Subject: Multics vs Unix
Posted by Anonymous on Tue, 17 Dec 2024 20:41:32 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

I was reading the introductory “Multics Concepts and Utilization” book
< http://bitsavers.trailing-edge.com/pdf/honeywell/large_syste
ms/multics/F01_multicsIntroCourseOct78.pdf>
over at Bitsavers. Multics (the “MULTiplexed Information and Computing
Service”) was, for its time, an extremely ambitious operating system
project. It was first introduced in 1965, in the form of a series of
papers at the AFIPS Fall Joint Computer Conference of that year
< https://www.computer.org/csdl/proceedings/1965/afips/12OmNzS h1au>.

Of course, it took far too long (7 years) to reach production quality.
In that time, a small group of researchers at AT&T Bell Labs grew
tired of waiting, and decided to create their own, less ambitious
system, which they called “UNIX” as a tongue-in-cheek homage to
“Multics”. And the rest, as they say, is history.

But, nevertheless, Multics remained an influential system. There are
even some present-day fans of it gathered at
<https://multicians.org/>. Apparently they have got the OS booting on
an emulator of the original GE 645 hardware. Though it was mostly
written in a high-level language (PL/I), Multics was never a portable
OS; to support its advanced virtual-memory and security features, it
required special processor hardware support which was not common in
those days.

Even today, Multics has some features which can be considered
innovative and uncommon. It may be true that, for example, SELinux can
match all of its security capabilities and more. But some aspects of
its file-protection system seem, to me, to make sharing of data
between users a bit easier than your typical Linux/POSIX-type system.

For a start, there seems to be no concept of file “ownership” as such.
Or even of POSIX-style file protection modes (read/write/execute for
owner/group/world). Instead, all file and directory access is
controlled via access-control lists (ACLs). Directories have a
permission called “modify”, which effectively gives a matching entity
(user, group, process) owner-type rights over that directory; except
that more than one entity can have that permission at once. Thus, a
group of users working on a common project can all be given this
“modify” access to a shared directory for that project, allowing them
all to put data there, read it back again, control access to it,
delete it etc on a completely equal basis. Contrast this with
POSIX/Linux, where every file has to have exactly one owner; even if

they create that file in a shared directory, it still gives the creating user a special status over that file, that others with write access to the containing directory do not have.

(Multics also offers a separate “append” permission, that allows the possessor to create an item in a directory, without having the ability to remove an item once it’s there.)

One radical idea introduced in Unix was its profligate use of multiple processes. Every new command you executed (except for the ones built into the shell) required the creation of a new process, often several processes. Other OSes tended to look askance at this; it seemed somehow wasteful, perhaps even sinful to spawn so many processes so readily and discard them so casually. The more conventional approach was to create a single process at user login, and execute nearly all commands within the context of that. There were special commands for explicitly creating additional processes (e.g. for background command execution), but such process creation did not simply happen as a matter of course.

Gradually, over time, the limitations of the single-process approach became too much to ignore, and the versatility of the Unix approach won over (nearly) everybody. Multics, however, is of the old school. More than that, the process even preserves global state, including static storage, in-between runs of programs, and this applies across different programs, not just reruns of the same one. For example, in FORTRAN, there is the concept of a “common block”. If you run two different programs that both refer to the same common block, then the second one will see values left in the block by the first one. To completely reinitialize everything, you need to invoke the “new_proc” command, which effectively deletes your process and gives you a fresh one.

One common irritation I find on POSIX/Linux systems is the convention that every directory has to have an entry called “.”, pointing to itself, and one called “..”, pointing to its parent. This way these names can be used in relative pathnames to reach any point in the directory hierarchy. But surely it is unnecessary to have explicit entries for these names cluttering up every directory; why not just build their recognition as a special case into the pathname-parsing logic in the kernel, once and for all? That way, directory-traversal routines in user programs don’t have to be specially coded to look for, and skip these entries, every single time.

Multics doesn’t seem to have this problem. An absolute pathname begins with “>” (which is the separator for pathname components, equivalent to POSIX “/”), while a relative pathname doesn’t. Furthermore, a relative pathname can begin with one or more “<” characters, indicating the corresponding number of steps up from the current working directory. Unlike POSIX “..”, you can’t have “<” characters in the middle of the pathname, which is probably not a big loss.

It is interesting to see other features which are nearly, but not quite, the same as, corresponding features in Unix. For example, there is a search path for executables, to save you typing the entire pathname to run the program. However, this does not seem as flexible as the $PATH environment-variable convention observed by Unix/POSIX shells. In particular, it does not seem possible to remove the current directory from the search path, which we now know can be a security risk.

Another one is the concept of “active functions” and “active strings”. These allow you to perform substitutions of dynamically-computed values into a command line. However, they are not as general as the Unix/POSIX concept of “command substitution”, where an entire shell command can supply its output to be interpolated into another command. Instead of having a completely separate vocabulary of “active functions” which can only be used for such substitutions, Unix/POSIX unifies this with the standard set of commands, any of which can be used in this way.

There are other features of Multics that others more familiar with it might want to see mentioned (the single-level store concept, where “everything is a memory segment”, versus Unix “everything is a file”? I/O redirection based on “switches”—symbolic references to files, versus Unix integer “file descriptors”?). But then, this long-winded essay would become even longer-winded :). So if you are interested in this particular piece of computing history, feel free to follow up the links above.

In summary, Multics is very much a museum piece, not something you would want to use today for regular work—not in its original form. But I think there are still one or two ideas there that we could usefully copy and adapt to a present-day OS, particularly a versatile one like Linux.

Subject: Re: Multics vs Unix
Posted by Anonymous on Wed, 18 Dec 2024 01:13:42 GMT
View Forum Message <> Reply to Message

Originally posted by: Grant Taylor

On 12/17/24 14:41, Lawrence D'Oliveiro wrote:
> But surely it is unnecessary to have explicit entries for these names
> cluttering up every directory; why not just build their recognition

> as a special case into the pathname-parsing logic in the kernel,
> once and for all?

Based on my limited understanding, the "in the kernel" bit is where your question runs off the rails.

My understanding is that the kernel doesn't know / care where what the path to the file is. Instead it cares about the file identifier, or inode. Remember, you can have the same file / inode appear in multiple directories a la hard links.

The directory path is largely a user space construct only with minimal kernel support in support of the user space.

As such, I don't think you can special case "." and ".." into the pathname-parsing logic in the kernel.

> That way, directory-traversal routines in user
> programs don’t have to be specially coded to look for, and skip
> these entries, every single time.

I question why you are wanting to treat "." and ".." special when you are working with dot files. It seems to me like if you are explicitly looking for dot files, then you'd want to see "." and "..".

--
Grant. . . .

Subject: Re: Multics vs Unix
Posted by Anonymous on Wed, 18 Dec 2024 01:20:09 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 17 Dec 2024 19:13:42 -0600, Grant Taylor wrote:

> My understanding is that the kernel doesn't know / care where what the
> path to the file is.

Oh, but it does.

> Instead it cares about the file identifier, or
> inode. Remember, you can have the same file / inode appear in multiple
> directories a la hard links.

Yes you can. But there is no userland API in POSIX/*nix to let you identify a file directly by inode. You always have to specify some path that gets to it. This is by design.

> As such, I don't think you can special case "." and ".." into the
> pathname-parsing logic in the kernel.

You already have to, for “..” at least. Think about what happens when “..” would take you to a different filesystem.

> I question why you are wanting to treat "." and ".." special when you
> are working with dot files. It seems to me like if you are explicitly
> looking for dot files, then you'd want to see "." and "..".

Typically, no.

Subject: Re: Multics vs Unix
Posted by Anonymous on Wed, 18 Dec 2024 13:04:28 GMT
View Forum Message <> Reply to Message

Originally posted by: antispam

>
> Of course, it took far too long (7 years) to reach production quality.
> In that time, a small group of researchers at AT&T Bell Labs grew
> tired of waiting, and decided to create their own, less ambitious
> system, which they called “UNIX” as a tongue-in-cheek homage t

What I read is a bit different: AT&T management got tired and decided to quit the project. Researchers at Bell Labs liked Multics features, did not want to loose them, so decided to do their own simpler variant.

--
Waldek Hebisch

Subject: Re: Multics vs Unix
Posted by Rich Alderson on Wed, 18 Dec 2024 23:26:21 GMT
View Forum Message <> Reply to Message

antispam@fricas.org (Waldek Hebisch) writes:

>> Of course, it took far too long (7 years) to reach production quality.
>> In that time, a small group of researchers at AT&T Bell Labs grew
>> tired of waiting, and decided to create their own, less ambitious

>> system, which they called "UNIX" as a tongue-in-cheek homage t

> What I read is a bit different: AT&T management got tired and
> decided to quit the project. Researchers at Bell Labs liked
> Multics features, did not want to loose them, so decided to
> do their own simpler variant.

That is the usual story; the poster to whom you responded often gets this kind
of thing wrong...

--
Rich Alderson   news@alderson.users.panix.com
    Audendum est, et veritas investiganda; quam etiamsi non assequamur,
  omnino tamen proprius, quam nunc sumus, ad eam perveniemus.
      --Galen

Subject: Re: Multics vs Unix
Posted by Niklas Karlsson on Thu, 19 Dec 2024 05:22:37 GMT
View Forum Message <> Reply to Message

On 2024-12-18, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Tue, 17 Dec 2024 19:13:42 -0600, Grant Taylor wrote:
>
>> Instead it cares about the file identifier, or
>> inode. Remember, you can have the same file / inode appear in multiple
>> directories a la hard links.
>
> Yes you can. But there is no userland API in POSIX/*nix to let you
> identify a file directly by inode. You always have to specify some path
> that gets to it. This is by design.

Hmm. I haven't gone grovelling through the API documentation, but I note
that find(1) on this machine (Ubuntu 18.04.6 LTS) has the -inum option
to find files by inode number. So unless find(1) does some serious
acrobatics for that, I do think there's a userland API to identify a
file by inode.

Niklas
--
All software sucks. Everybody is considered a jerk by somebody. The sun
rises, the sun sets, the Sun crashes, lusers are LARTed, BOFHs get drunk.
It is the way of things.   -- sconley@summit.bor.ohio.gov (Steve Conley)

Subject: Re: Multics vs Unix
Posted by Anonymous on Thu, 19 Dec 2024 07:11:35 GMT

View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On 19 Dec 2024 05:22:37 GMT, Niklas Karlsson wrote:

> Hmm. I haven't gone grovelling through the API documentation, but I note
> that find(1) on this machine (Ubuntu 18.04.6 LTS) has the -inum option
> to find files by inode number. So unless find(1) does some serious
> acrobatics for that ...

It calls stat(2) or lstat(2) and checks the st_ino field in the returned
data.

<https://savannah.gnu.org/projects/findutils/>

Subject: Re: Multics vs Unix
Posted by Niklas Karlsson on Thu, 19 Dec 2024 09:41:37 GMT
View Forum Message <> Reply to Message

On 2024-12-19, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On 19 Dec 2024 05:22:37 GMT, Niklas Karlsson wrote:
>
>> Hmm. I haven't gone grovelling through the API documentation, but I note
>> that find(1) on this machine (Ubuntu 18.04.6 LTS) has the -inum option
>> to find files by inode number. So unless find(1) does some serious
>> acrobatics for that ...
>
> It calls stat(2) or lstat(2) and checks the st_ino field in the returned
> data.

Looks like you're right; the only way to directly manipulate a file by
its inode number is through direct manipulation of the filesystem via
some FS-dependent tool (debugfs in the case of ext*).

Niklas
--
You know what the chain of command is? It's the chain I go get and beat
you with 'til you understand who's in ruttin' command here!
-- Jayne Cobb, _Firefly_

Subject: Re: Multics vs Unix
Posted by cross on Thu, 19 Dec 2024 12:28:11 GMT
View Forum Message <> Reply to Message

In article <mddbjx8wp2a.fsf@panix5.panix.com>,

Rich Alderson <news@alderson.users.panix.com> wrote:
> antispam@fricas.org (Waldek Hebisch) writes:
>
>>> Of course, it took far too long (7 years) to reach production quality.
>>> In that time, a small group of researchers at AT&T Bell Labs grew
>>> tired of waiting, and decided to create their own, less ambitious
>>> system, which they called "UNIX" as a tongue-in-cheek homage t
>
>> What I read is a bit different: AT&T management got tired and
>> decided to quit the project. Researchers at Bell Labs liked
>> Multics features, did not want to loose them, so decided to
>> do their own simpler variant.
>
> That is the usual story; the poster to whom you responded often gets this kind
> of thing wrong...

Indeed. It's honestly best not to engage with him; he's
a known troll.

 - Dan C.

Subject: Re: Multics vs Unix
Posted by scott on Thu, 19 Dec 2024 13:47:14 GMT

View Forum Message <> Reply to Message

Niklas Karlsson <nikke.karlsson@gmail.com> writes:
> On 2024-12-18, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>> On Tue, 17 Dec 2024 19:13:42 -0600, Grant Taylor wrote:
>>
>>> Instead it cares about the file identifier, or
>>> inode. Remember, you can have the same file / inode appear in multiple
>>> directories a la hard links.
>>
>> Yes you can. But there is no userland API in POSIX/*nix to let you
>> identify a file directly by inode. You always have to specify some path
>> that gets to it. This is by design.
>
> Hmm. I haven't gone grovelling through the API documentation, but I note
> that find(1) on this machine (Ubuntu 18.04.6 LTS) has the -inum option
> to find files by inode number. So unless find(1) does some serious
> acrobatics for that, I do think there's a userland API to identify a
> file by inode.

Admins used to use ncheck(8) to map inode numbers to path name(s)
in bell labs versions of Unix.

find(1) uses 'stat(2)' to get the inode number when walking the filesystem tree, so it's more a brute force method than an API.

Subject: Re: Multics vs Unix
Posted by Niklas Karlsson on Thu, 19 Dec 2024 14:13:01 GMT
View Forum Message <> Reply to Message

On 2024-12-19, Scott Lurndal <scott@slp53.sl.home> wrote:
> Niklas Karlsson <nikke.karlsson@gmail.com> writes:
>>
>> Hmm. I haven't gone grovelling through the API documentation, but I note
>> that find(1) on this machine (Ubuntu 18.04.6 LTS) has the -inum option
>> to find files by inode number. So unless find(1) does some serious
>> acrobatics for that, I do think there's a userland API to identify a
>> file by inode.
>
> Admins used to use ncheck(8) to map inode numbers to path name(s)
> in bell labs versions of Unix.

Oh, hello. That's an interesting fact. I like it when things like this
come up in discussions here. Thank you!

I found https://illumos.org/man/8/ncheck - interesting!

A quick web search suggests that at least some commercial UNIXes still
have it; Illumos is of course a Solaris derivative, and I found a
reference to it existing on AIX as well. On Linux, assuming ext*, you
apparently have to use debugfs to do something similar.

Niklas
--
> I've wondered recently why it's not feasible to make large passenger planes
> capable of water landing.
Well, they keep having to replace all the seat cushions, for one
thing... -- Mike Sphar and Mark Hughes in asr

Subject: Re: Multics vs Unix
Posted by scott on Thu, 19 Dec 2024 14:47:37 GMT
View Forum Message <> Reply to Message

Niklas Karlsson <nikke.karlsson@gmail.com> writes:
> On 2024-12-19, Scott Lurndal <scott@slp53.sl.home> wrote:
>> Niklas Karlsson <nikke.karlsson@gmail.com> writes:
>>>
>>> Hmm. I haven't gone grovelling through the API documentation, but I note
>>> that find(1) on this machine (Ubuntu 18.04.6 LTS) has the -inum option

>>> to find files by inode number. So unless find(1) does some serious
>>> acrobatics for that, I do think there's a userland API to identify a
>>> file by inode.
>>
>> Admins used to use ncheck(8) to map inode numbers to path name(s)
>> in bell labs versions of Unix.
>
> Oh, hello. That's an interesting fact. I like it when things like this
> come up in discussions here. Thank you!
>
> I found https://illumos.org/man/8/ncheck - interesting!

Unix V7 version:

```
$ PAGER= man /reference/usl/unix/v7/usr/man/man1/ncheck.1m
NCHECK(1M)                                          NCHECK(1M)



NAME
     ncheck  -  generate names from i-numbers

SYNOPSIS
     ncheck [ -i numbers ]  [ -a ] [ -s ]  [ filesystem ]

DESCRIPTION
     Ncheck  with  no argument generates a pathname vs. i-number list of all
     files on a set of default file systems.  Names of directory  files  are
     followed by `/.'.  The -i option reduces the report to only those files
     whose i-numbers follow.  The -a option allows printing of the names `.'
     and  `..', which are ordinarily suppressed.  suppressed.  The -s option
     reduces the report to special files and files with set-user-ID mode; it
     is intended to discover concealed violations of security policy.

     A file system may be specified.

     The report is in no useful order, and probably should be sorted.

SEE ALSO
     dcheck(1), icheck(1), sort(1)

DIAGNOSTICS
     When the filesystem structure is improper, `??' denotes the `parent' of
     a parentless file and a pathname beginning with `...' denotes a loop.
```

Subject: Re: Multics vs Unix

Posted by Anonymous on Thu, 19 Dec 2024 15:03:55 GMT
View Forum Message <> Reply to Message

Originally posted by: Bob Eager

On Thu, 19 Dec 2024 13:47:14 +0000, Scott Lurndal wrote:

> Admins used to use ncheck(8) to map inode numbers to path name(s)
> in bell labs versions of Unix.
>
> find(1) uses 'stat(2)' to get the inode number when walking the
> filesystem tree, so it's more a brute force method than an API.

Yes, I remember that on Sixth Edition.

Not to mention icheck.

--
Using UNIX since v6 (1975)...

Use the BIG mirror service in the UK:
 http://www.mirrorservice.org

Subject: Re: Multics vs Unix
Posted by cross on Thu, 19 Dec 2024 15:04:59 GMT
View Forum Message <> Reply to Message

In article <lsinvdFrpltU1@mid.individual.net>,
Niklas Karlsson <nikke.karlsson@gmail.com> wrote:
> On 2024-12-19, Scott Lurndal <scott@slp53.sl.home> wrote:
>> Niklas Karlsson <nikke.karlsson@gmail.com> writes:
>>>
>>> Hmm. I haven't gone grovelling through the API documentation, but I note
>>> that find(1) on this machine (Ubuntu 18.04.6 LTS) has the -inum option
>>> to find files by inode number. So unless find(1) does some serious
>>> acrobatics for that, I do think there's a userland API to identify a
>>> file by inode.
>>
>> Admins used to use ncheck(8) to map inode numbers to path name(s)
>> in bell labs versions of Unix.
>
> Oh, hello. That's an interesting fact. I like it when things like this
> come up in discussions here. Thank you!
>

> I found https://illumos.org/man/8/ncheck - interesting!
>
> A quick web search suggests that at least some commercial UNIXes still
> have it; Illumos is of course a Solaris derivative, and I found a
> reference to it existing on AIX as well. On Linux, assuming ext*, you
> apparently have to use debugfs to do something similar.

Various systems have had extensions to address files by inum
over time. From memory, systems that supported AFS used to add
an `openi` system call that allowed a user to open a file by
inode number (one presumes it also took some kind of reference
to the filesystem that the inode was relative to, since the name
space of inodes is per-fs, and not globally unique).

Hmm. Maybe that was Coda, and not AFS.

 - Dan C.

Subject: Re: Multics vs Unix
Posted by Anonymous on Thu, 19 Dec 2024 19:48:46 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On 19 Dec 2024 09:41:37 GMT, Niklas Karlsson wrote:

> On 2024-12-19, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>
>> On 19 Dec 2024 05:22:37 GMT, Niklas Karlsson wrote:
>>
>>> Hmm. I haven't gone grovelling through the API documentation, but I
>>> note that find(1) on this machine (Ubuntu 18.04.6 LTS) has the -inum
>>> option to find files by inode number. So unless find(1) does some
>>> serious acrobatics for that ...
>>
>> It calls stat(2) or lstat(2) and checks the st_ino field in the
>> returned data.
>
> Looks like you're right; the only way to directly manipulate a file by
> its inode number is through direct manipulation of the filesystem via
> some FS-dependent tool (debugfs in the case of ext*).

Having said that, Linux does offer “handle” calls
<https://manpages.debian.org/2/open_by_handle_at.2.en.html>, which
very likely include inode info somewhere in that opaque structure.

But note that accessing files in this way can only be done by

suitably-privileged processes. Otherwise it would break the POSIX
security model.

Subject: Re: Multics vs Unix
Posted by Anonymous on Thu, 19 Dec 2024 21:56:50 GMT
View Forum Message <> Reply to Message

Originally posted by: OrangeFish

On 2024-12-19 09:13, Niklas Karlsson wrote:
> On 2024-12-19, Scott Lurndal <scott@slp53.sl.home> wrote:
>> Niklas Karlsson <nikke.karlsson@gmail.com> writes:
>>>
>>> Hmm. I haven't gone grovelling through the API documentation, but I note
>>> that find(1) on this machine (Ubuntu 18.04.6 LTS) has the -inum option
>>> to find files by inode number. So unless find(1) does some serious
>>> acrobatics for that, I do think there's a userland API to identify a
>>> file by inode.
>>
>> Admins used to use ncheck(8) to map inode numbers to path name(s)
>> in bell labs versions of Unix.
>
> Oh, hello. That's an interesting fact. I like it when things like this
> come up in discussions here. Thank you!
>
> I found https://illumos.org/man/8/ncheck - interesting!
>
> A quick web search suggests that at least some commercial UNIXes still
> have it; Illumos is of course a Solaris derivative, and I found a
> reference to it existing on AIX as well. On Linux, assuming ext*, you
> apparently have to use debugfs to do something similar.
>
> Niklas

Solaris 11 still has it in /usr/sbin:

```
System Administration Commands                    ncheck(1M)

NAME
       ncheck - generate a list of path names versus i-numbers

SYNOPSIS
       ncheck [-F FSType] [-V] [generic_options]
           [-o FSType-specific_options] [special]...

DESCRIPTION
       ncheck with no options generates a path-name versus
```

```
        i-number list of all files on special. If special is not
        specified on the command line the list is generated for
        all specials in /etc/vfstab which have a numeric
        fsckpass. special is the raw device on which the file
        system exists.

OPTIONS
        -F   Specify the  FSType on which to operate. The  FSType
             should either be specified here  or be  determinable
             from  /etc/vfstab  by  finding an entry in the table
             that has a numeric  fsckpass field and  an   fsckdev
             that matches special.

        -V   Echo  the  complete command line, but do not execute
             the command. The command line is generated by  using
             the  options  and arguments provided by the user and
             adding   to   them    information    derived    from
             /etc/vfstab.  This  option may be used to verify and
             validate the command line.

[and so on]
```

OF

Subject: Re: Multics vs Unix
Posted by Anonymous on Mon, 23 Dec 2024 18:20:46 GMT
View Forum Message <> Reply to Message

Originally posted by: Sarr Blumson

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> I was reading the introductory ???Multics Concepts and Utilization??? book
> < http://bitsavers.trailing-edge.com/pdf/honeywell/large_syste
ms/multics/F01_multicsIntroCourseOct78.pdf>
> over at Bitsavers. Multics (the ???MULTiplexed Information and Computing
> Service???) was, for its time, an extremely ambitious operating system
> project. It was first introduced in 1965, in the form of a series of
> papers at the AFIPS Fall Joint Computer Conference of that year
> < https://www.computer.org/csdl/proceedings/1965/afips/12OmNzS h1au>.
>
> Of course, it took far too long (7 years) to reach production quality.
> In that time, a small group of researchers at AT&T Bell Labs grew
> tired of waiting, and decided to create their own, less ambitious
> system, which they called ???UNIX??? as a tongue-in-cheek homage to
> ???Multics???. And the rest, as they say, is history.

Of course it was a decade or two before UNIX could support and AT&T sized organization.

IBM got pissed when Bell Labs chose GE and cut a similar deal with the
University of Michigan. IBM tried to build TSS/360 on their own; UM
gave up after a similar delay but took the simple system route but
built their simple system on the 360/70. And ran it on IBM hardware for
40 years. An IBM Labs group built CP67/CMS->VM370 with good commercial
success.

Multics put a whole series of hardware vnedors out of business (GE, Honeywell,
Bull) but IBM had infinite resources.

--
sarr@sdf.org
SDF Public Access UNIX System - http://sdf.org

Subject: Re: Multics vs Unix
Posted by Anonymous on Mon, 23 Dec 2024 20:16:48 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Mon, 23 Dec 2024 18:20:46 -0000 (UTC), Sarr Blumson wrote:

> An IBM Labs group built CP67/CMS->VM370 with good commercial
> success.

That was a pretty crummy way of building a timesharing system, though; CMS
was “interactive” (insofar as IBM understood the term), but it was not
multiuser. So CP (later VM) was tacked on as an extra layer underneath to
allow each user to run their own instance of CMS.

This is normally hailed as “IBM pioneered virtualization”. But it was just
a less flexible, higher-overhead way of supporting multiple users than
other vendors, like DEC, were able to do far more efficiently.

Subject: Re: Multics vs Unix
Posted by Peter Flass on Mon, 23 Dec 2024 20:21:42 GMT
View Forum Message <> Reply to Message

Sarr Blumson <sarr@sdf.org> wrote:
> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>> I was reading the introductory ???Multics Concepts and Utilization??? book
>> < http://bitsavers.trailing-edge.com/pdf/honeywell/large_syste
ms/multics/F01_multicsIntroCourseOct78.pdf>

>> over at Bitsavers. Multics (the ???MULTiplexed Information and Computing
>> Service???) was, for its time, an extremely ambitious operating system
>> project. It was first introduced in 1965, in the form of a series of
>> papers at the AFIPS Fall Joint Computer Conference of that year
>> < https://www.computer.org/csdl/proceedings/1965/afips/12OmNzS h1au>.
>>
>> Of course, it took far too long (7 years) to reach production quality.
>> In that time, a small group of researchers at AT&T Bell Labs grew
>> tired of waiting, and decided to create their own, less ambitious
>> system, which they called ???UNIX??? as a tongue-in-cheek homage to
>> ???Multics???. And the rest, as they say, is history.
>
> Of course it was a decade or two before UNIX could support and AT&T sized
> organization.
>
> IBM got pissed when Bell Labs chose GE and cut a similar deal with the
> University of Michigan. IBM tried to build TSS/360 on their own; UM
> gave up after a similar delay but took the simple system route but
> built their simple system on the 360/70. And ran it on IBM hardware for
> 40 years. An IBM Labs group built CP67/CMS->VM370 with good commercial
> success.
>
> Multics put a whole series of hardware vnedors out of business (GE, Honeywell,
> Bull) but IBM had infinite resources.
>

You can still run MTS today.

--
Pete

Subject: Re: Multics vs Unix
Posted by Anonymous on Mon, 23 Dec 2024 21:54:48 GMT
View Forum Message <> Reply to Message

Originally posted by: Grant Taylor

On 12/23/24 14:16, Lawrence D'Oliveiro wrote:
> This is normally hailed as “IBM pioneered virtualization”. But
> it was just a less flexible, higher-overhead way of supporting
> multiple users than other vendors, like DEC, were able to do far
> more efficiently.

I don't see either of those statements as being incompatible with each
other.

I'm not aware of any prior efforts that could be described as virtualization in the sense that VM / VMware / KVM mean it.

I agree that separate (virtual) systems for people is an inefficient way
to support multiple users.

--
Grant. . . .

Subject: Re: Multics vs Unix
Posted by Peter Flass on Tue, 24 Dec 2024 00:35:15 GMT
View Forum Message <> Reply to Message

Grant Taylor <gtaylor@tnetconsulting.net> wrote:
> On 12/23/24 14:16, Lawrence D'Oliveiro wrote:
>> This is normally hailed as “IBM pioneered virtualization”. But
>> it was just a less flexible, higher-overhead way of supporting
>> multiple users than other vendors, like DEC, were able to do far
>> more efficiently.
>
> I don't see either of those statements as being incompatible with each
> other.
>
> I'm not aware of any prior efforts that could be described as
> virtualization in the sense that VM / VMware / KVM mean it.
>
> I agree that separate (virtual) systems for people is an inefficient way
> to support multiple users.
>

Great for security, though. Service bureaus loved CP and VM to make sure
customer data stayed secure.

--
Pete

Subject: Re: Multics vs Unix
Posted by Anonymous on Tue, 24 Dec 2024 02:22:04 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Mon, 23 Dec 2024 17:35:15 -0700, Peter Flass wrote:

> Service bureaus loved CP and VM to make sure customer data stayed
> secure.

Service bureaus commonly used multiuser timeshared systems, relying on OS protections to keep users out of each other's data.

Subject: Re: Multics vs Unix
Posted by Anonymous on Tue, 24 Dec 2024 02:57:00 GMT
View Forum Message <> Reply to Message

Originally posted by: Grant Taylor

On 12/23/24 18:35, Peter Flass wrote:
> Great for security, though. Service bureaus loved CP and VM to make
> sure customer data stayed secure.

Was it better for security than physically separate machines?

Or was it a compromise that behaved like separate machines without paying for multiple machines? ;-)

--
Grant. . . .

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Tue, 24 Dec 2024 03:44:06 GMT
View Forum Message <> Reply to Message

Grant Taylor <gtaylor@tnetconsulting.net> writes:
> Was it better for security than physically separate machines?
>
> Or was it a compromise that behaved like separate machines without
> paying for multiple machines? ;-)

CP(VM) kernel was relatively small amount of source code with well defined interface and relatively simple to modify and audit ... which govs & service bureaus tended to further restrict (user group presentations about "padded cells" for general users ... only allowing full virtual machine capability for specific purposes). Because of the clean separation it tended to reduce amount and complexity of code .... so it was easier to address both performance and security issues.

In the 80s, as mainframes got larger, there appeared CP/VM subset

functions implemented directly in hardware&microcode to partition
machines ... LPAR & PR/SM
https://en.wikipedia.org/wiki/Logical_partition
which now can be found on many platforms, not just IBM mainframes ...
heavily leveraged by large cloud datacenter operations
https://aws.amazon.com/what-is/virtualization/
How is virtualization different from cloud computing?

Cloud computing is the on-demand delivery of computing resources over
the internet with pay-as-you-go pricing. Instead of buying, owning, and
maintaining a physical data center, you can access technology services,
such as computing power, storage, and databases, as you need them from a
cloud provider.

Virtualization technology makes cloud computing possible. Cloud
providers set up and maintain their own data centers. They create
different virtual environments that use the underlying hardware
resources. You can then program your system to access these cloud
resources by using APIs. Your infrastructure needs can be met as a fully
managed service.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: CP/67 Multics vs Unix
Posted by John Levine on Tue, 24 Dec 2024 04:07:10 GMT
View Forum Message <> Reply to Message

According to Grant Taylor <gtaylor@tnetconsulting.net>:
> I'm not aware of any prior efforts that could be described as
> virtualization in the sense that VM / VMware / KVM mean it.

I am fairly sure that CP/40 and CP/67 were the first virtual machine operating systems.

It was apparently a stroke of luck that S/360 had a clean enough separation between
system and user modes that it was possible to virtualizze. The PDP-6/10 were designed
at about the same time but couldn't

> I agree that separate (virtual) systems for people is an inefficient way
> to support multiple users.

True, but it was an extremely cost efficient way to do system program development.

I think Lynn will confirm that the CP system was so well designed that it got
good performance even without the memory sharing other systems might do. The IBM

channel architecture made the overhead of simulating I/O fairly cheap, since
each I/O operation did a lot of work, read an entire card, print an entire line,
seek and read or write a disk block.
--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Tue, 24 Dec 2024 05:48:56 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 24 Dec 2024 04:07:10 -0000 (UTC), John Levine wrote:

> According to Grant Taylor <gtaylor@tnetconsulting.net>:
>
>> I agree that separate (virtual) systems for people is an inefficient
>> way to support multiple users.
>
> True, but it was an extremely cost efficient way to do system program
> development.

There is an even more cost-efficient way: containers.

> The IBM channel architecture made the overhead of simulating
> I/O fairly cheap, since each I/O operation did a lot of work, read an
> entire card, print an entire line, seek and read or write a disk block.

I understand there were security holes in that: the hypervisor tended to
trust that the channel programs submitted by the individual VMs were well-
behaved, even while VM users had full control over their particular VMs.

Subject: Re: CP/67 Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Tue, 24 Dec 2024 08:06:06 GMT
View Forum Message <> Reply to Message

John Levine <johnl@taugh.com> writes:
> I think Lynn will confirm that the CP system was so well designed that it got
> good performance even without the memory sharing other systems might do. The IBM
> channel architecture made the overhead of simulating I/O fairly cheap, since
> each I/O operation did a lot of work, read an entire card, print an entire line,
> seek and read or write a disk block.

Melinda's virtual machine history info
https://www.leeandmelindavarian.com/Melinda#VMHist

Univ. had got 360/67 to replace 709/1401 but ran as 360/65 with os/360 and I was still undergraduate but hired fulltime responsible for OS/360. Univ. shutdown datacenter on weekends and I would have it dedicated although 48hrs w/o sleep made monday classes hard.

CSC then came out and installed CP67 (3rd after CSC itself and MIT Lincoln labs) and I mostly played with it in my weekend dedicated time. Initially I mostly concentrated on pathlengths to improving running OS/360 in virtual machine. My OS/360 job stream ran 322secs on real machine, initially virtually ran 856secs (534secs CP67 CPU). After a couple months I had CP67 CPU down to 113secs (from 534). I then start redoing other parts of CP67, page replacement, dynamic adaptive resource management, scheduling and page thrashing controls, ordered arm seek queueing (from FIFO), multiple chained page transfers maximizing transfer/revolution (2301 paging drum improved from 80/sec able to do 270/sec peak), etc. Most of this CSC picks up for distribution in standard CP67.

After graduation, I join CSC and one of my hobbies was enhanced production operating systems for internal datacenters (the world-wide, online, sales&marketing support HONE systems was early and long-time customer). After decision to add virtual memory to all 370s, the morph of CP67->VM370 dropped or simplified a lot of stuff. During 1974 and early 1975, I was able to get most of it back into VM370R2 and then VM370R3.

In the wake of Future System implosion, Endicott ropes me into helping with VM/370 ECPS microcode assist for 370 138/148 ... basically identify 6kbytes of highest executed VM370 kernel paths for moving into microcode. 138/148 avg. 10 native instruction per emulated 370 instruction and kernel 370 instruction would translate approx. one-for-one into native ... getting 10 times speed up. Old archived a.f.c post with initial analysis
https://www.garlic.com/~lynn/94.html#21

6kbytes instructions accounted for 79.55% of kernel execution (moved to native running ten times faster) ... a lot of it involved simulated I/O (it had to make a copy of the virtual channel programs substituting real addresses for virtual, the corresponding virtual pages also had to be "fixed" in real storage until the VM I/O had completed)

Science Center was on 4th flr and Multics was on the 5th ... looking at some amount of Multics, I figured I could do page mapped filesystem with lots of sharing features (which was faster and much less CPU than the standard requiring I/O emulation). Note that "Future System" did

single-level-store ala Multics and (IBM) TSS/360 ... I had joked I had
learned what not to do from TSS/360. However when FS imploded it gave
anything that even slightly related to single-level-store a bad
reputation (and I had trouble even getting my CMS paged-mapped
filesystem used internally inside IBM).

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: CP/67 Multics vs Unix
Posted by Peter Flass on Tue, 24 Dec 2024 13:50:00 GMT

View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Tue, 24 Dec 2024 04:07:10 -0000 (UTC), John Levine wrote:
>
>> According to Grant Taylor <gtaylor@tnetconsulting.net>:
>>
>>> I agree that separate (virtual) systems for people is an inefficient
>>> way to support multiple users.
>>
>> True, but it was an extremely cost efficient way to do system program
>> development.
>
> There is an even more cost-efficient way: containers.
>
>> The IBM channel architecture made the overhead of simulating
>> I/O fairly cheap, since each I/O operation did a lot of work, read an
>> entire card, print an entire line, seek and read or write a disk block.
>
> I understand there were security holes in that: the hypervisor tended to
> trust that the channel programs submitted by the individual VMs were well-
> behaved, even while VM users had full control over their particular VMs.
>
>

All disks are either minidisks or dedicated packs, so it's easy for VM to
ensure that users stay within their limits. Cards and printers are spooled,
and tapes are attached to a one user at a time, so it's very difficult to
break security.

Channels fave a "file mask" that blocks seeks to another cylinder. The
first thing a dasd channel program does is seek cylinder and then set file
mask. After that any attempt to go elsewhere is trapped so the OS can
validate the new cylinder address before continuing.

--

Pete

Subject: Re: CP/67 Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Tue, 24 Dec 2024 18:40:50 GMT
View Forum Message <> Reply to Message

trivia: CSC CP67 had 1052&2741 support, but univ. had some number of
TTY/ASCII terminals, so I added TTY/ASCII support ... and CSC picked up
and distributed with standard CP67 (as well as lots of my other
stuff). I had done a hack with one byte values for TTY line
input/output. Tale of MIT Urban Lab having CP/67 (in tech sq bldg across
quad from 545, multics & science center). Somebody down at Harvard got
an ascii device with 1200(?) char length ... they modified CP67 field
for max. lengths ... but didn't adjust my one-byte hack.
https://www.multicians.org/thvv/360-67.html

A user at Harvard School of Public Health had connected a plotter to a
TTY line and was sending graphics to it, and every time he did, the
whole system crashed. (It is a tribute to the CP/CMS recovery system
that we could get 27 crashes in in a single day; recovery was fast and
automatic, on the order of 4-5 minutes. Multics was also crashing quite
often at that time, but each crash took an hour to recover because we
salvaged the entire file system. This unfavorable comparison was one
reason that the Multics team began development of the New Storage
System.)

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Tue, 24 Dec 2024 20:52:18 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 24 Dec 2024 08:40:50 -1000, Lynn Wheeler wrote:

> Multics was also crashing quite
> often at that time, but each crash took an hour to recover because we
> salvaged the entire file system. This unfavorable comparison was one
> reason that the Multics team began development of the New Storage
> System.)

Was the difference to do with storing allocation bitmaps on disk, instead
of only in the OS memory? Or just fixing up inconsistencies between the

two?

I recall our main campus PDP-11/70 system, back in 1979/1980 or so, would take about a quarter of an hour to recover from a crash. The error you got from trying to mount an improperly dismounted volume was “?Disk pack needs CLEANing”.

I think journalling filesystems were being developed through the 1990s. Then DEC took the idea to its logical conclusion in the early 1990s with something called “Spiralog”, which was a filesystem that was all journal and no actual (conventional) filesystem.

Subject: Re: CP/67 Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Tue, 24 Dec 2024 23:49:54 GMT
View Forum Message <> Reply to Message

Science Center and couple of the commercial online CP67 service spin-offs in the 60s did a lot of work for 7x24, dark-room, unattended operation. Also 60s was when IBM leased/rented machines with charges based on the "system meter" that ran whenever any cpu or any channel (I/O) was busy ... and a lot of work was done allowing system meter to stop when system was otherwise idle (there had to be no activity at all for at least 400ms before system meter would stop). One was special terminal I/O channel programs that would go idle (allowing system meter to stop), but immediately start up whenever characters were arriving.

trivia: long after IBM had switched to selling machines, (IBM batch) MVS system still had a 400ms timer event that guaranteed that system meter never stopped.

Late 80s, for IBM Austin RS/6000, AIX filesystem was modified to journal filesystem metadata changes using transaction memory (RIOS hardware that tracked changed memory) ... in part claiming it was more efficient.

Got the HA/6000 project in the late 80s, originally for NYTimes to move their newspaper system (ATEX) off DEC VAXCluster to RS/6000. AIX JFS enabled hot-standby unix filesystem take-over ... and some of RDBMS vendors supported (raw unix) concurrent shared disks.

Then IBM Palo Alto was porting journaled filesystem to machines that didn't have transaction memory and found that transaction journaling calls outperformed transaction memory (even when ported back to RS/6000).

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: CP/67 Multics vs Unix
Posted by Peter Flass on Thu, 26 Dec 2024 01:14:12 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Tue, 24 Dec 2024 08:40:50 -1000, Lynn Wheeler wrote:
>
>> Multics was also crashing quite
>> often at that time, but each crash took an hour to recover because we
>> salvaged the entire file system. This unfavorable comparison was one
>> reason that the Multics team began development of the New Storage
>> System.)
>
> Was the difference to do with storing allocation bitmaps on disk, instead
> of only in the OS memory? Or just fixing up inconsistencies between the
> two?
>
> I recall our main campus PDP-11/70 system, back in 1979/1980 or so, would
> take about a quarter of an hour to recover from a crash. The error you got
> from trying to mount an improperly dismounted volume was “?Disk pack needs
> CLEANing”.
>
> I think journalling filesystems were being developed through the 1990s.
> Then DEC took the idea to its logical conclusion in the early 1990s with
> something called “Spiralog”, which was a filesystem that was all journal
> and no actual (conventional) filesystem.
>

Obviously with minidisks no recovery was needed. I think the CMS filesystem
kept everything on disk, so no recovery of user data either, though I could
be wrong here. (I wrote the Wikipedia article about the CMS filesystem, but
I don’t recall the details at the present)

--
Pete

Subject: Re: CP/67 Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Thu, 26 Dec 2024 19:55:24 GMT
View Forum Message <> Reply to Message

Peter Flass <peter_flass@yahoo.com> writes:
> Obviously with minidisks no recovery was needed. I think the CMS
> filesystem kept everything on disk, so no recovery of user data
> either, though I could be wrong here. (I wrote the Wikipedia article
> about the CMS filesystem, but I don’t recall the details at the
> present)

when CMS was updating filesystem metadata (allocated blocks, allocated files, location of files and associated records), it was always to new disk record locations ... and last thing was to rewrite master record that switched from the old metadata to the new metadata in single record write.

Around the time of transition from CP67/CMS to VM370/CMS, it was found that IBM 360&370 (CKD) disk I/O had a particular failure mode during power failure, the system memory could have lost all power but the CKD disk and channel could have enough power to finish a write operation in progress ... but since there was no power to memory it would finish the write with all zeros and then write record error check based on the propagated zeros. CMS was enhanced to have a pair of master records and update would alternate between the two with basically a version number appended at the end (so any partial zero write wouldn't identify it was most recent & valid).

This was later fixed for fixed-block disks ... where a write wouldn't start until it had all the data from memory (i.e. countermeasure to partial record writes with trailing zeros) ... but CKD disks and other IBM operating systems (that didn't have FBA disk support) tended to still be vulnerable to this particular power failure problem.

other trivia: 60s, IBM rented/leased computers, charges based on "system meter" that ran whenever any cpu or channel was busy. CSC and a couple of the CSC CP67 commercial online spinoffs, did a lot of work for 7x24, dark room, unattended operation, optimized processing and channel programs so "system meter" could stop during idle periods (including special terminal channel programs that would release the channel with no activity, but instantly on with arriving characters). "System Meter" needed 400ms of complete idle before it would stop .... long after IBM had switched to selling computers, MVS still had a 400ms timer event that would guarantee system meter never stopped.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: CP/67 Multics vs Unix
Posted by scott on Fri, 27 Dec 2024 17:35:50 GMT

View Forum Message <> Reply to Message

Peter Flass <peter_flass@yahoo.com> writes:
> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>> On Tue, 24 Dec 2024 08:40:50 -1000, Lynn Wheeler wrote:
>>
>>> Multics was also crashing quite
>>> often at that time, but each crash took an hour to recover because we

>>> salvaged the entire file system. This unfavorable comparison was one
>>> reason that the Multics team began development of the New Storage
>>> System.)
>>
>> Was the difference to do with storing allocation bitmaps on disk, instead
>> of only in the OS memory? Or just fixing up inconsistencies between the
>> two?
>>
>> I recall our main campus PDP-11/70 system, back in 1979/1980 or so, would
>> take about a quarter of an hour to recover from a crash. The error you got
>> from trying to mount an improperly dismounted volume was “?Disk pack needs
>> CLEANing”.
>>
>> I think journalling filesystems were being developed through the 1990s.
>> Then DEC took the idea to its logical conclusion in the early 1990s with
>> something called “Spiralog”, which was a filesystem that was all journal
>> and no actual (conventional) filesystem.
>>
>
> Obviously with minidisks no recovery was needed. I think the CMS filesystem
> kept everything on disk, so no recovery of user data either, though I could
> be wrong here. (I wrote the Wikipedia article about the CMS filesystem, but
> I don’t recall the details at the present)

Veritas was an early player in the journaled filesystem space.

https://en.wikipedia.org/wiki/Veritas_File_System

Subject: Re: CP/67 Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Fri, 27 Dec 2024 19:35:21 GMT
View Forum Message <> Reply to Message

Lynn Wheeler <lynn@garlic.com> writes:
> Got the HA/6000 project in the late 80s, originally for NYTimes to move
> their newspaper system (ATEX) off DEC VAXCluster to RS/6000. AIX JFS
> enabled hot-standby unix filesystem take-over ... and some of RDBMS
> vendors supported (raw unix) concurrent shared disks.
>
> Then IBM Palo Alto was porting journaled filesystem to machines that
> didn't have transaction memory and found that transaction journaling
> calls outperformed transaction memory (even when ported back to
> RS/6000).

RS/6000 AIX with Journal Filesystem released in 1990
https://en.wikipedia.org/wiki/IBM_AIX

AIX was the first operating system to implement a journaling file system. IBM has continuously enhanced the software with features such as processor, disk, and network virtualization, dynamic hardware resource allocation (including fractional processor units), and reliability engineering concepts derived from its mainframe designs.[8]

In 1990, AIX Version 3 was released for the POWER-based RS/6000 platform.[16] It became the primary operating system for the RS/6000 series, which was later renamed IBM eServer pSeries, IBM System p, and finally IBM Power Systems.

.....

Nick Donofrio approved HA/6000 1988 (required the journal filesystem that would be part of RS/6000 1990 release) ... and started at the IBM Los Gatos lab Jan1989 (I rename it HA/CMP when start doing technical/scientific cluster scaleup with national labs, LLNL, LANL, NCAR, etc and commercial cluster scaleup with RDBMS vendors, Oracle, Sybase, Ingres, Informix.

27 Years of IBM RISC
http://ps-2.kev009.com/rootvg/column_risc.htm
1990 POWER

IBM announces its new RISC-based computer line, the RISC System/6000 (later named RS/6000, nowadays eServer pSeries), running AIX Version 3. The architecture of the systems is given the name POWER (now commonly referred to as POWER1), standing for Performance Optimization With Enhanced RISC. They where based on a multiple chip implementation of the 32-bit POWER architecture. The models introduced included an 8 KB instruction cache (I-cache) and either a 32 KB or 64 KB data cache (D-cache). They had a single floating-point unit capable of issuing one compound floating-point multiply-add (FMA) operation each cycle, with a latency of only two cycles and optimized 3-D graphics capabilities.

The model 7013-540 (30 MHz) processed 30 million instructions per second. Its electronic logic circuitry had up to 800,000 transistors per silicon chip. The maximum memory size was 256 Mbytes and its internal disk storage capacity was 2.5 GBytes.

Links: (for URLs see web page)
RISC System/6000 POWERstation/POWERserver 320
RISC System/6000 POWERstations/POWERservers 520 AND 530
RISC System/6000 POWERserver 540
RISC System/6000 POWERstation 730
RISC System/6000 POWERserver 930

AIX Version 3

AIX Version 3 is announced.

Links: (for URLs see web page)

AIX Version 3 (Februari, 1990)
Overview: IBM RISC System/6000 and related announcements

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Sat, 28 Dec 2024 23:25:07 GMT
View Forum Message <> Reply to Message

Originally posted by: Grant Taylor

On 12/27/24 13:35, Lynn Wheeler wrote:
> In 1990, AIX Version 3 was released for the POWER-based RS/6000
> platform.[16] It became the primary operating system for the RS/6000
> series, which was later renamed IBM eServer pSeries, IBM System p,
> and finally IBM Power Systems.

What other OSs ran on the RS/6000 in 1990?

I'm confident that NetBSD and know that Linux eventually made it to the
RS/6000. But I have no idea what else would run on it in '90.

--
Grant. . . .

Subject: Re: CP/67 Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Sun, 29 Dec 2024 03:04:15 GMT
View Forum Message <> Reply to Message

Grant Taylor <gtaylor@tnetconsulting.net> writes:
> I'm confident that NetBSD and know that Linux eventually made it to
> the RS/6000. But I have no idea what else would run on it in '90.

(AT&T unix port) AIXV2 and (UCB BSD port) AOS ran on PC/RT. They then
added bunch of BSD'isms for AIXV3 for 1990 RS/6000 (RIOS power chipset).
Then start AIM (apple, IBM, Motorola) & Somerset, single chip
power/pc.

https://en.wikipedia.org/wiki/IBM_RS/6000
https://www.ibm.com/docs/en/power4?topic=rs6000-systems

so must of non-AIX systems are going to be for power/pc ... and then
power & power/pc eventually merge.
https://en.wikipedia.org/wiki/IBM_Power_microprocessors

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: CP/67 Multics vs Unix
Posted by cb on Sun, 29 Dec 2024 10:28:03 GMT
View Forum Message <> Reply to Message

In article <vkq1cj$rur$1@tncsrv09.home.tnetconsulting.net>,
Grant Taylor <gtaylor@tnetconsulting.net> wrote:
> On 12/27/24 13:35, Lynn Wheeler wrote:
>> In 1990, AIX Version 3 was released for the POWER-based RS/6000
>> platform.[16] It became the primary operating system for the RS/6000
>> series, which was later renamed IBM eServer pSeries, IBM System p,
>> and finally IBM Power Systems.
>
> What other OSs ran on the RS/6000 in 1990?

Well, technically ... IBM licensed the NeXTStep operating system from
NeXT to put on the RS/6000 line, though it never made it to market:

 https://www.techmonitor.ai/technology/ibms_rs6000_announceme nts

"There are no fewer than three interfaces offered on the operating
 system licensed separately. First is Steve Jobs' NeXTStep – dubbed
 the AIX Graphic User Environment/6000."

 https://simson.net/ref/NeXT/nextworld/NextWorld_Extra/92.08.
Aug.NWE/92.08.Aug.NWExtra05.html

"In 1988, NeXT licensed NeXTSTEP 1.0 to IBM for use on the
 RS/6000 workstation."

// Christian

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Sun, 29 Dec 2024 15:56:54 GMT
View Forum Message <> Reply to Message

Originally posted by: Grant Taylor

On 12/28/24 17:25, Grant Taylor wrote:
> What other OSs ran on the RS/6000 in 1990?

Thank you Lynn and Christian. Today I learned something new to me. :-)

--
Grant. . . .

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Sun, 29 Dec 2024 21:17:27 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 29 Dec 2024 10:28:03 -0000 (UTC), Christian Brunschen wrote:

> ... IBM licensed the NeXTStep operating system ...
>
> "There are no fewer than three interfaces offered on the operating
> system licensed separately. First is Steve Jobs' NeXTStep – dubbed the
> AIX Graphic User Environment/6000."

So it was just an alternative GUI on top of AIX, not an OS in itself.

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Sun, 29 Dec 2024 21:39:58 GMT
View Forum Message <> Reply to Message

Originally posted by: rbowman

On Sun, 29 Dec 2024 09:56:54 -0600, Grant Taylor wrote:

> On 12/28/24 17:25, Grant Taylor wrote:
>> What other OSs ran on the RS/6000 in 1990?
>
> Thank you Lynn and Christian. Today I learned something new to me. :-)

I was only familiar with AIX. Historically our clients used RS/6000
machines with AIX. I can't remember the version numbers but the Y2K
patched AIX wouldn't run on older RS/6000 systems. Our clients looked at
the costs of upgrading to new servers versus Intel/Windows servers and

switched. We used MKS NutCracker to go cross platform and a few years later shut down our RS/6000 machines since all clients had switched.

I don't know how common that was industry wide but PSAPs (public safety answering point) don't have deep pockets.

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Sun, 29 Dec 2024 23:50:43 GMT

View Forum Message <> Reply to Message

Originally posted by: Grant Taylor

On 12/29/24 15:39, rbowman wrote:
> I was only familiar with AIX. Historically our clients used RS/6000
> machines with AIX. I can't remember the version numbers but the
> Y2K patched AIX wouldn't run on older RS/6000 systems. Our clients
> looked at the costs of upgrading to new servers versus Intel/Windows
> servers and switched. We used MKS NutCracker to go cross platform and
> a few years later shut down our RS/6000 machines since all clients
> had switched.

I was one degree removed from the AIX / RS/6000 to Windows / Intel exploration a few times.

> I don't know how common that was industry wide but PSAPs (public
> safety answering point) don't have deep pockets.

The first such migration was for a PSAP that was my first introduction to AIX. The two people that had been running their setup were able to do things and achieve service uptime with a pair of systems that the Windows / Intel vendor said could never be done.

Unidata ran on top of AIX and was sending (redo) logs from the active system to the passive system. When it came time to switch, everybody logged out, the admins did their shtick, and people logged back in a few (read: < 5) minutes later with everything running on the other system.

The Windows vendor was saying that you would require hours of down time every month for patching (patch Tuesday) and had no way to script the process of switching.

In hindsight, it was a quick leader / follower swap and service IP move between the systems. As long as the two systems were close to in sync (< 10 minutes out) switch over (including catch up) was something like a 90 second ordeal. Sanity check, and then give users all clear to log back in.

It was apparently amazing how many times my friend caused the vendors
pitching their Windows based product's jaw hit the floor when my friend
would ask "how do you do ${THING}", they'd say "you don't", he'd say "we
do it whenever we ant" / "why are we even bothering to look at your
product again?". He was (is) a salty AIX admin.

--
Grant. . . .

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Mon, 30 Dec 2024 07:19:12 GMT
View Forum Message <> Reply to Message

Originally posted by: rbowman

On Sun, 29 Dec 2024 17:50:43 -0600, Grant Taylor wrote:

> to AIX. The two people that had been running their setup were able to
> do things and achieve service uptime with a pair of systems that the
> Windows / Intel vendor said could never be done.
>
> Unidata ran on top of AIX and was sending (redo) logs from the active
> system to the passive system. When it came time to switch, everybody
> logged out, the admins did their shtick, and people logged back in a few
> (read: < 5) minutes later with everything running on the other system.
>
> The Windows vendor was saying that you would require hours of down time
> every month for patching (patch Tuesday) and had no way to script the
> process of switching.

Saying five nines and Windows in the same sentence is a joke. According to
our support people the answer is to never patch the server. When it
finally has to be updated there is great weeping. The site is generally
configured with a backup server paralleling the live server, preferably in
a different physical location. Switchover for maintenance is easy and if
the dispatch center winds up under 2' of water they can move to high
ground.

My favorite was a bug report that something was happening to the CJIN
interface around 2 AM every day. It turned out the server was configured
to automatically update.

The transition between AIX and Linux was very easy. In fact I still use a
Linux box for some development before moving to Windows. There were two
sites that used Linux for the backend services because the administrator

liked Linux. Once they moved up or on, the new guy would move to Windows.
When most sites switched to Esri for GIS data that sealed the deal.

Subject: Re: CP/67 Multics vs Unix
Posted by Peter Flass on Mon, 30 Dec 2024 19:03:02 GMT
View Forum Message <> Reply to Message

Grant Taylor <gtaylor@tnetconsulting.net> wrote:
> On 12/27/24 13:35, Lynn Wheeler wrote:
>> In 1990, AIX Version 3 was released for the POWER-based RS/6000
>> platform.[16] It became the primary operating system for the RS/6000
>> series, which was later renamed IBM eServer pSeries, IBM System p,
>> and finally IBM Power Systems.
>
> What other OSs ran on the RS/6000 in 1990?
>
> I'm confident that NetBSD and know that Linux eventually made it to the
> RS/6000. But I have no idea what else would run on it in '90.
>

IBM had OS/2 running on it, but they never officially released it

--
Pete

Subject: Re: CP/67 Multics vs Unix
Posted by Peter Flass on Mon, 30 Dec 2024 19:03:05 GMT
View Forum Message <> Reply to Message

rbowman <bowman@montana.com> wrote:
> On Sun, 29 Dec 2024 09:56:54 -0600, Grant Taylor wrote:
>
>> On 12/28/24 17:25, Grant Taylor wrote:
>>> What other OSs ran on the RS/6000 in 1990?
>>
>> Thank you Lynn and Christian. Today I learned something new to me. :-)
>
> I was only familiar with AIX. Historically our clients used RS/6000
> machines with AIX. I can't remember the version numbers but the Y2K
> patched AIX wouldn't run on older RS/6000 systems. Our clients looked at
> the costs of upgrading to new servers versus Intel/Windows servers and
> switched. We used MKS NutCracker to go cross platform and a few years
> later shut down our RS/6000 machines since all clients had switched.
>
> I don't know how common that was industry wide but PSAPs (public safety

> answering point) don't have deep pockets.
>

Our /6000 had a Microchannel bus, and when we wanted to upgrade it we
found out we couldn’t.

--
Pete

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Tue, 31 Dec 2024 03:10:15 GMT
View Forum Message <> Reply to Message

Originally posted by: Grant Taylor

On 12/30/24 13:03, Peter Flass wrote:
> IBM had OS/2 running on it, but they never officially released it

I hadn't even considered OS/2 b/c it's POWER PC support was almost
non-existent.

Were those PREP POWER PC systems?

I believe copies of OS/2 for POWER PC / PREP systems has made it out
into the wild.

--
Grant. . . .

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Tue, 31 Dec 2024 03:51:39 GMT
View Forum Message <> Reply to Message

Originally posted by: Dave Yeo

Grant Taylor wrote:
> On 12/30/24 13:03, Peter Flass wrote:
>> IBM had OS/2 running on it, but they never officially released it
>
> I hadn't even considered OS/2 b/c it's POWER PC support was almost
> non-existent.
>
> Were those PREP POWER PC systems?

>
> I believe copies of OS/2 for POWER PC / PREP systems has made it out
> into the wild.

Michal at the os2museum has a good write up on it,
https://www.os2museum.com/wp/os2-history/os2-warp-powerpc-ed ition/

Interestingly, JFS2 was first written for OS/2 and then ported to AIX
and Linux
Dave

Subject: Re: CP/67 Multics vs Unix
Posted by Alexander Schreiber on Tue, 31 Dec 2024 12:51:35 GMT
View Forum Message <> Reply to Message

Dave Yeo <dave.r.yeo@gmail.com> wrote:
> Grant Taylor wrote:
>> On 12/30/24 13:03, Peter Flass wrote:
>>> IBM had OS/2 running on it, but they never officially released it
>>
>> I hadn't even considered OS/2 b/c it's POWER PC support was almost
>> non-existent.
>>
>> Were those PREP POWER PC systems?
>>
>> I believe copies of OS/2 for POWER PC / PREP systems has made it out
>> into the wild.
>
> Michal at the os2museum has a good write up on it,
> https://www.os2museum.com/wp/os2-history/os2-warp-powerpc-ed ition/
>
> Interestingly, JFS2 was first written for OS/2 and then ported to AIX
> and Linux

The Linux-Port of JFS2 was ... not great. I used JFS as the backing store
filesystem of an AFS server during the pre-production test phase in the
early 2000s. Quickly switched back to ext3 when it reliably lost files.
That setup ran reliably for years with ext3 as backing store once it
was in production.

Kind regards,
Alex.
--
"Opportunity is missed by most people because it is dressed in overalls and
looks like work." -- Thomas A. Edison

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Tue, 31 Dec 2024 18:06:42 GMT
View Forum Message <> Reply to Message

Originally posted by: drb

> The Linux-Port of JFS2 was ... not great. I used JFS as the backing store
> filesystem of an AFS server during the pre-production test phase in the
> early 2000s. Quickly switched back to ext3 when it reliably lost files.
> That setup ran reliably for years with ext3 as backing store once it
> was in production.

A friend had more than one experience with the Linux port of Irix XFS
along the line of "well, fsck, I guess an empty filesystem _is_
consistent, but that's not really a _useful_ version of consistent."

After hearing those, I stuck with ext3/4 too, until ZFS was an option.

De

Subject: Re: CP/67 Multics vs Unix
Posted by Peter Flass on Tue, 31 Dec 2024 19:08:35 GMT
View Forum Message <> Reply to Message

Grant Taylor <gtaylor@tnetconsulting.net> wrote:
> On 12/30/24 13:03, Peter Flass wrote:
>> IBM had OS/2 running on it, but they never officially released it
>
> I hadn't even considered OS/2 b/c it's POWER PC support was almost
> non-existent.
>
> Were those PREP POWER PC systems?
>
> I believe copies of OS/2 for POWER PC / PREP systems has made it out
> into the wild.
>

I don’t think it did. By the time it was ready IBM had decided to decommit
OS/2. I think it was running internally, though, I’ve seen screenshots.
I’ll have to ask.

--
Pete

Subject: Re: CP/67 Multics vs Unix

Posted by Anonymous on Tue, 31 Dec 2024 22:13:06 GMT
View Forum Message <> Reply to Message

Originally posted by: Grant Taylor

On 12/31/24 13:08, Peter Flass wrote:
> I don't think it did. By the time it was ready IBM had decided to decommit
> OS/2. I think it was running internally, though, I've seen screenshots.
> I'll have to ask.

Take a look at this:

Link - OS/2 Warp, PowerPC Edition | OS/2 Museum
 - https://www.os2museum.com/wp/os2-history/os2-warp-powerpc-ed ition/

Maybe it is an internal copy that eventually leaked out to the world
years -> decades later.

Or maybe I'm conflating two things.

--
Grant. . . .

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Wed, 01 Jan 2025 17:51:11 GMT
View Forum Message <> Reply to Message

Originally posted by: George Musk

On Tue, 31 Dec 2024 18:06:42 +0000, Dennis Boone wrote:

>> The Linux-Port of JFS2 was ... not great. I used JFS as the backing store
>> filesystem of an AFS server during the pre-production test phase in the
>> early 2000s. Quickly switched back to ext3 when it reliably lost files.
>> That setup ran reliably for years with ext3 as backing store once it
>> was in production.
>
> A friend had more than one experience with the Linux port of Irix XFS
> along the line of "well, fsck, I guess an empty filesystem _is_
> consistent, but that's not really a _useful_ version of consistent."
>

Was it long time ago? XFS seems to be the default filesystem for RHEL...

Subject: Re: CP/67 Multics vs Unix
Posted by Anonymous on Wed, 01 Jan 2025 18:46:32 GMT
View Forum Message <> Reply to Message

Originally posted by: drb

> Was it long time ago? XFS seems to be the default filesystem for RHEL...

Probably at least 15 years. Maybe more.

De

Subject: Re: CP/67 Multics vs Unix
Posted by Alexander Schreiber on Thu, 02 Jan 2025 14:50:00 GMT
View Forum Message <> Reply to Message

George Musk <grgmusk@skiff.com> wrote:
> On Tue, 31 Dec 2024 18:06:42 +0000, Dennis Boone wrote:
>
>>> The Linux-Port of JFS2 was ... not great. I used JFS as the backing store
>>> filesystem of an AFS server during the pre-production test phase in the
>>> early 2000s. Quickly switched back to ext3 when it reliably lost files.
>>> That setup ran reliably for years with ext3 as backing store once it
>>> was in production.
>>
>> A friend had more than one experience with the Linux port of Irix XFS
>> along the line of "well, fsck, I guess an empty filesystem _is_
>> consistent, but that's not really a _useful_ version of consistent."
>>
>
> Was it long time ago? XFS seems to be the default filesystem for RHEL...

And it is the recommendation as backing store filesystem for GlusterFS,
where it seems to Just Work(TM) - I've been running a GlusterFS cluster
for some time now without any FS related problems.

Kind regards,
          Alex.
--
"Opportunity is missed by most people because it is dressed in overalls and
 looks like work."   -- Thomas A. Edison

Subject: Re: Multics vs Unix
Posted by Anonymous on Wed, 22 Jan 2025 01:17:26 GMT
View Forum Message <> Reply to Message

Originally posted by: antispam

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Mon, 23 Dec 2024 18:20:46 -0000 (UTC), Sarr Blumson wrote:
>
>> An IBM Labs group built CP67/CMS->VM370 with good commercial
>> success.
>
> That was a pretty crummy way of building a timesharing system, though; CMS
> was "interactive" (insofar as IBM understood the term), but it was not
> multiuser. So CP (later VM) was tacked on as an extra layer underneath to
> allow each user to run their own instance of CMS.

AFAICS concerning multiuser aspect VM/CMS combination was almost
as good as Multix: it seems that in Multix user normally had
a single process and "absentee" processes were quite detached,
almost as a separate virtual machine.

> This is normally hailed as "IBM pioneered virtualization". But it was just
> a less flexible, higher-overhead way of supporting multiple users than
> other vendors, like DEC, were able to do far more efficiently.

Lynn gave figures that suggest otherwise: that on similar hardware
VM could handle more users than competition. Looking for reasons
I can see the following:
- VM quite early got good paging algorithm
- VM had relatively small amout of non-pagable memory
- IIUC VM supported memory mapping of disk files/areas. Using
  this in principle one could get similar amount of memory
  sharing as good non-VM system

In other words, in general virtial machines does not look like
good way to implement multitasking, but it seems that VM
implemented some parts better than competition, and was able
to largely mitigate potential inefficiency of virtualization.

--
                              Waldek Hebisch

Subject: Re: Multics vs Unix
Posted by Anonymous on Wed, 22 Jan 2025 01:37:28 GMT
View Forum Message <> Reply to Message

Originally posted by: antispam

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>

> Even today, Multics has some features which can be considered
> innovative and uncommon. It may be true that, for example, SELinux can
> match all of its security capabilities and more. But some aspects of
> its file-protection system seem, to me, to make sharing of data
> between users a bit easier than your typical Linux/POSIX-type system.
>
> For a start, there seems to be no concept of file “ownership” as such.

Hmm, Multix file has "creator" which seem to have similar
role as Unix owner.

> One radical idea introduced in Unix was its profligate use of multiple
> processes. Every new command you executed (except for the ones built
> into the shell) required the creation of a new process, often several
> processes. Other OSes tended to look askance at this; it seemed
> somehow wasteful, perhaps even sinful to spawn so many processes so
> readily and discard them so casually. The more conventional approach
> was to create a single process at user login, and execute nearly all
> commands within the context of that. There were special commands for
> explicitly creating additional processes (e.g. for background command
> execution), but such process creation did not simply happen as a
> matter of course.
>
> Gradually, over time, the limitations of the single-process approach
> became too much to ignore, and the versatility of the Unix approach
> won over (nearly) everybody.

Well, IIUC "copy on the write" first appeared in Berkely Unix.
That greatly reduced cost of process creation and allowed more
sharing. Cost of separate address spaces is significant also
in modern times, so Unix got threads.

Without improvement like those Unix approach would be unattractive.

> One common irritation I find on POSIX/Linux systems is the convention
> that every directory has to have an entry called “.”, pointing to
> itself, and one called “..”, pointing to its parent. This way these
> names can be used in relative pathnames to reach any point in the
> directory hierarchy. But surely it is unnecessary to have explicit
> entries for these names cluttering up every directory; why not just
> build their recognition as a special case into the pathname-parsing
> logic in the kernel, once and for all?

Consider pathname like 'a/../b'. If 'a' is a symlink, than this
is likely to give different result than just 'b'. This may look
like an unnatural case, but symlinks are widely used and programs
create pathnames like this. So changing that would require
adjustment to many programs.

> There are other features of Multics that others more familiar with it
> might want to see mentioned (the single-level store concept, where
> “everything is a memory segment”, versus Unix “everything is a file”?

Hmm, it does not look like realy single-level: IIUC name space
for segments is bigger than what machine supports and one needs
explicitely map/activate segments to use them with memory access
instructions. This does not look very far from Unix memory
mapped files: you map a file and can access content like memory.

--
Waldek Hebisch

Subject: Re: Multics vs Unix
Posted by Anonymous on Wed, 22 Jan 2025 02:05:14 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Wed, 22 Jan 2025 01:17:26 -0000 (UTC), Waldek Hebisch wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>
>> On Mon, 23 Dec 2024 18:20:46 -0000 (UTC), Sarr Blumson wrote:
>>
>>> An IBM Labs group built CP67/CMS->VM370 with good commercial success.
>>
>> That was a pretty crummy way of building a timesharing system, though;
>> CMS was “interactive” (insofar as IBM understood the term), but it was
>> not multiuser. So CP (later VM) was tacked on as an extra layer
>> underneath to allow each user to run their own instance of CMS.
>
> AFAICS concerning multiuser aspect VM/CMS combination was almost as good
> as [MULTICS]: it seems that in [MULTICS] user normally had a single
> process and "absentee" processes were quite detached, almost as a
> separate virtual machine.

MULTICS also considered that different users might want to cooperate,
share files etc, not be shut off in their own separate VMs. It had a
sophisticated access-control system to allow this.

>> This is normally hailed as “IBM pioneered virtualization”. But it was
>> just a less flexible, higher-overhead way of supporting multiple users
>> than other vendors, like DEC, were able to do far more efficiently.
>
> Lynn gave figures that suggest otherwise: that on similar hardware VM

> could handle more users than competition.

Not likely. Their minimum hardware requirements for running a
“timesharing” system were about twice what an equivalent DEC OS needed on
a PDP-11, for example.

Subject: Re: Multics vs Unix
Posted by Anonymous on Wed, 22 Jan 2025 02:09:06 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Wed, 22 Jan 2025 01:37:28 -0000 (UTC), Waldek Hebisch wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>
>> For a start, there seems to be no concept of file “ownership” as such.
>
> Hmm, [MULTICS] file has "creator" which seem to have similar role as
> Unix owner.

There was no concept of a single file owner, though; multiple entities
could have “creator” privileges over a file/directory. That was my point.

> Well, IIUC "copy on the write" first appeared in Berkely Unix.

All the virtual-memory systems had that. Unix was not the first.

> Consider pathname like 'a/../b'. If 'a' is a symlink, than this
> is likely to give different result than just 'b'. This may look
> like an unnatural case, but symlinks are widely used and programs
> create pathnames like this. So changing that would require
> adjustment to many programs.

No, it wouldn’t require any change at all.

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Wed, 22 Jan 2025 03:27:13 GMT
View Forum Message <> Reply to Message

antispam@fricas.org (Waldek Hebisch) writes:
> Lynn gave figures that suggest otherwise: that on similar hardware
> VM could handle more users than competition. Looking for reasons
> I can see the following:
> - VM quite early got good paging algorithm

> - VM had relatively small amout of non-pagable memory
> - IIUC VM supported memory mapping of disk files/areas. Using
>   this in principle one could get similar amount of memory
>   sharing as good non-VM system

Some of the MIT CTSS/7094 people go to the 5th flr for Multics. Others
go to the ibm cambridge science center on the 4th flr and do virtual
machines, internal network, lots of performance and interactive apps,
invent GML in 1969, etc. There was some amount of friendly rivalry
between 4th and 5th flrs.

Then some CSC come out to the univ to install CP67 (3rd after CSC itself
and MIT Lincoln Labs).

as undergraduate I did global LRU and some other paging related stuff
(thrashing control, etc), dynamic adaptive resource managerment and
scheduling, optimzing pathlengths, etc as undergraduate ... circa 1968
.... when there was academic literature about local LRU and other kinds
of implementations. CMS filesystem was 360 channel program I/O ... and
I modify CMS & CP67 to provide a virtual machine simplfied I/O interface
that significantly cuts the emulation pathlength.

when I graduate and join IBM CSC and get much of undergaduate CP67 stuff
up and running on CSC, 768kbyte 360/67 (104 pageable pages after fixed
storage). Then IBM Grenoble Science Center has a 1mbyte 360/67 (155
pageable pages after fixed storage) and modify CP67 according to the 60s
ACM academic literature. I have 75-80 users on the CSC system with
better response and throughput then Grenoble with 35 users (all users
running similar workloads).

One of my hobbies after joining IBM was enhanced production operating
systems for internal datacenters. Somewhat the 4th/5th flr rivalry I
then redo CMS filesystem to be page/memory mapped with lots of new
features (and further vastly simplified API and pathlength
implementation, I joke that I learned what not to do from observing
the IBM TSS/360 single-level-store implementation). Then early
70s, IBM had Future System project that was total different from
360/370 and was going to completely replaced ... and had adapted
a TSS/360-like single-level-store. When FS implodes, a side-effect
was it gave page-oriented filesystems a bad reputation. As a result
while I deployed my implementation internally on lots of systems,
never got approval to ship to customers.

Then at Dec81 ACM, Jim Gray asks me to help a Tandem co-worker of his
get his Stanford phd ... which heavily involved global LRU and the local
LRU forces from the 60s were lobbying Stanford hard to not award Phd
involving global LRU. Jim knew I had a large amount of performance data
from both the Cambridge and Grenoble systems giving close A:B comparison

on otherwise very similar hardware and software.

Back in the early 70s, after the IBM decision to add virtual memory to
all 370s, the initial morph of CP67->VM370 dropped/simplified
features. Starting w/VM370R2, I start moving lots of CP67 to VM370R2 for
my internal CSC/VM system.

Later in the 70s, decision was made to release some number of my CSC/VM
features to customers (not including CMS page-mapped filesystem)

In he mid-70s, Endicott (mid-range 370s) cons me helping with ECPS,
moving lots of VM370 into 138/148. Mid-range averaged 10 native
instruction for every 370 instructions, and most of the vm370 kernel 370
instructions move to native on 1-for-1 ... giving a 10:1 performance
improvement. This was carried over to the 4300. In Jan1979, branch
office cons me into doing a vm/4341 benchmark for national lab
that was looking at getting 70 for compute farm (sort of leading edge of
the coming cluster supercomputing tsunami). Then in the 80s, large
corporations were ordering hundreds of vm/4341s at a time for placing
out in departmental areas (sort of the leading edge of the coming
distributed computing tsunami).

This is archived (afc) post with a decade of VAX sales, sliced&diced by year,
model, us/non-us
https://www.garlic.com/~lynn/2002f.html#0
vax and 4300s sold in the same mid-range market and 4300s sold in
similar numbers as VAX in small unit orders ... big difference the large
orders of scores and hundreds of 4300s.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Anonymous on Wed, 22 Jan 2025 10:19:03 GMT
View Forum Message <> Reply to Message

Originally posted by: antispam

Lynn Wheeler <lynn@garlic.com> wrote:
> antispam@fricas.org (Waldek Hebisch) writes:
>> Lynn gave figures that suggest otherwise: that on similar hardware
>> VM could handle more users than competition. Looking for reasons
>> I can see the following:
>> - VM quite early got good paging algorithm
>> - VM had relatively small amout of non-pagable memory
>> - IIUC VM supported memory mapping of disk files/areas. Using
>> this in principle one could get similar amount of memory

>>   sharing as good non-VM system
>
> Some of the MIT CTSS/7094 people go to the 5th flr for Multics. Others
> go to the ibm cambridge science center on the 4th flr and do virtual
> machines, internal network, lots of performance and interactive apps,
> invent GML in 1969, etc. There was some amount of friendly rivalry
> between 4th and 5th flrs.
>
> Then some CSC come out to the univ to install CP67 (3rd after CSC itself
> and MIT Lincoln Labs).
>
> as undergraduate I did global LRU and some other paging related stuff
> (thrashing control, etc), dynamic adaptive resource managerment and
> scheduling, optimzing pathlengths, etc as undergraduate ... circa 1968
> ... when there was academic literature about local LRU and other kinds
> of implementations. CMS filesystem was 360 channel program I/O ... and
> I modify CMS & CP67 to provide a virtual machine simplfied I/O interface
> that significantly cuts the emulation pathlength.
>
> when I graduate and join IBM CSC and get much of undergaduate CP67 stuff
> up and running on CSC, 768kbyte 360/67 (104 pageable pages after fixed
> storage). Then IBM Grenoble Science Center has a 1mbyte 360/67 (155
> pageable pages after fixed storage) and modify CP67 according to the 60s
> ACM academic literature. I have 75-80 users on the CSC system with
> better response and throughput then Grenoble with 35 users (all users
> running similar workloads).

Do you remember what kind of backing store was in use (drum, disk ?)
on those machines?

--
                              Waldek Hebisch

Subject: Re: Multics vs Unix
Posted by Anonymous on Wed, 22 Jan 2025 10:45:40 GMT

View Forum Message <> Reply to Message

Originally posted by: antispam

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Wed, 22 Jan 2025 01:37:28 -0000 (UTC), Waldek Hebisch wrote:
>
>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>>
>>> For a start, there seems to be no concept of file “ownership” as such.
>>
>> Hmm, [MULTICS] file has "creator" which seem to have similar role as

>> Unix owner.
>
> There was no concept of a single file owner, though; multiple entities
> could have “creator” privileges over a file/directory. That was my point.

My reading of Multics documentaion is that there was single creator.
Other users may had priviledges due to ACL or due to directory
settings, but IIUC creator had right to change ACL even when
he/she was not mentioned in other mechanizms.

>> Well, IIUC "copy on the write" first appeared in Berkely Unix.
>
> All the virtual-memory systems had that. Unix was not the first.

AFAIK Unix before Berkeley added it had no "copy on the write".
IIUC early IBM mainframes also had no "copy on the write" and
it was awkward to implement (rather late, when they wanted Posix
compatibility they added appropriate hardware extention).

--
Waldek Hebisch

Subject: Re: Multics vs Unix
Posted by Anonymous on Wed, 22 Jan 2025 10:58:03 GMT
View Forum Message <> Reply to Message

Originally posted by: antispam

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Wed, 22 Jan 2025 01:17:26 -0000 (UTC), Waldek Hebisch wrote:
>
>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>
>>> On Mon, 23 Dec 2024 18:20:46 -0000 (UTC), Sarr Blumson wrote:
>>>
>>>> An IBM Labs group built CP67/CMS->VM370 with good commercial success.
>>>
>>> That was a pretty crummy way of building a timesharing system, though;
>>> CMS was “interactive” (insofar as IBM understood the term), but it was
>>> not multiuser. So CP (later VM) was tacked on as an extra layer
>>> underneath to allow each user to run their own instance of CMS.
>>
>> AFAICS concerning multiuser aspect VM/CMS combination was almost as good
>> as [MULTICS]: it seems that in [MULTICS] user normally had a single
>> process and "absentee" processes were quite detached, almost as a
>> separate virtual machine.
>

> MULTICS also considered that different users might want to cooperate,
> share files etc, not be shut off in their own separate VMs. It had a
> sophisticated access-control system to allow this.

Clearly Multics was more advanced. I was simply pointing out that
the same "single task for single user" mindset seem to be present
in Multics.

>>> This is normally hailed as "IBM pioneered virtualization". But it was
>>> just a less flexible, higher-overhead way of supporting multiple users
>>> than other vendors, like DEC, were able to do far more efficiently.
>>
>> Lynn gave figures that suggest otherwise: that on similar hardware VM
>> could handle more users than competition.
>
> Not likely. Their minimum hardware requirements for running a
> "timesharing" system were about twice what an equivalent DEC OS needed on
> a PDP-11, for example.

It seem that you had to get "right" version of VM (one enhanced by
Lynn). However, note that main overhead of virtualization is due
to nested page tables. Since CMS was unpaged there was no such
overhead for VM/CMS combination.

Different way of looking at this is that normal operating system
(say Linux) presents a virtual machine to user programs. This
virtual machine does not try to look like any real machine.
So, the question is if 360 machine interface is worse than
modern OS interfaces. Well, it is worse, but not that much.
And in case of VM/CMS worst parts were smoothed out by using
DIAGNOSE as a way to do syscalls and (later) because parts
of VM were moved to microcode (not applicable to modern
architecters, but useful in 370 era).

--
Waldek Hebisch

Subject: Re: Multics vs Unix
Posted by cross on Wed, 22 Jan 2025 13:59:34 GMT
View Forum Message <> Reply to Message

In article <vmqivp$1ldng$3@paganini.bofh.team>,
Waldek Hebisch <antispam@fricas.org> wrote:
> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:

Meta note: please don't feed the troll.

>> [snip]
>>
>> MULTICS also considered that different users might want to cooperate,
>> share files etc, not be shut off in their own separate VMs. It had a
>> sophisticated access-control system to allow this.
>
> Clearly Multics was more advanced. I was simply pointing out that
> the same "single task for single user" mindset seem to be present
> in Multics.

Not precisely, just that the notion of a "process" was rather
different than in e.g. Unix. In Multics, processes tend to be
long-lived things and one may run many programs in a the context
of a single process, with the text and data of those programs
coming and going over time: to run something, I link it into the
address space of the current process, it does whatever it does,
and when it's done it is unlinked.

Many of the DEC OSes do more or less the same thing. VMS calls
this process "running up" or "running down" an "image".

Of course, to work properly, one needs support from the hardware
so that an errant program does not clobber the process itself;
in Multics this involved the ring architecture and how it worked
with respedct to segmentsand call gates, which were provided by
the GE and Honeyll/Bull hardware in the 645-6180-DPS/8M series
of machines. A program error could be trapped by the system and
dealt with one way or another and in all cases the process
itself was protcted from these shenanigans since the process
spanned multiple protection rings on a per-segment basis and
access to higher rings was controlled via the call-gating
mechanism.

On VMS, page tables contained protection bits for all four CPU
protection levels, and were used to control access in a vaguely
similar way. It wasn't the single-level store of Multics by
any means, but the process was protected from errant programs
(say) trashing the command interpreter.

In all of these sytems, processes are rather heavy-weight
constructs, and creating and deleting them is expensive.

VM was different yet again: each user session gets its own VM
and then boots whatever OS it wants in that VM. Most
interactive users ran CMS, which was a single-user system, but
paravirtualized (that is, it "knew" it was running in a VM) and
let the user interact with other users, sending them messages
and so forth. Users were (usually?) allocated "minidisks" for

storage; this was a typically small virtual disk that they
initialized with a filesystem that CMS understood for their
own files. Such disks could be shared and linked between
different VMs, allowing easy file sharing.

My sense was that VM was highly interactive and collaborative.
Maybe it was just our local site, but users sent each other
messages _all the time_, in a way that they did not on Unix, VMS
or the PDP-10. We had locally written scripts that let multiple
users "chat"; as I mentioned, disks could be shared ("linked")
between users, and there was a mechanism to query which users
were linked to a disk. One then queried that list and send each
such linked user a short message.

The terminal interface was generally clever about how these were
displayed; if you were in (say) the editor, you didn't get
notifications until you took some action where the interface
would switch (perhaps you had to exit the editor; I no longer
recall).

Personally, I found Unix and VMS far more pleasant for program
development, but VM/CMS a lot more social, though that probably
had a lot more to do with the local community than anything
else.

>>> Lynn gave figures that suggest otherwise: that on similar hardware VM
>>> could handle more users than competition.
>>
>> Not likely. Their minimum hardware requirements for running a
>> “timesharing” system were about twice what an equivalent DEC OS needed on
>> a PDP-11, for example.
>
> It seem that you had to get "right" version of VM (one enhanced by
> Lynn). However, note that main overhead of virtualization is due
> to nested page tables. Since CMS was unpaged there was no such
> overhead for VM/CMS combination.

Comparisons between a PDP-11 and a mainframe are silly. The
PDP-11 is a small departmental computer, and hardly DEC's only
offering. Mainframes are much larger and more capable machines;
a better comparison to contemporary DEC hardware would be
the PDP-10 or (later) one of the larger VAX models. Even within
DEC's various lines, there are major differences between
systems: there's a big difference between RT-11 and TOPS-20, for
instance.

> Different way of looking at this is that normal operating system
> (say Linux) presents a virtual machine to user programs. This

> virtual machine does not try to look like any real machine.

Eh.... A couple of things.

First, when discussing things in an historical context, I don't
know that I would characterize "Linux" as "normal", in so far as
back in those days there _was_ no "normal". Indeedk Unix was
often looked at as the outlier in the way it did things. I
remember a VMS administrator bristling at someone saying that
Unix was the "textbook OS": "Yeah, because the textbook was
written about Unix!"

Second, the Unix process model is of an augmented virtual
machine. The idea is that each "process" sees a virtual
computer with some restrictions (one doesn't get unfettered
access to physical memory, but rather, the virtual address space
that has been constructored for that process) and some additions
(system calls may be thought of as virtual instructions) and of
course a bunch of useful resources (files and so forth). But of
course the vast bulk of what a program depends on, and
importantly the instructions what it executes, is in fact
defined by the underlying hardware, so processes very much _do_
try to look like the underlying machine, at least in so far as
the CPU, RAM, etc, is concerned. IO is obviously different.

> So, the question is if 360 machine interface is worse than
> modern OS interfaces. Well, it is worse, but not that much.

Well, the IO architecture was much maligned in its time.

> And in case of VM/CMS worst parts were smoothed out by using
> DIAGNOSE as a way to do syscalls and (later) because parts
> of VM were moved to microcode (not applicable to modern
> architecters, but useful in 370 era).

The VM architecture was actually quite clever. There are
important classes of problems not handled well by the Unix
one-size-fits-all process model where a set of global
resources are imperfectly multiplexed across all processes on
the machine that one could imagine would be better served by
custom systems running on dedicated resources allocated to
light-weight virtual machines. Modern Linux (and other)
systems give the program a bunch of knobs to twist to try and
approximate something optimal here, but they often don't
interact super well and the result is never perfect. A
different abstraction might lead to a system much simpler and
better overall performance and utilization.

- Dan C.

Subject: Re: Multics vs Unix
Posted by scott on Wed, 22 Jan 2025 15:06:33 GMT
View Forum Message <> Reply to Message

cross@spitfire.i.gajendra.net (Dan Cross) writes:
> In article <vmqivp$1ldng$3@paganini.bofh.team>,
> Waldek Hebisch <antispam@fricas.org> wrote:

>> So, the question is if 360 machine interface is worse than
>> modern OS interfaces. Well, it is worse, but not that much.
>
> Well, the IO architecture was much maligned in its time.

Still is, per Lynn.

Subject: Re: Multics vs Unix
Posted by scott on Wed, 22 Jan 2025 15:08:23 GMT
View Forum Message <> Reply to Message

antispam@fricas.org (Waldek Hebisch) writes:
> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>> On Wed, 22 Jan 2025 01:37:28 -0000 (UTC), Waldek Hebisch wrote:
>>
>>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>>>
>>>> For a start, there seems to be no concept of file “ownership” as such.
>>>
>>> Hmm, [MULTICS] file has "creator" which seem to have similar role as
>>> Unix owner.
>>
>> There was no concept of a single file owner, though; multiple entities
>> could have “creator” privileges over a file/directory. That was my point.
>
> My reading of Multics documentaion is that there was single creator.
> Other users may had priviledges due to ACL or due to directory
> settings, but IIUC creator had right to change ACL even when
> he/she was not mentioned in other mechanizms.
>
>>> Well, IIUC "copy on the write" first appeared in Berkely Unix.
>>
>> All the virtual-memory systems had that. Unix was not the first.
>
> AFAIK Unix before Berkeley added it had no "copy on the write".

> IIUC early IBM mainframes also had no "copy on the write" and
> it was awkward to implement (rather late, when they wanted Posix
> compatibility they added appropriate hardware extention).

IIRC, it was SunOS (Pre-solaris it was based on BSD) that did CoW first.

Subject: Re: Multics vs Unix
Posted by cross on Wed, 22 Jan 2025 16:10:00 GMT
View Forum Message <> Reply to Message

In article <vmpi4m$1jqc1$2@paganini.bofh.team>,
Waldek Hebisch <antispam@fricas.org> wrote:
>> [snip]
>
> Well, IIUC "copy on the write" first appeared in Berkely Unix.
> That greatly reduced cost of process creation and allowed more
> sharing. Cost of separate address spaces is significant also
> in modern times, so Unix got threads.

Perhaps you are thinking about `vfork`, a fork variant with, er,
"odd" semantics.

Early virtual memory BSD on the VAX, like the PDP-11 before it,
would copy the entire parent process to the child on `fork`.
The observation was that this was unnecessarily wasteful, as
(probably) the most common pattern for `fork` was that it was
immediately followed by either `exec` or `exit`.

This meant that the kernel would have to allocate space to hold
a copy of the parent, as well as set up its address space
(requiring the allocation of more memory for page tables and so
on), simply for it all to be torn down and discarded nearly
immediately afterwards.

`vfork` was introduced as an optimization: like fork, it creates
a new process, but unlike `fork`, on success, the system
guarantees that a) the child has its "user" area in the kernel,
b) the parent will be suspended until the child either does an
`exec` or exits, and c) the child will otherwise run _in the
address space of the parent_. If the sequence is something
like:

```
    pid_t pid = vfork();
    if (pid < 0) {
        perror("vfork failed");
        exit(1);
    }
    if (pid == 0) {
        execl("/what/ever", "what", NULL);
        perror("execl of /what/ever failed");
        exit(1);
    } else {
        wait(NULL);
    }
```

Then, assuming `execl` is successful, the only things that
happen in the child are the conditional and the exec call
itself, all of which can run in the address space of the parent
plus a private kstack just fine. The kernel will only allocate
a new address space--beyond the minimum required for the kernel
to have its own kstack (in the user area)--for the `exec`.

Of course, if the child does a lot of work before exec (or
exit), then it's potentially modifying a lot of the state of
the parent, which is ill-advised.

CoW is a much cleaner way to eliminate a lot of the cost of fork
though it still requires allocating an address space for the
child: for a very large process that can be expensive, so modern
Unixes still sometimes provide (and use) `vfork`; e.g., NetBSD.

> Without improvement like those Unix approach would be unattractive.

A ton of work had to be put into Unix to make it attractive for
most of the uses it's put to today.

> [snip]
>> There are other features of Multics that others more familiar with it
>> might want to see mentioned (the single-level store concept, where
>> “everything is a memory segment”, versus Unix “everything is a file”?
>
> Hmm, it does not look like realy single-level: IIUC name space
> for segments is bigger than what machine supports and one needs
> explicitely map/activate segments to use them with memory access
> instructions. This does not look very far from Unix memory
> mapped files: you map a file and can access content like memory.

The point in Multics was that de-facto all IO for e.g. disks is
memory mapped; Unix got `mmap` (modeled after TENEX/TOPS-20
`PMAP`) much, much later. Of course, Multics got something
analogous to `read`/`write` much later, as well.

Multics is "single-level" in the sense that the concept of a
file is unified with the notion of a segment as described by the
hardware itself. That segments could be named and exist in a

hierarchical namespace, but still be linked into a program in a
natural way was very attractive. That those segments were
paged and the system exerted considerable effort to make that
transparent to the user was a detail.

 - Dan C.

Subject: Re: Multics vs Unix
Posted by Peter Flass on Wed, 22 Jan 2025 18:03:45 GMT
View Forum Message <> Reply to Message

Waldek Hebisch <antispam@fricas.org> wrote:
> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>> On Mon, 23 Dec 2024 18:20:46 -0000 (UTC), Sarr Blumson wrote:
>>
>>> An IBM Labs group built CP67/CMS->VM370 with good commercial
>>> success.
>>
>> That was a pretty crummy way of building a timesharing system, though; CMS
>> was “interactive” (insofar as IBM understood the term), but it was not
>> multiuser. So CP (later VM) was tacked on as an extra layer underneath to
>> allow each user to run their own instance of CMS.
>
> AFAICS concerning multiuser aspect VM/CMS combination was almost
> as good as Multix: it seems that in Multix user normally had
> a single process and "absentee" processes were quite detached,
> almost as a separate virtual machine.
>
>> This is normally hailed as “IBM pioneered virtualization”. But it was just
>> a less flexible, higher-overhead way of supporting multiple users than
>> other vendors, like DEC, were able to do far more efficiently.
>
> Lynn gave figures that suggest otherwise: that on similar hardware
> VM could handle more users than competition. Looking for reasons
> I can see the following:
> - VM quite early got good paging algorithm
> - VM had relatively small amout of non-pagable memory
> - IIUC VM supported memory mapping of disk files/areas. Using
>   this in principle one could get similar amount of memory
>   sharing as good non-VM system
>
> In other words, in general virtial machines does not look like
> good way to implement multitasking, but it seems that VM
> implemented some parts better than competition, and was able
> to largely mitigate potential inefficiency of virtualization.
>

VM/CMS was fast because it was so simple. VM did the minimum necessary to ensure separation of users, allowing almost no sharing. CMS was about as sophisticated as PCDOS. Basically a loader for a single program.

--
Pete

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Wed, 22 Jan 2025 18:07:39 GMT
View Forum Message <> Reply to Message

antispam@fricas.org (Waldek Hebisch) writes:
> Do you remember what kind of backing store was in use (drum, disk ?)
> on those machines?

360/67 had 2301 fixed head drums for primary paging (similar to 2303 drum but transferred four heads in parallel, four times the data rate, 1/4th the number of "track", each "track" four times larger"). Overflow paging, spool file system, and CMS filesystem (as well as os/360 dasd) were all 2314.

when CP67 was initially installed at univ ... all queued I/O was FIFO and paging were single transfer channel programs. I redid 2314 to be ordered seek and page I/Os were single channel program for all queued requests for 2301 and for same 2314 arm position/cylinder (optimized for max transfers/revolution). 2301 originally had throughput of around 70 page requests/sec ... my rewrite improved it to 270/sec peak.

after joining CSC, CSC was in processing of adding 2314s strings, eventually with five 8+1 drive strings and one 5 drive string (45 2314 total).

after decision to add virtual memory to all 370s, did a enhancement with CP67 option to emulate 370 (virtual) machines. had to demonstrate extremely strong security ... since 370 virtual memory hadn't been announced and was still classified ... and CSC also had profs, staff, and students from Boston/Cambridge institutions using the system.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Wed, 22 Jan 2025 18:11:42 GMT
View Forum Message <> Reply to Message

Peter Flass <peter_flass@yahoo.com> writes:
> VM/CMS was fast because it was so simple. VM did the minimum necessary to
> ensure separation of users, allowing almost no sharing. CMS was about as
> sophisticated as PCDOS. Basically a loader for a single program.

got more sophisticated over time ... sort of started off as upgrade of
CTSS.

trivia ... before ms/dos
https://en.wikipedia.org/wiki/MS-DOS
there was Seattle computer
https://en.wikipedia.org/wiki/Seattle_Computer_Products
before Seattle computer, there was CP/M
https://en.wikipedia.org/wiki/CP/M
before developing CP/M, kildall worked on IBM CP/67 at npg
https://en.wikipedia.org/wiki/Naval_Postgraduate_School

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Wed, 22 Jan 2025 18:16:19 GMT
View Forum Message <> Reply to Message

other trivia: tss/360 was suppose to be the official system for 360/67,
at the time tss/360 was "decomitted", there were 1200-some people
involved with tss/360 and 12 people at CSC in the combined CP67/CMS
group.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Wed, 22 Jan 2025 22:52:16 GMT
View Forum Message <> Reply to Message

antispam@fricas.org (Waldek Hebisch) writes:
> AFAIK Unix before Berkeley added it had no "copy on the write".
> IIUC early IBM mainframes also had no "copy on the write" and
> it was awkward to implement (rather late, when they wanted Posix
> compatibility they added appropriate hardware extention).

decade+ ago I was asked to track down executive decision to add virtual
memory to all 370s ... found staff to executive making the decision,
basically (OS/360) MVT storage management was so bad that region sizes

had to be specified four times larger than used used ... as result
common 1mbyte 370/165 only ran four regions concurrently, insufficient
to keep 165 busy and justified. Mapping MVT to a 16mbyte virtual memory
allowed number of concurrent executing regions to be increased by a
factor of four (modulo capped at 15 for 4bit storage protect keys) ...
similar to running MVT in a CP67 16mbyte virtual machine. Archive of
a.f.c. 2011 post with pieces of email exchange
https://www.garlic.com/~lynn/2011d.html#73

VM370 was planning on taking advantage of 370 virtual memory R/O segment
protect for sharing common CMS code. Then the 165 engineers were
complaining that if they had to retrofit full 370 virtual memory
architecture to 165, it would slip 370 virtual memory by six months. It
was then decided to regress 370 virtual memory to 165 subset (and r/o
segment protect was one of the casualties).

All other 370s that had already retrofitted full 370 virtual memory
architecture to the 165 subset and any software written to use full
architecture had to be redone for the 165 subset. VM370 to simulate R/O
segment protect for CMS ... then had to deploy a hack using (360)
storage protect keys ... CMS with shared segments ran with a PSW that
never had key=0 (can store anywhere) and all CMS non-shared pages had
non-zero protect key and CMS shared pages had zero protect key (i.e.
CMS PSW key and non-shared page protect keys matched for stores, but
shared pages keys never matched.

Then comes VM370R2 where 158 & 168 got VMASSIST ... load VMBLOK pointer
into CR6 and if virtual PSW was in virtual supervisor mode, the machine
would directly emulate priviliged mode for some number of privileged
mode (w/o requiring simulation by vm370). The problem was VMASSIST
didn't support the LPSW, ISK, & SSK hack for CMS shared R/O protect. For
VM370R3, they come up with a hack that allowed CMS w/share R/O protect
to be run in VMASSIST mode (by eliminating the protect key hack). When
ever there was a task switch (from CMS with R/O protect) all the
associated shared pages in memory, would be scanned for changed (aka
some store) pages. Any changed (shared pages) would be marked as no
longer shared ... and associated shared pages would be flagged as not in
memory ... the next reference results in page fault and a (non-changed)
copy be refreshed from backing store (selective copy on write).

Then comes VM370R4 with release of multiprocessor support (in the
original morph of CP67->VM370 lots of stuff was simplified and/or
dropped ... including multiprocessor support). For my VM370R2-based
internal release had done the kernel reorg needed by multiprocessor
support (but not actual SMP support) ... and then for VM370R3-based
internal release included multiprocessor support ... initially for the
internal branch office sales&marketing online HONE system (one my
original internal customers, enhanced operating systems) ... and the US

HONE datacenters had been recently consolidated in Palo Alto (trivia
when FACEBOOK 1st moved into silicon valley, it was into new bldg built
next door to the former US consolidated HONE datacenter). Initial had
eight single processor systems configured in one of the largest
single-system image, shared DASD operation with load-balancing and
fall-over across the complex. Then I add multiprocessor back in so they
can add 2nd processor to each system.

At least for my internal VM370R3-based release, I revert to the storage
protect key hack, because scanning for changed, shared pages only showed
VMASSIST benefit justified for original base VM370 with 16 shared pages
.... and I had increased the number of shared pages by factor of at least
2-3 times ... and depending on what CMS was doing could be a great deal
move (greatly increasing shared page scan overhead swamping the VMASSIST
benefit).

The VMASSIST hack also got worse w/multiprocessor, which required unique
set of share pages for each processor and each time a CMS user with
shared pages, was dispatched, had to make sure their page table pointers
to shared pages had to match the processor being dispatched on.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Peter Flass on Wed, 22 Jan 2025 23:12:19 GMT
View Forum Message <> Reply to Message

Lynn Wheeler <lynn@garlic.com> wrote:
> antispam@fricas.org (Waldek Hebisch) writes:
>> AFAIK Unix before Berkeley added it had no "copy on the write".
>> IIUC early IBM mainframes also had no "copy on the write" and
>> it was awkward to implement (rather late, when they wanted Posix
>> compatibility they added appropriate hardware extention).
>
> decade+ ago I was asked to track down executive decision to add virtual
> memory to all 370s ... found staff to executive making the decision,
> basically (OS/360) MVT storage management was so bad that region sizes
> had to be specified four times larger than used used ... as result
> common 1mbyte 370/165 only ran four regions concurrently, insufficient
> to keep 165 busy and justified. Mapping MVT to a 16mbyte virtual memory
> allowed number of concurrent executing regions to be increased by a
> factor of four (modulo capped at 15 for 4bit storage protect keys) ...
> similar to running MVT in a CP67 16mbyte virtual machine. Archive of
> a.f.c. 2011 post with pieces of email exchange
> https://www.garlic.com/~lynn/2011d.html#73
>

> VM370 was planning on taking advantage of 370 virtual memory R/O segment
> protect for sharing common CMS code. Then the 165 engineers were
> complaining that if they had to retrofit full 370 virtual memory
> architecture to 165, it would slip 370 virtual memory by six months. It
> was then decided to regress 370 virtual memory to 165 subset (and r/o
> segment protect was one of the casualties).
>
> All other 370s that had already retrofitted full 370 virtual memory
> architecture to the 165 subset and any software written to use full
> architecture had to be redone for the 165 subset. VM370 to simulate R/O
> segment protect for CMS ... then had to deploy a hack using (360)
> storage protect keys ... CMS with shared segments ran with a PSW that
> never had key=0 (can store anywhere) and all CMS non-shared pages had
> non-zero protect key and CMS shared pages had zero protect key (i.e.
> CMS PSW key and non-shared page protect keys matched for stores, but
> shared pages keys never matched.
>
> Then comes VM370R2 where 158 & 168 got VMASSIST ... load VMBLOK pointer
> into CR6 and if virtual PSW was in virtual supervisor mode, the machine
> would directly emulate priviliged mode for some number of privileged
> mode (w/o requiring simulation by vm370). The problem was VMASSIST
> didn't support the LPSW, ISK, & SSK hack for CMS shared R/O protect. For
> VM370R3, they come up with a hack that allowed CMS w/share R/O protect
> to be run in VMASSIST mode (by eliminating the protect key hack). When
> ever there was a task switch (from CMS with R/O protect) all the
> associated shared pages in memory, would be scanned for changed (aka
> some store) pages. Any changed (shared pages) would be marked as no
> longer shared ... and associated shared pages would be flagged as not in
> memory ... the next reference results in page fault and a (non-changed)
> copy be refreshed from backing store (selective copy on write).
>
> Then comes VM370R4 with release of multiprocessor support (in the
> original morph of CP67->VM370 lots of stuff was simplified and/or
> dropped ... including multiprocessor support). For my VM370R2-based
> internal release had done the kernel reorg needed by multiprocessor
> support (but not actual SMP support) ... and then for VM370R3-based
> internal release included multiprocessor support ... initially for the
> internal branch office sales&marketing online HONE system (one my
> original internal customers, enhanced operating systems) ... and the US
> HONE datacenters had been recently consolidated in Palo Alto (trivia
> when FACEBOOK 1st moved into silicon valley, it was into new bldg built
> next door to the former US consolidated HONE datacenter). Initial had
> eight single processor systems configured in one of the largest
> single-system image, shared DASD operation with load-balancing and
> fall-over across the complex. Then I add multiprocessor back in so they
> can add 2nd processor to each system.
>
> At least for my internal VM370R3-based release, I revert to the storage

> protect key hack, because scanning for changed, shared pages only showed
> VMASSIST benefit justified for original base VM370 with 16 shared pages
> ... and I had increased the number of shared pages by factor of at least
> 2-3 times ... and depending on what CMS was doing could be a great deal
> move (greatly increasing shared page scan overhead swamping the VMASSIST
> benefit).
>
> The VMASSIST hack also got worse w/multiprocessor, which required unique
> set of share pages for each processor and each time a CMS user with
> shared pages, was dispatched, had to make sure their page table pointers
> to shared pages had to match the processor being dispatched on.
>

That’s what happens with VM the tail to the MVS dog.

--
Pete

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Thu, 23 Jan 2025 00:06:27 GMT
View Forum Message <> Reply to Message

Peter Flass <peter_flass@yahoo.com> writes:
> That’s what happens with VM the tail to the MVS dog.

after decision add virtual memory to all 370s ... came the future system
effort, completely different than 370 and was going to replace 370
(during FS period, internal politics was shutting down 370 projects,
claim was that lack of new 370 during FS period gave the clone 370
makers their market foothold). then when FS implodes there is mad rush
to get stuff back into the product pipelines, including kicking off
quick&dirty 3033&3081 in parallel.
http://www.jfsowa.com/computer/memo125.htm
https://people.computing.clemson.edu/~mark/fs.html
https://en.wikipedia.org/wiki/IBM_Future_Systems_project

also the head of pok (high-end 370) manages to convince corporate to
kill the vm370 product, shutdown the development group and transfer all
the people to pok to work on MVS/XA (370xa, 31bit and various MVS
specific features). Endicott (midrange 370) manages to save the vm370
product mission but has to recreate a development group from scratch.

Besides endicott con'ing me into (VM370) ECPS (reference upthread,
initially 138&148 and then 4331&/4341), I also get con'ed into working
on a 16-cpu 370 multiprocessor (having done multiprocessor for HONE on
VM370R3 base, reference upthread, ... HONE 2-cpu 370 systems getting
twice the throughput of single CPU system) and we con the 3033 processor

engineers into working on it in their spare time (lot more interesting
than remapping 168 logic to 20% faster chips).

Everybody thought it was great until somebody tells the head of POK that
it could be decades before the POK favorite son operating system ("MVS")
had (effective) 16-cpu support (IBM documents at the time said MVS 2-cpu
support had 1.2-1.5 times the throughput of single CPU system) and some
of us were invited to never visit POK again and 3033 processor engineers
directed to heads down on 3033 and no distractions (note POK doesn't
ship a 16-cpu system until after the turn of the century).

trivia: along with ECPS, Endicott tried to get corporate to approve
preinstalling VM370 on every machine shipped (but POK influence trying
to totally kill VM370, wouldn't allow it) ... somewhat analogous to
Amdahl;s microcode hypervisor (multiple domain, no vm370 software)
https://ieeexplore.ieee.org/document/7054
https://www.computinghistory.org.uk/det/15016/amdahl-580-Mul tiple-Domain-Feature-Overview/
https://archive.org/details/bitsavers_amdahl580AainFeatureOv erviewJul86_3237564
and nearly decade later IBM 3090 microcode LPAR&PR/SM
https://en.wikipedia.org/wiki/Logical_partition

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Rich Alderson on Thu, 23 Jan 2025 00:37:30 GMT
View Forum Message <> Reply to Message

scott@slp53.sl.home (Scott Lurndal) writes:

> IIRC, it was SunOS (Pre-solaris it was based on BSD) that did CoW first.

That would not surprise me, given the number of Stanford people at Sun. CoW
was a long time feature of TENEX/TOPS-20 from the early 1970s, more than a
decade before Sun was founded.

--
Rich Alderson news@alderson.users.panix.com
Audendum est, et veritas investiganda; quam etiamsi non assequamur,
omnino tamen proprius, quam nunc sumus, ad eam perveniemus.
--Galen

Subject: Re: vfork history, Multics vs Unix
Posted by John Levine on Thu, 23 Jan 2025 03:35:20 GMT
View Forum Message <> Reply to Message

According to Dan Cross <cross@spitfire.i.gajendra.net>:
> `vfork` was introduced as an optimization: like fork, it creates
> a new process, but unlike `fork`, on success, the system
> [lends the parent address space to the child until it exec's or exits]

> CoW is a much cleaner way to eliminate a lot of the cost of fork ...

The VAX-11/750 had microcode bugs, one of which made read-only pages in the stack segment not work reliably. VMS didn't use them so DEC wasn't interested in fixing the bug, but without read-only pages that give a page fault if you try to write to them, you can't do CoW. I believe vfork() was as much a workaround for that bug as anything else.

In practice, most of the time if a program touches a stack page at all it's going to write to it, so copy-on-touch would have worked about as well without needing read-only stack pages. IBM mainframes didn't do page fault on write until quite late (S/390?) so some of their systems did CoT.

--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: Multics vs Unix
Posted by Anonymous on Thu, 23 Jan 2025 03:54:28 GMT
View Forum Message <> Reply to Message

Originally posted by: antispam

Dan Cross <cross@spitfire.i.gajendra.net> wrote:
> In article <vmqivp$1ldng$3@paganini.bofh.team>,
> Waldek Hebisch <antispam@fricas.org> wrote:
>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>
> Meta note: please don't feed the troll.
>
>>> [snip]
>>>
>>> MULTICS also considered that different users might want to cooperate,
>>> share files etc, not be shut off in their own separate VMs. It had a
>>> sophisticated access-control system to allow this.
>>
>> Clearly Multics was more advanced. I was simply pointing out that
>> the same "single task for single user" mindset seem to be present
>> in Multics.
>
> Not precisely, just that the notion of a "process" was rather

> different than in e.g. Unix. In Multics, processes tend to be
> long-lived things and one may run many programs in a the context
> of a single process, with the text and data of those programs
> coming and going over time: to run something, I link it into the
> address space of the current process, it does whatever it does,
> and when it's done it is unlinked.

Well, but is it essentially different than CMS, DOS or loading
shared libraries on Unix? In each case code and data is loaded,
run und then unloaded. Loading of new thing means that old
thing does not run. Multics documentation is not entirely
clear on this, but it seems that at given protection level
there is single stack, which would make things like coroutines
hairy.

Multics hardware have some advantage here, because segments
simplify relocation.

IIUC only tradition and need for multtasking prevents Unix shell
from taking commands from shared library and executing them in
a single process.

> Many of the DEC OSes do more or less the same thing. VMS calls
> this process "running up" or "running down" an "image".
>
> Of course, to work properly, one needs support from the hardware
> so that an errant program does not clobber the process itself;
> in Multics this involved the ring architecture and how it worked
> with respedct to segmentsand call gates, which were provided by
> the GE and Honeyll/Bull hardware in the 645-6180-DPS/8M series
> of machines. A program error could be trapped by the system and
> dealt with one way or another and in all cases the process
> itself was protcted from these shenanigans since the process
> spanned multiple protection rings on a per-segment basis and
> access to higher rings was controlled via the call-gating
> mechanism.

To say the truth some details are not clear. More precisely,
parts running in a single security context (the same ring, user
and whatever Multics considers relevant) seem to have equal rights,
so writable memory of one part seem to be unprotected from changes
by other parts. I am not saying that this is bad, simply given fact
that a lot of things may run together there seem to be
less protection than suggested by Multics propaganda.

Another thing is what changes security context? There are
gates, that at first looks clear. However, there seem to be
segment table and ATM it is not clear to me which part of

OS is responsible for setting up segment table.

> On VMS, page tables contained protection bits for all four CPU
> protection levels, and were used to control access in a vaguely
> similar way. It wasn't the single-level store of Multics by
> any means, but the process was protected from errant programs
> (say) trashing the command interpreter.
>
> In all of these sytems, processes are rather heavy-weight
> constructs, and creating and deleting them is expensive.
>
> VM was different yet again: each user session gets its own VM
> and then boots whatever OS it wants in that VM. Most
> interactive users ran CMS, which was a single-user system, but
> paravirtualized (that is, it "knew" it was running in a VM) and
> let the user interact with other users, sending them messages
> and so forth. Users were (usually?) allocated "minidisks" for
> storage; this was a typically small virtual disk that they
> initialized with a filesystem that CMS understood for their
> own files. Such disks could be shared and linked between
> different VMs, allowing easy file sharing.
>
> My sense was that VM was highly interactive and collaborative.
> Maybe it was just our local site, but users sent each other
> messages _all the time_, in a way that they did not on Unix, VMS
> or the PDP-10. We had locally written scripts that let multiple
> users "chat"; as I mentioned, disks could be shared ("linked")
> between users, and there was a mechanism to query which users
> were linked to a disk. One then queried that list and send each
> such linked user a short message.
>
> The terminal interface was generally clever about how these were
> displayed; if you were in (say) the editor, you didn't get
> notifications until you took some action where the interface
> would switch (perhaps you had to exit the editor; I no longer
> recall).
>
> Personally, I found Unix and VMS far more pleasant for program
> development, but VM/CMS a lot more social, though that probably
> had a lot more to do with the local community than anything
> else.

I think so. I used VM/CMS for accessing e-mail, but all other
things were done on other machines (some people tried to make
more use of it but that was rare). This was for communication
with remote sites, locally there was non-computer communication.

>>>> Lynn gave figures that suggest otherwise: that on similar hardware VM

>>>> could handle more users than competition.
>>>
>>> Not likely. Their minimum hardware requirements for running a
>>> â??timesharingâ?? system were about twice what an equivalent DEC OS needed on
>>> a PDP-11, for example.
>>
>> It seem that you had to get "right" version of VM (one enhanced by
>> Lynn). However, note that main overhead of virtualization is due
>> to nested page tables. Since CMS was unpaged there was no such
>> overhead for VM/CMS combination.
>
> Comparisons between a PDP-11 and a mainframe are silly. The
> PDP-11 is a small departmental computer, and hardly DEC's only
> offering. Mainframes are much larger and more capable machines;
> a better comparison to contemporary DEC hardware would be
> the PDP-10 or (later) one of the larger VAX models. Even within
> DEC's various lines, there are major differences between
> systems: there's a big difference between RT-11 and TOPS-20, for
> instance.

IIRC Lynn in the past gave comparison with big machines.

>> Different way of looking at this is that normal operating system
>> (say Linux) presents a virtual machine to user programs. This
>> virtual machine does not try to look like any real machine.
>
> Eh.... A couple of things.
>
> First, when discussing things in an historical context, I don't
> know that I would characterize "Linux" as "normal", in so far as
> back in those days there _was_ no "normal". Indeedk Unix was
> often looked at as the outlier in the way it did things. I
> remember a VMS administrator bristling at someone saying that
> Unix was the "textbook OS": "Yeah, because the textbook was
> written about Unix!"

OK. I mentione Linux because in description of Unix systems
"division of labor" seem to be more explicit than in other
systems: there is kernel vesrus userland. Multics with its
rings splits this into layers and at least overview/introductory
documentation do not make clear what are responsibilites of
various layers.

> Second, the Unix process model is of an augmented virtual
> machine. The idea is that each "process" sees a virtual
> computer with some restrictions (one doesn't get unfettered
> access to physical memory, but rather, the virtual address space
> that has been constructored for that process) and some additions

> (system calls may be thought of as virtual instructions) and of
> course a bunch of useful resources (files and so forth). But of
> course the vast bulk of what a program depends on, and
> importantly the instructions what it executes, is in fact
> defined by the underlying hardware, so processes very much _do_
> try to look like the underlying machine, at least in so far as
> the CPU, RAM, etc, is concerned. IO is obviously different.
>
>> So, the question is if 360 machine interface is worse than
>> modern OS interfaces. Well, it is worse, but not that much.
>
> Well, the IO architecture was much maligned in its time.
>
>> And in case of VM/CMS worst parts were smoothed out by using
>> DIAGNOSE as a way to do syscalls and (later) because parts
>> of VM were moved to microcode (not applicable to modern
>> architecters, but useful in 370 era).
>
> The VM architecture was actually quite clever. There are
> important classes of problems not handled well by the Unix
> one-size-fits-all process model where a set of global
> resources are imperfectly multiplexed across all processes on
> the machine that one could imagine would be better served by
> custom systems running on dedicated resources allocated to
> light-weight virtual machines. Modern Linux (and other)
> systems give the program a bunch of knobs to twist to try and
> approximate something optimal here, but they often don't
> interact super well and the result is never perfect. A
> different abstraction might lead to a system much simpler and
> better overall performance and utilization.

--
Waldek Hebisch

Subject: Re: Multics vs Unix
Posted by Anonymous on Thu, 23 Jan 2025 04:10:19 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Wed, 22 Jan 2025 10:58:03 -0000 (UTC), Waldek Hebisch wrote:

> Clearly Multics was more advanced. I was simply pointing out that the
> same "single task for single user" mindset seem to be present in
> Multics.

All the OSes of the time tried to minimize process creation. Unix was the

one going against the trend, by its seemingly profligate creation of new
processes for doing every single thing.

One big change was that the CLI was not some part of the resident kernel,
but just another user process. Hard to realize this was seen as quite
radical at the time.

> Different way of looking at this is that normal operating system (say
> Linux) presents a virtual machine to user programs.

The term is “abstract machine”, not “virtual machine”. It’s a “machine”
which is better than the underlying hardware because it has more
convenient capabilities for doing common things (e.g. storage management,
networking), at the cost of additional software overhead.

We can have multiple such “abstract machine” layers. For example, one
program we write to run on top of the Linux kernel might be a “shell”,
which itself is not particularly specialized for doing some specific task,
but more a range of tasks; and on top of that we build another abstract
machine, in the form of a “shell script”, that the shell can interpret to
perform a more specific task.

Subject: Re: Multics vs Unix
Posted by Anonymous on Thu, 23 Jan 2025 04:13:03 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Wed, 22 Jan 2025 11:03:45 -0700, Peter Flass wrote:

> VM/CMS was fast because it was so simple.

I’m not sure you could say it was “fast” unless you were comparing it to
alternatives.

For example, DEC’s various OSes for its PDP-11 range could do interactive
timesharing for less overhead than IBM’s offerings, because they
integrated multiuser support into the same OS kernel, instead of faking it
via a separate hypervisor layer.

Subject: Re: vfork history, Multics vs Unix
Posted by Anonymous on Thu, 23 Jan 2025 04:14:25 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 23 Jan 2025 03:35:20 -0000 (UTC), John Levine wrote:

> The VAX-11/750 had microcode bugs, one of which made read-only pages in
> the stack segment not work reliably. VMS didn't use them so DEC wasn't
> interested in fixing the bug, but without read-only pages that give a
> page fault if you try to write to them, you can't do CoW. I believe
> vfork() was as much a workaround for that bug as anything else.

vfork(2) was still needed anyway, bug or no bug. That’s why Linux still
has it, for example.

Subject: Re: vfork history, Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Thu, 23 Jan 2025 04:15:23 GMT
View Forum Message <> Reply to Message

John Levine <johnl@taugh.com> writes:
> In practice, most of the time if a program touches a stack page at all it's
> going to write to it, so copy-on-touch would have worked about as well without
> needing read-only stack pages. IBM mainframes didn't do page fault on write
> until quite late (S/390?) so some of their systems did CoT.

.... but 360 key protect had both (real memory) store (& 360 optional
fetch) protect ... which would be applied to all processes (the original
370 virtual memory architecture before regression to 165 subset, had
segment r/o protect ... so could have a mixture of virtual address
spaces some with no protect and some with r/o protect). VM370 was
originally going to use the original segment protect for share segments,
but with the retrenching to 165 subset, initially had to fall back to
the storage key protect hack.

original 360 was key protect was 2kbytes but later changed to 4k,
a later feature was special storage r/o page protect ... to prevent
even authorized kernel (key=0) from storing
https://www.ibm.com/docs/en/zos-basic-skills?topic=integrity -what-is-storage-protection

360 program interrupt .... "04" storage key protection (instead of page
fault interrupt)
https://en.wikipedia.org/wiki/IBM_System/360_architecture#Pr ogram_interruption

possible to trap on program interrupt (and simulate copy-on-write).

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Anonymous on Thu, 23 Jan 2025 04:24:10 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Wed, 22 Jan 2025 10:45:40 -0000 (UTC), Waldek Hebisch wrote:

> My reading of Multics documentaion is that there was single creator.

Read it again.

From the MULTICS Intro Course, for example (chapter 11), you see on page
11-5 the access modes for directories: “modify” (ability to change
attributes of entries, including their ACLs) versus “append” (ability to
add new entries). Then from 11-6, the ACL is a sequence of 3-part entries,
and the accessing entity is compared against each entry in sequence until
a match is found (with corresponding access granted) or the end of the
list is reached (which means no access).

There is no special ACL entry for the item creator; all ACL entries can be
modified/deleted.

Subject: Re: vfork history, Multics vs Unix
Posted by cross on Thu, 23 Jan 2025 04:29:59 GMT
View Forum Message <> Reply to Message

In article <vmsddo$25vl$1@gal.iecc.com>, John Levine <johnl@taugh.com> wrote:
> According to Dan Cross <cross@spitfire.i.gajendra.net>:
>> `vfork` was introduced as an optimization: like fork, it creates
>> a new process, but unlike `fork`, on success, the system
>> [lends the parent address space to the child until it exec's or exits]
>
>> CoW is a much cleaner way to eliminate a lot of the cost of fork ...
>
> The VAX-11/750 had microcode bugs, one of which made read-only pages in the
> stack segment not work reliably. VMS didn't use them so DEC wasn't interested in
> fixing the bug, but without read-only pages that give a page fault if you
> try to write to them, you can't do CoW. I believe vfork() was as much a
> workaround for that bug as anything else.

I've never heard this before, and I've seen no contemporary
sources that suggest this bug (which I can believe existed) was
a motivation for `vfork`. Do you have a source?

Every source I've seen suggests that `vfork` was introduced for
efficiency in the case I outlined in my previous reply, but that

it was understood to be inferior to CoW, which would have been
difficult to implement for other reasons. For instance, "Design
and Implementation of the Berkeley Virtual Memory Extensions to
the UNIX Operating System" by Baboglu, Joy, and Porcar notes:

|It should be noted that an implementation of fork using a _Copy
|On Write_ mechanism would be completely transparent and nearly
|as efficient as vfork. Such a mechanism would rely on more
|general sharing primitives and data structures than are present
|in the current version of the system, so it has not been
|implemented.

This mentions nothing about hardware bugs, which seems like it
would have been important to note, and suggests that the CoW
work was deferred for other reasons having to do with the
overall structure of the kernel, not the hardware.

"The Design And Implementation of the 4.3BSD Unix Operating
System" by Leffler et al says that "vfork" is "generally
considered to be an architectural blemish." One would think
that if a hardware bug was one of the princple motivations for
vfork, this would at least be mentioned in self-defense.

Moreover, Reiser and London's VM system for VAX Unix, done at
Bell Labs in 1979 or 1980 and independent of the Berkeley work,
had copy-on-write. Granted that primarily targeted the 11/780
but there's some evidence it survived on into System V which
surely ran on the 750.

Finally, if limited solely to the stack, this would be easy
enough to work around: stacks in those days were relatively
small, so just copy them on `fork` and use CoW and R/O sharing
for the (larger) data and text segments.

> In practice, most of the time if a program touches a stack page at all it's
> going to write to it, so copy-on-touch would have worked about as well without
> needing read-only stack pages. IBM mainframes didn't do page fault on write
> until quite late (S/390?) so some of their systems did CoT.

This also seems like a reasonable approach, and could be applied
solely to the stack if the problem were, indeed, limited to that
segment.

 - Dan C.

Subject: Re: Multics vs Unix

Posted by Anne &amp; Lynn Wheel on Thu, 23 Jan 2025 04:30:55 GMT
View Forum Message <> Reply to Message

antispam@fricas.org (Waldek Hebisch) writes:
> Well, but is it essentially different than CMS, DOS or loading
> shared libraries on Unix? In each case code and data is loaded,
> run und then unloaded. Loading of new thing means that old
> thing does not run. Multics documentation is not entirely
> clear on this, but it seems that at given protection level
> there is single stack, which would make things like coroutines
> hairy.

when I originally did page-mapped CMS filesystem ... it included support
for independent location segments ... but was quite crippled because CMS
heavily borrowed OS/360 assemblers and compilers. While OS/360 made
reference to program relocation ... address referenses had to be changed
to fixed/absolute before execution. As a result, default program
segments required fixed address loading.

TSS/360 had kept "addresses" as offset (from the segment base
address). To get address independnt segments (including shared segments
where the same shared segment could appear concurrently at different
virtual addresses in different virtual address spaces), I had to hack
the (application) code to something that resembled the TSS/360
convention.

A small subset of this was released in VM370R3 as DCSS (but without the
page-mapped filesystem and w/o the independent address location support)
.... where the shared segment images were saved in special defined VM370
disk areas

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by cross on Thu, 23 Jan 2025 04:36:22 GMT
View Forum Message <> Reply to Message

In article <mddmsfi1i2d.fsf@panix5.panix.com>,
Rich Alderson <news@alderson.users.panix.com> wrote:
> scott@slp53.sl.home (Scott Lurndal) writes:
>> IIRC, it was SunOS (Pre-solaris it was based on BSD) that did CoW first.
>
> That would not surprise me, given the number of Stanford people at Sun. CoW
> was a long time feature of TENEX/TOPS-20 from the early 1970s, more than a
> decade before Sun was founded.

London and Reiser had it working on Unix at Bell Labs in 1979 or
1980, before Sun was founded. Sun's initial Unix offering was
a port of V7 done by UniSoft. SunOS (based on 4.2BSD) came in
1982, if Wikipedia is to believed; this suggests that the London
and Reiser VM at BTL was the first verison of Unix with CoW
virtual memory.

Contemporary observations were that it was better than BSD, but
Research re-ported 4.1 as the basis of what became 8th Edition,
picking up the BSD VM, and the London and Reiser work never had
the same impact. I understand it formed the basis for the VM
system in an early version of System V, but by the time of SVR4
that was mostly gone, replaced by a "new" VM system mostly taken
from SunOS.

 - Dan C.

Subject: Re: vfork history, Multics vs Unix
Posted by John Levine on Thu, 23 Jan 2025 04:43:41 GMT
View Forum Message <> Reply to Message

According to Lynn Wheeler <lynn@garlic.com>:
> John Levine <johnl@taugh.com> writes:
>> In practice, most of the time if a program touches a stack page at all it's
>> going to write to it, so copy-on-touch would have worked about as well without
>> needing read-only stack pages. IBM mainframes didn't do page fault on write
>> until quite late (S/390?) so some of their systems did CoT.
>
> ... but 360 key protect had both (real memory) store (& 360 optional
> fetch) protect ...

Right, but on a fetch protect violation the instruction might be suppressed or
it might be terminated. In the latter case the supervisor can't fix and
restart the offending instruction.

S/390 added the suppression-on-protection feature which made fetch protection
traps restartable. A note in the POO says it's for copy-on-write in AIX/ESA.
--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: vfork history, Multics vs Unix
Posted by John Levine on Thu, 23 Jan 2025 04:48:53 GMT
View Forum Message <> Reply to Message

According to Dan Cross <cross@spitfire.i.gajendra.net>:
>> The VAX-11/750 had microcode bugs, one of which made read-only pages in the
>> stack segment not work reliably. VMS didn't use them so DEC wasn't interested in
>> fixing the bug, but without read-only pages that give a page fault if you
>> try to write to them, you can't do CoW. I believe vfork() was as much a
>> workaround for that bug as anything else.
>
> I've never heard this before, and I've seen no contemporary
> sources that suggest this bug (which I can believe existed) was
> a motivation for `vfork`. Do you have a source?

Nothing written that I can find. Yale got a /750 about the same time that
Berkeley did, and Bill Joy and I both discovered that a bug in the MOVTUC
instruction made printf() not work. I was cross compiling from a PDP-11 which
was slow and tedious so he fixed it before I did. My recolection is that he
discovered the read only stack bug then too.

--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: Multics vs Unix
Posted by cross on Thu, 23 Jan 2025 04:52:20 GMT
View Forum Message <> Reply to Message

In article <vmsehi$1tnl3$1@paganini.bofh.team>,
Waldek Hebisch <antispam@fricas.org> wrote:
> Dan Cross <cross@spitfire.i.gajendra.net> wrote:
>> In article <vmqivp$1ldng$3@paganini.bofh.team>,
>> Waldek Hebisch <antispam@fricas.org> wrote:
>>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>
>> Meta note: please don't feed the troll.
>>
>>>> [snip]
>>>>
>>>> MULTICS also considered that different users might want to cooperate,
>>>> share files etc, not be shut off in their own separate VMs. It had a
>>>> sophisticated access-control system to allow this.
>>>
>>> Clearly Multics was more advanced. I was simply pointing out that
>>> the same "single task for single user" mindset seem to be present
>>> in Multics.
>>
>> Not precisely, just that the notion of a "process" was rather
>> different than in e.g. Unix. In Multics, processes tend to be

>> long-lived things and one may run many programs in a the context
>> of a single process, with the text and data of those programs
>> coming and going over time: to run something, I link it into the
>> address space of the current process, it does whatever it does,
>> and when it's done it is unlinked.
>
> Well, but is it essentially different than CMS, DOS or loading
> shared libraries on Unix?

CMS and Unix are rather different, as Lynn has described. But
for the others, I'd say, yes, quite, because all of those things
run at the same protection level in a single address space.
That is the critical part missing in other systems.

> In each case code and data is loaded, run und then unloaded.

True, reductive to the point of not being useful.

> Loading of new thing means that old thing does not run.

Not true; loading a new thing in (say) Multics doesn't mean that
you can't refer to other segments that had been linked earlier,
for instance. Things can be suspended and resumed later, etc.

> [snip]
> IIUC only tradition and need for multtasking prevents Unix shell
> from taking commands from shared library and executing them in
> a single process.

It's deeper than that. Nothing prevents a Unix shared library
from scribbling all over the memory of whatever program loads
it; in the worst case, it may corrupt some critical data
structure that the shell needs causing _it_ to crash, perhaps
long after the shared image had been unloaded.

 - Dan C.

Subject: Re: Multics vs Unix
Posted by Anonymous on Thu, 23 Jan 2025 05:40:20 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Wed, 22 Jan 2025 18:30:55 -1000, Lynn Wheeler wrote:

> TSS/360 had kept "addresses" as offset (from the segment base address).
> To get address independnt segments (including shared segments where the

> same shared segment could appear concurrently at different virtual
> addresses in different virtual address spaces), I had to hack the
> (application) code to something that resembled the TSS/360 convention.

How much of this was enforced by the OS? I have a suspicion that there is
a dependence on well-behaved applications here, to maintain the integrity
of the whole system.

Subject: Re: vfork history, Multics vs Unix
Posted by scott on Thu, 23 Jan 2025 14:32:39 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:
> On Thu, 23 Jan 2025 03:35:20 -0000 (UTC), John Levine wrote:
>
>> The VAX-11/750 had microcode bugs, one of which made read-only pages in
>> the stack segment not work reliably. VMS didn't use them so DEC wasn't
>> interested in fixing the bug, but without read-only pages that give a
>> page fault if you try to write to them, you can't do CoW. I believe
>> vfork() was as much a workaround for that bug as anything else.
>
> vfork(2) was still needed anyway, bug or no bug. That’s why Linux still
> has it, for example.

Linux has it to support legacy BSD code. Not because it's "needed"
for some abstract reason.

Subject: Re: vfork history, Multics vs Unix
Posted by cross on Thu, 23 Jan 2025 15:55:13 GMT
View Forum Message <> Reply to Message

In article <bqskP.1852771$bYV2.353023@fx17.iad>,
Scott Lurndal <slp53@pacbell.net> wrote:
> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>> On Thu, 23 Jan 2025 03:35:20 -0000 (UTC), John Levine wrote:
>>
>>> The VAX-11/750 had microcode bugs, one of which made read-only pages in
>>> the stack segment not work reliably. VMS didn't use them so DEC wasn't
>>> interested in fixing the bug, but without read-only pages that give a
>>> page fault if you try to write to them, you can't do CoW. I believe
>>> vfork() was as much a workaround for that bug as anything else.
>>
>> vfork(2) was still needed anyway, bug or no bug. That’s why Linux still
>> has it, for example.
>

> Linux has it to support legacy BSD code. Not because it's "needed"
> for some abstract reason.

Yup. vfork is one of those things that was obviously a mistake
in hindsight, but the semantics were sufficiently different that
once it was out there, you're stuck with it forever.

 - Dan C.

Subject: Re: vfork history, Multics vs Unix
Posted by scott on Thu, 23 Jan 2025 17:24:19 GMT
View Forum Message <> Reply to Message

cross@spitfire.i.gajendra.net (Dan Cross) writes:
> In article <bqskP.1852771$bYV2.353023@fx17.iad>,
> Scott Lurndal <slp53@pacbell.net> wrote:
>> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>>> On Thu, 23 Jan 2025 03:35:20 -0000 (UTC), John Levine wrote:
>>>
>>>> The VAX-11/750 had microcode bugs, one of which made read-only pages in
>>>> the stack segment not work reliably. VMS didn't use them so DEC wasn't
>>>> interested in fixing the bug, but without read-only pages that give a
>>>> page fault if you try to write to them, you can't do CoW. I believe
>>>> vfork() was as much a workaround for that bug as anything else.
>>>
>>> vfork(2) was still needed anyway, bug or no bug. That’s why Linux still
>>> has it, for example.
>>
>> Linux has it to support legacy BSD code. Not because it's "needed"
>> for some abstract reason.
>
> Yup. vfork is one of those things that was obviously a mistake
> in hindsight, but the semantics were sufficiently different that
> once it was out there, you're stuck with it forever.

vfork(2) was obsolete in Open Group Standard Base Specification Issue 6,
and has been completely removed from Issue 8.

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Thu, 23 Jan 2025 18:34:23 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:
> How much of this was enforced by the OS? I have a suspicion that there is
> a dependence on well-behaved applications here, to maintain the integrity

> of the whole system.

well supervisor/problem separation to protect the OS.

Note original 801/RISC had cp.r operating system and pl.8 programming
language and claimed it needed no hardware supervisor/problem mode
separation ... the pl.8 language would only generate valid programs and
cp.r would only load valid pl.8 programs for execution (as a result
inline library code could execute all operations, w/o requiring
supervisor calls).

The ROMP chip (w/cp.r & pl.8) was originally going to be be used for the
followon to the displaywriter. When that got canceled they decided to
pivot to the unix workstation market and got the company that had done
PC/IX for the IBM/PC to do port for ROMP ... becoming PC/RT and
AIX. However ROMP chip had to have hardware supervisor/problem mode for
the UNIX paradigm

CMS XMAS tree exec ... displayed xmas greeting and (on 3270 terminal)
blinking light xmas tree ... something like this simulation
https://www.garlic.com/~lynn/2007v.html#54

but the exec also resent a copy to everybody in the person's name/contact file
https://en.wikipedia.org/wiki/Christmas_Tree_EXEC
... swamping bitnet
https://en.wikipedia.org/wiki/BITNET

.... this was dec87, a year before the 88 morris worm
https://en.wikipedia.org/wiki/Morris_worm

At the 1996 m'soft MDC at Moscone ... all the banner's said "Internet"
.... but the constant refrain in all the sessions was "preserve your
investment" ... which was visual basic automagic execution in data files
.... including email messages ... giving rise to enormous explosion in
virus attacks.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Peter Flass on Thu, 23 Jan 2025 18:52:01 GMT
View Forum Message <> Reply to Message

Waldek Hebisch <antispam@fricas.org> wrote:
> Dan Cross <cross@spitfire.i.gajendra.net> wrote:
>> In article <vmqivp$1ldng$3@paganini.bofh.team>,

>> Waldek Hebisch <antispam@fricas.org> wrote:
>>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>
>> Meta note: please don't feed the troll.
>>
>>>> [snip]
>>>>
>>>> MULTICS also considered that different users might want to cooperate,
>>>> share files etc, not be shut off in their own separate VMs. It had a
>>>> sophisticated access-control system to allow this.
>>>
>>> Clearly Multics was more advanced. I was simply pointing out that
>>> the same "single task for single user" mindset seem to be present
>>> in Multics.
>>
>> Not precisely, just that the notion of a "process" was rather
>> different than in e.g. Unix. In Multics, processes tend to be
>> long-lived things and one may run many programs in a the context
>> of a single process, with the text and data of those programs
>> coming and going over time: to run something, I link it into the
>> address space of the current process, it does whatever it does,
>> and when it's done it is unlinked.
>
> Well, but is it essentially different than CMS, DOS or loading
> shared libraries on Unix? In each case code and data is loaded,
> run und then unloaded. Loading of new thing means that old
> thing does not run. Multics documentation is not entirely
> clear on this, but it seems that at given protection level
> there is single stack, which would make things like coroutines
> hairy.

[sorry for not pruning more, but it’s harder on an iPad]

Not multithreading, but there’s an interesting story about a program
getting an exception due to a missing segment (subroutine.) System
invokes,the debugger (if I have this right), and gives control back to the
user at the terminal. User writes the code in question, compiles it, and
returns to the interrupted program, which then continues on its merry way.

I believe that, toward the end, Multics was getting threads.

>
> Multics hardware have some advantage here, because segments
> simplify relocation.
>
> IIUC only tradition and need for multtasking prevents Unix shell
> from taking commands from shared library and executing them in
> a single process.

>
>> Many of the DEC OSes do more or less the same thing. VMS calls
>> this process "running up" or "running down" an "image".
>>
>> Of course, to work properly, one needs support from the hardware
>> so that an errant program does not clobber the process itself;
>> in Multics this involved the ring architecture and how it worked
>> with respedct to segmentsand call gates, which were provided by
>> the GE and Honeyll/Bull hardware in the 645-6180-DPS/8M series
>> of machines. A program error could be trapped by the system and
>> dealt with one way or another and in all cases the process
>> itself was protcted from these shenanigans since the process
>> spanned multiple protection rings on a per-segment basis and
>> access to higher rings was controlled via the call-gating
>> mechanism.
>
> To say the truth some details are not clear. More precisely,
> parts running in a single security context (the same ring, user
> and whatever Multics considers relevant) seem to have equal rights,
> so writable memory of one part seem to be unprotected from changes
> by other parts. I am not saying that this is bad, simply given fact
> that a lot of things may run together there seem to be
> less protection than suggested by Multics propaganda.
>
> Another thing is what changes security context? There are
> gates, that at first looks clear. However, there seem to be
> segment table and ATM it is not clear to me which part of
> OS is responsible for setting up segment table.
>
>> On VMS, page tables contained protection bits for all four CPU
>> protection levels, and were used to control access in a vaguely
>> similar way. It wasn't the single-level store of Multics by
>> any means, but the process was protected from errant programs
>> (say) trashing the command interpreter.
>>
>> In all of these sytems, processes are rather heavy-weight
>> constructs, and creating and deleting them is expensive.
>>
>> VM was different yet again: each user session gets its own VM
>> and then boots whatever OS it wants in that VM. Most
>> interactive users ran CMS, which was a single-user system, but
>> paravirtualized (that is, it "knew" it was running in a VM) and
>> let the user interact with other users, sending them messages
>> and so forth. Users were (usually?) allocated "minidisks" for
>> storage; this was a typically small virtual disk that they
>> initialized with a filesystem that CMS understood for their
>> own files. Such disks could be shared and linked between
>> different VMs, allowing easy file sharing.

>>
>> My sense was that VM was highly interactive and collaborative.
>> Maybe it was just our local site, but users sent each other
>> messages _all the time_, in a way that they did not on Unix, VMS
>> or the PDP-10. We had locally written scripts that let multiple
>> users "chat"; as I mentioned, disks could be shared ("linked")
>> between users, and there was a mechanism to query which users
>> were linked to a disk. One then queried that list and send each
>> such linked user a short message.
>>
>> The terminal interface was generally clever about how these were
>> displayed; if you were in (say) the editor, you didn't get
>> notifications until you took some action where the interface
>> would switch (perhaps you had to exit the editor; I no longer
>> recall).
>>
>> Personally, I found Unix and VMS far more pleasant for program
>> development, but VM/CMS a lot more social, though that probably
>> had a lot more to do with the local community than anything
>> else.
>
> I think so. I used VM/CMS for accessing e-mail, but all other
> things were done on other machines (some people tried to make
> more use of it but that was rare). This was for communication
> with remote sites, locally there was non-computer communication.
>
>>>> > Lynn gave figures that suggest otherwise: that on similar hardware VM
>>>> > could handle more users than competition.
>>>>
>>>> Not likely. Their minimum hardware requirements for running a
>>>> â??timesharingâ?? system were about twice what an equivalent DEC OS needed on
>>>> a PDP-11, for example.
>>>
>>> It seem that you had to get "right" version of VM (one enhanced by
>>> Lynn). However, note that main overhead of virtualization is due
>>> to nested page tables. Since CMS was unpaged there was no such
>>> overhead for VM/CMS combination.
>>
>> Comparisons between a PDP-11 and a mainframe are silly. The
>> PDP-11 is a small departmental computer, and hardly DEC's only
>> offering. Mainframes are much larger and more capable machines;
>> a better comparison to contemporary DEC hardware would be
>> the PDP-10 or (later) one of the larger VAX models. Even within
>> DEC's various lines, there are major differences between
>> systems: there's a big difference between RT-11 and TOPS-20, for
>> instance.
>
> IIRC Lynn in the past gave comparison with big machines.

>
>>> Different way of looking at this is that normal operating system
>>> (say Linux) presents a virtual machine to user programs. This
>>> virtual machine does not try to look like any real machine.
>>
>> Eh.... A couple of things.
>>
>> First, when discussing things in an historical context, I don't
>> know that I would characterize "Linux" as "normal", in so far as
>> back in those days there _was_ no "normal". Indeedk Unix was
>> often looked at as the outlier in the way it did things. I
>> remember a VMS administrator bristling at someone saying that
>> Unix was the "textbook OS": "Yeah, because the textbook was
>> written about Unix!"
>
> OK. I mentione Linux because in description of Unix systems
> "division of labor" seem to be more explicit than in other
> systems: there is kernel vesrus userland. Multics with its
> rings splits this into layers and at least overview/introductory
> documentation do not make clear what are responsibilites of
> various layers.
>
>> Second, the Unix process model is of an augmented virtual
>> machine. The idea is that each "process" sees a virtual
>> computer with some restrictions (one doesn't get unfettered
>> access to physical memory, but rather, the virtual address space
>> that has been constructored for that process) and some additions
>> (system calls may be thought of as virtual instructions) and of
>> course a bunch of useful resources (files and so forth). But of
>> course the vast bulk of what a program depends on, and
>> importantly the instructions what it executes, is in fact
>> defined by the underlying hardware, so processes very much _do_
>> try to look like the underlying machine, at least in so far as
>> the CPU, RAM, etc, is concerned. IO is obviously different.
>>
>>> So, the question is if 360 machine interface is worse than
>>> modern OS interfaces. Well, it is worse, but not that much.
>>
>> Well, the IO architecture was much maligned in its time.
>>
>>> And in case of VM/CMS worst parts were smoothed out by using
>>> DIAGNOSE as a way to do syscalls and (later) because parts
>>> of VM were moved to microcode (not applicable to modern
>>> architecters, but useful in 370 era).
>>
>> The VM architecture was actually quite clever. There are
>> important classes of problems not handled well by the Unix
>> one-size-fits-all process model where a set of global

>> resources are imperfectly multiplexed across all processes on
>> the machine that one could imagine would be better served by
>> custom systems running on dedicated resources allocated to
>> light-weight virtual machines. Modern Linux (and other)
>> systems give the program a bunch of knobs to twist to try and
>> approximate something optimal here, but they often don't
>> interact super well and the result is never perfect. A
>> different abstraction might lead to a system much simpler and
>> better overall performance and utilization.
>

--
Pete

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Thu, 23 Jan 2025 19:00:18 GMT
View Forum Message <> Reply to Message

---

cross@spitfire.i.gajendra.net (Dan Cross) writes:
> CMS and Unix are rather different, as Lynn has described. But
> for the others, I'd say, yes, quite, because all of those things
> run at the same protection level in a single address space.
> That is the critical part missing in other systems.

note: OS/360 used 360 privileged instruction and supervisor/problem mode and for concurrent application program separation with (4bit) storage protection keys (for up to 15 concurrent applications plus kernel/supervisor).

This initial move of OS/360 MVT to VS2 was into a single 16mbyte virtual address space (something like running MVT in a CP67 16mbyte virtual machine). However, as systems got larger and more powerful ... they need to go past 15 cap. The VS2/SVS to VS2/MVS was to give each concurrent executing program its own 16mbyte virtual address space.

However, the OS/360 heritage was heavily pointer-passing API. In order for the kernel calls to access program parameter list they mapped an 8mbyte image of the MVS kernel into every 16mbyte virtual address space (so kernel call processing was done in the same virtual address space as the caller).

VS2/MVS also mapped kernel subsystems into their own seperate 16mbyte virtual address spaces ... so application calls to subsystems, paraemters were now in different address space. For this they created the Common Segment Area ("CSA") that was mapped into every 16mbyte

virtual adress space for subsystem API call parameters. The issue was
subsystem parameter list space requirement was somewhat proportional to
number subsystems and number of concurrently executing programs ... and
by the time of 370 3033 processor CSA had morphed into 5-6mbyte "Common
System Area" (CSA) leaving only 2-3 mbytes for programs, but was
threatening to become 8mbytes (leaving zero for applictions).

This was part of POK (high-end 370s) VS2/MVS mad rush to 370/XA
architecture, 31-bit addressing, access register, Program Call and
Program Return instructions.

Program Call was akin to kernel call ... table of all the MVS subsystems
and their associated address space pointers. A Program Call would select
the specific subsystem entry, move the caller's address space to
secondary and load the subsystem's address space as primary. A
semi-privileged subsystem would access parameter list in the caller's
"secondary address space" (no longer needing "CSA" space). Program
Return would move the caller's secondary address space pointer to
primary and return (task switches would have to save/restore both the
primary and any secondary address space pointers).

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Peter Flass on Thu, 23 Jan 2025 19:07:17 GMT
View Forum Message <> Reply to Message

Lynn Wheeler <lynn@garlic.com> wrote:
> antispam@fricas.org (Waldek Hebisch) writes:
>> Well, but is it essentially different than CMS, DOS or loading
>> shared libraries on Unix? In each case code and data is loaded,
>> run und then unloaded. Loading of new thing means that old
>> thing does not run. Multics documentation is not entirely
>> clear on this, but it seems that at given protection level
>> there is single stack, which would make things like coroutines
>> hairy.
>
> when I originally did page-mapped CMS filesystem ... it included support
> for independent location segments ... but was quite crippled because CMS
> heavily borrowed OS/360 assemblers and compilers. While OS/360 made
> reference to program relocation ... address referenses had to be changed
> to fixed/absolute before execution. As a result, default program
> segments required fixed address loading.
>
> TSS/360 had kept "addresses" as offset (from the segment base

> address). To get address independnt segments (including shared segments
> where the same shared segment could appear concurrently at different
> virtual addresses in different virtual address spaces), I had to hack
> the (application) code to something that resembled the TSS/360
> convention.
>
> A small subset of this was released in VM370R3 as DCSS (but without the
> page-mapped filesystem and w/o the independent address location support)
> ... where the shared segment images were saved in special defined VM370
> disk areas
>

It was fun mapping the DCSS to specific memory addresses, and making sure that a program didn't need two mapped to the same address. This, I think, was S/370, so 31-bit (16MB) address space. Didn't DCSS sizes have to be a multiple of 64K? It's been a while, so I don't recall the details.

--
Pete

Subject: Re: Multics vs Unix
Posted by Peter Flass on Thu, 23 Jan 2025 19:07:18 GMT

View Forum Message <> Reply to Message

Dan Cross <cross@spitfire.i.gajendra.net> wrote:
> In article <vmsehi$1tnl3$1@paganini.bofh.team>,
> Waldek Hebisch <antispam@fricas.org> wrote:
>> Dan Cross <cross@spitfire.i.gajendra.net> wrote:
>>> In article <vmqivp$1ldng$3@paganini.bofh.team>,
>>> Waldek Hebisch <antispam@fricas.org> wrote:
>>>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>>
>>> Meta note: please don't feed the troll.
>>>
>>>> > [snip]
>>>> >
>>>> > MULTICS also considered that different users might want to cooperate,
>>>> > share files etc, not be shut off in their own separate VMs. It had a
>>>> > sophisticated access-control system to allow this.
>>>>
>>>> Clearly Multics was more advanced. I was simply pointing out that
>>>> the same "single task for single user" mindset seem to be present
>>>> in Multics.
>>>
>>> Not precisely, just that the notion of a "process" was rather
>>> different than in e.g. Unix. In Multics, processes tend to be
>>> long-lived things and one may run many programs in a the context

>>> of a single process, with the text and data of those programs
>>> coming and going over time: to run something, I link it into the
>>> address space of the current process, it does whatever it does,
>>> and when it's done it is unlinked.
>>
>> Well, but is it essentially different than CMS, DOS or loading
>> shared libraries on Unix?
>
> CMS and Unix are rather different, as Lynn has described. But
> for the others, I'd say, yes, quite, because all of those things
> run at the same protection level in a single address space.
> That is the critical part missing in other systems.
>
>> In each case code and data is loaded, run und then unloaded.
>
> True, reductive to the point of not being useful.
>
>> Loading of new thing means that old thing does not run.
>
> Not true; loading a new thing in (say) Multics doesn't mean that
> you can't refer to other segments that had been linked earlier,
> for instance. Things can be suspended and resumed later, etc.
>
>> [snip]
>> IIUC only tradition and need for multtasking prevents Unix shell
>> from taking commands from shared library and executing them in
>> a single process.
>
> It's deeper than that. Nothing prevents a Unix shared library
> from scribbling all over the memory of whatever program loads
> it; in the worst case, it may corrupt some critical data
> structure that the shell needs causing _it_ to crash, perhaps
> long after the shared image had been unloaded.

I love these kinds of bugs, they're hell to find. Some program leaves a
bomb in memory and it goes off sometime later. You can look and say "oh,
look, this field should be zero", but you have no idea who did it.

--
Pete

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Thu, 23 Jan 2025 19:22:52 GMT
View Forum Message <> Reply to Message

Lynn Wheeler <lynn@garlic.com> writes:
> This was part of POK (high-end 370s) VS2/MVS mad rush to 370/XA

> architecture, 31-bit addressing, access register, Program Call and
> Program Return instructions.

.... VS2/MVS was also getting quite bloated and getting to 370/XA
was taking too long ... so there were pieces retrofitted to 3033.

VS2/MVS needed more real storage and cramped in 16mbyte real. The 16bit
page table entry had 12bit real page numbers (for 16mbyte) ... and flag
bits ... but there were two unused bits. They did a 3033 hack where the
two unused bits could be prepended to the 12bit real page number for
14bits (64mbyte) mapping 16mbyte virtual address spaces into 64mbyte
real (all instructions were still 24bit/16mbyte).

370 IDALs (full word) had I/O channel program extension for storage
address ... so it was possible to use extend 3033 I/O to 31bit.

there was still periodic problem where kernel code had to access virtual
pages at real addresses ... so there was a "bring down" process where a
virtual page was moved from a real address above the 16mbyte line to
real address "below the line".

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by scott on Thu, 23 Jan 2025 19:48:13 GMT
View Forum Message <> Reply to Message

Peter Flass <peter_flass@yahoo.com> writes:
> Dan Cross <cross@spitfire.i.gajendra.net> wrote:
>> In article <vmsehi$1tnl3$1@paganini.bofh.team>,
>> Waldek Hebisch <antispam@fricas.org> wrote:
>>> Dan Cross <cross@spitfire.i.gajendra.net> wrote:
>>>> In article <vmqivp$1ldng$3@paganini.bofh.team>,
>>>> Waldek Hebisch <antispam@fricas.org> wrote:
>>>> > Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>>>
>>>> Meta note: please don't feed the troll.
>>>>
>>>> >> [snip]
>>>> >>
>>>> >> MULTICS also considered that different users might want to cooperate,
>>>> >> share files etc, not be shut off in their own separate VMs. It had a
>>>> >> sophisticated access-control system to allow this.
>>>> >
>>>> > Clearly Multics was more advanced. I was simply pointing out that
>>>> > the same "single task for single user" mindset seem to be present

>>>> > in Multics.
>>>>
>>>> Not precisely, just that the notion of a "process" was rather
>>>> different than in e.g. Unix. In Multics, processes tend to be
>>>> long-lived things and one may run many programs in a the context
>>>> of a single process, with the text and data of those programs
>>>> coming and going over time: to run something, I link it into the
>>>> address space of the current process, it does whatever it does,
>>>> and when it's done it is unlinked.
>>>
>>> Well, but is it essentially different than CMS, DOS or loading
>>> shared libraries on Unix?
>>
>> CMS and Unix are rather different, as Lynn has described. But
>> for the others, I'd say, yes, quite, because all of those things
>> run at the same protection level in a single address space.
>> That is the critical part missing in other systems.
>>
>>> In each case code and data is loaded, run und then unloaded.
>>
>> True, reductive to the point of not being useful.
>>
>>> Loading of new thing means that old thing does not run.
>>
>> Not true; loading a new thing in (say) Multics doesn't mean that
>> you can't refer to other segments that had been linked earlier,
>> for instance. Things can be suspended and resumed later, etc.
>>
>>> [snip]
>>> IIUC only tradition and need for multtasking prevents Unix shell
>>> from taking commands from shared library and executing them in
>>> a single process.
>>
>> It's deeper than that. Nothing prevents a Unix shared library
>> from scribbling all over the memory of whatever program loads
>> it; in the worst case, it may corrupt some critical data
>> structure that the shell needs causing _it_ to crash, perhaps
>> long after the shared image had been unloaded.
>
> I love these kinds of bugs, they’re hell to find. Some program leaves a
> bomb in memory and it goes off sometime later. You can look and say “oh,
> look, this field should be zero”, but you have no idea who did it.

We have tools to help find those bugs.   valgrind and various address sanitizers
have become quite good at finding memory corruption.

Subject: Re: Multics vs Unix
Posted by Anonymous on Thu, 23 Jan 2025 21:28:15 GMT
View Forum Message <> Reply to Message

Originally posted by: drb

> Not multithreading, but there’s an interesting story about a program
> getting an exception due to a missing segment (subroutine.) System
> invokes,the debugger (if I have this right), and gives control back to the
> user at the terminal. User writes the code in question, compiles it, and
> returns to the interrupted program, which then continues on its merry way.

The Prime folks, when they got their EPF stuff fully implemented,
had gotten close enough to "Multics in a matchbox" that they could
do this demo too, and did. Take linkage fault. Write and compile
the missing routine, resume the original program where it left off,
system snaps the missing links, win.

De

Subject: Re: Multics vs Unix
Posted by Anonymous on Fri, 24 Jan 2025 00:51:16 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 23 Jan 2025 12:07:18 -0700, Peter Flass wrote:

>> Nothing prevents a Unix shared library from
>> scribbling all over the memory of whatever program loads it ...
>
> I love these kinds of bugs, they’re hell to find.

This is why I think a lot of current apps don’t do plugins/addons as DSOs
any more. For example, GIMP and Inkscape run them as separate processes,
with some standardized IPC protocol for passing data back and forth.

This also has the advantage that you can use whatever language you like to
write the addon, if it can wrap the IPC part. This is how I managed to
write plugins in Python 3 to work with GIMP 2.x, for example
<https://gitlab.com/ldo/pylibgimp2/>.

Subject: Re: Multics vs Unix
Posted by Anonymous on Fri, 24 Jan 2025 00:53:23 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 23 Jan 2025 09:00:18 -1000, Lynn Wheeler wrote:

> note: OS/360 used 360 privileged instruction and supervisor/problem mode
> and for concurrent application program separation with (4bit) storage
> protection keys (for up to 15 concurrent applications plus
> kernel/supervisor).

So each (physical?) page belonged to exactly one user process (or the
kernel)?

How did this handle shared sections accessible by more than one process?

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Fri, 24 Jan 2025 02:19:10 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:
> So each (physical?) page belonged to exactly one user process (or the
> kernel)?
>
> How did this handle shared sections accessible by more than one process?

PSW key=0 (nominally kernel/supervisor) allowed access to all pages,
nonzero PSW only could store (and/or fetch if the fetch protect feature
was installed and fetch set) to pages with matching storage protect key.

shared code, tended to be only store protected (but not fetch protected)
.... and run with the PSW key of the invoking process (so could stored in
the invoking process pages).

originally 360 this were 2kbytes ... (in 4k paging environment ...
pairs of 2k storage keys were managed) ... after 370, moved to 4k.

os/360 running concurrent "regions" could do up to 15 ... protected from
each other. VS2 started with with single 16mbyte address space (VS2/SVS)
.... but had to move unique address space (for each "region") to move
past 15 and provide separation.

Storage Protect
https://en.wikipedia.org/wiki/IBM_System/360_architecture#St orage_protection

If the storage protection feature[2]17-17.1 is installed, then there is
a 4-bit storage key associated with every 2,048-byte block of storage
and that key is checked when storing into any address in that block by
either a CPU or an I/O channel. A CPU or channel key of 0 disables the

check; a nonzero CPU or channel key allows data to be stored only in a
block with the matching key.

Storage Protection was used to prevent a defective application from
writing over storage belonging to the operating system or another
application. This permitted testing to be performed along with
production. Because the key was only four bits in length, the maximum
number of different applications that could be run simultaneously was
15.

An additional option available on some models was fetch protection. It
allowed the operating system to specify that blocks were protected from
fetching as well as from storing.

.... snip ...

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Anonymous on Fri, 24 Jan 2025 06:54:22 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 23 Jan 2025 16:19:10 -1000, Lynn Wheeler wrote:

> shared code, tended to be only store protected (but not fetch protected)
> ... and run with the PSW key of the invoking process (so could stored in
> the invoking process pages).

So only shared code, no shared data? Was there any copy-on-write?

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Fri, 24 Jan 2025 07:41:05 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:
> So only shared code, no shared data? Was there any copy-on-write?

nothing precluded shared data ... os/360, still relative small real
storage ... so shared data tended to be disk resident and only what was
needed was specifically brought in for relatively short periods.

transaction processing systems ran business critical specific data from

disk nominally for very brief duration for transaction measured in
microseconds ... or batch processing that serially/sequentially
processed (relatively) large amounts of data.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Bill Findlay on Fri, 24 Jan 2025 12:02:39 GMT
View Forum Message <> Reply to Message

On 23 Jan 2025, Lynn Wheeler wrote
(in article <87tt9ppd8d.fsf@localhost>):

> cross@spitfire.i.gajendra.net (Dan Cross) writes:
>> CMS and Unix are rather different, as Lynn has described. But
>> for the others, I'd say, yes, quite, because all of those things
>> run at the same protection level in a single address space.
>> That is the critical part missing in other systems.
>
> note: OS/360 used 360 privileged instruction and supervisor/problem mode
> and for concurrent application program separation with (4bit) storage
> protection keys (for up to 15 concurrent applications plus
> kernel/supervisor).
>
> This initial move of OS/360 MVT to VS2 was into a single 16mbyte virtual
> address space (something like running MVT in a CP67 16mbyte virtual
> machine). However, as systems got larger and more powerful ... they need
> to go past 15 cap. The VS2/SVS to VS2/MVS was to give each concurrent
> executing program its own 16mbyte virtual address space.
>
> However, the OS/360 heritage was heavily pointer-passing API. In order
> for the kernel calls to access program parameter list they mapped an
> 8mbyte image of the MVS kernel into every 16mbyte virtual address space
> (so kernel call processing was done in the same virtual address space as
> the caller).
>
> VS2/MVS also mapped kernel subsystems into their own seperate 16mbyte
> virtual address spaces ... so application calls to subsystems,
> paraemters were now in different address space. For this they created
> the Common Segment Area ("CSA") that was mapped into every 16mbyte
> virtual adress space for subsystem API call parameters. The issue was
> subsystem parameter list space requirement was somewhat proportional to
> number subsystems and number of concurrently executing programs ... and
> by the time of 370 3033 processor CSA had morphed into 5-6mbyte "Common
> System Area" (CSA) leaving only 2-3 mbytes for programs, but was
> threatening to become 8mbytes (leaving zero for applictions).

>
> This was part of POK (high-end 370s) VS2/MVS mad rush to 370/XA
> architecture, 31-bit addressing, access register, Program Call and
> Program Return instructions.
>
> Program Call was akin to kernel call ... table of all the MVS subsystems
> and their associated address space pointers. A Program Call would select
> the specific subsystem entry, move the caller's address space to
> secondary and load the subsystem's address space as primary. A
> semi-privileged subsystem would access parameter list in the caller's
> "secondary address space" (no longer needing "CSA" space). Program
> Return would move the caller's secondary address space pointer to
> primary and return (task switches would have to save/restore both the
> primary and any secondary address space pointers).

It's quite a history of bungling, isn't it?

--
Bill Findlay

Subject: Re: vfork history, Multics vs Unix
Posted by cross on Fri, 24 Jan 2025 13:00:46 GMT
View Forum Message <> Reply to Message

In article <7XukP.83392$G93a.13310@fx05.iad>,
Scott Lurndal <slp53@pacbell.net> wrote:
> cross@spitfire.i.gajendra.net (Dan Cross) writes:
>> In article <bqskP.1852771$bYV2.353023@fx17.iad>,
>> Scott Lurndal <slp53@pacbell.net> wrote:
>>> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>>>> On Thu, 23 Jan 2025 03:35:20 -0000 (UTC), John Levine wrote:
>>>>
>>>> > The VAX-11/750 had microcode bugs, one of which made read-only pages in
>>>> > the stack segment not work reliably. VMS didn't use them so DEC wasn't
>>>> > interested in fixing the bug, but without read-only pages that give a
>>>> > page fault if you try to write to them, you can't do CoW. I believe
>>>> > vfork() was as much a workaround for that bug as anything else.
>>>>
>>>> vfork(2) was still needed anyway, bug or no bug. That’s why Linux still
>>>> has it, for example.
>>>
>>> Linux has it to support legacy BSD code. Not because it's "needed"
>>> for some abstract reason.
>>
>> Yup. vfork is one of those things that was obviously a mistake
>> in hindsight, but the semantics were sufficiently different that
>> once it was out there, you're stuck with it forever.

>
> vfork(2) was obsolete in Open Group Standard Base Specification Issue 6,
> and has been completely removed from Issue 8.

Good. Or, as the kids say, "Bye, Felicia."

With `posix_spawn` there's much less of an impetus for it, so
it's well past time.

I suspect individual systems are going to support it forever,
though. Somewhat perplexingly (to me), NetBSD re-implemented it
and sprinkled it throughout their userspace ... for efficiency.
https://www.netbsd.org/docs/kernel/vfork.html

There are not enough tears in the universe to decry such shame.

 - Dan C.

Subject: Re: Multics vs Unix
Posted by scott on Fri, 24 Jan 2025 13:55:48 GMT
View Forum Message <> Reply to Message

Lynn Wheeler <lynn@garlic.com> writes:
> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>> So each (physical?) page belonged to exactly one user process (or the
>> kernel)?
>>
>> How did this handle shared sections accessible by more than one process?
>
> PSW key=0 (nominally kernel/supervisor) allowed access to all pages,
> nonzero PSW only could store (and/or fetch if the fetch protect feature
> was installed and fetch set) to pages with matching storage protect key.

   rodtsasdt 111111report*

The book _The Adolescence of P-1_ describes a student at U Waterloo
finding a timing window that allowed him to set the PSW key to zero,
thus allowing full access to the kernel/supervisor.   Which he used
to create a worm which spread via teleprocessing ports and eventually
became a self-aware AI.   The book was written in 1977.   It's still
a great read.

https://en.wikipedia.org/wiki/The_Adolescence_of_P-1

Subject: Re: Multics vs Unix

Posted by Anonymous on Fri, 24 Jan 2025 14:32:18 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

cross@spitfire.i.gajendra.net (Dan Cross) writes:
>>> CMS and Unix are rather different, as Lynn has described. But
>>> for the others, I'd say, yes, quite, because all of those things
>>> run at the same protection level in a single address space.
>>> That is the critical part missing in other systems.
>>
On 23 Jan 2025, Lynn Wheeler wrote
>> [...]
>> This initial move of OS/360 MVT to VS2 was into a single 16mbyte virtual
>> address space (something like running MVT in a CP67 16mbyte virtual
>> machine). However, as systems got larger and more powerful ... they need
>> to go past 15 cap. The VS2/SVS to VS2/MVS was to give each concurrent
>> executing program its own 16mbyte virtual address space.
>>
>> However, the OS/360 heritage was heavily pointer-passing API. In order
>> for the kernel calls to access program parameter list they mapped an
>> 8mbyte image of the MVS kernel into every 16mbyte virtual address space
>> (so kernel call processing was done in the same virtual address space as
>> the caller).
>>
>> VS2/MVS also mapped kernel subsystems into their own seperate 16mbyte
>> virtual address spaces ... so application calls to subsystems,
>> paraemters were now in different address space. For this they created
>> the Common Segment Area ("CSA") that was mapped into every 16mbyte
>> virtual adress space for subsystem API call parameters. The issue was
>> subsystem parameter list space requirement was somewhat proportional to
>> number subsystems and number of concurrently executing programs ... and
>> by the time of 370 3033 processor CSA had morphed into 5-6mbyte "Common
>> System Area" (CSA) leaving only 2-3 mbytes for programs, but was
>> threatening to become 8mbytes (leaving zero for applictions).
>>
>> This was part of POK (high-end 370s) VS2/MVS mad rush to 370/XA
>> architecture, 31-bit addressing, access register, Program Call and
>> Program Return instructions.
>>
>> Program Call was akin to kernel call ... table of all the MVS subsystems
>> and their associated address space pointers. A Program Call would select
>> the specific subsystem entry, move the caller's address space to
>> secondary and load the subsystem's address space as primary. A
>> semi-privileged subsystem would access parameter list in the caller's
>> "secondary address space" (no longer needing "CSA" space). Program
>> Return would move the caller's secondary address space pointer to
>> primary and return (task switches would have to save/restore both the
>> primary and any secondary address space pointers).

On 2025-01-24, Bill Findlay <findlaybill@blueyonder.co.uk> wrote:
> It's quite a history of bungling, isn't it?

More of a story of how a seemingly quick fix ends up snowballing until
it becomes very hard to backtrack and do the slightly larger fix that
really solves the problem.

It is hard to appreciate just how expensive memory was in the old days.
I remember I was at a systems programming class in London around 1974(?)
and in the class we had someone from an IBM group in France, who told us
his datacenter had a 360/MFT with 16 MegaBytes of main storage. Our jaws
dropped: What ever could you possibly need that much storage for?

Subject: Re: vfork history, Multics vs Unix
Posted by Anonymous on Fri, 24 Jan 2025 14:42:00 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

In article <bqskP.1852771$bYV2.353023@fx17.iad>,
Scott Lurndal <slp53@pacbell.net> wrote:
>>>> Linux has it to support legacy BSD code. Not because it's "needed"
>>>> for some abstract reason.

cross@spitfire.i.gajendra.net (Dan Cross) writes:
>>> Yup. vfork is one of those things that was obviously a mistake
>>> in hindsight, but the semantics were sufficiently different that
>>> once it was out there, you're stuck with it forever.

vfork(2) looks odd for those who grew up in the unix traditions,
where processes are cheap throwaway things. But on VMS, where processes
are big, heavy and expensive, vfork saves a lot. It is more or less what
makes POSIX portability practical. Supporting it in (u|lin)ux allows
that portabilty to be a two-way street.

In article <7XukP.83392$G93a.13310@fx05.iad>,
Scott Lurndal <slp53@pacbell.net> wrote:
>> vfork(2) was obsolete in Open Group Standard Base Specification Issue 6,
>> and has been completely removed from Issue 8.

On 2025-01-24, Dan Cross <cross@spitfire.i.gajendra.net> wrote:
> Good. Or, as the kids say, "Bye, Felicia."

"Felicia försvann ... kan någon säga hur?" (Cornelius Vreeswijk)

Subject: Re: Multics vs Unix
Posted by John Levine on Fri, 24 Jan 2025 21:33:52 GMT
View Forum Message <> Reply to Message

According to Dan Cross <cross@spitfire.i.gajendra.net>:
> It's deeper than that. Nothing prevents a Unix shared library
> from scribbling all over the memory of whatever program loads
> it; in the worst case, it may corrupt some critical data
> structure that the shell needs causing _it_ to crash, perhaps
> long after the shared image had been unloaded.

Really? The executable files including librares are mapped read-only or
copy-on-write. Static data obviously has to be copy-on-write. The mmap() call
won't let you do a shared write mapping of a file unless you have write
permission, which for shared libraries you don't. A process can only screw up
its own memory unless there are explicitly shared writable files mapped in.

The shell starts programs as separate processes partly out of tradtion, but also
because there's a lot more to the process context than what's mapped into
memory, e.g., I/O redirection and signals.

--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

Subject: Re: Multics vs Unix
Posted by scott on Fri, 24 Jan 2025 22:01:17 GMT
View Forum Message <> Reply to Message

John Levine <johnl@taugh.com> writes:
> According to Dan Cross <cross@spitfire.i.gajendra.net>:
>> It's deeper than that. Nothing prevents a Unix shared library
>> from scribbling all over the memory of whatever program loads
>> it; in the worst case, it may corrupt some critical data
>> structure that the shell needs causing _it_ to crash, perhaps
>> long after the shared image had been unloaded.
>
> Really?

It is worth noting that the korn shell, for example, supports
dynamically loaded shared objects - in such a case, a rogue
shared object can easily corrupt any writable pages, and has
access to readable pages in the process, not to mention direct
access to the mprotect system call.

  "On Linux it is always permissible to call mprotect() on

any address in a process's address space (except for the
kernel vsyscall area). In particular it can be used to
change existing code mappings to be writable."

Granted it would be rather foolish to install an untrusted
shared object, but Dan's point seems valid.

There are various sandboxing facilities available that will reduce
the exposure surface or minimize adverse consequences (e.g. containers).

> The executable files including librares are mapped read-only or
> copy-on-write. Static data obviously has to be copy-on-write. The mmap() call
> won't let you do a shared write mapping of a file unless you have write
> permission, which for shared libraries you don't. A process can only screw up
> its own memory unless there are explicitly shared writable files mapped in.

Or system V shared memory is in use (shmat, et alia).

Subject: Re: Multics vs Unix
Posted by cross on Fri, 24 Jan 2025 22:24:28 GMT

View Forum Message <> Reply to Message

In article <vn1100$1uml$1@gal.iecc.com>, John Levine <johnl@taugh.com> wrote:
> According to Dan Cross <cross@spitfire.i.gajendra.net>:
>> It's deeper than that. Nothing prevents a Unix shared library
>> from scribbling all over the memory of whatever program loads
>> it; in the worst case, it may corrupt some critical data
>> structure that the shell needs causing _it_ to crash, perhaps
>> long after the shared image had been unloaded.
>
> Really?

Of course. Note what I said: "Nothing prevents a Unix shared
library from scribbling all over the memory _of whatever program
loads it_." (Emphasis added.)

In fairness, I should have said "proces" not "program", though
that should have been clear from the surrounding context, as
surely a program can only load a shared library in a meaningful
way while executing in the context of some process. Also, I
should have been more precise and said that code or data
provided by the shared library would cause the corruption, not
the mere act of dynamic loading (presumably via `dlopen` or
similar) itself.

> The executable files including librares are mapped read-only or

> copy-on-write. Static data obviously has to be copy-on-write. The mmap() call
> won't let you do a shared write mapping of a file unless you have write
> permission, which for shared libraries you don't.

Code invoked from a shared library has full access to the
address space of the process it is loaded into; if invoked,
nothing prevents that code from corrupting memory in that
process, either deliberately or through error. Similarly, the
shared library itself may contain garbage data (unterminated
string values, uninitialized integers or pointers, the linked
list with the `next` pointer that points to some random location
in the text segment; whatever), that it provides to the larger
program that causes it to crash.

For that matter, nothing prevents that code from calling `exit`
or munging the virtual address (via `mmap` or `munmap`) in some
way that would similarly corrupt the process, or any number of
other weird things. It could also open or close files
arbitrarily in ways the user doesn't appreciate.

The point was that the shared library can easily crash the
program that loads it, or cause other undesireous behavior.

Indeed, this is true of any library, and we can see this
trivially if, say, we get a coredump when we try to pass a bad
pointer value as the destination argument to `memcpy`. But
whereas, if a program is written against a known set of
libraries, tested against them, and so on, we may have some
confidence of that program's correct operation. Once we start
dynamically loading arbitrary code at runtime, however, all
bets are off and we can no longer reasonably afford any optimism
that the program will execute properly.

> A process can only screw up
> its own memory unless there are explicitly shared writable files mapped in.

Yes. But if that process is your shell, you're likely to have a
bad time.

The original context was a poster asking why Unix shells haven't
been implemented in a VMS/TOPS-20/Multics style, where
"commands" are invoked by mapping some some shared object that
implements the command into the address space of the shell
process and invoking it by calling its `main` or whatever; that
is, instead of `fork` and `exec`, map and call.

In fact, this was sort of how the original PDP-7 version of Unix
worked; the shell directly loaded each program and then jumped

to its entry point, but only after fork'ing itself (with the
parent waiting for the child to exit).

> The shell starts programs as separate processes partly out of tradtion, but also
> because there's a lot more to the process context than what's mapped into
> memory, e.g., I/O redirection and signals.

IO redirection and similar things _could_ theoretically be
worked around, but only imperfectly: `posix_spawn` exists and
has to handle many of the same sorts of issues, for instance.
but yeah: it gets gnarly really fast.

A much harder thing to emulate in this model would be pipelines.

 - Dan C.

Subject: Re: vfork history, Multics vs Unix
Posted by cross on Fri, 24 Jan 2025 22:25:41 GMT
View Forum Message <> Reply to Message

In article <slrnvp79lo.3tdt4.lars@cleo.beagle-ears.com>,
Lars Poulsen <lars@cleo.beagle-ears.com> wrote:
> In article <bqskP.1852771$bYV2.353023@fx17.iad>,
>   Scott Lurndal <slp53@pacbell.net> wrote:
>>>> >Linux has it to support legacy BSD code. Not because it's "needed"
>>>> >for some abstract reason.
>
> cross@spitfire.i.gajendra.net (Dan Cross) writes:
>>>> Yup. vfork is one of those things that was obviously a mistake
>>>> in hindsight, but the semantics were sufficiently different that
>>>> once it was out there, you're stuck with it forever.
>
> vfork(2) looks odd for those who grew up in the unix traditions,
> where processes are cheap throwaway things. But on VMS, where processes
> are big, heavy and expensive, vfork saves a lot. It is more or less what
> makes POSIX portability practical. Supporting it in (u|lin)ux allows
> that portabilty to be a two-way street.

Hmm. I think `posix_spawn` fits the VMS model much better than
`vfork`.

 - Dan C.

Subject: Re: Multics vs Unix
Posted by scott on Fri, 24 Jan 2025 22:27:27 GMT

View Forum Message <> Reply to Message

cross@spitfire.i.gajendra.net (Dan Cross) writes:
> In article <vn1100$1uml$1@gal.iecc.com>, John Levine <johnl@taugh.com> wrote:

>
> The original context was a poster asking why Unix shells haven't
> been implemented in a VMS/TOPS-20/Multics style, where
> "commands" are invoked by mapping some some shared object that
> implements the command into the address space of the shell
> process and invoking it by calling its `main` or whatever; that
> is, instead of `fork` and `exec`, map and call.

The Korn shell does support this - extending the built-in command
set by loading user provided shared objects at runtime (e.g. dlopen()).

Subject: Re: Multics vs Unix
Posted by ted@loft.tnolan.com (on Fri, 24 Jan 2025 22:30:12 GMT
View Forum Message <> Reply to Message

In article <jtUkP.928265$2xE6.227106@fx18.iad>,
Scott Lurndal <slp53@pacbell.net> wrote:
> cross@spitfire.i.gajendra.net (Dan Cross) writes:
>> In article <vn1100$1uml$1@gal.iecc.com>, John Levine <johnl@taugh.com> wrote:
>
>>
>> The original context was a poster asking why Unix shells haven't
>> been implemented in a VMS/TOPS-20/Multics style, where
>> "commands" are invoked by mapping some some shared object that
>> implements the command into the address space of the shell
>> process and invoking it by calling its `main` or whatever; that
>> is, instead of `fork` and `exec`, map and call.
>
> The Korn shell does support this - extending the built-in command
> set by loading user provided shared objects at runtime (e.g. dlopen()).

It is one of the main use cases for Tcl's tclsh, though I suspect few
people use that as their login shell (though you could).
--
columbiaclosings.com
What's not in Columbia anymore..

Subject: Re: Multics vs Unix
Posted by cross on Fri, 24 Jan 2025 23:00:59 GMT
View Forum Message <> Reply to Message

In article <jtUkP.928265$2xE6.227106@fx18.iad>,
Scott Lurndal <slp53@pacbell.net> wrote:
> cross@spitfire.i.gajendra.net (Dan Cross) writes:
>> In article <vn1100$1uml$1@gal.iecc.com>, John Levine <johnl@taugh.com> wrote:
>
>>
>> The original context was a poster asking why Unix shells haven't
>> been implemented in a VMS/TOPS-20/Multics style, where
>> "commands" are invoked by mapping some some shared object that
>> implements the command into the address space of the shell
>> process and invoking it by calling its `main` or whatever; that
>> is, instead of `fork` and `exec`, map and call.
>
> The Korn shell does support this - extending the built-in command
> set by loading user provided shared objects at runtime (e.g. dlopen()).

Cool. Presumably this is relatively rare, though, and not meant
as a subtitute for the normal path of program execution; I can
sorta see the use case for extending the shell's set of builtins
and of course other languages that allow dynamically loadable
modules do similar things (Python; Tcl was mentioned; Perl; Ruby
et al). A well-defined FFI is useful.

 - Dan C.

Subject: Re: Multics vs Unix
Posted by scott on Fri, 24 Jan 2025 23:47:37 GMT
View Forum Message <> Reply to Message

cross@spitfire.i.gajendra.net (Dan Cross) writes:
> In article <jtUkP.928265$2xE6.227106@fx18.iad>,
> Scott Lurndal <slp53@pacbell.net> wrote:
>> cross@spitfire.i.gajendra.net (Dan Cross) writes:
>>> In article <vn1100$1uml$1@gal.iecc.com>, John Levine <johnl@taugh.com> wrote:
>>
>>>
>>> The original context was a poster asking why Unix shells haven't
>>> been implemented in a VMS/TOPS-20/Multics style, where
>>> "commands" are invoked by mapping some some shared object that
>>> implements the command into the address space of the shell
>>> process and invoking it by calling its `main` or whatever; that
>>> is, instead of `fork` and `exec`, map and call.
>>
>> The Korn shell does support this - extending the built-in command
>> set by loading user provided shared objects at runtime (e.g. dlopen()).
>
> Cool. Presumably this is relatively rare, though, and not meant

> as a subtitute for the normal path of program execution;

I expect so.   I've not found a use for it myself.   I think
it would be most useful exporting some system specific
functionality as native compiled functions for performance
in shell scripts.

I do use the dynamically loaded shell functions ($FPATH)
feature of ksh93 extensively.

e.g.

```
$ typeset -fu sdkrun
```

when the user subsequently enters 'sdkrun', it will check all the paths
listed in $FPATH and load and execute the named function in that file;
it will remain loaded and used on subsequent references until
unset -f sdkrun, after which it will be reloaded automatically
when referenced.

Subject: Re: Multics vs Unix
Posted by Charlie Gibbs on Fri, 24 Jan 2025 23:57:12 GMT
View Forum Message <> Reply to Message

On 2025-01-24, Scott Lurndal <scott@slp53.sl.home> wrote:

```
>  Lynn Wheeler <lynn@garlic.com> writes:
>
>>  Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>>
>>>  So each (physical?) page belonged to exactly one user process (or the
>>>  kernel)?
>>>
>>>  How did this handle shared sections accessible by more than one process?
>>
>>  PSW key=0 (nominally kernel/supervisor) allowed access to all pages,
>>  nonzero PSW only could store (and/or fetch if the fetch protect feature
>>  was installed and fetch set) to pages with matching storage protect key.
>
>    rodtsasdt 111111report*
>
>  The book _The Adolescence of P-1_ describes a student at U Waterloo
>  finding a timing window that allowed him to set the PSW key to zero,
>  thus allowing full access to the kernel/supervisor.   Which he used
>  to create a worm which spread via teleprocessing ports and eventually
>  became a self-aware AI.   The book was written in 1977.   It's still
>  a great read.
```

>
> https://en.wikipedia.org/wiki/The_Adolescence_of_P-1

I probably still have a copy it around somewhere.

CALL GREGORY

```
--
/~\  Charlie Gibbs                  |  Growth for the sake of
\ /  <cgibbs@kltpzyxm.invalid>      |  growth is the ideology
 X   I'm really at ac.dekanfrus     |  of the cancer cell.
/ \  if you read it the right way.  |    -- Edward Abbey
```

Subject: Re: Multics vs Unix
Posted by Charlie Gibbs on Fri, 24 Jan 2025 23:57:13 GMT
View Forum Message <> Reply to Message

On 2025-01-24, Lars Poulsen <lars@cleo.beagle-ears.com> wrote:

> On 23 Jan 2025, Lynn Wheeler wrote

<history of IBM 360 development>

> On 2025-01-24, Bill Findlay <findlaybill@blueyonder.co.uk> wrote:
>
>> It's quite a history of bungling, isn't it?

It could more charitably be called a learning process - at least
if management and time pressures allow you to learn.

> More of a story of how a seemingly quick fix ends up snowballing until
> it becomes very hard to backtrack and do the slightly larger fix that
> really solves the problem.

"If at first you don't succeed, you might as well forget it."

> It is hard to appreciate just how expensive memory was in the old days.
> I remember I was at a systems programming class in London around 1974(?)
> and in the class we had someone from an IBM group in France, who told us
> his datacenter had a 360/MFT with 16 MegaBytes of main storage. Our jaws
> dropped: What ever could you possibly need that much storage for?

Wow. I never heard of a 360 outfitted with a full 16MB. Who could
afford it?

I once read in a trade rag, in the early days of the 370, of how IBM
rocked the industry by slashing the price of one megabyte of memory

from $75,000 to a mere $15,000. Then I look at that thumb drive
in my pocket...

In 1970 I was working in a small shop on a machine with 16K of memory.
But that didn't stop us from running accounts receivables, general
ledgers, payrolls, etc. for small companies all over town who couldn't
afford a computer at all. When we upgraded the machine to 32K, we too
had to figure out what to do with all that space. (We succeeded.)

```
--
/~\  Charlie Gibbs                  |  Growth for the sake of
\ /  <cgibbs@kltpzyxm.invalid>      |  growth is the ideology
 X   I'm really at ac.dekanfrus     |  of the cancer cell.
/ \  if you read it the right way.  |    -- Edward Abbey
```

Subject: Re: Multics vs Unix
Posted by Anonymous on Sat, 25 Jan 2025 00:16:35 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 23 Jan 2025 21:41:05 -1000, Lynn Wheeler wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>>
>> So only shared code, no shared data? Was there any copy-on-write?
>
> nothing precluded shared data ...

But could more than one process write to the same memory segment?

Subject: Re: Multics vs Unix
Posted by Anonymous on Sat, 25 Jan 2025 00:21:07 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Fri, 24 Jan 2025 14:32:18 -0000 (UTC), Lars Poulsen wrote:

> It is hard to appreciate just how expensive memory was in the old days.

But note that the trend of falling memory prices was already becoming
clear by the 1970s, if not earlier. The earliest batch systems only kept
one program in memory at one time, and swapped it out for another one when
it went into any kind of I/O wait, to keep the CPU busy; later on, when

memory became large enough (and cheap enough), it made sense to keep
multiple programs resident at once (“multiprogramming”, this was called),
to allow the scheduling switches to happen more quickly.

Subject: Re: Multics vs Unix
Posted by Anonymous on Sat, 25 Jan 2025 00:24:55 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Fri, 24 Jan 2025 12:02:39 +0000, Bill Findlay wrote:

> It's quite a history of bungling, isn't it?

Large corporates did, and do, tend to become unwieldy in their development
processes. IBM had a long history of doing innovative research (and
patenting the results); but if you looked at their actual product line,
very little of this innovation actually seemed to make it into that.

One example I recall is SNA, their “Systems Network Architecture”. For a
long time this was not what we would understand as a “network” at all: it
was primarily a way for large, expensive central machines to control
remote, cheaper machines at branch offices.

IBM didn’t discover peer-to-peer networking until the 1980s, years after
other companies were already doing it routinely.

Subject: Re: Multics vs Unix
Posted by Anonymous on Sat, 25 Jan 2025 00:27:07 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Fri, 24 Jan 2025 21:33:52 -0000 (UTC), John Levine wrote:

> The executable files including librares are mapped read-only or
> copy-on-write.

But once some external code has been loaded into a process address space,
it is quite free to write into any writeable area in that address space,
whether it came from the same library or not.

> The shell starts programs as separate processes partly out of
> tradtion ...

It wasn’t “tradition” back when Unix started doing it.

Subject: Re: vfork history, Multics vs Unix
Posted by Anonymous on Sat, 25 Jan 2025 00:35:08 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Fri, 24 Jan 2025 14:42:00 -0000 (UTC), Lars Poulsen wrote:

> vfork(2) looks odd for those who grew up in the unix traditions,

But it was added by BSD, very much as part of their Unix tradition. It was defined in POSIX (at least for a while), and Linux still supports it. The man page <https://manpages.debian.org/vfork(2)> says:

> Some consider the semantics of vfork() to be an architectural blemish, and the 4.2BSD man page stated: “This system call will be eliminated when proper system sharing mechanisms are implemented. Users should not depend on the memory sharing semantics of vfork as it will, in that case, be made synonymous to fork.” However, even though modern memory management hardware has decreased the performance difference between fork(2) and vfork(), there are various reasons why Linux and other systems have retained vfork():
>
> • Some performance-critical applications require the small performance advantage conferred by vfork().
>
> • vfork() can be implemented on systems that lack a memory-management unit (MMU), but fork(2) can't be implemented on such systems. (POSIX.1-2008 removed vfork() from the standard; the POSIX rationale for the posix_spawn(3) function
>
> lent to fork(2)+exec(3), is designed to be implementable on systems that lack an MMU.)
>
> • On systems where memory is constrained, vfork() avoids the need to temporarily commit memory (see the description of /proc/sys/vm/overcommit_memory in proc(5)) in order to execute a new program. (This can be especially beneficial where a large parent process wishes to execute a small helper program in a child process.) By contrast, using fork(2) in this scenario requires either committing an amount of memory equal to the size of the parent process (if strict overcommitting is in force) or overcommitting memory with the risk that a process is terminated by the out-of-memory (OOM) killer.

Subject: Re: vfork history, Multics vs Unix
Posted by ted@loft.tnolan.com (on Sat, 25 Jan 2025 01:05:43 GMT
View Forum Message <> Reply to Message

In article <vn1bjs$2fink$5@dont-email.me>,
Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Fri, 24 Jan 2025 14:42:00 -0000 (UTC), Lars Poulsen wrote:
>
>> vfork(2) looks odd for those who grew up in the unix traditions,
>
> But it was added by BSD, very much as part of their Unix tradition. It was
> defined in POSIX (at least for a while), and Linux still supports it. The
> man page <https://manpages.debian.org/vfork(2)> says:
>
> Some consider the semantics of vfork() to be an architectural
> blemish, and the 4.2BSD man page stated: “This system call will be
> eliminated when proper system sharing mechanisms are implemented.
> Users should not depend on the memory sharing semantics of vfork
> as it will, in that case, be made synonymous to fork.” However,
> even though modern memory management hardware has decreased the
> performance difference between fork(2) and vfork(), there are
> various reasons why Linux and other systems have retained vfork():
>
> • Some performance-critical applications require the small
> performance advantage conferred by vfork().
>
> • vfork() can be implemented on systems that lack a
> memory-management unit (MMU), but fork(2) can't be implemented
> on such systems. (POSIX.1-2008 removed vfork() from the
> standard; the POSIX rationale for the posix_spawn(3) function

> lent to fork(2)+exec(3), is designed to be implementable on
> systems that lack an MMU.)
>

I don't think that's true, is it? Venix & PC/IX ran on the 8086 which
had no MMU and had fork().
--
columbiaclosings.com
What's not in Columbia anymore..

Subject: Re: vfork history, Multics vs Unix
Posted by Anonymous on Sat, 25 Jan 2025 08:06:47 GMT
View Forum Message <> Reply to Message

Originally posted by: Bob Eager

On Sat, 25 Jan 2025 01:05:43 +0000, Ted Nolan <tednolan> wrote:

```
>  In article <vn1bjs$2fink$5@dont-email.me>,
>  Lawrence D'Oliveiro  <ldo@nz.invalid> wrote:
>> On Fri, 24 Jan 2025 14:42:00 -0000 (UTC), Lars Poulsen wrote:
>>
>>>  vfork(2) looks odd for those who grew up in the unix traditions,
>>
>> But it was added by BSD, very much as part of their Unix tradition. It
>> was defined in POSIX (at least for a while), and Linux still supports
>> it. The man page <https://manpages.debian.org/vfork(2)> says:
>>
>>     Some consider the semantics of vfork() to be an architectural
>>     blemish, and the 4.2BSD man page stated: "This system call will be
>>     eliminated when proper system sharing mechanisms are implemented.
>>     Users should not depend on the memory sharing semantics of vfork as
>>     it will, in that case, be made synonymous to fork." However, even
>>     though modern memory management hardware has decreased the
>>     performance difference between fork(2) and vfork(), there are
>>     various reasons why Linux and other systems have retained vfork():
>>
>>     • Some performance-critical applications require the small
>>       performance advantage conferred by vfork().
>>
>>     • vfork() can be implemented on systems that lack a
>>       memory-management unit (MMU), but fork(2) can't be implemented on
>>       such systems. (POSIX.1-2008 removed vfork() from the standard; the
>>       POSIX rationale for the posix_spawn(3) function notes that that

>>       lent to fork(2)+exec(3), is designed to be implementable on
>>       systems that lack an MMU.)
>>
>>
>  I don't think that's true, is it?  Venix & PC/IX ran on the 8086 which
>  had no MMU and had fork().
```

And so did Mini-UNIX, back around 1978. (I am NOT referring to MINIX, but the Bell Labs system)

--
Using UNIX since v6 (1975)...

Use the BIG mirror service in the UK:
 http://www.mirrorservice.org

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Sat, 25 Jan 2025 09:43:57 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:
> Large corporates did, and do, tend to become unwieldy in their development
> processes. IBM had a long history of doing innovative research (and
> patenting the results); but if you looked at their actual product line,
> very little of this innovation actually seemed to make it into that.
>
> One example I recall is SNA, their “Systems Network Architecture”. For a
> long time this was not what we would understand as a “network” at all: it
> was primarily a way for large, expensive central machines to control
> remote, cheaper machines at branch offices.
>
> IBM didn’t discover peer-to-peer networking until the 1980s, years after
> other companies were already doing it routinely.

In the 70s about same time SNA appeared, my wife was co-author of AWP39, "Peer-to=Peer Networking" architecture ... peer-to-peer qualification necessary because the communication group had co-opted "network" .... joke was that SNA was "not a System", "not a Network", and "not an Architecture".

My wife was then con'ed into going to POK (IBM high-end mainframe) responsible for "loosely-coupled" (mainframe for "cluster") shared DASD architecture. She didn't remain long because 1) periodic battles with the SNA/VTAM forces trying to force her into using SNA/VTAM for loosely-coupled operation and 2) little uptake, until much later with SYSPLEX and Parallel SYSPLEX
https://en.wikipedia.org/wiki/IBM_Parallel_Sysplex
except for IMS hot-standby.

Co-worker at science center was responsible for the CP67-based (precursor to VM370) wide-area network, that morphs into the corproate internal network (larger than arpanet/internet from the beginning until sometime mid/late 80s, about the time the SNA-org forced the internal network to be converted to SNA/VTAM). Technology also used for the corporate sponsored univ BITNET (also for a time larger than arpanet/internet): https://en.wikipedia.org/wiki/BITNET

Account by one of the inventors of GML (in 1969) at the science center:
https://web.archive.org/web/20230402212558/http://www.sgmlso urce.com/history/jasis.htm
Actually, the law office application was the original motivation for the project, something I was allowed to do part-time because of my knowledge of the user requirements. My real job was to encourage the staffs of the various scientific centers to make use of the CP-67-based Wide Area Network that was centered in Cambridge.

.... trivia: a decade later GML morphs into ISO standard SGML and after
another decade morphs into HTML at CERN and 1st webserver in the
use is CERN sister location, Stanford SLAC (on their VM370 sstem)
https://ahro.slac.stanford.edu/wwwslac-exhibit
https://ahro.slac.stanford.edu/wwwslac-exhibit/early-web-chr onology-and-documents-1991-1994

Edson (passed Aug2020)
https://en.wikipedia.org/wiki/Edson_Hendricks
In June 1975, MIT Professor Jerry Saltzer accompanied Hendricks to
DARPA, where Hendricks described his innovations to the principal
scientist, Dr. Vinton Cerf. Later that year in September 15-19 of 75,
Cerf and Hendricks were the only two delegates from the United States,
to attend a workshop on Data Communications at the International
Institute for Applied Systems Analysis, 2361 Laxenburg Austria where
again, Hendricks spoke publicly about his innovative design which paved
the way to the Internet as we know it today.

We then transfer out to San Jose Research and in early 80s, I get HSDT,
T1 and faster computer links, some amount of conflict with SNA forces
(note in 60s, IBM had 2701 telecommunication controller that supported
T1 links, however IBM's move to SNA in the mid-70s and associated issues
seem to have capped links at 56kbits/sec).

trivia: at the time of arpanet/internet cut-over to internetworking
protocol on 1Jan1983, there were 100 IMPs and approx 255 hosts ... at a
time when the internal network was rapidly approach 1000 hosts all over
the world (approval of IMPs somewhat held back arpanet growth; for
internal network growth it was corporate requirement that all links be
encrypted ... which could be real problem with various country
gov. agencies, especially when links cross national
boundaries). Archived post with list of world-wide corporate locations
that got one or more hew host networking connections during 1983:
https://www.garlic.com/~lynn/2006k.html#8

For HSDT, was involved in working with the NSF director and was suppose to
get $20M to interconnect the NSF Supercomputer centers. Then congress
cuts the budget, some other things happen and eventually an RFP is
released (in part based on what we already had running). NSF 28Mar1986
Preliminary Announcement:
https://www.garlic.com/~lynn/2002k.html#12
The OASC has initiated three programs: The Supercomputer Centers Program
to provide Supercomputer cycles; the New Technologies Program to foster
new supercomputer software and hardware developments; and the Networking
Program to build a National Supercomputer Access Network - NSFnet.

IBM internal politics was not allowing us to bid (being blamed for
online computer conferencing inside IBM likely contributed). The NSF
director tried to help by writing the company a letter (3Apr1986, NSF

Director to IBM Chief Scientist and IBM Senior VP and director of
Research, copying IBM CEO) with support from other gov. agencies ... but
that just made the internal politics worse (as did claims that what we
already had operational was at least 5yrs ahead of the winning bid), as
regional networks connect in, it becomes the NSFNET backbone, precursor
to modern internet.

One of HSDT first long-haul T1 links was between the IBM Los Gatos lab
(on the west coast) and Clementi's
https://en.wikipedia.org/wiki/Enrico_Clementi
E&S lab in Kingston, NY, that had lots of floating point systems boxes
that included 40mbyte/sec disk arrays.
https://en.wikipedia.org/wiki/Floating_Point_Systems
Cornell University, led by physicist Kenneth G. Wilson, made a
supercomputer proposal to NSF with IBM to produce a processor array of
FPS boxes attached to an IBM mainframe with the name lCAP.

SJMerc article about Edson, "IBM'S MISSED OPPORTUNITY WITH THE INTERNET"
(gone behind paywall but lives free at wayback machine),
https://web.archive.org/web/20000124004147/http://www1.sjmer
cury.com/svtech/columns/gillmor/docs/dg092499.htm
Also from wayback machine, some additional (IBM missed) references from
Ed's website
https://web.archive.org/web/20000115185349/http://www.edh.ne t/bungle.htm

For awhile I reported to same executive as the person behind AWP164
(which becomes APPN), needling that he should come work on real
networking ... becauses the SNA forces would never appreciated what he
was doing ... the SNA forces veto the announcement of APPN ... and the
APPN announcement letter was carefully rewritten to not imply and
relationship between APPN and SNA.

SNA forces had been fiercely fighting off client/server and distributed
computing and trying to block the announcement of mainframe TCP/IP. When
that failed, they change their tactic and claim that since they have
corporate strategic responsibility for everything that crosses
datacenter walls, it has to be released through them. What ships get
44kbyte/sec aggregate using nearly whole 3090 processor. I then do the
support for RFC1044 and in some turning tests at Cray Research between a
Cray and a 4341, get sustained 4341 channel I/O throughput, using only a
modest amount of 4341 processor (something like 500 times improvement in
bytes moved per instruction executed).

In 1988 we get last product we did at IBM, HA/CMP.
https://en.wikipedia.org/wiki/IBM_High_Availability_Cluster_ Multiprocessing
It starts out HA/6000 for the NYTimes to port their newspaper system
(ATEX) off DEC VAXCluster to RS/6000. I rename it HA/CMP when I start
doing technical/scientific cluster scale-up with national labs and

commercial cluster scale-up with RDBMS vendors that have VAXCluster
support in the same source base with UNIX (Oracle, Sybase, Ingres,
Informix).

Early Jan1992, have meeting with Oracle CEO where IBM/AWD Hester tells
Ellison that by mid92 there would be 16-system clusters and by ye92,
128-system clusters. Then late Jan1992, cluster scale-up is transferred
for announce as IBM Supercomputer (for technical/scientific *ONLY*) and
we are told we can't work on anything with more than four systems (we
leave IBM a few months later). Note: IBM mainframe DB2 had been
complaining if we were allowed to go ahead, it would be years ahead of
them.

trivia: when 1st transferred to IBM SJR, I did some work with Jim Gray
and Vera Watson on the original SQL/relational, System/R ... and while
corporation was preoccupied with the next great DBMS ("EAGLE"), we were
able to do tech transfer to Endicott for release as SQL/DS. Then when
"EAGLE" implodes there was a request for how fast could System/R be
ported to MVS ... which is eventually released as "DB2" (originally for
decision support only).

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: vfork history, Multics vs Unix
Posted by cross on Sat, 25 Jan 2025 15:19:04 GMT
View Forum Message <> Reply to Message

In article <lvjkcnF4lq8U1@mid.individual.net>,
Bob Eager <news0009@eager.cx> wrote:
> On Sat, 25 Jan 2025 01:05:43 +0000, Ted Nolan <tednolan> wrote:
>
>> In article <vn1bjs$2fink$5@dont-email.me>,
>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>> On Fri, 24 Jan 2025 14:42:00 -0000 (UTC), Lars Poulsen wrote:
>>>
>>>> vfork(2) looks odd for those who grew up in the unix traditions,
>>>
>>> But it was added by BSD, very much as part of their Unix tradition. It
>>> was defined in POSIX (at least for a while), and Linux still supports
>>> it. The man page <https://manpages.debian.org/vfork(2)> says:
>>>
>>>   Some consider the semantics of vfork() to be an architectural
>>>   blemish, and the 4.2BSD man page stated: “This system call will be
>>>   eliminated when proper system sharing mechanisms are implemented.
>>>   Users should not depend on the memory sharing semantics of vfork as
>>>   it will, in that case, be made synonymous to fork.” However, even

>>>   though modern memory management hardware has decreased the
>>>   performance difference between fork(2) and vfork(), there are
>>>   various reasons why Linux and other systems have retained vfork():
>>>
>>>   • Some performance-critical applications require the small
>>>   performance advantage conferred by vfork().
>>>
>>>   • vfork() can be implemented on systems that lack a
>>>   memory-management unit (MMU), but fork(2) can't be implemented on
>>>   such systems. (POSIX.1-2008 removed vfork() from the standard; the
>>>   POSIX rationale for the posix_spawn(3) function notes that that

>>>   lent to fork(2)+exec(3), is designed to be implementable on
>>>   systems that lack an MMU.)
>>
>> I don't think that's true, is it? Venix & PC/IX ran on the 8086 which
>> had no MMU and had fork().
>
> And so did Mini-UNIX, back around 1978. (I am NOT referring to MINIX, but
> the Bell Labs system)

Indeed. But speaking of Minix, early Minix didn't use the MMU,
either. I understand this is one of the reasons Torvalds was
dissatisfied with it, and started working on Linux.

You can do a Unix-like fork without an MMU if you're willing to
live with some constraints; for instance, only one process can
be resident in memory at a time, perhaps.

 - Dan C.

Subject: Re: vfork history, Multics vs Unix
Posted by ted@loft.tnolan.com (on Sat, 25 Jan 2025 17:21:32 GMT
View Forum Message <> Reply to Message

In article <vn2vd8$q4k$1@reader2.panix.com>,
Dan Cross <cross@spitfire.i.gajendra.net> wrote:
> In article <lvjkcnF4lq8U1@mid.individual.net>,
> Bob Eager <news0009@eager.cx> wrote:
>> On Sat, 25 Jan 2025 01:05:43 +0000, Ted Nolan <tednolan> wrote:
>>
>>> In article <vn1bjs$2fink$5@dont-email.me>,
>>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>>> On Fri, 24 Jan 2025 14:42:00 -0000 (UTC), Lars Poulsen wrote:
>>>>
>>>> > vfork(2) looks odd for those who grew up in the unix traditions,
>>>>

>>>> But it was added by BSD, very much as part of their Unix tradition. It
>>>> was defined in POSIX (at least for a while), and Linux still supports
>>>> it. The man page <https://manpages.debian.org/vfork(2)> says:
>>>>
>>>>   Some consider the semantics of vfork() to be an architectural
>>>>   blemish, and the 4.2BSD man page stated: “This system call will be
>>>>   eliminated when proper system sharing mechanisms are implemented.
>>>>   Users should not depend on the memory sharing semantics of vfork as
>>>>   it will, in that case, be made synonymous to fork.” However, even
>>>>   though modern memory management hardware has decreased the
>>>>   performance difference between fork(2) and vfork(), there are
>>>>   various reasons why Linux and other systems have retained vfork():
>>>>
>>>>   • Some performance-critical applications require the small
>>>>   performance advantage conferred by vfork().
>>>>
>>>>   • vfork() can be implemented on systems that lack a
>>>>   memory-management unit (MMU), but fork(2) can't be implemented on
>>>>   such systems. (POSIX.1-2008 removed vfork() from the standard; the
>>>>   POSIX rationale for the posix_spawn(3) function notes that that

>>>>   lent to fork(2)+exec(3), is designed to be implementable on
>>>>   systems that lack an MMU.)
>>>
>>> I don't think that's true, is it? Venix & PC/IX ran on the 8086 which
>>> had no MMU and had fork().
>>
>> And so did Mini-UNIX, back around 1978. (I am NOT referring to MINIX, but
>> the Bell Labs system)
>
> Indeed. But speaking of Minix, early Minix didn't use the MMU,
> either. I understand this is one of the reasons Torvalds was
> dissatisfied with it, and started working on Linux.
>
> You can do a Unix-like fork without an MMU if you're willing to
> live with some constraints; for instance, only one process can
> be resident in memory at a time, perhaps.
>

I never used PC/IX, but my memory from reading about it in the press
and here on Usenet at the time is that it definitely was a full,
AT&T SYSIII Unix implementation, with more than one process resident
at a time.

I think the way they tried to finesse memory protection was that
the C compiler would not emit code for changing the base register,
so even if a program went wild, it would be within one 64k segment
only. Of course this was not foolproof and would be easy to defeat,

but the idea was, who would try given the platform.
--
columbiaclosings.com
What's not in Columbia anymore..

Subject: Re: vfork history, Multics vs Unix
Posted by Anonymous on Sat, 25 Jan 2025 19:00:45 GMT
View Forum Message <> Reply to Message

Originally posted by: Bob Eager

On Sat, 25 Jan 2025 15:19:04 +0000, Dan Cross wrote:

```
>  In article <lvjkcnF4lq8U1@mid.individual.net>,
>  Bob Eager  <news0009@eager.cx> wrote:
>> On Sat, 25 Jan 2025 01:05:43 +0000, Ted Nolan <tednolan> wrote:
>>
>>>  In article <vn1bjs$2fink$5@dont-email.me>,
>>>  Lawrence D'Oliveiro  <ldo@nz.invalid> wrote:
>>>> On Fri, 24 Jan 2025 14:42:00 -0000 (UTC), Lars Poulsen wrote:
>>>>
>>>> > vfork(2) looks odd for those who grew up in the unix traditions,
>>>>
>>>> But it was added by BSD, very much as part of their Unix tradition. It
>>>> was defined in POSIX (at least for a while), and Linux still supports
>>>> it. The man page <https://manpages.debian.org/vfork(2)> says:
>>>>
>>>>     Some consider the semantics of vfork() to be an architectural
>>>>     blemish, and the 4.2BSD man page stated: “This system call will be
>>>>     eliminated when proper system sharing mechanisms are implemented.
>>>>     Users should not depend on the memory sharing semantics of vfork
>>>>     as it will, in that case, be made synonymous to fork.” However,
>>>>     even though modern memory management hardware has decreased the
>>>>     performance difference between fork(2) and vfork(), there are
>>>>     various reasons why Linux and other systems have retained vfork():
>>>>
>>>>     • Some performance-critical applications require the small
>>>>       performance advantage conferred by vfork().
>>>>
>>>>     • vfork() can be implemented on systems that lack a
>>>>       memory-management unit (MMU), but fork(2) can't be implemented
>>>>       on such systems. (POSIX.1-2008 removed vfork() from the
>>>>       standard; the POSIX rationale for the posix_spawn(3) function

>>>>       lent to fork(2)+exec(3), is designed to be implementable on
>>>>       systems that lack an MMU.)
>>>
```

>>> I don't think that's true, is it? Venix & PC/IX ran on the 8086 which
>>> had no MMU and had fork().
>>
>> And so did Mini-UNIX, back around 1978. (I am NOT referring to MINIX,
>> but the Bell Labs system)
>
> Indeed. But speaking of Minix, early Minix didn't use the MMU,
> either. I understand this is one of the reasons Torvalds was
> dissatisfied with it, and started working on Linux.
>
> You can do a Unix-like fork without an MMU if you're willing to live
> with some constraints; for instance, only one process can be resident in
> memory at a time, perhaps.

Which is what Mini-UNIX did. I have complete installation details if
anyone wants to try it!

--
Using UNIX since v6 (1975)...

Use the BIG mirror service in the UK:
 http://www.mirrorservice.org

Subject: Re: PC/IX, was vfork history, Multics vs Unix
Posted by John Levine on Sat, 25 Jan 2025 20:21:18 GMT
View Forum Message <> Reply to Message

It appears that Ted Nolan <tednolan> <tednolan> said:
> I never used PC/IX, but my memory from reading about it in the press
> and here on Usenet at the time is that it definitely was a full,
> AT&T SYSIII Unix implementation, with more than one process resident
> at a time.

I worked on it. It was full System III, similar to what you'd get on a PDP-11.

> I think the way they tried to finesse memory protection was that
> the C compiler would not emit code for changing the base register,

Well, segment register, not base register, but that's right, programs ran in
small model, one segment for code, another segment for stack and data, and the C
compiler never emitted code to mess with the segment registers. It worked quite
well. We got a bug report for something that only failed if the system had been
up continuously for over a year.

When we first shipped it a lot of people asked how to make bigger programs

and the answer was you couldn't other than by the usual Unix trick of running
multiple programs that talked through pipes. In retrospect, even though the
contract didn't call for it we should have added something to let you allocate
muliple data segments and copy stuff in and out. It wouldn't have been hard and
it would have let more people port programs to it.

--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: Multics vs Unix
Posted by Anonymous on Sat, 25 Jan 2025 20:40:56 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>> IBM didn’t discover peer-to-peer networking until the 1980s, years after
>> other companies were already doing it routinely.

It was not that they did not discover it, but that it was ideologically
against "the IBM way". IBM was solidly backing strictly top-down
hierachies, thus there had to be ONE TRUE CENTER of any network.
Aligned with the IBM sales philosophy of marketing to the C-suite, not
to the actual departments needing the resources and having hands-on
understanding of the needs.

On 2025-01-25, Lynn Wheeler <lynn@garlic.com> wrote:
> In the 70s about same time SNA appeared, my wife was co-author of AWP39,
> "Peer-to=Peer Networking" architecture ... peer-to-peer qualification
> necessary because the communication group had co-opted "network"
> ... joke was that SNA was "not a System", "not a Network", and "not an
> Architecture".
>
> My wife was then con'ed into going to POK (IBM high-end mainframe)
> responsible for "loosely-coupled" (mainframe for "cluster") shared DASD
> architecture. She didn't remain long because 1) periodic battles with
> the SNA/VTAM forces trying to force her into using SNA/VTAM for
> loosely-coupled operation and 2) little uptake, until much later with
> SYSPLEX and Parallel SYSPLEX
> https://en.wikipedia.org/wiki/IBM_Parallel_Sysplex
> except for IMS hot-standby.

Of course, a tight cluster with survivability cannot have a strict
hierarchy, since the top position would have to be renegotiated if

the "master" node failed.

> Co-worker at science center was responsible for the CP67-based
> (precursor to VM370) wide-area network, that morphs into the corproate
> internal network (larger than arpanet/internet from the beginning until
> sometime mid/late 80s, about the time the SNA-org forced the internal
> network to be converted to SNA/VTAM). Technology also used for the
> corporate sponsored univ BITNET (also for a time larger than
> arpanet/internet): https://en.wikipedia.org/wiki/BITNET

The wide-area network was in effect an internal application, not driven
by the C-suite of IBM, but by the needs of technical departments needing
to link up with their peers across division boundaries.

> Account by one of the inventors of GML (in 1969) at the science center:
> https://web.archive.org/web/20230402212558/http://www.sgmlso urce.com/history/jasis.htm
> Actually, the law office application was the original motivation for the
> project, something I was allowed to do part-time because of my knowledge
> of the user requirements. My real job was to encourage the staffs of the
> various scientific centers to make use of the CP-67-based Wide Area
> Network that was centered in Cambridge.

A beautiful piece of history

> ... [much more great stuff]

Subject: Re: vfork history, Multics vs Unix
Posted by scott on Sat, 25 Jan 2025 20:48:37 GMT

View Forum Message <> Reply to Message

ted@loft.tnolan.com (Ted Nolan <tednolan>) writes:
> In article <vn1bjs$2fink$5@dont-email.me>,
> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>> On Fri, 24 Jan 2025 14:42:00 -0000 (UTC), Lars Poulsen wrote:
>>
>>> vfork(2) looks odd for those who grew up in the unix traditions,
>>
>> But it was added by BSD, very much as part of their Unix tradition. It was
>> defined in POSIX (at least for a while), and Linux still supports it. The
>> man page <https://manpages.debian.org/vfork(2)> says:
>>
>> Some consider the semantics of vfork() to be an architectural
>> blemish, and the 4.2BSD man page stated: “This system call will be
>> eliminated when proper system sharing mechanisms are implemented.
>> Users should not depend on the memory sharing semantics of vfork
>> as it will, in that case, be made synonymous to fork.” However,
>> even though modern memory management hardware has decreased the

```
>>     performance difference between fork(2) and vfork(), there are
>>     various reasons why Linux and other systems have retained vfork():
>>
>>     • Some performance-critical applications require the small
>>       performance advantage conferred by vfork().
>>
>>     • vfork() can be implemented on systems that lack a
>>       memory-management unit (MMU), but fork(2) can't be implemented
>>       on such systems. (POSIX.1-2008 removed vfork() from the
>>       standard; the POSIX rationale for the posix_spawn(3) function

>>       lent to fork(2)+exec(3), is designed to be implementable on
>>       systems that lack an MMU.)
>>
>
> I don't think that's true, is it?  Venix & PC/IX ran on the 8086 which
> had no MMU and had fork().
```

Unix V6 ran on a PDP-11/34, with no MMU. Long before vfork was
conceived.

Subject: Re: vfork history, Multics vs Unix
Posted by scott on Sat, 25 Jan 2025 20:50:00 GMT

View Forum Message <> Reply to Message

cross@spitfire.i.gajendra.net (Dan Cross) writes:

```
> In article <lvjkcnF4lq8U1@mid.individual.net>,
> Bob Eager  <news0009@eager.cx> wrote:
>> On Sat, 25 Jan 2025 01:05:43 +0000, Ted Nolan <tednolan> wrote:
>>
>>>  In article <vn1bjs$2fink$5@dont-email.me>,
>>>  Lawrence D'Oliveiro  <ldo@nz.invalid> wrote:
>>>> On Fri, 24 Jan 2025 14:42:00 -0000 (UTC), Lars Poulsen wrote:
>>>>
>>>> > vfork(2) looks odd for those who grew up in the unix traditions,
>>>>
>>>> But it was added by BSD, very much as part of their Unix tradition. It
>>>> was defined in POSIX (at least for a while), and Linux still supports
>>>> it. The man page <https://manpages.debian.org/vfork(2)> says:
>>>>
>>>>     Some consider the semantics of vfork() to be an architectural
>>>>     blemish, and the 4.2BSD man page stated: "This system call will be
>>>>     eliminated when proper system sharing mechanisms are implemented.
>>>>     Users should not depend on the memory sharing semantics of vfork as
>>>>     it will, in that case, be made synonymous to fork." However, even
>>>>     though modern memory management hardware has decreased the
>>>>     performance difference between fork(2) and vfork(), there are
```

>>>>   various reasons why Linux and other systems have retained vfork():
>>>>
>>>>   • Some performance-critical applications require the small
>>>>   performance advantage conferred by vfork().
>>>>
>>>>   • vfork() can be implemented on systems that lack a
>>>>   memory-management unit (MMU), but fork(2) can't be implemented on
>>>>   such systems. (POSIX.1-2008 removed vfork() from the standard; the
>>>>   POSIX rationale for the posix_spawn(3) function notes that that

>>>>   lent to fork(2)+exec(3), is designed to be implementable on
>>>>   systems that lack an MMU.)
>>>
>>> I don't think that's true, is it? Venix & PC/IX ran on the 8086 which
>>> had no MMU and had fork().
>>
>> And so did Mini-UNIX, back around 1978. (I am NOT referring to MINIX, but
>> the Bell Labs system)
>
> Indeed. But speaking of Minix, early Minix didn't use the MMU,
> either. I understand this is one of the reasons Torvalds was
> dissatisfied with it, and started working on Linux.
>
> You can do a Unix-like fork without an MMU if you're willing to
> live with some constraints; for instance, only one process can
> be resident in memory at a time, perhaps.

IIRC, there were some gyrations around the U area (see the lions
commentary).   And the famous comment "You aren't expected to
understand this" in the fork code.

Subject: Re: vfork history, Multics vs Unix
Posted by Anonymous on Sat, 25 Jan 2025 22:27:17 GMT
View Forum Message <> Reply to Message

Originally posted by: Bob Eager

On Sat, 25 Jan 2025 20:48:37 +0000, Scott Lurndal wrote:

> ted@loft.tnolan.com (Ted Nolan <tednolan>) writes:
>> In article <vn1bjs$2fink$5@dont-email.me>,
>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>> On Fri, 24 Jan 2025 14:42:00 -0000 (UTC), Lars Poulsen wrote:
>>>
>>>> vfork(2) looks odd for those who grew up in the unix traditions,
>>>
>>> But it was added by BSD, very much as part of their Unix tradition. It

```
>>> was defined in POSIX (at least for a while), and Linux still supports
>>> it. The man page <https://manpages.debian.org/vfork(2)> says:
>>>
>>>     Some consider the semantics of vfork() to be an architectural
>>>     blemish, and the 4.2BSD man page stated: “This system call will be
>>>     eliminated when proper system sharing mechanisms are implemented.
>>>     Users should not depend on the memory sharing semantics of vfork as
>>>     it will, in that case, be made synonymous to fork.” However, even
>>>     though modern memory management hardware has decreased the
>>>     performance difference between fork(2) and vfork(), there are
>>>     various reasons why Linux and other systems have retained vfork():
>>>
>>>     • Some performance-critical applications require the small
>>>       performance advantage conferred by vfork().
>>>
>>>     • vfork() can be implemented on systems that lack a
>>>       memory-management unit (MMU), but fork(2) can't be implemented on
>>>       such systems. (POSIX.1-2008 removed vfork() from the standard;
>>>       the POSIX rationale for the posix_spawn(3) function notes that

>>>       lent to fork(2)+exec(3), is designed to be implementable on
>>>       systems that lack an MMU.)
>>>
>>>
>> I don't think that's true, is it?  Venix & PC/IX ran on the 8086 which
>> had no MMU and had fork().
>
>  Unix V6 ran on a PDP-11/34, with no MMU.  Long before vfork was
>  conceived.
```

I do not believe that. And I used it - a lot. But Mini-UNIX (a stripped
down V6) would run even on an 11/20.

  https://www.tavi.co.uk/unixhistory/mini-unix.html

--
Using UNIX since v6 (1975)...

Use the BIG mirror service in the UK:
 http://www.mirrorservice.org

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Sat, 25 Jan 2025 22:27:29 GMT
View Forum Message <> Reply to Message

Lars Poulsen <lars@cleo.beagle-ears.com> writes:
> Of course, a tight cluster with survivability cannot have a strict
> hierarchy, since the top position would have to be renegotiated if
> the "master" node failed.

.... when I was out marketing HA/CMP I coined the termas "disaster survivability" and "geographic survivability". The IBM S/88 product administrator also started taking us around to their customers ... and got me to write a section for the corporate continuous availability strategy document (it got pulled when both Rochester/AS400 and POK/mainframe complained they couldn't meet the objectives).
https://www.pcmag.com/encyclopedia/term/system88

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Bill Findlay on Sun, 26 Jan 2025 00:05:16 GMT
View Forum Message <> Reply to Message

On 25 Jan 2025, Lawrence D'Oliveiro wrote
(in article <vn1apj$2fink$2@dont-email.me>):
>
> But note that the trend of falling memory prices was already becoming
> clear by the 1970s, if not earlier. The earliest batch systems only kept
> one program in memory at one time, and swapped it out for another one when
> it went into any kind of I/O wait, to keep the CPU busy...
They did no such thing.

--
Bill Findlay

Subject: Re: vfork history, Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 00:12:31 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

ted@loft.tnolan.com (Ted Nolan <tednolan>) writes:
>>> I don't think that's true, is it? Venix & PC/IX ran on the 8086 which
>>> had no MMU and had fork().
>>
On Sat, 25 Jan 2025 20:48:37 +0000, Scott Lurndal wrote:
>> Unix V6 ran on a PDP-11/34, with no MMU. Long before vfork was

>> conceived.

11/34 was like 11/35 and 11/40. Had an 18-bit MMU (22-bit UNIBUS map).
Unlike the 11/45 (and 11/44 and 11/70) it did niot have I-space and
D-space.

Subject: Re: Multics vs Unix
Posted by Rich Alderson on Sun, 26 Jan 2025 00:23:31 GMT
View Forum Message <> Reply to Message

Peter Flass <peter_flass@yahoo.com> writes:

> Lynn Wheeler <lynn@garlic.com> wrote:

>> A small subset of this was released in VM370R3 as DCSS (but without the
>> page-mapped filesystem and w/o the independent address location support)
>> ... where the shared segment images were saved in special defined VM370
>> disk areas

> It was fun mapping the DCSS to specific memory addresses, and making sure
> that a program didn't need two mapped to the same address. This, I think,
> was S/370, so 31-bit (16MB) address space. Didn't DCSS sizes have to be a
> multiple of 64K? It's been a while, so I don't recall the details.

The 370, like the 360 before it, was a 24 bit address architecture, for an
address space of 16,777,216 locations (16MB).

31-bit addressing is 303x/308x or 390, as I remember things.

(Yes, I've been a PDP-10 guy for more than 45 years, but I was a 36-370 guy for
10 years befreo that.)

--
Rich Alderson   news@alderson.users.panix.com
      Audendum est, et veritas investiganda; quam etiamsi non assequamur,
   omnino tamen proprius, quam nunc sumus, ad eam perveniemus.
         --Galen

Subject: Re: Multics vs Unix
Posted by Rich Alderson on Sun, 26 Jan 2025 00:46:01 GMT
View Forum Message <> Reply to Message

cross@spitfire.i.gajendra.net (Dan Cross) writes:

> The original context was a poster asking why Unix shells haven't been

> implemented in a VMS/TOPS-20/Multics style, where "commands" are invoked by
> mapping some some shared object that implements the command into the address
> space of the shell process and invoking it by calling its `main` or whatever;
> that is, instead of `fork` and `exec`, map and call.

Hey, Dan,

Your characterization of TOPS-20 here is incorrect.

First off, commands (no need for quotes) are pieces of code assembled into the command processor (EXEC.EXE), accessed by lookup tables handled by the COMND% JSYS. This system call deals with everything (guide words, ESCape recognition, subcommand tables, etc.). There is nothing external to EXEC in a command.

In additions to builtin commands, there are programs. Early on, in TENEX and early TOPS-20, programs were executed via the RUN or R commands (where RUN defaults to the connected directory, and R executes from the SYS: logical path (which the user can customize for herself). In later TOPS-20, at least at Stanford, an unrecognized command (in the sense of builtins) causes a search on the SYS: path for an executable program by that name.

But when an executable is wanted, the mechanism is to CFORK% create a process, GET% a memory image of the executable file into *that* process, and SFORK% start the process. The calling program (including the EXEC) can WFORK% wait for process execution to conclude, or move on to other things.

This is not the (early?) Unix model, and it may not model the VMS operation, which grew up from RSX-11M.

--
Rich Alderson news@alderson.users.panix.com
Audendum est, et veritas investiganda; quam etiamsi non assequamur,
omnino tamen proprius, quam nunc sumus, ad eam perveniemus.
--Galen

Subject: Re: Multics vs Unix
Posted by Rich Alderson on Sun, 26 Jan 2025 00:49:40 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:

> On Wed, 22 Jan 2025 10:58:03 -0000 (UTC), Waldek Hebisch wrote:

>> Clearly Multics was more advanced. I was simply pointing out that the
>> same "single task for single user" mindset seem to be present in
>> Multics.

> All the OSes of the time tried to minimize process creation. Unix was the
> one going against the trend, by its seemingly profligate creation of new
> processes for doing every single thing.

> One big change was that the CLI was not some part of the resident kernel,
> but just another user process. Hard to realize this was seen as quite
> radical at the time.

Once again...

The separation of the command processor from the kernel was a feature of TENEX
on the PDP-10 when UNICS was still being written on the PDP-7.

--
Rich Alderson   news@alderson.users.panix.com
    Audendum est, et veritas investiganda; quam etiamsi non assequamur,
  omnino tamen proprius, quam nunc sumus, ad eam perveniemus.
      --Galen

Subject: Re: Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 01:51:08 GMT

View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On 25 Jan 2025 19:46:01 -0500, Rich Alderson wrote:

> But when an executable is wanted, the mechanism is to CFORK% create a
> process, GET% a memory image of the executable file into *that* process,
> and SFORK% start the process. The calling program (including the EXEC)
> can WFORK% wait for process execution to conclude, or move on to other
> things.

That sounds like a TOPS-20/TENEX thing, rather than TOPS-10.

I don’t think any home-grown DEC OS created new processes to run new
commands as a matter of course, with the possible exception of RSX-11,
with its rather convoluted notion of “tasks”.

Subject: Re: Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 01:53:01 GMT

View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 26 Jan 2025 00:05:16 +0000, Bill Findlay wrote:

> On 25 Jan 2025, Lawrence D'Oliveiro wrote (in article
> <vn1apj$2fink$2@dont-email.me>):
>>
>> But note that the trend of falling memory prices was already
>> becoming clear by the 1970s, if not earlier. The earliest batch
>> systems only kept one program in memory at one time, and swapped it
>> out for another one when it went into any kind of I/O wait, to keep
>> the CPU busy; later on, when memory became large enough (and cheap
>> enough), it made sense to keep multiple programs resident at once
>> ("multiprogramming", this was called), to allow the scheduling
>> switches to happen more quickly.
>
> They did no such thing.

They didn't have enough RAM to do anything else. That's why a special term had to be invented for the new feature.

Subject: Re: Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 01:59:14 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On 25 Jan 2025 19:49:40 -0500, Rich Alderson wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>
>> One big change was that the CLI was not some part of the resident
>> kernel, but just another user process.
>
> The separation of the command processor from the kernel was a feature of
> TENEX on the PDP-10 when UNICS was still being written on the PDP-7.

Did the "command processor" have any special privileges or status on those systems?

On Unix it did not.

Subject: Re: vfork history, Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 02:01:23 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 26 Jan 2025 00:12:31 -0000 (UTC), Lars Poulsen wrote:

> 11/34 was like 11/35 and 11/40. Had an 18-bit MMU (22-bit UNIBUS map).

Those earlier/smaller PDP-11 models only had 18-bit physical addresses,
and that applied to UNIBUS accesses as well.

22-bit physical addressing came later.

Subject: Re: Multics vs Unix
Posted by Rich Alderson on Sun, 26 Jan 2025 02:10:09 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:

> On 25 Jan 2025 19:46:01 -0500, Rich Alderson wrote:

>> But when an executable is wanted, the mechanism is to CFORK% create a
>> process, GET% a memory image of the executable file into *that* process,
>> and SFORK% start the process. The calling program (including the EXEC)
>> can WFORK% wait for process execution to conclude, or move on to other
>> things.

> That sounds like a TOPS-20/TENEX thing, rather than TOPS-10.

In the portion quoted from Dan Cross which you snipped, he wrote the string

        VMS/TOPS-20/Multics

Try to keep up.

> I don't think any home-grown DEC OS created new processes to run new
> commands as a matter of course, with the possible exception of RSX-11,
> with its rather convoluted notion of "tasks".

--
Rich Alderson   news@alderson.users.panix.com
    Audendum est, et veritas investiganda; quam etiamsi non assequamur,
  omnino tamen proprius, quam nunc sumus, ad eam perveniemus.
        --Galen

Subject: Re: Multics vs Unix
Posted by Rich Alderson on Sun, 26 Jan 2025 02:14:47 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:

> On 25 Jan 2025 19:49:40 -0500, Rich Alderson wrote:

>> Lawrence D'Oliveiro <ldo@nz.invalid> writes:

>>> One big change was that the CLI was not some part of the resident
>>> kernel, but just another user process.

>> The separation of the command processor from the kernel was a feature of
>> TENEX on the PDP-10 when UNICS was still being written on the PDP-7.

> Did the "command processor" have any special privileges or status on those
> systems?

> On Unix it did not.

It did not. The program EXEC.EXE was a vanilla an executable as you could wish
for.

TENEX/TOPS-20 does not have a concept of "program privileges" ("suid/sgid" or
whatever), but of USER privileges. Certain system calls check for appropriate
user privileges before running, or (as in the case of EXEC) before performing
a privileged action.

--
Rich Alderson news@alderson.users.panix.com
Audendum est, et veritas investiganda; quam etiamsi non assequamur,
omnino tamen proprius, quam nunc sumus, ad eam perveniemus.
--Galen

Subject: Re: Multics vs Unix
Posted by cross on Sun, 26 Jan 2025 02:43:34 GMT
View Forum Message <> Reply to Message

In article <mddwmei4e4c.fsf@panix5.panix.com>,
Rich Alderson <news@alderson.users.panix.com> wrote:
> Peter Flass <peter_flass@yahoo.com> writes:
>
>> Lynn Wheeler <lynn@garlic.com> wrote:
>
>>> A small subset of this was released in VM370R3 as DCSS (but without the
>>> page-mapped filesystem and w/o the independent address location support)
>>> ... where the shared segment images were saved in special defined VM370
>>> disk areas
>
>> It was fun mapping the DCSS to specific memory addresses, and making sure

>> that a program didn't need two mapped to the same address. This, I think,
>> was S/370, so 31-bit (16MB) address space. Didn't DCSS sizes have to be a
>> multiple of 64K? It's been a while, so I don't recall the details.
>
> The 370, like the 360 before it, was a 24 bit address architecture, for an
> address space of 16,777,216 locations (16MB).
>
> 31-bit addressing is 303x/308x or 390, as I remember things.

I remember it as 370-XA; 308x is right. ESA/370 (the infamous
3090) was a few years later.

> (Yes, I've been a PDP-10 guy for more than 45 years, but I was a 36-370 guy for
> 10 years befreo that.)

I'm glad people are keeping the -10 alive. :-)

 - Dan C.

Subject: Re: Multics vs Unix
Posted by cross on Sun, 26 Jan 2025 02:48:32 GMT
View Forum Message <> Reply to Message

In article <mddtt9m4d2u.fsf@panix5.panix.com>,
Rich Alderson <news@alderson.users.panix.com> wrote:
> cross@spitfire.i.gajendra.net (Dan Cross) writes:
>
>> The original context was a poster asking why Unix shells haven't been
>> implemented in a VMS/TOPS-20/Multics style, where "commands" are invoked by
>> mapping some some shared object that implements the command into the address
>> space of the shell process and invoking it by calling its `main` or whatever;
>> that is, instead of `fork` and `exec`, map and call.
>
> Hey, Dan,
>
> Your characterization of TOPS-20 here is incorrect.

I stand corrected.

> First off, commands (no need for quotes) are pieces of code assembled into the
> command processor (EXEC.EXE), accessed by lookup tables handled by the COMND%
> JSYS. This system call deals with everything (guide words, ESCape recognition,
> subcommand tables, etc.). There is nothing external to EXEC in a command.
>
> In additions to builtin commands, there are programs. Early on, in TENEX and
> early TOPS-20, programs were executed via the RUN or R commands (where RUN
> defaults to the connected directory, and R executes from the SYS: logical path

> (which the user can customize for herself). In later TOPS-20, at least at
> Stanford, an unrecognized command (in the sense of builtins) causes a search on
> the SYS: path for an executable program by that name.
>
> But when an executable is wanted, the mechanism is to CFORK% create a process,
> GET% a memory image of the executable file into *that* process, and SFORK%
> start the process. The calling program (including the EXEC) can WFORK% wait
> for process execution to conclude, or move on to other things.
>
> This is not the (early?) Unix model, and it may not model the VMS operation,
> which grew up from RSX-11M.

Thanks, Rich, that was very informative!

 - Dan C.

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Sun, 26 Jan 2025 03:11:24 GMT
View Forum Message <> Reply to Message

Peter Flass <peter_flass@yahoo.com> writes:
> It was fun mapping the DCSS to specific memory addresses, and making sure
> that a program didn’t need two mapped to the same address. This, I think,
> was S/370, so 31-bit (16MB) address space. Didn’t DCSS sizes have to be a
> multiple of 64K? It’s been a while, so I don’t recall the details.

370 virtual memory architecture allowed for two page sizes, 2k and 4k
.... and two segment sizes, 64k and 1mbyte.

for vm370 CMS, used 4k page size and 64k (16 4k pages) segment size.

as i've periodic mentioned, DCSS ("discontiguous shared segments") was
small piece of my CMS page mapped filesystem ... which included being
able to specify loading file with pieces as shared segment(s).

370 had two level virtual memory table ... a segment table where each
entry pointed to that (segment's) page table (for vm370, 16 4k page
entries). in the case of a shared segment ... each individual virtual
memory segment table entry point to a common, shared page table.

in the case of my CMS page mapped filesystem, filesystem control
information specified mapping to individual pages ... or for a
mapping(s) that specified 16 4k pages to be mapped as a "shared
segment".

Since the CMS page mapped filesystem wasn't picked up for VM370R3, all

the control information had to placed somewhere. The VM370 module name
was (kernel assembled) DMKSNT which had SNT entires information
specifying the number of pages, virtual addresses of the pages, which
portion of the virtual pages to be treated as shared segments, physical
disk locations of the saved pages, etc.

A sysadm or sysprog ... would typically load application into their
virtual memory and issue the kernel (privileged) "SAVESYS" command
specifying a SNT entry, the pages from their virtual memory would then
be written to the specified disk locations. Then users could issue the
"LOADSYS" command specifying a SNT entry, which would reverse the
process, mapping their virtual memory to the SNT entry specified pages
(and in case of shared segment, their corresponding segment table
entry/entries mapped to the shared page table(s)).

If wanting to change, add or delete a DMKSNT SNT entry, edit the
assemble source, assemble the changed DMKSNT file, rebuild the kernel
(w/changed DMKSNT) and re-ipl (note SNT entries were global system
wide).

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by cross on Sun, 26 Jan 2025 04:04:52 GMT

View Forum Message <> Reply to Message

In article <mddr04q4cwr.fsf@panix5.panix.com>,
Rich Alderson <news@alderson.users.panix.com> wrote:
> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>> On Wed, 22 Jan 2025 10:58:03 -0000 (UTC), Waldek Hebisch wrote:
>>> Clearly Multics was more advanced. I was simply pointing out that the
>>> same "single task for single user" mindset seem to be present in
>>> Multics.
>>
>> All the OSes of the time tried to minimize process creation. Unix was the
>> one going against the trend, by its seemingly profligate creation of new
>> processes for doing every single thing.
>>
>> One big change was that the CLI was not some part of the resident kernel,
>> but just another user process. Hard to realize this was seen as quite
>> radical at the time.
>
> Once again...
>
> The separation of the command processor from the kernel was a feature of TENEX
> on the PDP-10 when UNICS was still being written on the PDP-7.

You're point is well taken. It is perhaps worth a mention that
Multics also did this. Unix was hardly "radical" for separating
the shell and kernel.

That said, I'm not sure the timeline here is true.

According to the "Status of the TENEX Memos" document from 1972,
(http://www.bitsavers.org/pdf/bbn/tenex/TENEX_Memos_Mar72.pdf),
the TENEX memos were published as a set in 1970, but "at that
time, coding had not begun." It's not entirely clear to me if
this means that implementation had not begun by 1970, or if
coding had not begun until most of the TENEX memos were written
and the 1970 reference just marks the year the memos appeared as
a set, independent (at that point) of implementation. That same
memo mentions that "it has now been nearly two years since the
first version of the TENEX monitor was placed in operation."
Given that that memo was written in March 1972, this puts TENEX
going into service sometime in spring/summer of 1970; Dan
Murphy's TENEX/TOPS-20 history paper puts it as mid-June.
I know development was rapid; Murphy's paper makes mention of a,
"very tight schedule held since November, 1969." Elsewhere
Murphy makes mention of working on TENEX starting in 1968; the
"Origins and Development of TOPS-20" paper says that "most of
its architecture remained as it was designed in 1969" [at BBN]:
https://opost.com/tenex/hbook.html

UNICS on the PDP-7 was operational and supporting users in 1969,
if only in nascent form. As Dennis Ritchie put it, "by the
beginning of 1970, PDP-7 Unix was a going concern." It was
moved to the PDP-11 in late-summer 1970 (though they didn't have
a disk until the end of the year, so it was pretty anemic at the
time: https://www.bell-labs.com/usr/dmr/www/hist.html). Even on
the PDP-7, the shell was a separate program from the kernel, and
the Ritchie paper mentioned says clearly that,

So it appears to me that TENEX and Unix are mostly
contemporaneous, with the latter perhaps having a slight edge
over the former in terms of entering "production" first. Of
course, PDP-7 UNICS was a much simpler system than TENEX.

- Dan C.

Subject: Re: vfork history, Multics vs Unix
Posted by cross on Sun, 26 Jan 2025 04:18:23 GMT
View Forum Message <> Reply to Message

In article <lvkksrFncvgU1@mid.individual.net>,
Ted Nolan <tednolan> <tednolan> wrote:
> In article <vn2vd8$q4k$1@reader2.panix.com>,
> Dan Cross <cross@spitfire.i.gajendra.net> wrote:
>> In article <lvjkcnF4lq8U1@mid.individual.net>,
>>> And so did Mini-UNIX, back around 1978. (I am NOT referring to MINIX, but
>>> the Bell Labs system)
>>
>> Indeed. But speaking of Minix, early Minix didn't use the MMU,
>> either. I understand this is one of the reasons Torvalds was
>> dissatisfied with it, and started working on Linux.
>>
>> You can do a Unix-like fork without an MMU if you're willing to
>> live with some constraints; for instance, only one process can
>> be resident in memory at a time, perhaps.
>
> I never used PC/IX, but my memory from reading about it in the press
> and here on Usenet at the time is that it definitely was a full,
> AT&T SYSIII Unix implementation, with more than one process resident
> at a time.

Cool.

"One process resident in memory at a time" is just one way it
might be done; I didn't mean to imply it was the only option.

> I think the way they tried to finesse memory protection was that
> the C compiler would not emit code for changing the base register,
> so even if a program went wild, it would be within one 64k segment
> only. Of course this was not foolproof and would be easy to defeat,
> but the idea was, who would try given the platform.

On x86 pre-386, mucking about with segmentation is certainly a
way go. I assume Minix did something similar, but I'm not in a
spot where I can check the code easily at the moment.

Mini-Unix ran on the LSI-11, which didn't do x86-style
segmentation, so in some sense was more constrained.

 - Dan C.

Subject: Re: vfork history, Multics vs Unix
Posted by cross on Sun, 26 Jan 2025 04:43:45 GMT

View Forum Message <> Reply to Message

In article <Y7clP.1278404$Uup4.20588@fx10.iad>,
Scott Lurndal <slp53@pacbell.net> wrote:

> cross@spitfire.i.gajendra.net (Dan Cross) writes:
>> In article <lvjkcnF4lq8U1@mid.individual.net>,
>> Bob Eager <news0009@eager.cx> wrote:
>>> On Sat, 25 Jan 2025 01:05:43 +0000, Ted Nolan <tednolan> wrote:
>>>
>>>> In article <vn1bjs$2fink$5@dont-email.me>,
>>>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>>> >On Fri, 24 Jan 2025 14:42:00 -0000 (UTC), Lars Poulsen wrote:
>>>> >
>>>> >> vfork(2) looks odd for those who grew up in the unix traditions,
>>>> >
>>>> >But it was added by BSD, very much as part of their Unix tradition. It
>>>> >was defined in POSIX (at least for a while), and Linux still supports
>>>> >it. The man page <https://manpages.debian.org/vfork(2)> says:
>>>> >
>>>> >   Some consider the semantics of vfork() to be an architectural
>>>> >   blemish, and the 4.2BSD man page stated: “This system call will be
>>>> >   eliminated when proper system sharing mechanisms are implemented.
>>>> >   Users should not depend on the memory sharing semantics of vfork as
>>>> >   it will, in that case, be made synonymous to fork.” However, even
>>>> >   though modern memory management hardware has decreased the
>>>> >   performance difference between fork(2) and vfork(), there are
>>>> >   various reasons why Linux and other systems have retained vfork():
>>>> >
>>>> >   • Some performance-critical applications require the small
>>>> >   performance advantage conferred by vfork().
>>>> >
>>>> >   • vfork() can be implemented on systems that lack a
>>>> >   memory-management unit (MMU), but fork(2) can't be implemented on
>>>> >   such systems. (POSIX.1-2008 removed vfork() from the standard; the
>>>> >   POSIX rationale for the posix_spawn(3) function notes that that

>>>> >   lent to fork(2)+exec(3), is designed to be implementable on
>>>> >   systems that lack an MMU.)
>>>>
>>>> I don't think that's true, is it? Venix & PC/IX ran on the 8086 which
>>>> had no MMU and had fork().
>>>
>>> And so did Mini-UNIX, back around 1978. (I am NOT referring to MINIX, but
>>> the Bell Labs system)
>>
>> Indeed. But speaking of Minix, early Minix didn't use the MMU,
>> either. I understand this is one of the reasons Torvalds was
>> dissatisfied with it, and started working on Linux.
>>
>> You can do a Unix-like fork without an MMU if you're willing to
>> live with some constraints; for instance, only one process can
>> be resident in memory at a time, perhaps.

>
> IIRC, there were some gyrations around the U area (see the lions
> commentary).   And the famous comment "You aren't expected to
> understand this" in the fork code.

For vfork? Yeah; the kernel stack for a process lives in the u
area, so you've got to play some games to get that to match up
right. I think the 4.3BSD book talks about this; something that
describes vfork does if not that.

"You are not expected to understand this" is such a classic.
The fundamental problem was that they, essentially, used a
non-local goto to do a coroutine jump, but relied on a function
returning from a stack that wasn't its own behaving correctly.
Dennis Ritchie explained it here:
https://www.bell-labs.com/usr/dmr/www/odd.html

 - Dan C.

Subject: Re: Multics vs Unix
Posted by Bill Findlay on Sun, 26 Jan 2025 12:18:39 GMT
View Forum Message <> Reply to Message

On 26 Jan 2025, Lawrence D'Oliveiro wrote
(in article <vn44ht$37klv$2@dont-email.me>):

> On Sun, 26 Jan 2025 00:05:16 +0000, Bill Findlay wrote:
>
>> On 25 Jan 2025, Lawrence D'Oliveiro wrote (in article
>> <vn1apj$2fink$2@dont-email.me>):
>>>
>>> But note that the trend of falling memory prices was already
>>> becoming clear by the 1970s, if not earlier. The earliest batch
>>> systems only kept one program in memory at one time, and swapped it
>>> out for another one when it went into any kind of I/O wait, to keep
>>> the CPU busy; later on, when memory became large enough (and cheap
>>> enough), it made sense to keep multiple programs resident at once
>>> ("multiprogramming", this was called), to allow the scheduling
>>> switches to happen more quickly.
>>
>> They did no such thing.
>
> They didn´t have enough RAM to do anything else. That´s why a special term
> had to be invented for the new feature.

Nonsense.

--
Bill Findlay

Subject: The history of "multi-programming"
Posted by Anonymous on Sun, 26 Jan 2025 14:33:48 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

On 25 Jan 2025, Lawrence D'Oliveiro wrote (in article
 <vn1apj$2fink$2@dont-email.me>):
>>>> But note that the trend of falling memory prices was already
>>>> becoming clear by the 1970s, if not earlier. The earliest batch
>>>> systems only kept one program in memory at one time, and swapped it
>>>> out for another one when it went into any kind of I/O wait, to keep
>>>> the CPU busy; later on, when memory became large enough (and cheap
>>>> enough), it made sense to keep multiple programs resident at once
>>>> ("multiprogramming", this was called), to allow the scheduling
>>>> switches to happen more quickly.

On Sun, 26 Jan 2025 00:05:16 +0000, Bill Findlay wrote:
>>> They did no such thing.

On 26 Jan 2025, Lawrence D'Oliveiro wrote
 (in article <vn44ht$37klv$2@dont-email.me>):
>> They didn´t have enough RAM to do anything else. That´s why a special term
>> had to be invented for the new feature.

On 2025-01-26, Bill Findlay <findlaybill@blueyonder.co.uk> wrote:
> Nonsense.

Be nice, be respectful.

When Lawrence says "the earliest batch systems", he is not going far
enough back. The earliest batch systems only ran ONE JOB AT A TIME,
and the time spent in I/O wait ... well, there was often no IOWAIT
instruction. The I/O instruction did not complete until the transfer
was done. Then someone - I assume IBM - came up with the idea of a
an I/O co-processor called a "channel", to allow the I/O to "steal
memory cycles" from the CPU. At that point it became possible to do
"multiprogramming", what we today would call threads. (I think channels
were introduced on the 700 series in the mid-1950s).

To reduce the waste of an expensive fast CPU, it was common to
redirect the slowest peripherals (card read, print and card punch)
to tape, and use a separate offline device for them, eventually
a cheap slow processor, which at first might be a plugboard

device.

The earliest use of threads in TAPE operating systems would have been
overlapped read and write in tape sort programs.

The availability of threads, enabled the feature of Simultaneous
Peripheral Operations On-Line (SPOOL) processing, either as part of
the (resident) O/S, or in a separate memory partition as a second
active job.

With disk drives available, it might be possible to run a secondary
job stream which swapped jobs in and out as Lawrence describes,
but I have only ever seen that done with interactive jobs where
the long wait that triggered a swap was a read from the terminal.

But then again, I did not lay my hands on a computer until 1969.

How old are you Lawrence, and when and where was your first hands-on
computer use?

Subject: The evolution of MMUs (was: Multics vs Unix)
Posted by Anonymous on Sun, 26 Jan 2025 14:47:30 GMT

View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

Peter Flass <peter_flass@yahoo.com> writes:
>> It was fun mapping the DCSS to specific memory addresses, and making sure
>> that a program didn't need two mapped to the same address. This, I think,
>> was S/370, so 31-bit (16MB) address space. Didn't DCSS sizes have to be a
>> multiple of 64K? It's been a while, so I don't recall the details.

On 2025-01-26, Lynn Wheeler <lynn@garlic.com> wrote:
> 370 virtual memory architecture allowed for two page sizes, 2k and 4k
> ... and two segment sizes, 64k and 1mbyte.
>
> for vm370 CMS, used 4k page size and 64k (16 4k pages) segment size.
>
> as i've periodic mentioned, DCSS ("discontiguous shared segments") was
> small piece of my CMS page mapped filesystem ... which included being
> able to specify loading file with pieces as shared segment(s).
>
> 370 had two level virtual memory table ... a segment table where each
> entry pointed to that (segment's) page table (for vm370, 16 4k page
> entries). in the case of a shared segment ... each individual virtual
> memory segment table entry point to a common, shared page table.
>

> in the case of my CMS page mapped filesystem, filesystem control
> information specified mapping to individual pages ... or for a
> mapping(s) that specified 16 4k pages to be mapped as a "shared
> segment".
>
> Since the CMS page mapped filesystem wasn't picked up for VM370R3, all
> the control information had to placed somewhere. The VM370 module name
> was (kernel assembled) DMKSNT which had SNT entires information
> specifying the number of pages, virtual addresses of the pages, which
> portion of the virtual pages to be treated as shared segments, physical
> disk locations of the saved pages, etc.
> [... good stuff snipped ...]

It seems to have been a delicate dance between kernel programmaers and
hardware engineers to figure out how to help the O/S people make better
features to benefit the users, on one side, and figuring out how to use
the new hardware to overcome previous limitations.

In the original 360 design the company seemed to take great pride in
having the two teams work together, and that is also reported as having
been very much the case in VAX/VMS. But in IBM, that process seems to
have been the downfall of TSS/360, which turned from a "this is our
future" project like 360, into an expensive overgrown research lab.

Does anyone have thoughts on that?

Subject: Re: vfork history, Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 14:56:49 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

On Sun, 26 Jan 2025 00:12:31 -0000 (UTC), Lars Poulsen wrote:
>> 11/34 was like 11/35 and 11/40. Had an 18-bit MMU (22-bit UNIBUS map).

On 2025-01-26, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> Those earlier/smaller PDP-11 models only had 18-bit physical addresses,
> and that applied to UNIBUS accesses as well.
>
> 22-bit physical addressing came later.

I should have have added a "but" inside the parenthesis.

I.e., you are right that the earlier (11/40 and its 11/35 cousin) were
purely 18-bits physical addressing, but according to WikiPedia,
the 11/34 had a 22-bit address space somewhere, though I don't know
what it did with it. Personally, I only knew the 11/34 as a drop-in

replacement for the 11/35 systems on which I did my early industrial applications.

Subject: Re: vfork history, Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 15:07:26 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

On 2025-01-26, Dan Cross <cross@spitfire.i.gajendra.net> wrote:
> "You are not expected to understand this" is such a classic.
> The fundamental problem was that they, essentially, used a
> non-local goto to do a coroutine jump, but relied on a function
> returning from a stack that wasn't its own behaving correctly.
> Dennis Ritchie explained it here:
> https://www.bell-labs.com/usr/dmr/www/odd.html

Quoting DMR from there:
 So we tried to explain what was going on. "You
 are not expected to understand this" was intended
 as a remark in the spirit of "This won't be on
 the exam," rather than as an impudent challenge.

 The real problem is that we didn't understand what
 was going on either. The savu/retu mechanism for
 doing process exchange was fundamentally broken
 because it depended on switching to a previous
 stack frame and executing function return code in
 a different procedure from the one that saved the
 earlier state. This worked on the PDP-11 because
 its compiler always used the same context-save
 mechanism; with the Interdata compiler, the
 procedure return code differed depending on which
 registers were saved.

Very interesting. The modern "portable" version of context switching in the RTOSes I have looked at internals for, uses "longjmp()".

Subject: Re: vfork history, Multics vs Unix
Posted by Charlie Gibbs on Sun, 26 Jan 2025 18:32:58 GMT
View Forum Message <> Reply to Message

On 2025-01-26, Lars Poulsen <lars@cleo.beagle-ears.com> wrote:

> Quoting DMR from there:

> So we tried to explain what was going on. "You
> are not expected to understand this" was intended
> as a remark in the spirit of "This won't be on
> the exam," rather than as an impudent challenge.

Sadly, the attitude of modern-day software developers toward
users has morphed into "You are expected not to understand this."

Ignorance is strength.
-- George Orwell: Nineteen Eighty-Four

```
--
/~\  Charlie Gibbs                  |  Growth for the sake of
\ /  <cgibbs@kltpzyxm.invalid>      |  growth is the ideology
 X   I'm really at ac.dekanfrus     |  of the cancer cell.
/ \  if you read it the right way.  |    -- Edward Abbey
```

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Sun, 26 Jan 2025 18:52:26 GMT
View Forum Message <> Reply to Message

jgd@cix.co.uk (John Dallman) writes:
> On the IBM 360 family, working with multiple memory partitions required
> 256+KB RAM, which was a lot in the 1960s. Nowadays, that's L2 cache on a
> mobile device processor, but it was a fairly big mainframe 55 years ago,
> and the programs it ran were small enough - mostly written in assembler -
> to fit several into memory.

Starting in the middle 70s, I would periodically claim tht the original
1965 360 CKD DASD offloadeded function searching for information in the
I/O rather than keeping it cached in limited/small real storage ... but
by mid-70s that trade-off was starting to invert. CKD (multi-track
search) searching directories and data could involve multiple
full-cylinder searches, each one taking 1/3sec; which would not only
busy the disk, but also controller (blocking access to all other disks
on that controller) and channel (blocking access to all other disks on
that channel), limited memory met that matching compare had to refetch
the information from system memory for each compare ... limiting the
amount of concurrent I/O possible.

after transferring to SJR in 77, I got to wander around lots of (IBM &
non-IBM) datacenters in silicon valley, including disk
bldg14/engineering and bldg15/product test across the street. At the
time they were had 7x24, prescheduled, mainframe testing ... and had
recently tried MVS ... but it had 15min MTBF (requiring manual re-ipl)
in that environment. I offered to rewrite I/O subsystem making it bullet
proof and never crash, allowing any amount of concurrent disk testing,

greatly improving productivity. I also wrote an internal research report on all the things (to the I/O subsystem) that needed to be done and happen to mention the MVS MTBF (bringing down the wrath of the MVS organization on my head).

I also worked with Jim Gray and Vera Watson on the original SQL/relational implementation, System/R. The major company IMS DBMS product was stressing that System/R was significantly slower (than IMS). IMS kept physical disk addresses internally in data records for related data and IMS would say that System/R record twice the disk space for data indexes and 4-5 times the I/O (transversing the physical disk records). System/R counter was that IMS had much greater sysadmin human time ... managing exposed data record addresses in data records.

By the early 80s, that had started to change, disk price/bit was significantly dropping (offsetting the doubling space requirement for indexes) and system memories sizes were significantly allowing indexes to be cached, significantly reducing I/O ... while overall dataprocessing dropping costs was greatly expanding computer installs and use of DBMS ... putting stress on available human skills and resources.

Early 80s, I wrote a tome that the relative system disk throughput had dropped by an order of magnitude since 1964 (disks had gotten 3-5 times faster while systems got 40-50 times faster, requiring increasing amount of concurrent I/O (and multi-tasking) ... but the extensive use of CKD multi-track search limited its effectiveness. A disk division executive took exception and assigned the division performance group to refute the claims ... after a few weeks, they came back and explained that I had slightly understated the problem..

There was a recent observation that the (current) latency of cache misses and main memory operations, when measured in in count of current processor cycles is comparable to 60s disk latency when measured in count of 60s processor cycles (memory is the new disk).

This is also the explanation given for decision to add virtual memory to all 370s, because MVT needed to increasingly increase the number of concurrent tasks to maintain system utilization and justification aka MVT storage management was so bad that region storage size requirements had to be specified four times larger than actually used limiting typical 1mbyte 370/165 to four tasks, going to 16mbyte virtual memory (VS2/SVS) would allow number of concurrently executing tasks to be increased by nearly a factor of four times (with little or no paging)
.... although capped at 15 because of 4bit storage protect key
.... keeping regions separated/isolated from each other. As systems increased processing power ... had to bypass 15 cap by going to separate virtual address space for each region (VS2/MVS).

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix
Posted by John Levine on Sun, 26 Jan 2025 21:02:52 GMT
View Forum Message <> Reply to Message

According to Dan Cross <cross@spitfire.i.gajendra.net>:
>> 31-bit addressing is 303x/308x or 390, as I remember things.
>
> I remember it as 370-XA; 308x is right. ESA/370 (the infamous
> 3090) was a few years later.

It was 370/XA, according to the manual I have here.

>> (Yes, I've been a PDP-10 guy for more than 45 years, but I was a 36-370 guy for
>> 10 years befreo that.)

I started programming a PDP-6 in about 1969, was programming -10 until I left
grad school in 1981.

--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: ancient hacks, was Multics vs Unix
Posted by John Levine on Sun, 26 Jan 2025 21:12:04 GMT
View Forum Message <> Reply to Message

According to John Dallman <jgd@cix.co.uk>:
> In article <87wmehmmqd.fsf@localhost>, lynn@garlic.com (Lynn Wheeler)
> wrote:
>
>> IMS kept physical disk addresses internally in data records for
>> related data
>
> Ouch!

I hear that in the original SABRE airline system, the locator
ID for your reservation was actually a disk address.
--
Regards,

John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: vfork history, Multics vs Unix
Posted by cross on Sun, 26 Jan 2025 21:17:11 GMT
View Forum Message <> Reply to Message

In article <slrnvpcjte.18si6.lars@cleo.beagle-ears.com>,
Lars Poulsen <lars@cleo.beagle-ears.com> wrote:
> On 2025-01-26, Dan Cross <cross@spitfire.i.gajendra.net> wrote:
>> "You are not expected to understand this" is such a classic.
>> The fundamental problem was that they, essentially, used a
>> non-local goto to do a coroutine jump, but relied on a function
>> returning from a stack that wasn't its own behaving correctly.
>> Dennis Ritchie explained it here:
>> https://www.bell-labs.com/usr/dmr/www/odd.html
>
> Quoting DMR from there:
> So we tried to explain what was going on. "You
> are not expected to understand this" was intended
> as a remark in the spirit of "This won't be on
> the exam," rather than as an impudent challenge.
>
> The real problem is that we didn't understand what
> was going on either. The savu/retu mechanism for
> doing process exchange was fundamentally broken
> because it depended on switching to a previous
> stack frame and executing function return code in
> a different procedure from the one that saved the
> earlier state. This worked on the PDP-11 because
> its compiler always used the same context-save
> mechanism; with the Interdata compiler, the
> procedure return code differed depending on which
> registers were saved.

If one looks at the 7th Edition Unix sources, one sees that the
`savu`/`retu` mechanism was replaced with `save` and `resume`,
which are effectively `setjmp`/`lonjmp`: these save enough
context that a `longjmp` makes `setjmp` appear to return twice;
once for the initial call, and once again as a result of
`longjmp`. The return value from `setjmp` can be used to
differentiate between the two cases. But note that `longjmp`
itself never returns to its caller.

The difference with `setu` and `retu` was that the former just
squirreled away some register state and a stack pointer,
similarly to what `setjmp` does, and the latter restored that

state including, critically, the stack pointer, but then
resumed execution _in its caller, but not with that caller's
stack_. So on return from _that_ function (that is, whatever
called `retu`), execution would be resumed elsewhere,
immediately after the corresponding call to `setu`.

Hence the bug exposed by the Interdata compiler: since the exit
path from a function on that system depended on whatever
registers were saved by that function (which could vary),
returning from `retu` and then relying on the return from
`retu`'s caller to properly restore state from `savu`'s caller
wasn't reliable. Since function returns on the PDP-11 always
looked the same, this didn't matter there. Since `resume`, like
`longjmp`, doesn't return to its caller, the problem is avoided
entirely in the 7th Ed kernel.

> Very interesting. The modern "portable" version of context switching in
> the RTOSes I have looked at internals for, uses "longjmp()".

There's little reason to do it substantially differently, though
whether it is actually `setjmp`/`longjmp` (as provided via
compiler intrinsics or whatever) is, IMHO, immaterial.

Personally I prefer to implement it myself, as knowing exactly
what is in the context structure, and it's layout, can be
useful.

 - Dan C.

Subject: Re: The history of "multi-programming"
Posted by Anonymous on Sun, 26 Jan 2025 21:20:25 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 26 Jan 2025 14:33:48 -0000 (UTC), Lars Poulsen wrote:

> To reduce the waste of an expensive fast CPU, it was common to redirect
> the slowest peripherals (card read, print and card punch)
> to tape, and use a separate offline device for them, eventually a cheap
> slow processor, which at first might be a plugboard device.

There was a term for this: it was called “spooling”.

Which might, or might not, have stood for “Simultaneous Peripheral
Operation On-Line”. Or maybe “On-Line” should be “Off-Line”.

> With disk drives available, it might be possible to run a secondary
> job stream which swapped jobs in and out as Lawrence describes ...

Remember also that the disparity between disk I/O speeds and main memory speeds was not so great then. Disks (or drums) were still considered “fast” peripherals.

Subject: Re: Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 21:23:52 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 26 Jan 2025 11:05 +0000 (GMT Standard Time), John Dallman wrote:

> In article <vn1apj$2fink$2@dont-email.me>, ldo@nz.invalid (Lawrence
> D'Oliveiro) wrote:
>
>> But note that the trend of falling memory prices was already becoming
>> clear by the 1970s, if not earlier. The earliest batch systems only
>> kept one program in memory at one time, and swapped it out for another
>> one when it went into any kind of I/O wait, to keep the CPU busy;
>
> Not so. That demands a plentiful supply of fast disk or drum space, and
> the earliest batch systems often didn't have disks or drums.

The small, cheap(er) ones would not have them of course. But you had a more powerful (i.e. more expensive) CPU, it behooved you to try to keep that as busy as possible, to recoup your investment.

Remember, the whole point of batch systems was that computer hardware was expensive, while human programmer time (back then) was cheap.

> The solutions to this problem included spooling to tape, and having
> multiple programs resident in memory and switching between them.

But main memory was even more expensive than disks or drums. So swapping would have been cheaper (looking at overall costs of peripherals vs CPU idle time) than multiprogramming.

Later, as main memory costs continued to fall, the equation changed in favour of multiprogramming.

Subject: Re: ancient hacks, was Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 21:24:44 GMT

View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 26 Jan 2025 21:12:04 -0000 (UTC), John Levine wrote:

> I hear that in the original SABRE airline system, the locator ID for
> your reservation was actually a disk address.

Hmmm ... How much validation was done on these IDs?

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 21:33:16 GMT
View Forum Message <> Reply to Message

Originally posted by: Bob Eager

On Sun, 26 Jan 2025 21:02:52 +0000, John Levine wrote:

> According to Dan Cross <cross@spitfire.i.gajendra.net>:
>>> 31-bit addressing is 303x/308x or 390, as I remember things.
>>
>> I remember it as 370-XA; 308x is right. ESA/370 (the infamous 3090) was
>> a few years later.
>
> It was 370/XA, according to the manual I have here.

Later on, Amdahl had it too. I was heavily involved in porting a non-IBM
operating system to the XA architecture, but from Amdahl. Some differences
of course, but the main one was the different I/O architecture. We had
been limited by the 24 bit address space, but 31 bits was fine.

As I recall, it was a 4381.

--
Using UNIX since v6 (1975)...

Use the BIG mirror service in the UK:
http://www.mirrorservice.org

Subject: Re: The evolution of MMUs (was: Multics vs Unix)
Posted by Anonymous on Sun, 26 Jan 2025 21:35:29 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 26 Jan 2025 14:47:30 -0000 (UTC), Lars Poulsen wrote:

> In the original 360 design the company seemed to take great pride in
> having the two teams work together ...

If you read Fred Brooks' "The Mythical Man-Month", that doesn't quite seem to have been the case. The project was divided among multiple teams working on different parts, yes, but the communication between them was somewhat lacking.

For example, a lower-level program loader module was being implemented with all kinds of complicated whiz-bang features that made it quite slow. But the team working on that assumed that the module would not be used all that much (at least by the rest of the system), because of clever design features being worked on elsewhere.

Meanwhile, various other groups working on higher-level components found that their code was quite disk-intensive, but they didn't worry too much because they figured the lower-level loader module would handle all this quite efficiently.

You see the problem?

> ... and that is also reported as having been very much the case in
> VAX/VMS.

Certainly the hardware was designed very much with the needs of the software in mind, e.g. the FFS instruction was used in a clever way in the process dispatcher to find the next highest-priority process that was ready to run -- it searched a 32-bit flag array to find the highest-priority process queue that had something in it (represented by a 1-bit). Then the bit number found was used as an index into the array of process queues with the REMQUE instruction to remove the process from the front of the corresponding queue (doubly-linked list).

Then the actual resumption of process execution took just 2 instructions:

```
    LDPCTX ; load process context
    REI    ; return from exception/interrupt
```

Of course, whether all this translated into speed or not was quite another matter ...

Subject: Re: vfork history, Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 21:44:52 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 26 Jan 2025 14:56:49 -0000 (UTC), Lars Poulsen wrote:

> I.e., you are right that the earlier (11/40 and its 11/35 cousin) were
> purely 18-bits physical addressing, but according to WikiPedia,
> the 11/34 had a 22-bit address space somewhere, though I don't know what
> it did with it. Personally, I only knew the 11/34 as a drop-in
> replacement for the 11/35 systems on which I did my early industrial
> applications.

Looking at the 11/34 User's Guide at Bitsavers, I see only 18-bit physical addressing. The maximum memory expansion is 124K words (the top 4K being reserved for I/O addressing).

The 11/44, on the other hand, has 22-bit memory addresses.

Subject: Re: vfork history, Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 21:48:15 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 26 Jan 2025 15:07:26 -0000 (UTC), Lars Poulsen wrote:

> Quoting DMR:
>   The real problem is that we didn't understand what was going on
>   either. The savu/retu mechanism for doing process exchange was
>   fundamentally broken because it depended on switching to a
>   previous stack frame and executing function return code in a
>   different procedure from the one that saved the earlier state.
>   This worked on the PDP-11 because its compiler always used the
>   same context-save mechanism; with the Interdata compiler, the
>   procedure return code differed depending on which registers were
>   saved.

This shouldn't have been broken. When the current process is being suspended, *all* its registers should have been saved. So when it is later resumed, all of them can be restored.

Subject: Re: Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 21:50:28 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On 25 Jan 2025 21:14:47 -0500, Rich Alderson wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>
>> On 25 Jan 2025 19:49:40 -0500, Rich Alderson wrote:
>
>>> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>>>
>>>> One big change was that the CLI was not some part of the resident
>>>> kernel, but just another user process.
>>>>
>>> The separation of the command processor from the kernel was a feature
>>> of TENEX on the PDP-10 when UNICS was still being written on the
>>> PDP-7.
>>
>> Did the "command processor" have any special privileges or status on
>> those systems?
>>
>> On Unix it did not.
>
> It did not. The program EXEC.EXE was a vanilla an executable as you
> could wish for.

How easy was it to replace it with something else?

Subject: Re: old pharts, Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Sun, 26 Jan 2025 22:24:09 GMT
View Forum Message <> Reply to Message

Bob Eager <news0009@eager.cx> writes:
> Later on, Amdahl had it too. I was heavily involved in porting a non-IBM
> operating system to the XA architecture, but from Amdahl. Some differences
> of course, but the main one was the different I/O architecture. We had
> been limited by the 24 bit address space, but 31 bits was fine.
>
> As I recall, it was a 4381.

Tom Simpson (from Simpson & Crabtree, responsible for HASP spooling), in early 70s had done RASP, a MFT-11 adaption for virtual memory, but with page-mapped filesystem (rather retaining 360 channel programs) ... which IBM wasn't interested in. He leaves IBM for Amdahl and recreates RASP in "clean room" from scratch (IBM sued but court appointed experts were only able to find one or two similar instruction sequences).

Amdahl people like to keep me up on the latest Amdahl gossip after the (SLAC hosted) monthly BAYBUNCH meetings.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Sun, 26 Jan 2025 22:28:35 GMT
View Forum Message <> Reply to Message

John Levine <johnl@taugh.com> writes:
> It was 370/XA, according to the manual I have here.

370 was follow on to 360 ... then increasing the multiprogramming level motivated adding virtual memory quickly/initially to all 370s (and CP67 was able to morph into vm370 for virtual machines and online interactive).

but also relatively quickly, the "Future System" effort appeared and 370 efforts were being killed off (lack of new IBM 370 during FS is credited with giving the 370 system clone makers their market foothold). When FS imploded, there was mad rush to get stuff back into the product pipelines, including kicking off the quick & dirty 3033 (remapping 168 logic to 20% faster chips) and 3081 (leveraging some warmed over FS hardware, designed for 370/xa, a lot of architecture designed for addressing MVS short comings, but pending availability of MVS/XA ran 370 mode). trivia: 370/xa was referred to 811 for the nov1978 publication date of the internal design & architecture documentation.

part of 811 (370/xa) was as DASD I/O increasingly become major system throughput bottleneck, the MVS I/O redrive pathlength (from ending interrupt of previous I/O to start of next queued operation) was several thousand instructions (leaving device idle). I recently mentioned getting to play disk engineer and rewrite I/O supervisor for integrity and availability ... I also cut redrive pathlength to 1-to-2 hundred instructions ... being able to demonstrate 370 redrive very close to dedicated 811 hardware architecture (but then I had to bring the wrath of the MVS organization on my head).

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Sun, 26 Jan 2025 23:53:34 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 26 Jan 2025 12:28:35 -1000, Lynn Wheeler wrote:

> 370 was follow on to 360 ...

What was the motivation behind the branding change? The original “360” was
supposed to indicate 360°, i.e. coverage of the full circle of computer
applications. But this didn’t seem to last more than about a couple of
years.

I recall a comment that the first System/370 machines were not that
different from their System/360 precursors anyway.

And Amdahl added 1 to each nonzero digit of 360 to create its 470, and
then I think later 580, models.

Subject: Re: old pharts, Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Mon, 27 Jan 2025 00:31:11 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:
> What was the motivation behind the branding change? The original “360” was
> supposed to indicate 360°, i.e. coverage of the full circle of computer
> applications. But this didn’t seem to last more than about a couple of
> years.

i always understood some of it turned into 60s, 70s, 80s decades
.... 3090 doesn't fit (3081 kind of fit) but then they did 390 & 9000

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Rich Alderson on Mon, 27 Jan 2025 01:11:02 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:

> On 25 Jan 2025 21:14:47 -0500, Rich Alderson wrote:

>> Lawrence D'Oliveiro <ldo@nz.invalid> writes:

>>> On 25 Jan 2025 19:49:40 -0500, Rich Alderson wrote:

>>>> Lawrence D'Oliveiro <ldo@nz.invalid> writes:

>>>> > One big change was that the CLI was not some part of the resident
>>>> > kernel, but just another user process.

>>>> The separation of the command processor from the kernel was a feature
>>>> of TENEX on the PDP-10 when UNICS was still being written on the
>>>> PDP-7.

>>> Did the "command processor" have any special privileges or status on
>>> those systems?

>>> On Unix it did not.

>> It did not. The program EXEC.EXE was a vanilla an executable as you
>> could wish for.

> How easy was it to replace it with something else?

Pretty damn simple, as long as the replacement understands the I/O model.

For example, Dick Helliwell did the port of the GNU utilities to TOPS-20 for
the Toad-1 30+ years ago. For shits and giggles, he made an account for
himself which pointed to bash.exe as his command processor.

--
Rich Alderson   news@alderson.users.panix.com
      Audendum est, et veritas investiganda; quam etiamsi non assequamur,
   omnino tamen proprius, quam nunc sumus, ad eam perveniemus.
         --Galen

Subject: Re: Multics vs Unix
Posted by Anonymous on Mon, 27 Jan 2025 03:32:58 GMT
View Forum Message <> Reply to Message

Originally posted by: antispam

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Sun, 26 Jan 2025 11:05 +0000 (GMT Standard Time), John Dallman wrote:
>
>> In article <vn1apj$2fink$2@dont-email.me>, ldo@nz.invalid (Lawrence
>> D'Oliveiro) wrote:
>>
>>> But note that the trend of falling memory prices was already becoming
>>> clear by the 1970s, if not earlier. The earliest batch systems only
>>> kept one program in memory at one time, and swapped it out for another
>>> one when it went into any kind of I/O wait, to keep the CPU busy;

>>
>> Not so. That demands a plentiful supply of fast disk or drum space, and
>> the earliest batch systems often didn't have disks or drums.
>
> The small, cheap(er) ones would not have them of course. But you had a
> more powerful (i.e. more expensive) CPU, it behooved you to try to keep
> that as busy as possible, to recoup your investment.

Sure. But if you are doing I/O to disk it does not make much
sense to swap to disk: you wait on seek on one disk, to swap
you need seeks on two other disks (one for swap in, one for
swap out). For tapes too: resonably fast tape unit can read
a block of data in say 30ms, it makes little sense to swap
because new thing can start only after say 30ms (rather
optimistic estimate for early disks) and at that time data
will be available so you can continue with original program.
Even with cards swapping to disk can give you only marginal
gains.

Tapes and cards are seqential media and easy way to gain speed
is multiple buffering (and/or using bigger blocks in case of
tape). Similarly for printers. So swapping to disk makes sense
if you have devices which are slower, especially interative
terminals.

Theoretically it makes sense to swap to drum during other I/O
operations. But I am not aware of any _batch_ system doing this.
OTOH early _interactive_ system were reported to swap out
current user on interactive I/O and swap in next ready user.
Do you have any reference to batch system using swapping (and not
multiprogramming) on program I/O?

Some early systems used drum as main memory and probably some
of them did multiprogramming, but that is different than swapping
to drum: in such system code on drum is directly executable.

> Remember, the whole point of batch systems was that computer hardware was
> expensive, while human programmer time (back then) was cheap.
>
>> The solutions to this problem included spooling to tape, and having
>> multiple programs resident in memory and switching between them.
>
> But main memory was even more expensive than disks or drums. So swapping
> would have been cheaper (looking at overall costs of peripherals vs CPU
> idle time) than multiprogramming.

But you would need rather strange combinations of peripherial to
gain speed using _only_ swapping. And you still need memory for

multiple I/O buffers, in most cases this memory would give more
gain if used for multiple buffering for single program. Once
you have more memory than this you are likely to be able to fit
multiple programs into main memory (in particular significant
part of memory cost of a simple spooler is space for its buffers).

> Later, as main memory costs continued to fall, the equation changed in
> favour of multiprogramming.

--
                              Waldek Hebisch

Subject: Re: Multics vs Unix
Posted by Anonymous on Mon, 27 Jan 2025 05:46:31 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Mon, 27 Jan 2025 03:32:58 -0000 (UTC), Waldek Hebisch wrote:

> But if you are doing I/O to disk it does not make much sense to
> swap to disk ...

I/O would be typically to magtape, paper tape, cards, that kind of thing.
Swapping might be to disks or drums.

> But you would need rather strange combinations of peripherial to gain
> speed using _only_ swapping.

Disks/drums would be cheaper than core memory. The other peripherals would
typically be a lot slower, so the swapping would help increase
utilization.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Mon, 27 Jan 2025 07:59:54 GMT
View Forum Message <> Reply to Message

Originally posted by: Bob Eager

On Sun, 26 Jan 2025 12:24:09 -1000, Lynn Wheeler wrote:

> Bob Eager <news0009@eager.cx> writes:
>> Later on, Amdahl had it too. I was heavily involved in porting a
>> non-IBM operating system to the XA architecture, but from Amdahl. Some
>> differences of course, but the main one was the different I/O

>> architecture. We had been limited by the 24 bit address space, but 31
>> bits was fine.
>>
>> As I recall, it was a 4381.
>
> Tom Simpson (from Simpson & Crabtree, responsible for HASP spooling), in
> early 70s had done RASP, a MFT-11 adaption for virtual memory, but with
> page-mapped filesystem (rather retaining 360 channel programs) ... which
> IBM wasn't interested in. He leaves IBM for Amdahl and recreates RASP in
> "clean room" from scratch (IBM sued but court appointed experts were
> only able to find one or two similar instruction sequences).
>
> Amdahl people like to keep me up on the latest Amdahl gossip after the
> (SLAC hosted) monthly BAYBUNCH meetings.

THat's interesting. In my case, the operating system I was porting was a
university written one, from the ground up.

I remember having a problem testing it on VM/XA. The initial IPL brought
in a large program (a mini version of the full system), and it overwrote
the code that VM loaded to emulate the IPL hardware. I had to load in
portions around that.

--
Using UNIX since v6 (1975)...

Use the BIG mirror service in the UK:
 http://www.mirrorservice.org

Subject: Re: old pharts, Multics vs Unix
Posted by cross on Mon, 27 Jan 2025 12:28:32 GMT
View Forum Message <> Reply to Message

In article <lvosnqF4lq8U7@mid.individual.net>,
Bob Eager <news0009@eager.cx> wrote:
> On Sun, 26 Jan 2025 12:24:09 -1000, Lynn Wheeler wrote:
>
>> Bob Eager <news0009@eager.cx> writes:
>>> Later on, Amdahl had it too. I was heavily involved in porting a
>>> non-IBM operating system to the XA architecture, but from Amdahl. Some
>>> differences of course, but the main one was the different I/O
>>> architecture. We had been limited by the 24 bit address space, but 31
>>> bits was fine.
>>>
>>> As I recall, it was a 4381.

>>
>> Tom Simpson (from Simpson & Crabtree, responsible for HASP spooling), in
>> early 70s had done RASP, a MFT-11 adaption for virtual memory, but with
>> page-mapped filesystem (rather retaining 360 channel programs) ... which
>> IBM wasn't interested in. He leaves IBM for Amdahl and recreates RASP in
>> "clean room" from scratch (IBM sued but court appointed experts were
>> only able to find one or two similar instruction sequences).
>>
>> Amdahl people like to keep me up on the latest Amdahl gossip after the
>> (SLAC hosted) monthly BAYBUNCH meetings.
>
> THat's interesting. In my case, the operating system I was porting was a
> university written one, from the ground up.
>
> I remember having a problem testing it on VM/XA. The initial IPL brought
> in a large program (a mini version of the full system), and it overwrote
> the code that VM loaded to emulate the IPL hardware. I had to load in
> portions around that.

What OS, if I may ask? MTS, maybe?

 - Dan C.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Mon, 27 Jan 2025 12:44:22 GMT
View Forum Message <> Reply to Message

Originally posted by: Bob Eager

On Mon, 27 Jan 2025 12:28:32 +0000, Dan Cross wrote:

> In article <lvosnqF4lq8U7@mid.individual.net>,
> Bob Eager <news0009@eager.cx> wrote:
>> On Sun, 26 Jan 2025 12:24:09 -1000, Lynn Wheeler wrote:
>>
>>> Bob Eager <news0009@eager.cx> writes:
>>>> Later on, Amdahl had it too. I was heavily involved in porting a
>>>> non-IBM operating system to the XA architecture, but from Amdahl.
>>>> Some differences of course, but the main one was the different I/O
>>>> architecture. We had been limited by the 24 bit address space, but 31
>>>> bits was fine.
>>>>
>>>> As I recall, it was a 4381.
>>>
>>> Tom Simpson (from Simpson & Crabtree, responsible for HASP spooling),
>>> in early 70s had done RASP, a MFT-11 adaption for virtual memory, but
>>> with page-mapped filesystem (rather retaining 360 channel programs)

>>> ... which IBM wasn't interested in. He leaves IBM for Amdahl and
>>> recreates RASP in "clean room" from scratch (IBM sued but court
>>> appointed experts were only able to find one or two similar
>>> instruction sequences).
>>>
>>> Amdahl people like to keep me up on the latest Amdahl gossip after the
>>> (SLAC hosted) monthly BAYBUNCH meetings.
>>
>> THat's interesting. In my case, the operating system I was porting was a
>> university written one, from the ground up.
>>
>> I remember having a problem testing it on VM/XA. The initial IPL brought
>> in a large program (a mini version of the full system), and it overwrote
>> the code that VM loaded to emulate the IPL hardware. I had to load in
>> portions around that.
>
> What OS, if I may ask? MTS, maybe?

No, I didn't say because few will have heard of it. There were only a few
instances.

The Edinburgh Multi Access System (EMAS). There isn't much in Wikipedia,
but here it is:

  https://en.wikipedia.org/wiki/Edinburgh_Multiple_Access_Syst em

I have a page about it:|

 https://emas.bobeager.uk

--
Using UNIX since v6 (1975)...

Use the BIG mirror service in the UK:
 http://www.mirrorservice.org

Subject: Re: old pharts, Multics vs Unix
Posted by cross on Mon, 27 Jan 2025 13:07:20 GMT

View Forum Message <> Reply to Message

In article <lvpdd6F4lq8U8@mid.individual.net>,
Bob Eager <news0009@eager.cx> wrote:
> On Mon, 27 Jan 2025 12:28:32 +0000, Dan Cross wrote:
>

>> In article <lvosnqF4lq8U7@mid.individual.net>,
>> Bob Eager <news0009@eager.cx> wrote:
>>> On Sun, 26 Jan 2025 12:24:09 -1000, Lynn Wheeler wrote:
>>>
>>>> Bob Eager <news0009@eager.cx> writes:
>>>> > Later on, Amdahl had it too. I was heavily involved in porting a
>>>> > non-IBM operating system to the XA architecture, but from Amdahl.
>>>> > Some differences of course, but the main one was the different I/O
>>>> > architecture. We had been limited by the 24 bit address space, but 31
>>>> > bits was fine.
>>>> >
>>>> > As I recall, it was a 4381.
>>>>
>>>> Tom Simpson (from Simpson & Crabtree, responsible for HASP spooling),
>>>> in early 70s had done RASP, a MFT-11 adaption for virtual memory, but
>>>> with page-mapped filesystem (rather retaining 360 channel programs)
>>>> ... which IBM wasn't interested in. He leaves IBM for Amdahl and
>>>> recreates RASP in "clean room" from scratch (IBM sued but court
>>>> appointed experts were only able to find one or two similar
>>>> instruction sequences).
>>>>
>>>> Amdahl people like to keep me up on the latest Amdahl gossip after the
>>>> (SLAC hosted) monthly BAYBUNCH meetings.
>>>
>>> THat's interesting. In my case, the operating system I was porting was a
>>> university written one, from the ground up.
>>>
>>> I remember having a problem testing it on VM/XA. The initial IPL brought
>>> in a large program (a mini version of the full system), and it overwrote
>>> the code that VM loaded to emulate the IPL hardware. I had to load in
>>> portions around that.
>>
>> What OS, if I may ask? MTS, maybe?
>
> No, I didn't say because few will have heard of it. There were only a few
> instances.
>
> The Edinburgh Multi Access System (EMAS).

I'm familiar! I take it you're referring to EMAS-3,
specifically? I mostly associate it with ICL machines.

> There isn't much in Wikipedia, but here it is:
>
>   https://en.wikipedia.org/wiki/Edinburgh_Multiple_Access_Syst em
>
> I have a page about it:|
>

> https://emas.bobeager.uk

This is great; thanks for that.

I've been interested in EMAS since reading about it in this
paper (which is also topical and interesting in its own right):
https://www.terzarima.net/doc/taste.pdf

The paper that Forsyth references, "An Experiment in Doing it
Again, But Very Well This Time" should be required reading.
https://www.gtoal.com/history.dcs.ed.ac.uk/archive/docs/Expe riment/again.html

- Dan C.

Subject: Re: old pharts, Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Mon, 27 Jan 2025 17:09:48 GMT
View Forum Message <> Reply to Message

Bob Eager <news0009@eager.cx> writes:
> THat's interesting. In my case, the operating system I was porting was a
> university written one, from the ground up.
>
> I remember having a problem testing it on VM/XA. The initial IPL brought
> in a large program (a mini version of the full system), and it overwrote
> the code that VM loaded to emulate the IPL hardware. I had to load in
> portions around that.

about the same time was "gold" from recent univ. graduate (tried to hire
him at IBM, but no takers) that had ported unix to 370. ("gold" code
name taken from "Au" - "Amdahl Unix")

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Mon, 27 Jan 2025 17:30:47 GMT
View Forum Message <> Reply to Message

cross@spitfire.i.gajendra.net (Dan Cross) writes:
> What OS, if I may ask? MTS, maybe?

during IBM's Future System period (and internal politics killing off 370
efforts, giving clone 370 system makers their market foothold) I had
recently joined IBM and got to continue going to SHARE and visiting
customers.

http://www.jfsowa.com/computer/memo125.htm
https://people.computing.clemson.edu/~mark/fs.html
https://en.wikipedia.org/wiki/IBM_Future_Systems_project

The director of one of the largest (east coast) financial datacenters
liked me to stop by and talk technology. At some point the IBM branch
manager horribly offended the customer and in retaliation, the customer
ordered an Amdahl system (it would be a lone Amdahl in vast sea of blue
systems). This was in period when Amdahl was still primarily selling
into the univ/technical/scientific market and had yet to make a
commercial, "true blue" customer. I was asked to go onsite at the
customer for 6-12months (apparently to help obfuscate why the customer
was ordering an Amdahl system). I talk it over with the customer and
decide to not accept IBM's offer. I was then told the branch manager was
a good sailing buddy of IBM's CEO and if I didn't do this, I could
forget career, promotions, raises.

MTS trivia:
https://en.wikipedia.org/wiki/Michigan_Terminal_System
https://web.archive.org/web/20221216212415/http://archive.mi chigan-terminal-system.org/
https://web.archive.org/web/20050212073808/www.itd.umich.edu /~doc/Digest/0596/feat01.html
https://web.archive.org/web/20050212073808/www.itd.umich.edu /~doc/Digest/0596/feat02.html
https://web.archive.org/web/20050212183905/www.itd.umich.edu /~doc/Digest/0596/feat03.html
http://www.eecis.udel.edu/~mills/gallery/gallery7.html
http://www.eecis.udel.edu/~mills/gallery/gallery8.html

MTS folklore is it started out being scaffolding off MIT Lincoln Labs
(2nd CP67 installation after CSC itself) LLMPS
https://web.archive.org/web/20200926144628/michigan-terminal
-system.org/discussions/anecdotes-comments-observations/8-1s omeinformationaboutllmps

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix
Posted by cross on Mon, 27 Jan 2025 21:29:16 GMT
View Forum Message <> Reply to Message

In article <87cyg8npyb.fsf@localhost>, Lynn Wheeler <lynn@garlic.com> wrote:
> Bob Eager <news0009@eager.cx> writes:
>> THat's interesting. In my case, the operating system I was porting was a
>> university written one, from the ground up.
>>
>> I remember having a problem testing it on VM/XA. The initial IPL brought
>> in a large program (a mini version of the full system), and it overwrote
>> the code that VM loaded to emulate the IPL hardware. I had to load in
>> portions around that.

>
> about the same time was "gold" from recent univ. graduate (tried to hire
> him at IBM, but no takers) that had ported unix to 370. ("gold" code
> name taken from "Au" - "Amdahl Unix")

First name Tom, perhaps?

 - Dan C.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Mon, 27 Jan 2025 21:33:52 GMT
View Forum Message <> Reply to Message

Originally posted by: Bob Eager

On Mon, 27 Jan 2025 13:07:20 +0000, Dan Cross wrote:

> In article <lvpdd6F4lq8U8@mid.individual.net>,
> Bob Eager <news0009@eager.cx> wrote:
>> On Mon, 27 Jan 2025 12:28:32 +0000, Dan Cross wrote:
>>
>>> In article <lvosnqF4lq8U7@mid.individual.net>,
>>> Bob Eager <news0009@eager.cx> wrote:
>>>> On Sun, 26 Jan 2025 12:24:09 -1000, Lynn Wheeler wrote:
>>>>
>>>> > Bob Eager <news0009@eager.cx> writes:
>>>> >> Later on, Amdahl had it too. I was heavily involved in porting a
>>>> >> non-IBM operating system to the XA architecture, but from Amdahl.
>>>> >> Some differences of course, but the main one was the different I/O
>>>> >> architecture. We had been limited by the 24 bit address space, but
>>>> >> 31 bits was fine.
>>>> >>
>>>> >> As I recall, it was a 4381.
>>>> >
>>>> > Tom Simpson (from Simpson & Crabtree, responsible for HASP
>>>> > spooling), in early 70s had done RASP, a MFT-11 adaption for virtual
>>>> > memory, but with page-mapped filesystem (rather retaining 360
>>>> > channel programs) ... which IBM wasn't interested in. He leaves IBM
>>>> > for Amdahl and recreates RASP in "clean room" from scratch (IBM sued
>>>> > but court appointed experts were only able to find one or two
>>>> > similar instruction sequences).
>>>> >
>>>> > Amdahl people like to keep me up on the latest Amdahl gossip after
>>>> > the (SLAC hosted) monthly BAYBUNCH meetings.
>>>>
>>>> THat's interesting. In my case, the operating system I was porting was
>>>> a university written one, from the ground up.

>>>>
>>>> I remember having a problem testing it on VM/XA. The initial IPL
>>>> brought in a large program (a mini version of the full system), and it
>>>> overwrote the code that VM loaded to emulate the IPL hardware. I had
>>>> to load in portions around that.
>>>
>>> What OS, if I may ask? MTS, maybe?
>>
>> No, I didn't say because few will have heard of it. There were only a
>> few instances.
>>
>> The Edinburgh Multi Access System (EMAS).
>
> I'm familiar! I take it you're referring to EMAS-3,
> specifically? I mostly associate it with ICL machines.

Yes. EMAS-3 was easily done to the Amdahl, as it was originally for the
English Electric (later ICL) System 4, an IBM 'clone'. The XA version was
the one I worked on, taking IPL tapes to London three times a week! I
built these on EMAS-2 (ICL 2900), of which I know quite a lot.

>> There isn't much in Wikipedia, but here it is:
>>
>>   https://en.wikipedia.org/wiki/Edinburgh_Multiple_Access_Syst em
>>
>> I have a page about it:|
>>
>> https://emas.bobeager.uk
>
> This is great; thanks for that.
>
> I've been interested in EMAS since reading about it in this paper (which
> is also topical and interesting in its own right):
> https://www.terzarima.net/doc/taste.pdf

Thanks; I'll take a look.

> The paper that Forsyth references, "An Experiment in Doing it Again, But
> Very Well This Time" should be required reading.
>   https://www.gtoal.com/history.dcs.ed.ac.uk/archive/docs/Expe riment/
again.html

Indeed, and that is on EMAS-2, of course. I have that also as a physical
copy of the internal report; see:

 http://www.ancientgeek.org.uk/EMAS/EMAS_Booklets/
An_Experiment_In_Doing_It_Again_But_Very_Well_This_Time.pdf

I gave Graham Toal quite a lot of stuff.
--
Using UNIX since v6 (1975)...

Use the BIG mirror service in the UK:
 http://www.mirrorservice.org

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Mon, 27 Jan 2025 21:38:59 GMT
View Forum Message <> Reply to Message

Originally posted by: Bob Eager

On Mon, 27 Jan 2025 13:07:20 +0000, Dan Cross wrote:

> I've been interested in EMAS since reading about it in this paper (which
> is also topical and interesting in its own right):
> https://www.terzarima.net/doc/taste.pdf

Ah yes, I have that somewhere but didn't recognise it from the URL. It
should be on that web page; I'll fix that.

Incidentally, the last few minutes or so of that video may interest you,
even if you don't look at the rest:

 https://youtu.be/khnu7R3Pffo?si=dRwQ68xSnqWqN1o9&t=3658

--
Using UNIX since v6 (1975)...

Use the BIG mirror service in the UK:
 http://www.mirrorservice.org

Subject: Re: Multics vs Unix
Posted by Peter Flass on Mon, 27 Jan 2025 22:45:14 GMT
View Forum Message <> Reply to Message

Bill Findlay <findlaybill@blueyonder.co.uk> wrote:
> On 23 Jan 2025, Lynn Wheeler wrote
> (in article <87tt9ppd8d.fsf@localhost>):
>
>> cross@spitfire.i.gajendra.net (Dan Cross) writes:
>>> CMS and Unix are rather different, as Lynn has described. But
>>> for the others, I'd say, yes, quite, because all of those things
>>> run at the same protection level in a single address space.

>>> That is the critical part missing in other systems.
>>
>> note: OS/360 used 360 privileged instruction and supervisor/problem mode
>> and for concurrent application program separation with (4bit) storage
>> protection keys (for up to 15 concurrent applications plus
>> kernel/supervisor).
>>
>> This initial move of OS/360 MVT to VS2 was into a single 16mbyte virtual
>> address space (something like running MVT in a CP67 16mbyte virtual
>> machine). However, as systems got larger and more powerful ... they need
>> to go past 15 cap. The VS2/SVS to VS2/MVS was to give each concurrent
>> executing program its own 16mbyte virtual address space.
>>
>> However, the OS/360 heritage was heavily pointer-passing API. In order
>> for the kernel calls to access program parameter list they mapped an
>> 8mbyte image of the MVS kernel into every 16mbyte virtual address space
>> (so kernel call processing was done in the same virtual address space as
>> the caller).
>>
>> VS2/MVS also mapped kernel subsystems into their own seperate 16mbyte
>> virtual address spaces ... so application calls to subsystems,
>> paraemters were now in different address space. For this they created
>> the Common Segment Area ("CSA") that was mapped into every 16mbyte
>> virtual adress space for subsystem API call parameters. The issue was
>> subsystem parameter list space requirement was somewhat proportional to
>> number subsystems and number of concurrently executing programs ... and
>> by the time of 370 3033 processor CSA had morphed into 5-6mbyte "Common
>> System Area" (CSA) leaving only 2-3 mbytes for programs, but was
>> threatening to become 8mbytes (leaving zero for applictions).
>>
>> This was part of POK (high-end 370s) VS2/MVS mad rush to 370/XA
>> architecture, 31-bit addressing, access register, Program Call and
>> Program Return instructions.
>>
>> Program Call was akin to kernel call ... table of all the MVS subsystems
>> and their associated address space pointers. A Program Call would select
>> the specific subsystem entry, move the caller's address space to
>> secondary and load the subsystem's address space as primary. A
>> semi-privileged subsystem would access parameter list in the caller's
>> "secondary address space" (no longer needing "CSA" space). Program
>> Return would move the caller's secondary address space pointer to
>> primary and return (task switches would have to save/restore both the
>> primary and any secondary address space pointers).
>
> It's quite a history of bungling, isn't it?
>

More like kludging. How can I get from x to y?

--
Pete

Subject: Re: Multics vs Unix
Posted by Peter Flass on Mon, 27 Jan 2025 22:45:16 GMT
View Forum Message <> Reply to Message

Lynn Wheeler <lynn@garlic.com> wrote:
> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>> Large corporates did, and do, tend to become unwieldy in their development
>> processes. IBM had a long history of doing innovative research (and
>> patenting the results); but if you looked at their actual product line,
>> very little of this innovation actually seemed to make it into that.
>>
>> One example I recall is SNA, their "Systems Network Architecture". For a
>> long time this was not what we would understand as a "network" at all: it
>> was primarily a way for large, expensive central machines to control
>> remote, cheaper machines at branch offices.
>>
>> IBM didn't discover peer-to-peer networking until the 1980s, years after
>> other companies were already doing it routinely.
>
> In the 70s about same time SNA appeared, my wife was co-author of AWP39,
> "Peer-to=Peer Networking" architecture ... peer-to-peer qualification
> necessary because the communication group had co-opted "network"
> ... joke was that SNA was "not a System", "not a Network", and "not an
> Architecture".
>
> My wife was then con'ed into going to POK (IBM high-end mainframe)
> responsible for "loosely-coupled" (mainframe for "cluster") shared DASD
> architecture. She didn't remain long because 1) periodic battles with
> the SNA/VTAM forces trying to force her into using SNA/VTAM for
> loosely-coupled operation and 2) little uptake, until much later with
> SYSPLEX and Parallel SYSPLEX
> https://en.wikipedia.org/wiki/IBM_Parallel_Sysplex
> except for IMS hot-standby.
>
> Co-worker at science center was responsible for the CP67-based
> (precursor to VM370) wide-area network, that morphs into the corproate
> internal network (larger than arpanet/internet from the beginning until
> sometime mid/late 80s, about the time the SNA-org forced the internal
> network to be converted to SNA/VTAM). Technology also used for the
> corporate sponsored univ BITNET (also for a time larger than
> arpanet/internet): https://en.wikipedia.org/wiki/BITNET
>
> Account by one of the inventors of GML (in 1969) at the science center:

> https://web.archive.org/web/20230402212558/http://www.sgmlso urce.com/history/jasis.htm
> Actually, the law office application was the original motivation for the
> project, something I was allowed to do part-time because of my knowledge
> of the user requirements. My real job was to encourage the staffs of the
> various scientific centers to make use of the CP-67-based Wide Area
> Network that was centered in Cambridge.
>
> ... trivia: a decade later GML morphs into ISO standard SGML and after
> another decade morphs into HTML at CERN and 1st webserver in the
> use is CERN sister location, Stanford SLAC (on their VM370 sstem)
> https://ahro.slac.stanford.edu/wwwslac-exhibit
> https://ahro.slac.stanford.edu/wwwslac-exhibit/early-web-chr
onology-and-documents-1991-1994
>
> Edson (passed Aug2020)
> https://en.wikipedia.org/wiki/Edson_Hendricks
> In June 1975, MIT Professor Jerry Saltzer accompanied Hendricks to
> DARPA, where Hendricks described his innovations to the principal
> scientist, Dr. Vinton Cerf. Later that year in September 15-19 of 75,
> Cerf and Hendricks were the only two delegates from the United States,
> to attend a workshop on Data Communications at the International
> Institute for Applied Systems Analysis, 2361 Laxenburg Austria where
> again, Hendricks spoke publicly about his innovative design which paved
> the way to the Internet as we know it today.
>
> We then transfer out to San Jose Research and in early 80s, I get HSDT,
> T1 and faster computer links, some amount of conflict with SNA forces
> (note in 60s, IBM had 2701 telecommunication controller that supported
> T1 links, however IBM's move to SNA in the mid-70s and associated issues
> seem to have capped links at 56kbits/sec).
>
> trivia: at the time of arpanet/internet cut-over to internetworking
> protocol on 1Jan1983, there were 100 IMPs and approx 255 hosts ... at a
> time when the internal network was rapidly approach 1000 hosts all over
> the world (approval of IMPs somewhat held back arpanet growth; for
> internal network growth it was corporate requirement that all links be
> encrypted ... which could be real problem with various country
> gov. agencies, especially when links cross national
> boundaries). Archived post with list of world-wide corporate locations
> that got one or more hew host networking connections during 1983:
> https://www.garlic.com/~lynn/2006k.html#8
>
> For HSDT, was involved in working with the NSF director and was suppose to
> get $20M to interconnect the NSF Supercomputer centers. Then congress
> cuts the budget, some other things happen and eventually an RFP is
> released (in part based on what we already had running). NSF 28Mar1986
> Preliminary Announcement:
> https://www.garlic.com/~lynn/2002k.html#12

> The OASC has initiated three programs: The Supercomputer Centers Program
> to provide Supercomputer cycles; the New Technologies Program to foster
> new supercomputer software and hardware developments; and the Networking
> Program to build a National Supercomputer Access Network - NSFnet.
>
> IBM internal politics was not allowing us to bid (being blamed for
> online computer conferencing inside IBM likely contributed). The NSF
> director tried to help by writing the company a letter (3Apr1986, NSF
> Director to IBM Chief Scientist and IBM Senior VP and director of
> Research, copying IBM CEO) with support from other gov. agencies ... but
> that just made the internal politics worse (as did claims that what we
> already had operational was at least 5yrs ahead of the winning bid), as
> regional networks connect in, it becomes the NSFNET backbone, precursor
> to modern internet.
>
> One of HSDT first long-haul T1 links was between the IBM Los Gatos lab
> (on the west coast) and Clementi's
> https://en.wikipedia.org/wiki/Enrico_Clementi
> E&S lab in Kingston, NY, that had lots of floating point systems boxes
> that included 40mbyte/sec disk arrays.
> https://en.wikipedia.org/wiki/Floating_Point_Systems
> Cornell University, led by physicist Kenneth G. Wilson, made a
> supercomputer proposal to NSF with IBM to produce a processor array of
> FPS boxes attached to an IBM mainframe with the name lCAP.
>
> SJMerc article about Edson, "IBM'S MISSED OPPORTUNITY WITH THE INTERNET"
> (gone behind paywall but lives free at wayback machine),
> https://web.archive.org/web/20000124004147/http://www1.sjmer
cury.com/svtech/columns/gillmor/docs/dg092499.htm
> Also from wayback machine, some additional (IBM missed) references from
> Ed's website
> https://web.archive.org/web/20000115185349/http://www.edh.ne t/bungle.htm
>
> For awhile I reported to same executive as the person behind AWP164
> (which becomes APPN), needling that he should come work on real
> networking ... becauses the SNA forces would never appreciated what he
> was doing ... the SNA forces veto the announcement of APPN ... and the
> APPN announcement letter was carefully rewritten to not imply and
> relationship between APPN and SNA.
>
> SNA forces had been fiercely fighting off client/server and distributed
> computing and trying to block the announcement of mainframe TCP/IP. When
> that failed, they change their tactic and claim that since they have
> corporate strategic responsibility for everything that crosses
> datacenter walls, it has to be released through them. What ships get
> 44kbyte/sec aggregate using nearly whole 3090 processor. I then do the
> support for RFC1044 and in some turning tests at Cray Research between a
> Cray and a 4341, get sustained 4341 channel I/O throughput, using only a

> modest amount of 4341 processor (something like 500 times improvement in
> bytes moved per instruction executed).
>
> In 1988 we get last product we did at IBM, HA/CMP.
> https://en.wikipedia.org/wiki/IBM_High_Availability_Cluster_ Multiprocessing
> It starts out HA/6000 for the NYTimes to port their newspaper system
> (ATEX) off DEC VAXCluster to RS/6000. I rename it HA/CMP when I start
> doing technical/scientific cluster scale-up with national labs and
> commercial cluster scale-up with RDBMS vendors that have VAXCluster
> support in the same source base with UNIX (Oracle, Sybase, Ingres,
> Informix).
>
> Early Jan1992, have meeting with Oracle CEO where IBM/AWD Hester tells
> Ellison that by mid92 there would be 16-system clusters and by ye92,
> 128-system clusters. Then late Jan1992, cluster scale-up is transferred
> for announce as IBM Supercomputer (for technical/scientific *ONLY*) and
> we are told we can't work on anything with more than four systems (we
> leave IBM a few months later). Note: IBM mainframe DB2 had been
> complaining if we were allowed to go ahead, it would be years ahead of
> them.
>
> trivia: when 1st transferred to IBM SJR, I did some work with Jim Gray
> and Vera Watson on the original SQL/relational, System/R ... and while
> corporation was preoccupied with the next great DBMS ("EAGLE"), we were
> able to do tech transfer to Endicott for release as SQL/DS. Then when
> "EAGLE" implodes there was a request for how fast could System/R be
> ported to MVS ... which is eventually released as "DB2" (originally for
> decision support only).
>

I was never thrilled with APPN. It seemed to me that it was cumbersome to
set up compared to,other networks of the time - Bitnet, TCP/IP or Netware.

--
Pete

Subject: Re: Multics vs Unix
Posted by Peter Flass on Mon, 27 Jan 2025 22:45:17 GMT
View Forum Message <> Reply to Message

Lynn Wheeler <lynn@garlic.com> wrote:
> Lars Poulsen <lars@cleo.beagle-ears.com> writes:
>> Of course, a tight cluster with survivability cannot have a strict
>> hierarchy, since the top position would have to be renegotiated if
>> the "master" node failed.
>

> ... when I was out marketing HA/CMP I coined the termas "disaster
> survivability" and "geographic survivability". The IBM S/88 product
> administrator also started taking us around to their customers ... and
> got me to write a section for the corporate continuous availability
> strategy document (it got pulled when both Rochester/AS400 and
> POK/mainframe complained they couldn't meet the objectives).
> https://www.pcmag.com/encyclopedia/term/system88
>
>

Still the way corporations work today. If your existing product can't meet
the specs, rewrite the specs. That's why large companies get clobbered by
small companies who don't have that problem.

--
Pete

Subject: Re: Multics vs Unix
Posted by Peter Flass on Mon, 27 Jan 2025 22:45:18 GMT
View Forum Message <> Reply to Message

Bill Findlay <findlaybill@blueyonder.co.uk> wrote:
> On 25 Jan 2025, Lawrence D'Oliveiro wrote
> (in article <vn1apj$2fink$2@dont-email.me>):
>>
>> But note that the trend of falling memory prices was already becoming
>> clear by the 1970s, if not earlier. The earliest batch systems only kept
>> one program in memory at one time, and swapped it out for another one when
>> it went into any kind of I/O wait, to keep the CPU busy...
> They did no such thing.
>

Well, early timesharing systems such as PDP-11 systems or TSO.

--
Pete

Subject: Re: Multics vs Unix
Posted by Peter Flass on Mon, 27 Jan 2025 22:45:19 GMT
View Forum Message <> Reply to Message

Rich Alderson <news@alderson.users.panix.com> wrote:
> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>
>> On Wed, 22 Jan 2025 10:58:03 -0000 (UTC), Waldek Hebisch wrote:

>
>>> Clearly Multics was more advanced. I was simply pointing out that the
>>> same "single task for single user" mindset seem to be present in
>>> Multics.
>
>> All the OSes of the time tried to minimize process creation. Unix was the
>> one going against the trend, by its seemingly profligate creation of new
>> processes for doing every single thing.
>
>> One big change was that the CLI was not some part of the resident kernel,
>> but just another user process. Hard to realize this was seen as quite
>> radical at the time.
>
> Once again...
>
> The separation of the command processor from the kernel was a feature of TENEX
> on the PDP-10 when UNICS was still being written on the PDP-7.
>

Multics shell was the inspiration for Unix version.

--
Pete

Subject: Re: Multics vs Unix
Posted by Peter Flass on Mon, 27 Jan 2025 22:45:20 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On 25 Jan 2025 19:46:01 -0500, Rich Alderson wrote:
>
>> But when an executable is wanted, the mechanism is to CFORK% create a
>> process, GET% a memory image of the executable file into *that* process,
>> and SFORK% start the process. The calling program (including the EXEC)
>> can WFORK% wait for process execution to conclude, or move on to other
>> things.
>
> That sounds like a TOPS-20/TENEX thing, rather than TOPS-10.
>
> I don’t think any home-grown DEC OS created new processes to run new
> commands as a matter of course, with the possible exception of RSX-11,
> with its rather convoluted notion of “tasks”.
>

I first encountered the “create a new process” model on Unix, and my
initial reaction was “that has to be extremely stupid and wasteful, who
would want to do that?”

--
Pete

Subject: Re: Multics vs Unix
Posted by Peter Flass on Mon, 27 Jan 2025 22:45:21 GMT
View Forum Message <> Reply to Message

Dan Cross <cross@spitfire.i.gajendra.net> wrote:
> In article <mddwmei4e4c.fsf@panix5.panix.com>,
> Rich Alderson <news@alderson.users.panix.com> wrote:
>> Peter Flass <peter_flass@yahoo.com> writes:
>>
>>> Lynn Wheeler <lynn@garlic.com> wrote:
>>
>>>> A small subset of this was released in VM370R3 as DCSS (but without the
>>>> page-mapped filesystem and w/o the independent address location support)
>>>> ... where the shared segment images were saved in special defined VM370
>>>> disk areas
>>
>>> It was fun mapping the DCSS to specific memory addresses, and making sure
>>> that a program didn't need two mapped to the same address. This, I think,
>>> was S/370, so 31-bit (16MB) address space. Didn't DCSS sizes have to be a
>>> multiple of 64K? It's been a while, so I don't recall the details.
>>
>> The 370, like the 360 before it, was a 24 bit address architecture, for an
>> address space of 16,777,216 locations (16MB).
>>
>> 31-bit addressing is 303x/308x or 390, as I remember things.
>
> I remember it as 370-XA; 308x is right. ESA/370 (the infamous
> 3090) was a few years later.
>
>> (Yes, I've been a PDP-10 guy for more than 45 years, but I was a 36-370 guy for
>> 10 years befreo that.)
>
> I'm glad people are keeping the -10 alive. :-)
>

Whatever happened to the -10 (and other hardware);from the LCM?

--
Pete

Subject: Re: Multics vs Unix

Posted by Peter Flass on Mon, 27 Jan 2025 22:45:22 GMT
View Forum Message <> Reply to Message

Lynn Wheeler <lynn@garlic.com> wrote:
> Peter Flass <peter_flass@yahoo.com> writes:
>> It was fun mapping the DCSS to specific memory addresses, and making sure
>> that a program didn't need two mapped to the same address. This, I think,
>> was S/370, so 31-bit (16MB) address space. Didn't DCSS sizes have to be a
>> multiple of 64K? It's been a while, so I don't recall the details.
>
>
> 370 virtual memory architecture allowed for two page sizes, 2k and 4k
> ... and two segment sizes, 64k and 1mbyte.
>
> for vm370 CMS, used 4k page size and 64k (16 4k pages) segment size.
>
> as i've periodic mentioned, DCSS ("discontiguous shared segments") was
> small piece of my CMS page mapped filesystem ... which included being
> able to specify loading file with pieces as shared segment(s).
>
> 370 had two level virtual memory table ... a segment table where each
> entry pointed to that (segment's) page table (for vm370, 16 4k page
> entries). in the case of a shared segment ... each individual virtual
> memory segment table entry point to a common, shared page table.
>
> in the case of my CMS page mapped filesystem, filesystem control
> information specified mapping to individual pages ... or for a
> mapping(s) that specified 16 4k pages to be mapped as a "shared
> segment".
>
> Since the CMS page mapped filesystem wasn't picked up for VM370R3, all
> the control information had to placed somewhere. The VM370 module name
> was (kernel assembled) DMKSNT which had SNT entires information
> specifying the number of pages, virtual addresses of the pages, which
> portion of the virtual pages to be treated as shared segments, physical
> disk locations of the saved pages, etc.
>
> A sysadm or sysprog ... would typically load application into their
> virtual memory and issue the kernel (privileged) "SAVESYS" command
> specifying a SNT entry, the pages from their virtual memory would then
> be written to the specified disk locations. Then users could issue the
> "LOADSYS" command specifying a SNT entry, which would reverse the
> process, mapping their virtual memory to the SNT entry specified pages
> (and in case of shared segment, their corresponding segment table
> entry/entries mapped to the shared page table(s)).
>
> If wanting to change, add or delete a DMKSNT SNT entry, edit the
> assemble source, assemble the changed DMKSNT file, rebuild the kernel
> (w/changed DMKSNT) and re-ipl (note SNT entries were global system

> wide).
>

At the point I got into it, I don’t recall any messing with the nucleus
being required. Of course the disadvantage was that, like windows shared
libraries, DCSS could only need loaded at the address they occupied when
saved.

--
Pete

Subject: Re: Multics vs Unix
Posted by Peter Flass on Mon, 27 Jan 2025 22:45:23 GMT

View Forum Message <> Reply to Message

John Dallman <jgd@cix.co.uk> wrote:
> In article <87wmehmmqd.fsf@localhost>, lynn@garlic.com (Lynn Wheeler)
> wrote:
>
>> IMS kept physical disk addresses internally in data records for
>> related data
>
> Ouch!

This was pretty standard. IMS was a hierarchical system, but CODASYL
specified a network model like IDMS. All pointers were kept in the records.
GE had a system that implemented this, and was the model for CODASYL.

>
> John
>

--
Pete

Subject: Re: ancient hacks, was Multics vs Unix
Posted by Peter Flass on Mon, 27 Jan 2025 22:45:24 GMT

View Forum Message <> Reply to Message

John Levine <johnl@taugh.com> wrote:
> According to John Dallman <jgd@cix.co.uk>:
>> In article <87wmehmmqd.fsf@localhost>, lynn@garlic.com (Lynn Wheeler)
>> wrote:

>>
>>> IMS kept physical disk addresses internally in data records for
>>> related data
>>
>> Ouch!
>
> I hear that in the original SABRE airline system, the locator
> ID for your reservation was actually a disk address.

IBM’s DOS DBOMP database did the same, which made data unmovable.

--
Pete

Subject: Re: Multics vs Unix
Posted by Anonymous on Mon, 27 Jan 2025 22:53:58 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Mon, 27 Jan 2025 15:45:17 -0700, Peter Flass wrote:

> That’s why large companies get clobbered by small companies who don’t
> have that problem.

You would think so. But too often, the large companies can counter with
marketing campaigns to tell everyone how wonderful they are.

IBM was the number-one master at this (before it was eclipsed by
Microsoft). People figured, they were so big, they must be doing something
right, even if their products were frequently more expensive and less
capable than the opposition.

This can also be put down to the “market for lemons” effect.

Subject: Re: Multics vs Unix
Posted by Anonymous on Mon, 27 Jan 2025 22:57:15 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Mon, 27 Jan 2025 15:45:18 -0700, Peter Flass wrote:

> Well, early timesharing systems such as PDP-11 systems or TSO.

Swapping was very common in those 1970s systems and earlier.

What happened to change things was 1) larger and cheaper RAM configurations, and 2) worsening of the disparity in speeds between RAM and backing store. So swapping was itself incurring increasing performance costs.

Subject: Re: Multics vs Unix
Posted by Anonymous on Mon, 27 Jan 2025 23:02:27 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Mon, 27 Jan 2025 15:45:20 -0700, Peter Flass wrote:

> I first encountered the “create a new process” model on Unix, and my
> initial reaction was “that has to be extremely stupid and wasteful, who
> would want to do that?”

Unix was, I am pretty sure, seen as a big and slow OS compared to the proprietary competition in its early days. Being a research project, performance was (at least initially) not a high concern.

However, the abstractions it offered turned out to be very useful. Like doing away with the traditional record blocking/deblocking layer, and have the kernel itself offer up the abstraction of a file as a stream of n

size”. And yes, creating lots of processes at the drop of a hat, which turned out to be quite an elegant way of performing many tasks.

In short, it succeeded because it made its users more productive than the alternative platforms. And that continues to today, though with Unix being supplanted by Linux.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Mon, 27 Jan 2025 23:04:18 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Mon, 27 Jan 2025 07:09:48 -1000, Lynn Wheeler wrote:

> about the same time was "gold" from recent univ. graduate (tried to hire
> him at IBM, but no takers) that had ported unix to 370. ("gold" code
> name taken from "Au" - "Amdahl Unix")

Was it running on bare metal or under the VM hypervisor?

Because I believe Linux on IBM mainframes to this day runs as a VM, not on bare metal. Not sure why.

Subject: Re: Multics vs Unix
Posted by Anonymous on Mon, 27 Jan 2025 23:05:35 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Mon, 27 Jan 2025 15:45:19 -0700, Peter Flass wrote:

> Multics shell was the inspiration for Unix version.

No, because MULTICS, for all its innovations, never had the concept of a command processor running as a separate user process.

Subject: Re: Multics vs Unix
Posted by Anonymous on Mon, 27 Jan 2025 23:58:27 GMT
View Forum Message <> Reply to Message

Originally posted by: Andy Walker

On 27/01/2025 22:45, Peter Flass wrote:
> I first encountered the “create a new process” model on Unix, and my
> initial reaction was “that has to be extremely stupid and wasteful, who
> would want to do that?”

So did I, and my initial reaction was "Thank goodness we don't have to write 20 or 100 lines of job control[*] just to compile and run a ten-line program" [which I guess answers your question].

____

* Admittedly largely packaged up into macros kindly written by our computer centre, who thereby kept tight control over what mere users were allowed to do.

--
Andy Walker, Nottingham.
Andy's music pages: www.cuboid.me.uk/andy/Music
Composer of the day: www.cuboid.me.uk/andy/Music/Composers/Godfrey

---

Subject: Re: old pharts, Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Tue, 28 Jan 2025 00:27:42 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:
> Was it running on bare metal or under the VM hypervisor?
>
> Because I believe Linux on IBM mainframes to this day runs as a VM, not on
> bare metal. Not sure why.

80s claim that both IBM AIX/370 (from UCLA Locus) and Amdahl's GOLD ....
ran under vm370 ... because field hardware support CEs demanded
mainframe EREP .... to add mainframe EREP to them was several times
larger effort than just getting them to run on mainframe (so they ran
under VM370, relying on VM370 to provide the mainframe EREP).

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Tue, 28 Jan 2025 00:55:58 GMT
View Forum Message <> Reply to Message

Peter Flass <peter_flass@yahoo.com> writes:
> At the point I got into it, I don’t recall any messing with the nucleus
> being required. Of course the disadvantage was that, like windows shared
> libraries, DCSS could only need loaded at the address they occupied when
> saved.

at some point between vm370 and now, they converted DCSS kernel DMKSNT
savesys and loadsys to use the spool file system.

the os/360 convention/heritage was executable images had address
constants and the executable images had RLD information appended, which
could turn all address constants into fixed values at the point of
bringing the (linkedit/loader loading) executable images into memory.

Original/Early CMS implementation would "save" memory image of a loaded
executable image as "MODULE" filetype. Since a "MODULE" filetype no
longer required any storage modifications ... it could be R/O shared
across multiple virtual address spaces ... but at same fixed address
location across all virtual address spaces.

I would do some hacks of os/360 convention software to change all the
"relative" adcons to "fixed" displacements ... which would be added to
virtual address space specific location (sort of simulating the TSS/360
convention of location independent code) ... and enabling the

specification of location independent option ... where it could use
location other than possible default.

For non-shared R/O, an image could execute store operations into its own
image (both CMS "MODULE" filetype) and DCSS savesys & loadsys. An early
variation was booting MVT into a virtual machine ... and stopping it
near the end of MVT IPL and doing a savesys (using savesys/loadsys sort
of like image checkpoint) for a fast reboot (akin to some of the current
PC fast container bringups).

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Tue, 28 Jan 2025 01:21:39 GMT
View Forum Message <> Reply to Message

Lynn Wheeler <lynn@garlic.com> writes:
> 80s claim that both IBM AIX/370 (from UCLA Locus) and Amdahl's GOLD ....
> ran under vm370 ... because field hardware support CEs demanded
> mainframe EREP .... to add mainframe EREP to them was several times
> larger effort than just getting them to run on mainframe (so they ran
> under VM370, relying on VM370 to provide the mainframe EREP).

in gossiping with Amdahl people I mentioned IBM had special project for
AT&T where IBM had done a strip down of TSS/370 kernel called SSUP
(which included mainframe erep) which higher level part of UNIX kernel
was being layered on top ... besides effectively adding mainframe EREP
(more device support, etc) to UNIX kernel (for running on bare hardware)
it also got mainframe multiprocessor support.

I ask if Amdahl might consider doing something with GOLD and possibly
Simpson's Amdahl RASP

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix
Posted by cross on Tue, 28 Jan 2025 02:10:27 GMT
View Forum Message <> Reply to Message

In article <lvqcnjF4lq8U10@mid.individual.net>,
Bob Eager <news0009@eager.cx> wrote:
> On Mon, 27 Jan 2025 13:07:20 +0000, Dan Cross wrote:
>

>> I've been interested in EMAS since reading about it in this paper (which
>> is also topical and interesting in its own right):
>> https://www.terzarima.net/doc/taste.pdf
>
> Ah yes, I have that somewhere but didn't recognise it from the URL. It
> should be on that web page; I'll fix that.
>
> Incidentally, the last few minutes or so of that video may interest you,
> even if you don't look at the rest:
>
> https://youtu.be/khnu7R3Pffo?si=dRwQ68xSnqWqN1o9&t=3658

Thank you; I will defnitely check this out.

 - Dan C.

Subject: Re: swapping, was Multics vs Unix
Posted by John Levine on Tue, 28 Jan 2025 03:20:09 GMT
View Forum Message <> Reply to Message

According to Peter Flass <peter_flass@yahoo.com>:
>>> But note that the trend of falling memory prices was already becoming
>>> clear by the 1970s, if not earlier. The earliest batch systems only kept
>>> one program in memory at one time, and swapped it out for another one when
>>> it went into any kind of I/O wait, to keep the CPU busy...
>> They did no such thing.
>
> Well, early timesharing systems such as PDP-11 systems or TSO.

Timesharing systems all swapped because programs spent most of their
time waiting for people at terminals. That meant delays of at least
seconds so it made sense to spend a fraction of a second swapping to
disk.

Genrally speaking, batch systems only waited for tapes and disks. Swapping
wouldn't make sense because the swap disk would be no faster than the
device a program was waiting for. Indeed it might be the same device.

Starting in the 1960s there were plenty of batch systems that ran multiple
programs, switching back and forth when the programs were I/O bound, but the
programs were all in memory so it was just context switching, no extra I/O. Once
virtual memory and paging arrived, we can argue about whether the I/O for paging
was like swapping, although it is my strong impression that if a system's memory
was so overcommitted that it did a lot of paging, it was unlikely to get much
useful work done.

--

Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: swapping, was Multics vs Unix
Posted by Anonymous on Tue, 28 Jan 2025 03:23:19 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 28 Jan 2025 03:20:09 -0000 (UTC), John Levine wrote:

> Genrally speaking, batch systems only waited for tapes and disks.
> Swapping wouldn't make sense because the swap disk would be no faster
> than the device a program was waiting for. Indeed it might be the same
> device.

However ...

> Starting in the 1960s there were plenty of batch systems that ran
> multiple programs, switching back and forth when the programs were I/O
> bound, but the programs were all in memory so it was just context
> switching, no extra I/O.

Batch systems were older than this, though.

Subject: Re: Multics vs Unix
Posted by Anonymous on Tue, 28 Jan 2025 03:34:17 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On 25 Jan 2025 21:14:47 -0500, Rich Alderson wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>
>> On 25 Jan 2025 19:49:40 -0500, Rich Alderson wrote:
>
>>> The separation of the command processor from the kernel was a feature
>>> of TENEX on the PDP-10 when UNICS was still being written on the
>>> PDP-7.
>
>> Did the "command processor" have any special privileges or status on
>> those systems?
>

>> On Unix it did not.
>
> It did not. The program EXEC.EXE was a vanilla an executable as you
> could wish for.

This article <https://gunkies.org/wiki/TENEX> says that TENEX was not
really innovative at all: it was basically a port of the OS for the SDS
940 that was created as part of “Project Genie” at UC Berkeley.

Subject: LCM+L PDP-10 systems [was Re: Multics vs Unix]
Posted by Rich Alderson on Tue, 28 Jan 2025 20:53:48 GMT
View Forum Message <> Reply to Message

Peter Flass <peter_flass@yahoo.com> writes:

> Dan Cross <cross@spitfire.i.gajendra.net> wrote:
>> In article <mddwmei4e4c.fsf@panix5.panix.com>,
>> Rich Alderson <news@alderson.users.panix.com> wrote:

>>> (Yes, I've been a PDP-10 guy for more than 45 years, but I was a 36-370 guy
>>> for 10 years befreo that.)

>> I'm glad people are keeping the -10 alive. :-)

> Whatever happened to the -10 (and other hardware);from the LCM?

Several of the high-end items were sold at auction by Christie's.

As much of the PDP-10 hardware as could be acquired was rescued by another
Seattle computer museum, ICM.org

I have nothing to do with either, of course.

--
Rich Alderson   news@alderson.users.panix.com
      Audendum est, et veritas investiganda; quam etiamsi non assequamur,
   omnino tamen proprius, quam nunc sumus, ad eam perveniemus.
         --Galen

Subject: Re: LCM+L PDP-10 systems [was Re: Multics vs Unix]
Posted by scott on Tue, 28 Jan 2025 21:05:18 GMT
View Forum Message <> Reply to Message

Rich Alderson <news@alderson.users.panix.com> writes:
> Peter Flass <peter_flass@yahoo.com> writes:

>
>> Dan Cross <cross@spitfire.i.gajendra.net> wrote:
>>> In article <mddwmei4e4c.fsf@panix5.panix.com>,
>>> Rich Alderson <news@alderson.users.panix.com> wrote:
>
>>>> (Yes, I've been a PDP-10 guy for more than 45 years, but I was a 36-370 guy
>>>> for 10 years befreo that.)
>
>>> I'm glad people are keeping the -10 alive. :-)
>
>> Whatever happened to the -10 (and other hardware);from the LCM?
>
> Several of the high-end items were sold at auction by Christie's.
>
> As much of the PDP-10 hardware as could be acquired was rescued by another
> Seattle computer museum, ICM.org
>
> I have nothing to do with either, of course.

Have you heard anything regarding the disposition of the V380?

Subject: Re: old pharts, Multics vs Unix
Posted by Peter Flass on Tue, 28 Jan 2025 23:45:22 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Mon, 27 Jan 2025 07:09:48 -1000, Lynn Wheeler wrote:
>
>> about the same time was "gold" from recent univ. graduate (tried to hire
>> him at IBM, but no takers) that had ported unix to 370. ("gold" code
>> name taken from "Au" - "Amdahl Unix")
>
> Was it running on bare metal or under the VM hypervisor?
>
> Because I believe Linux on IBM mainframes to this day runs as a VM, not on
> bare metal. Not sure why.
>

RAS. Lots of error reporting and recovery would otherwise have to be
duplicated. Technically it’s not VM, but an LPAR (I believe) which is part
of vm in microcode.

--
Pete

Subject: Re: Multics vs Unix
Posted by Peter Flass on Tue, 28 Jan 2025 23:45:24 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Mon, 27 Jan 2025 15:45:19 -0700, Peter Flass wrote:
>
>> Multics shell was the inspiration for Unix version.
>
> No, because MULTICS, for all its innovations, never had the concept of a
> command processor running as a separate user process.
>

https://multicians.org/shell.html

--
Pete

Subject: Re: Multics vs Unix
Posted by Peter Flass on Tue, 28 Jan 2025 23:48:27 GMT
View Forum Message <> Reply to Message

John Dallman <jgd@cix.co.uk> wrote:
> In article <87wmehmmqd.fsf@localhost>, lynn@garlic.com (Lynn Wheeler)
> wrote:
>
>> IMS kept physical disk addresses internally in data records for
>> related data
>
> Ouch!

This was pretty standard. IMS was a hierarchical system, but CODASYL
specified a network model like IDMS. All pointers were kept in the records.
GE had a system that implemented this, and was the model for CODASYL.
>
> John
>

[I don’t know what happened with this post. NewsTap had it stuck in the
outbox]

--
Pete

--
Pete

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Wed, 29 Jan 2025 01:13:28 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 28 Jan 2025 16:45:22 -0700, Peter Flass wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>
>> On Mon, 27 Jan 2025 07:09:48 -1000, Lynn Wheeler wrote:
>>
>>> about the same time was "gold" from recent univ. graduate (tried to
>>> hire him at IBM, but no takers) that had ported unix to 370. ("gold"
>>> code name taken from "Au" - "Amdahl Unix")
>>
>> Was it running on bare metal or under the VM hypervisor?
>>
>> Because I believe Linux on IBM mainframes to this day runs as a VM, not
>> on bare metal. Not sure why.
>>
> RAS. Lots of error reporting and recovery would otherwise have to be
> duplicated.

Look at it the other way: Linux already has extensive mechanisms for doing
exactly this sort of thing, which work across all its different platforms.

So the question becomes: would someone running Linux on an IBM mainframe
want to make a special case of that installation, or would they rather be
able to manage it with the same mechanisms as all their other Linux
installations?

Subject: Re: Multics vs Unix
Posted by Anonymous on Wed, 29 Jan 2025 01:14:34 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 28 Jan 2025 16:45:24 -0700, Peter Flass wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>
>> On Mon, 27 Jan 2025 15:45:19 -0700, Peter Flass wrote:
>>
>>> Multics shell was the inspiration for Unix version.
>>
>> No, because MULTICS, for all its innovations, never had the concept of

>> a command processor running as a separate user process.
>>
> https://multicians.org/shell.html

You got misled by the term “shell“, didn’t you?

A MULTICS user normally did everything in a single process.

Subject: Re: old pharts, Multics vs Unix
Posted by John Levine on Wed, 29 Jan 2025 03:01:54 GMT
View Forum Message <> Reply to Message

According to Peter Flass <peter_flass@yahoo.com>:
>>> about the same time was "gold" from recent univ. graduate (tried to hire
>>> him at IBM, but no takers) that had ported unix to 370. ("gold" code
>>> name taken from "Au" - "Amdahl Unix")
>>
>> Was it running on bare metal or under the VM hypervisor?
>>
>> Because I believe Linux on IBM mainframes to this day runs as a VM, not on
>> bare metal. Not sure why.
>
> RAS. Lots of error reporting and recovery would otherwise have to be
> duplicated. Technically it’s not VM, but an LPAR (I believe) which is part
> of vm in microcode.

Does anything run on a bare zSeries machine? It is my impression that everything
is in an LPAR, partly for the RAS reason you mention, partly for flexibility so
one can borrow part of the system for testing or a short term project.

--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: The history of "multi-programming"
Posted by Bill Findlay on Wed, 29 Jan 2025 13:33:42 GMT
View Forum Message <> Reply to Message

On 26 Jan 2025, Lars Poulsen wrote
(in article<slrnvpchuc.18si6.lars@cleo.beagle-ears.com>):

> On 25 Jan 2025, Lawrence D'Oliveiro wrote (in article
> <vn1apj$2fink$2@dont-email.me>):
>>>> > But note that the trend of falling memory prices was already

>>>> > becoming clear by the 1970s, if not earlier. The earliest batch
>>>> > systems only kept one program in memory at one time, and swapped it
>>>> > out for another one when it went into any kind of I/O wait, to keep
>>>> > the CPU busy; later on, when memory became large enough (and cheap
>>>> > enough), it made sense to keep multiple programs resident at once
>>>> > ("multiprogramming", this was called), to allow the scheduling
>>>> > switches to happen more quickly.
>
> On Sun, 26 Jan 2025 00:05:16 +0000, Bill Findlay wrote:
>>>> They did no such thing.
>
> On 26 Jan 2025, Lawrence D'Oliveiro wrote
> (in article <vn44ht$37klv$2@dont-email.me>):
>>> They didn´t have enough RAM to do anything else.
>>> That´s why a special term
>>> had to be invented for the new feature.
>
> On 2025-01-26, Bill Findlay <findlaybill@blueyonder.co.uk> wrote:
>> Nonsense.
>
> Be nice, be respectful.

Do try not to be patronising, Lars, my good fellow.
It would have made no sense to respond to an initial IO
waitby incurring a further 4 IO waits: swap out/in/out/in
since:

(a) the swaps out would not be possible while the
initial IO was still active in the initial job's core

(b) those systems did not have devices capable of swapping
the whole of core in significantly less time time than
the initial IO would take to finish.

So: literally, nonsense.

--
Bill Findlay

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Wed, 29 Jan 2025 15:02:14 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:

> On Mon, 27 Jan 2025 07:09:48 -1000, Lynn Wheeler wrote:
>

>> about the same time was "gold" from recent univ. graduate (tried to hire
>> him at IBM, but no takers) that had ported unix to 370. ("gold" code
>> name taken from "Au" - "Amdahl Unix")
>
> Was it running on bare metal or under the VM hypervisor?
>
> Because I believe Linux on IBM mainframes to this day runs as a VM, not on
> bare metal. Not sure why.

When working on MVS and needing Unix, it makes the most sense to use
z/OS Unix. Unix runs in a TSO type environment providing a very
functional Unix environment.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Wed, 29 Jan 2025 17:34:05 GMT
View Forum Message <> Reply to Message

Dan Espen <dan1espen@gmail.com> writes:
> When working on MVS and needing Unix, it makes the most sense to use
> z/OS Unix. Unix runs in a TSO type environment providing a very
> functional Unix environment.

Late 80s, a IBM senior disk enginner got a talk scheduled at annual,
world-wide, internal communication group conference, supposedly on 3174
performance, but opened the talk with the statement that the
communication group was going to be responsible for the demise of of
disk division. The disk division was seeing data fleeing datacenters
with a drop in disk sales to more distributed computing friendly
platforms. They came up with a number of solutions that were all being
vetoed by the communication froup (with their corporate strategic
ownership of overything that crossed the datacenter walls, fiercely
fighting off client/server and distributed computing).

The disk divison executive responsible for software was coming up with
some work-arounds to the communication group; including investing in
distributed computing startups that would use IBM disks, he would
periodically ask us to visit his investments to see if we could offer
some help). He also paid for development of (UNIX) Posix implementation
for MVS (software, wasn't IBM physical hardware transmission product).

a couple years later, IBM has one of the largest losses in the history
of US corporations (wasn't just disks, impacting whole mainframe
datacenter market) and was being re-orged into the 13 "baby blues"
(take-off on AT&T "baby bells" breakup a decade earlier)

https://web.archive.org/web/20101120231857/http://www.time.c
om/time/magazine/article/0,9171,977353,00.html
https://content.time.com/time/subscriber/article/0,33009,977 353-1,00.html

We had already left IBM but get a call from the bowels of Armonk
(hdqtrs) asking if we could help with the breakup. Before we get
started, the board brings in the former AMEX president as CEO to try and
save the company, who (somewhat) reverses the breakup (but it isn't long
before the disk division is gone).

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Wed, 29 Jan 2025 18:00:10 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

Dan Espen <dan1espen@gmail.com> writes:
>> When working on MVS and needing Unix, it makes the most sense to use
>> z/OS Unix. Unix runs in a TSO type environment providing a very
>> functional Unix environment.

On 2025-01-29, Lynn Wheeler <lynn@garlic.com> wrote:
> The disk divison executive responsible for software was coming up with
> some work-arounds to the communication group; including investing in
> distributed computing startups that would use IBM disks, he would
> periodically ask us to visit his investments to see if we could offer
> some help). He also paid for development of (UNIX) Posix implementation
> for MVS (software, wasn't IBM physical hardware transmission product).

What kinds of /unix/linux/aix are in common use in IBM mainframe
environments today?

I know of z/Linux, which runs on z-series processors (including cheaper
ones cripped to NOT be able to run z/OS). I think there is also a
p-series unix, running on the POWER version of the processors. I wonder
if that is A/IX based?

But what is this z/OS Unix described above?

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Wed, 29 Jan 2025 21:05:43 GMT
View Forum Message <> Reply to Message

Lars Poulsen <lars@cleo.beagle-ears.com> writes:

> Dan Espen <dan1espen@gmail.com> writes:
>>> When working on MVS and needing Unix, it makes the most sense to use
>>> z/OS Unix. Unix runs in a TSO type environment providing a very
>>> functional Unix environment.
>
> On 2025-01-29, Lynn Wheeler <lynn@garlic.com> wrote:
>> The disk divison executive responsible for software was coming up with
>> some work-arounds to the communication group; including investing in
>> distributed computing startups that would use IBM disks, he would
>> periodically ask us to visit his investments to see if we could offer
>> some help). He also paid for development of (UNIX) Posix implementation
>> for MVS (software, wasn't IBM physical hardware transmission product).
>
> What kinds of /unix/linux/aix are in common use in IBM mainframe
> environments today?
>
> I know of z/Linux, which runs on z-series processors (including cheaper
> ones cripped to NOT be able to run z/OS). I think there is also a
> p-series unix, running on the POWER version of the processors. I wonder
> if that is A/IX based?
>
> But what is this z/OS Unix described above?

Sounds to me like you are asking Lynn, but I'll throw   in my 2 cents.

I don't know of anyone using z/Linux.
Most mainframes are heavily committed to z/OS or z/DOS.

Z/OS Unix is a standard part of MVS.
You can run z/OS Unix commands under ISPF just like you can run TSO commands.
Just like invoking TSO from batch, you can also run z/OS Unix from batch jobs.

z/OS Unix is pretty much a full implementation of Unix.

In my work experience, I made some use of z/OS Unix for software
development. In general I found it to be more convenient to do Unix stuff
on my Linux desktop. Also batch TSO (or REXX) gives you just as much
function as Unix so I found z/OS Unix only slightly useful.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Wed, 29 Jan 2025 23:01:07 GMT

View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Wed, 29 Jan 2025 16:05:43 -0500, Dan Espen wrote:

> z/OS Unix is pretty much a full implementation of Unix.

Sounds like an emulation layer, like DG’s MV/UX and EUNICE for VMS of old.

I remember, I think it was the Perl build script, if it detected you were
running on a proper *nix system, the message would come up

    Congratulations! You’re not running EUNICE!

Basically, these emulation layers on top of proprietary OSes were
inevitably crap.

And yes, I guess WSL1 for Windows falls in the same bin.

Subject: Re: LCM+L PDP-10 systems [was Re: Multics vs Unix]
Posted by Rich Alderson on Thu, 30 Jan 2025 00:11:46 GMT

View Forum Message <> Reply to Message

scott@slp53.sl.home (Scott Lurndal) writes:

> Rich Alderson <news@alderson.users.panix.com> writes:
>> Peter Flass <peter_flass@yahoo.com> writes:

>>> Dan Cross <cross@spitfire.i.gajendra.net> wrote:
>>>> In article <mddwmei4e4c.fsf@panix5.panix.com>,
>>>> Rich Alderson <news@alderson.users.panix.com> wrote:

>>>> > (Yes, I've been a PDP-10 guy for more than 45 years, but I was a 36-370
>>>> > guy for 10 years befreo that.)

>>>> I'm glad people are keeping the -10 alive. :-)

>>> Whatever happened to the -10 (and other hardware);from the LCM?

>> Several of the high-end items were sold at auction by Christie's.

>> As much of the PDP-10 hardware as could be acquired was rescued by another
>> Seattle computer museum, ICM.org

>> I have nothing to do with either, of course.

> Have you heard anything regarding the disposition of the V380?

I have not. All of the engineering staff were laid off as of 1 July 2020, and
the two people who took over dealing with the remains of the museum (an
archivist and a manager) were explicitly forbidden to talk to any of us about
what was happening.

--
Rich Alderson   news@alderson.users.panix.com
      Audendum est, et veritas investiganda; quam etiamsi non assequamur,
   omnino tamen proprius, quam nunc sumus, ad eam perveniemus.
         --Galen

Subject: Re: Multics vs Unix
Posted by Anonymous on Thu, 30 Jan 2025 01:31:01 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Wed, 29 Jan 2025 01:14:34 -0000 (UTC), I wrote:

> On Tue, 28 Jan 2025 16:45:24 -0700, Peter Flass wrote:
>
>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>
>>> On Mon, 27 Jan 2025 15:45:19 -0700, Peter Flass wrote:
>>>
>>>> Multics shell was the inspiration for Unix version.
>>>
>>> No, because MULTICS, for all its innovations, never had the concept of
>>> a command processor running as a separate user process.
>>>
>> https://multicians.org/shell.html
>
> You got misled by the term “shell“, didn’t you?
>
> A MULTICS user normally did everything in a single process.

Look at section 1.6.1 here <https://multicians.org/exec-env.html>:

    1.6.1 How the Command processor uses processes

    The cost of creating a Multics process is substantial, and this
    led us to a design in which each user gets one process that is
    used to run many commands, as opposed to the Unix design where
    users get a fresh process for each command. The original plan for
    each user's Multics environment was that every logged in user

would have three processes: an overseer process, a worker process,
and a utility process that (I think) handled I/O and allowed
debugging. The three process environment was never run; by the
time of the Phase One milestone in 1967, and from then on, each
Multics user had a single process. Because this process has a
complex address space and is expensive to create, it gets reused
by each command. A Multics process starts up in the "process
overseer" procedure specified in its accounting file: the standard
one calls the standard command processor (shell), which reads and
executes command lines. It does so by parsing the command line and
finding the program named as the first token: the shell just
obtains a pointer to this program and calls it. The dynamic
linking mechanism does most of the work, and the command becomes a
known segment in the process; subsequent executions of the command
will be faster.

Reusing the process this way is efficient, but if a command
scribbles on the ring-4 combined linkage and static segments, or
scrambles the stack threading, the user process can malfunction
and have to be destroyed. The new_proc command signals the
answering service to destroy the current user's process and create
a fresh one, about the same effect as logging out and back in
again. The user's process can also be terminated by encountering a
"simulated fault" caused by dereferencing a special pointer, or by
running out of stack.

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Thu, 30 Jan 2025 15:12:15 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:

> On Wed, 29 Jan 2025 16:05:43 -0500, Dan Espen wrote:
>
>> z/OS Unix is pretty much a full implementation of Unix.
>
> Sounds like an emulation layer, like DG’s MV/UX and EUNICE for VMS of old.

I don't know what those things are but I wouldn't call z/OS an emulation
layer. It looked and acted like a Unix implementation.

One a mainframe, there are a few issues to deal with to run Unix.
The common use terminal, a 3270 is not character at a time, data is
transferred in blocks with a pretty complex protocol. z/OS unix
couldn't do things like run Emacs on a 3270 but it did a reasonably good
job of providing a working stdin/stdout.

Another issue is the native disk file system where data is read in
blocks with predefined block sizes. z/OS Unix also did a pretty good
job with this. Also "normal" Unix file systems were accessible.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Thu, 30 Jan 2025 15:56:22 GMT
View Forum Message <> Reply to Message

Originally posted by: drb

> I don't know what those things are but I wouldn't call z/OS an emulation
 > layer.   It looked and acted like a Unix implementation.

Seems like it might be time to inject a reminder that "unix" sometimes
refers to kernel (AIX is not unix, etc.) and sometimes to API (POSIX
compliance might be thought of as "unix"). IIRC the z/OS unix stuff is
API compliance.

De

Subject: Unix (was: Re: old pharts, Multics vs Unix)
Posted by Anonymous on Thu, 30 Jan 2025 16:18:16 GMT
View Forum Message <> Reply to Message

Originally posted by: vallor

On Thu, 30 Jan 2025 15:56:22 +0000, Dennis Boone wrote:

>> I don't know what those things are but I wouldn't call z/OS an
emulation
>> layer.   It looked and acted like a Unix implementation.
>
> Seems like it might be time to inject a reminder that "unix" sometimes
> refers to kernel (AIX is not unix, etc.) and sometimes to API (POSIX
> compliance might be thought of as "unix"). IIRC the z/OS unix stuff is
> API compliance.

I agree. "unix" or "Unix" is more about the API, and

""UNIX® is a registered trademark of The Open Group""

https://unix.org/trademark.html

So Linux is a Unix but not UNIX(R).

Oddly enough, MacOS is UNIX(R). I have a correspondent
who claims otherwise, but they haven't described what UNIX(R)
offers that MacOS doesn't. Nevertheless, I find it troublesome
to use because of its extra "security" features that hobble
ordinary usage. Example:

```
(base) Mac:~ scott$ cd Desktop/
(base) Mac:Desktop scott$ ls
ls: .: Operation not permitted
(base) Mac:Desktop scott$ ls -ld .
drwx------  8 scott  staff  256 Dec 13 13:45 .
(base) Mac:Desktop scott$ id
uid=502(scott) gid=20(staff) groups=20(staff),12(everyone),
 61(localaccounts),79(_appserverusr),80(admin),81(_appservera dm),
 98(_lpadmin),701(com.apple.sharepoint.group.1),33(_appstore) ,
100(_lpoperator),204(_developer),250(_analyticsusers),
 395(com.apple.access_ftp),398(com.apple.access_screensharing ),
399(com.apple.access_ssh),400(com.apple.access_remote_ae)
```

Oh, and see all those supplemental groups?

```
(base) Mac:Desktop scott$ getconf NGROUPS_MAX
16
```

So all those groups almost fill up the supplemental
groups array.

Back on friendly Linux:

```
$ getconf NGROUPS_MAX
65536
```

--
-Scott System76 Thelio Mega v1.1 x86_64 NVIDIA RTX 3090 Ti
   OS: Linux 6.13.0 Release: Mint 22.1 Mem: 258G
   Linux user since 1992.

Subject: Re: old pharts, Multics vs Unix
Posted by cross on Thu, 30 Jan 2025 16:29:31 GMT
View Forum Message <> Reply to Message

In article <zg-dnUdXgZm7PAb6nZ2dnZfqnPadnZ2d@giganews.com>,
Dennis Boone <drb@ihatespam.msu.edu> wrote:
>> I don't know what those things are but I wouldn't call z/OS an emulation

>> layer.   It looked and acted like a Unix implementation.
>
> Seems like it might be time to inject a reminder that "unix" sometimes
> refers to kernel (AIX is not unix, etc.) and sometimes to API (POSIX
> compliance might be thought of as "unix"). IIRC the z/OS unix stuff is
> API compliance.

"Unix" is a trademark, owned by The Open Group. Systems that
implement relevant standards and provide a common base of
utilities, libraries, etc, can apply for certification to use
the Unix trademark. Only a few still bother to do so: Apple,
IBM, HP and SCO. Of the IBM systems, versions of both AIX and
z/OS are certified Unix systems. So both AIX and z/OS are Unix.

Many other systems adhere closely to the the standards required
for Unix certification and provide the common utilities and so
forth, but don't bother with the laborious (and expensive)
certification process, and so can't technically call themselves
Unix. Most of these systems meet those criteria for all intents
and purposes and so are sort of de facto Unix, even if not
legally Unix. Among these are Linux, most of the BSD-derived
operating systems, QNX, and illumos (descendent of Solaris,
itself derived from SVR4).

Then there is the kernel lineage: at one point, "Unix" meant
sharing code with systems derived from early versions of
research Unix (e.g., 7th Edition, 32/V, 4BSD, SysIII and SysV,
etc). AIX and HP-UX both certainly fall into this category, as
did many of the now obsolete "commercial" Unixes of the 1980s
and 90s, but z/OS (as a whole) and Linux do not. The BSDs are
sort of weird in this regard: they are kind of the Ship of
Theseus of Unix in that they are directly descended from the
research Unix code base, but with all of the AT&T code rewritten
and replaced.

Then there are Unix-like systems that are designed for other
purposes, but neither descend from research Unix code nor try to
adhere to the standards mentioned before. Many of these are
research or pedagogical systems, such as Comer's Xinu, xv6 and
its derivatives (rxv64 [disclaimer: I wrote that]; sv6, and the
early versions of Minix. Mach and similar research systems fit
into a weird quasi-independent space, as Mach started from 4BSD
but is for all intents its own thing now.

Then there are hybrids like OSF/1, etc.

I've found that when one refers to "Unix" one is generally being
imprecise, and the exact meaning depends on context. Often it's

just a shorthand for "looks and behaves substantially as one
expects based on familiarity with other Unix systems." So when
one is talking generically about "Unix" one may also mean Linux,
BSD, etc, even though those are technically not "Unix" in the
sense of being certified to use the trademark, or sharing code
with Bell/AT&T/USL/whatever code.

- Dan C.

Subject: Re: old pharts, Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Thu, 30 Jan 2025 16:37:30 GMT
View Forum Message <> Reply to Message

Dan Espen <dan1espen@gmail.com> writes:
> One a mainframe, there are a few issues to deal with to run Unix.
> The common use terminal, a 3270 is not character at a time, data is
> transferred in blocks with a pretty complex protocol. z/OS unix
> couldn't do things like run Emacs on a 3270 but it did a reasonably good
> job of providing a working stdin/stdout.

70s 3272/3277 had .089sec hardware response ... then 80 came 3274/3278
where lots of electronics were moved out of terminal back into 3274
controller (reducing 3278 manufacturing cost) ... significantly driving
up the coax protocol chatter and latency ... and hardware response went
to .3sec-.5sec (depending on amount of data transferred). This was in
period of studies showing improved human productivity with .25sec
"response" (seen by human) ... with 3272/3277 required .161 system
response (plus terminal hardware .089) to give .25sec human response.

after joining ibm, one of my hobbies was enhanced production operating
systems for internal datacenters. circa 1980, rare MVS/TSO system was
lucky to see even 1sec interactive system response (and nearly all much
worse) ... some number of internal VM370 systems were claiming .25sec
interactive system response ... but I had lots of internal systems with
..11sec interactive system response (giving .199 response seeen by
human w/3277).

Letters to the 3278 product administrator complaining about 3278 for
interactive computing got reply that 3278 wasn't intended for
interactive computing, but "data entry" (aka electronic keypunch).

Later IBM/PC hardware 3277 emulation cards had 4-5 times the
upload/download throughput of 3278 emulation cards.

however, all 3270s were half-duplex and if you were unfortunate to hit a
key same time system went to write to screen, it would lock the keyboard
and would have to stop and hit the reset key. YKT developed a FIFO box

for 3277, unplug the keyboard from the 3277 head, plug the FIFO box into
the head and plug the 3277 keyboard into the fifo box ... eliminating
the unfortunate keyboad lock.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix
Posted by scott on Thu, 30 Jan 2025 16:48:41 GMT
View Forum Message <> Reply to Message

cross@spitfire.i.gajendra.net (Dan Cross) writes:
> In article <zg-dnUdXgZm7PAb6nZ2dnZfqnPadnZ2d@giganews.com>,
> Dennis Boone <drb@ihatespam.msu.edu> wrote:
>>> I don't know what those things are but I wouldn't call z/OS an emulation
>>> layer. It looked and acted like a Unix implementation.
>>

<snip>

> Many other systems adhere closely to the the standards required
> for Unix certification and provide the common utilities and so
> forth, but don't bother with the laborious (and expensive)
> certification process, and so can't technically call themselves
> Unix. Most of these systems meet those criteria for all intents
> and purposes and so are sort of de facto Unix, even if not
> legally Unix. Among these are Linux, most of the BSD-derived
> operating systems, QNX, and illumos (descendent of Solaris,
> itself derived from SVR4).

I wouldn't necessarily say 'derived'. Sun and USL worked
together on SVR4, with SVR4 adopting several features
from SunOS including a bunch of UCB usermode stuff.

> Then there are hybrids like OSF/1, etc.

And Chorus/MIX and SVR4/Mk (aka Amadeus[*])

[*] USL, Unisys, ICL (Fujitsu), Chorus Systemes, et al.

Subject: Re: old pharts, Multics vs Unix
Posted by Peter Flass on Thu, 30 Jan 2025 18:33:40 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:

> On Tue, 28 Jan 2025 16:45:22 -0700, Peter Flass wrote:
>
>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>>
>>> On Mon, 27 Jan 2025 07:09:48 -1000, Lynn Wheeler wrote:
>>>
>>>> about the same time was "gold" from recent univ. graduate (tried to
>>>> hire him at IBM, but no takers) that had ported unix to 370. ("gold"
>>>> code name taken from "Au" - "Amdahl Unix")
>>>
>>> Was it running on bare metal or under the VM hypervisor?
>>>
>>> Because I believe Linux on IBM mainframes to this day runs as a VM, not
>>> on bare metal. Not sure why.
>>>
>> RAS. Lots of error reporting and recovery would otherwise have to be
>> duplicated.
>
> Look at it the other way: Linux already has extensive mechanisms for doing
> exactly this sort of thing, which work across all its different platforms.
>
> So the question becomes: would someone running Linux on an IBM mainframe
> want to make a special case of that installation, or would they rather be
> able to manage it with the same mechanisms as all their other Linux
> installations?
>

You wouldn’t buy a mainframe to run Linux. I would think the typical
customer is already running a zOS, zVM, or zTPF workload and wants to add
Linux to it.

--
Pete

Subject: Re: old pharts, Multics vs Unix
Posted by Peter Flass on Thu, 30 Jan 2025 18:33:42 GMT
View Forum Message <> Reply to Message

Dennis Boone <drb@ihatespam.msu.edu> wrote:
>> I don't know what those things are but I wouldn't call z/OS an emulation
>> layer.   It looked and acted like a Unix implementation.
>
> Seems like it might be time to inject a reminder that "unix" sometimes
> refers to kernel (AIX is not unix, etc.) and sometimes to API (POSIX
> compliance might be thought of as "unix"). IIRC the z/OS unix stuff is
> API compliance.
>

> De
>

I think they got the right to call it “unix”, but I could be mistaken. It
used to be “unix system services”.

--
Pete

Subject: Re: old pharts, Multics vs Unix
Posted by Peter Flass on Thu, 30 Jan 2025 18:33:43 GMT
View Forum Message <> Reply to Message

Dan Cross <cross@spitfire.i.gajendra.net> wrote:
> In article <zg-dnUdXgZm7PAb6nZ2dnZfqnPadnZ2d@giganews.com>,
> Dennis Boone <drb@ihatespam.msu.edu> wrote:
>>> I don't know what those things are but I wouldn't call z/OS an emulation
>>> layer.   It looked and acted like a Unix implementation.
>>
>> Seems like it might be time to inject a reminder that "unix" sometimes
>> refers to kernel (AIX is not unix, etc.) and sometimes to API (POSIX
>> compliance might be thought of as "unix"). IIRC the z/OS unix stuff is
>> API compliance.
>
> "Unix" is a trademark, owned by The Open Group. Systems that
> implement relevant standards and provide a common base of
> utilities, libraries, etc, can apply for certification to use
> the Unix trademark. Only a few still bother to do so: Apple,
> IBM, HP and SCO. Of the IBM systems, versions of both AIX and
> z/OS are certified Unix systems. So both AIX and z/OS are Unix.
>
> Many other systems adhere closely to the the standards required
> for Unix certification and provide the common utilities and so
> forth, but don't bother with the laborious (and expensive)
> certification process, and so can't technically call themselves
> Unix. Most of these systems meet those criteria for all intents
> and purposes and so are sort of de facto Unix, even if not
> legally Unix. Among these are Linux, most of the BSD-derived
> operating systems, QNX, and illumos (descendent of Solaris,
> itself derived from SVR4).
>
> Then there is the kernel lineage: at one point, "Unix" meant
> sharing code with systems derived from early versions of
> research Unix (e.g., 7th Edition, 32/V, 4BSD, SysIII and SysV,
> etc). AIX and HP-UX both certainly fall into this category, as
> did many of the now obsolete "commercial" Unixes of the 1980s
> and 90s, but z/OS (as a whole) and Linux do not. The BSDs are

> sort of weird in this regard: they are kind of the Ship of
> Theseus of Unix in that they are directly descended from the
> research Unix code base, but with all of the AT&T code rewritten
> and replaced.
>
> Then there are Unix-like systems that are designed for other
> purposes, but neither descend from research Unix code nor try to
> adhere to the standards mentioned before. Many of these are
> research or pedagogical systems, such as Comer's Xinu, xv6 and
> its derivatives (rxv64 [disclaimer: I wrote that]; sv6, and the
> early versions of Minix. Mach and similar research systems fit
> into a weird quasi-independent space, as Mach started from 4BSD
> but is for all intents its own thing now.
>
> Then there are hybrids like OSF/1, etc.
>
> I've found that when one refers to "Unix" one is generally being
> imprecise, and the exact meaning depends on context. Often it's
> just a shorthand for "looks and behaves substantially as one
> expects based on familiarity with other Unix systems." So when
> one is talking generically about "Unix" one may also mean Linux,
> BSD, etc, even though those are technically not "Unix" in the
> sense of being certified to use the trademark, or sharing code
> with Bell/AT&T/USL/whatever code.
>

I like Unix vs. Unix(R), that’s a great way to differentiate. Just like it
used to be with IBM “standards”back in the day, I suspect Linux is driving
the bus these days.

--
Pete

Subject: Re: Unix (was: Re: old pharts, Multics vs Unix)
Posted by Anonymous on Thu, 30 Jan 2025 19:19:40 GMT
View Forum Message <> Reply to Message

Originally posted by: Mister Johnson

On 2025-01-30, vallor <vallor@cultnix.org> wrote:
[...]
```
> (base) Mac:~ scott$ cd Desktop/
> (base) Mac:Desktop scott$ ls
> ls: .: Operation not permitted
> (base) Mac:Desktop scott$ ls -ld .
> drwx------  8 scott  staff  256 Dec 13 13:45 .
```
[...]

that's odd!

```
% cd Desktop/
% ls
[lots of stuff]
% ls -ld .
drwx------@ 129 foo  staff  4128 22 Jan 22:17 .
% ls -lde .
drwx------@ 129 foo  staff  4128 22 Jan 22:17 .
 0: group:everyone deny delete
```

Subject: Re: old pharts, Multics vs Unix
Posted by John Levine on Thu, 30 Jan 2025 19:45:05 GMT
View Forum Message <> Reply to Message

According to Peter Flass <peter_flass@yahoo.com>:
> You wouldn’t buy a mainframe to run Linux. I would think the typical
> customer is already running a zOS, zVM, or zTPF workload and wants to add
> Linux to it.

You might think so but IBM has this thing called LinuxONE which is a
zSeries machine with microcode tweaks so it can ron linux but not
z/OS or other operating systems. I gather it is cheaper than
the untweaked Z:

https://www.ibm.com/products/linuxone-4

You'd use it for the same reason you'd use any other mainframe,
extremely high reliability with uptime measured in years and
sometimes decades. They can swap out entire hardware subsystems
without rebooting. It's the Computer of Theseus.

--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: Unix
Posted by Anonymous on Thu, 30 Jan 2025 20:24:23 GMT
View Forum Message <> Reply to Message

Originally posted by: geodandw

On 1/30/25 11:18, vallor wrote:
> On Thu, 30 Jan 2025 15:56:22 +0000, Dennis Boone wrote:
>
>>> I don't know what those things are but I wouldn't call z/OS an
> emulation
>>> layer.   It looked and acted like a Unix implementation.
>>
>> Seems like it might be time to inject a reminder that "unix" sometimes
>> refers to kernel (AIX is not unix, etc.) and sometimes to API (POSIX
>> compliance might be thought of as "unix"). IIRC the z/OS unix stuff is
>> API compliance.
>
> I agree. "unix" or "Unix" is more about the API, and
>
> ""UNIX® is a registered trademark of The Open Group""
>
> https://unix.org/trademark.html
>
> So Linux is a Unix but not UNIX(R).
>
> Oddly enough, MacOS is UNIX(R). I have a correspondent
> who claims otherwise, but they haven't described what UNIX(R)
> offers that MacOS doesn't. Nevertheless, I find it troublesome
> to use because of its extra "security" features that hobble
> ordinary usage. Example:
>

```
> (base) Mac:~ scott$ cd Desktop/
> (base) Mac:Desktop scott$ ls
> ls: .: Operation not permitted
> (base) Mac:Desktop scott$ ls -ld .
> drwx------  8 scott  staff  256 Dec 13 13:45 .
> (base) Mac:Desktop scott$ id
> uid=502(scott) gid=20(staff) groups=20(staff),12(everyone),
>  61(localaccounts),79(_appserverusr),80(admin),81(_appservera dm),
>  98(_lpadmin),701(com.apple.sharepoint.group.1),33(_appstore) ,
> 100(_lpoperator),204(_developer),250(_analyticsusers),
>  395(com.apple.access_ftp),398(com.apple.access_screensharing ),
> 399(com.apple.access_ssh),400(com.apple.access_remote_ae)
```

>
> Oh, and see all those supplemental groups?
>

```
> (base) Mac:Desktop scott$ getconf NGROUPS_MAX
> 16
```

>
> So all those groups almost fill up the supplemental
> groups array.
>
> Back on friendly Linux:

>
> $ getconf NGROUPS_MAX
> 65536
>
Since MacOS is based on BSD, why is this surprising?

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Thu, 30 Jan 2025 21:12:53 GMT
View Forum Message <> Reply to Message

Lynn Wheeler <lynn@garlic.com> writes:

> however, all 3270s were half-duplex and if you were unfortunate to hit a
> key same time system went to write to screen, it would lock the keyboard
> and would have to stop and hit the reset key. YKT developed a FIFO box
> for 3277, unplug the keyboard from the 3277 head, plug the FIFO box into
> the head and plug the 3277 keyboard into the fifo box ... eliminating
> the unfortunate keyboad lock.

Back in the late 60s I finished implementing my first online system on a
S/360-30 and IBM 2260s.

Then my boss came in with the manuals for the IBM 3270s.
He wanted my opinion on the terminals.
I read the manual and told my boss they were unnecessary complicated
crap.

Over the years I did lots of stuff with 3270s. I never changed my mind.

One project I did was using Bunker Ramo 3270 compatible terminals.
By mistake I wrote some code that put more than 80 characters on a line.
The Bunker Ramo CRT just squeezed the text a bit. If I recall you could
put around 120 characters on a line and still read the display.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by cross on Thu, 30 Jan 2025 22:12:45 GMT
View Forum Message <> Reply to Message

In article <725933673.759953355.073976.peter_flass-yahoo.com@news.eternal-september.org>,
Peter Flass <peter_flass@yahoo.com> wrote:
> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>> On Tue, 28 Jan 2025 16:45:22 -0700, Peter Flass wrote:

>>
>>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>>>
>>>> On Mon, 27 Jan 2025 07:09:48 -1000, Lynn Wheeler wrote:
>>>>
>>>> > about the same time was "gold" from recent univ. graduate (tried to
>>>> > hire him at IBM, but no takers) that had ported unix to 370. ("gold"
>>>> > code name taken from "Au" - "Amdahl Unix")
>>>>
>>>> Was it running on bare metal or under the VM hypervisor?
>>>>
>>>> Because I believe Linux on IBM mainframes to this day runs as a VM, not
>>>> on bare metal. Not sure why.
>>>>
>>> RAS. Lots of error reporting and recovery would otherwise have to be
>>> duplicated.
>>
>> Look at it the other way: Linux already has extensive mechanisms for doing
>> exactly this sort of thing, which work across all its different platforms.
>>
>> So the question becomes: would someone running Linux on an IBM mainframe
>> want to make a special case of that installation, or would they rather be
>> able to manage it with the same mechanisms as all their other Linux
>> installations?
>
> You wouldn't buy a mainframe to run Linux. I would think the typical
> customer is already running a zOS, zVM, or zTPF workload and wants to add
> Linux to it.

The troll's characterization of Linux with respect to RAS is
simply incorrect. While Linux on some architectures may have
some rudimentary support for RAS (e.g., x86 MCA or something,
support for basic PCIe hotplug), Linux comes nowhere near
mainframe-class RAS support.

- Dan C.

Subject: Re: old pharts, Multics vs Unix
Posted by cross on Thu, 30 Jan 2025 22:16:15 GMT
View Forum Message <> Reply to Message

In article
<1471164854.759954296.495287.peter_flass-yahoo.com@news.eternal-september.org>,
Peter Flass <peter_flass@yahoo.com> wrote:
> Dan Cross <cross@spitfire.i.gajendra.net> wrote:
>> In article <zg-dnUdXgZm7PAb6nZ2dnZfqnPadnZ2d@giganews.com>,
>> Dennis Boone <drb@ihatespam.msu.edu> wrote:

>>>> I don't know what those things are but I wouldn't call z/OS an emulation
>>>> layer.   It looked and acted like a Unix implementation.
>>>
>>> Seems like it might be time to inject a reminder that "unix" sometimes
>>> refers to kernel (AIX is not unix, etc.) and sometimes to API (POSIX
>>> compliance might be thought of as "unix"). IIRC the z/OS unix stuff is
>>> API compliance.
>>
>> "Unix" is a trademark, owned by The Open Group. Systems that
>> implement relevant standards and provide a common base of
>> utilities, libraries, etc, can apply for certification to use
>> the Unix trademark. Only a few still bother to do so: Apple,
>> IBM, HP and SCO. Of the IBM systems, versions of both AIX and
>> z/OS are certified Unix systems. So both AIX and z/OS are Unix.
>>
>> Many other systems adhere closely to the the standards required
>> for Unix certification and provide the common utilities and so
>> forth, but don't bother with the laborious (and expensive)
>> certification process, and so can't technically call themselves
>> Unix. Most of these systems meet those criteria for all intents
>> and purposes and so are sort of de facto Unix, even if not
>> legally Unix. Among these are Linux, most of the BSD-derived
>> operating systems, QNX, and illumos (descendent of Solaris,
>> itself derived from SVR4).
>>
>> Then there is the kernel lineage: at one point, "Unix" meant
>> sharing code with systems derived from early versions of
>> research Unix (e.g., 7th Edition, 32/V, 4BSD, SysIII and SysV,
>> etc). AIX and HP-UX both certainly fall into this category, as
>> did many of the now obsolete "commercial" Unixes of the 1980s
>> and 90s, but z/OS (as a whole) and Linux do not. The BSDs are
>> sort of weird in this regard: they are kind of the Ship of
>> Theseus of Unix in that they are directly descended from the
>> research Unix code base, but with all of the AT&T code rewritten
>> and replaced.
>>
>> Then there are Unix-like systems that are designed for other
>> purposes, but neither descend from research Unix code nor try to
>> adhere to the standards mentioned before. Many of these are
>> research or pedagogical systems, such as Comer's Xinu, xv6 and
>> its derivatives (rxv64 [disclaimer: I wrote that]; sv6, and the
>> early versions of Minix. Mach and similar research systems fit
>> into a weird quasi-independent space, as Mach started from 4BSD
>> but is for all intents its own thing now.
>>
>> Then there are hybrids like OSF/1, etc.
>>
>> I've found that when one refers to "Unix" one is generally being

>> imprecise, and the exact meaning depends on context. Often it's
>> just a shorthand for "looks and behaves substantially as one
>> expects based on familiarity with other Unix systems." So when
>> one is talking generically about "Unix" one may also mean Linux,
>> BSD, etc, even though those are technically not "Unix" in the
>> sense of being certified to use the trademark, or sharing code
>> with Bell/AT&T/USL/whatever code.
>>
>
> I like Unix vs. Unix(R), that's a great way to differentiate. Just like it
> used to be with IBM “standards”back in the day, I suspect Linux is driving
> the bus these days.

Very much so. Linux is, undoubtedly, the most important OS in
the world right now. I think this is why you see so few systems
listed as UNIX(R) certified: it simply doesn't matter all that
much anymore.

 - Dan C.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Thu, 30 Jan 2025 22:20:27 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 30 Jan 2025 19:45:05 -0000 (UTC), John Levine wrote:

> You'd use it for the same reason you'd use any other mainframe,
> extremely high reliability with uptime measured in years and sometimes
> decades. They can swap out entire hardware subsystems without
> rebooting.

That’s all a complete myth.

There is an article from 1986 on Bitsavers, talking about maintaining
correct time on IBM mainframes. It recommends rebooting to turn daylight
saving on and off.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Thu, 30 Jan 2025 22:21:09 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 30 Jan 2025 11:33:40 -0700, Peter Flass wrote:

> You wouldn’t buy a mainframe to run Linux.

You wouldn’t buy a mainframe for any rational reason. All the rational reasons for buying them disappeared during the 1980s.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Thu, 30 Jan 2025 22:23:38 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 30 Jan 2025 10:12:15 -0500, Dan Espen wrote:

> ... I wouldn't call z/OS an emulation layer. It looked and acted like a
> Unix implementation.

Not a very good one, by your own admission:

> One a mainframe, there are a few issues to deal with to run Unix.
> The common use terminal, a 3270 is not character at a time, data is
> transferred in blocks with a pretty complex protocol. z/OS unix
> couldn't do things like run Emacs on a 3270 but it did a reasonably good
> job of providing a working stdin/stdout.

Couldn’t even run Emacs?? What kind of “Unix” is this?

> Another issue is the native disk file system where data is read in
> blocks with predefined block sizes. z/OS Unix also did a pretty good
> job with this.

What did “pretty good job” mean, exactly? Could it reliably and efficiently handle files with sizes that were not an exact multiple of some block size, or not?

How efficiently could it fork multiple processes?

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Thu, 30 Jan 2025 22:24:17 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 30 Jan 2025 15:56:22 +0000, Dennis Boone wrote:

> Seems like it might be time to inject a reminder that "unix" sometimes
> refers to kernel (AIX is not unix, etc.) and sometimes to API (POSIX
> compliance might be thought of as "unix"). IIRC the z/OS unix stuff is
> API compliance.

In other words, it was an emulation layer. And an imperfect one, at that.

Subject: Re: old pharts, Multics vs Unix
Posted by cross on Thu, 30 Jan 2025 22:44:15 GMT
View Forum Message <> Reply to Message

In article <J3OmP.47191$HO1.4584@fx14.iad>,
Scott Lurndal <slp53@pacbell.net> wrote:
> cross@spitfire.i.gajendra.net (Dan Cross) writes:
>> In article <zg-dnUdXgZm7PAb6nZ2dnZfqnPadnZ2d@giganews.com>,
>> Dennis Boone <drb@ihatespam.msu.edu> wrote:
>>>> I don't know what those things are but I wouldn't call z/OS an emulation
>>>> layer. It looked and acted like a Unix implementation.
>>>
>
> <snip>
>
>> Many other systems adhere closely to the the standards required
>> for Unix certification and provide the common utilities and so
>> forth, but don't bother with the laborious (and expensive)
>> certification process, and so can't technically call themselves
>> Unix. Most of these systems meet those criteria for all intents
>> and purposes and so are sort of de facto Unix, even if not
>> legally Unix. Among these are Linux, most of the BSD-derived
>> operating systems, QNX, and illumos (descendent of Solaris,
>> itself derived from SVR4).
>
> I wouldn't necessarily say 'derived'. Sun and USL worked
> together on SVR4, with SVR4 adopting several features
> from SunOS including a bunch of UCB usermode stuff.

I would. As you say, Sun and AT&T collaborated to produce SVR4,
and Sun agreed to switch from the 4.2/4.3BSD-based SunOS 4 to
System V in exchange for a cache infusion from AT&T. But this
was all in the context of AT&T trying to enter the computer
market after the Bell System broke up and ended its monopoly
status in 1982; once Sun adopted Solaris, their agreement with
AT&T was terminated, and AT&T SVR4.x and Solaris followed
diverging paths. Anyway, by the time of OpenSolaris (and then
illumos with it's various distributions: OpenIndiana and OmniOS
and so on) Solaris was clearly derived from SVR4, but pretty

different internally.

>> Then there are hybrids like OSF/1, etc.
>
> And Chorus/MIX and SVR4/Mk (aka Amadeus[*])
>
> [*] USL, Unisys, ICL (Fujitsu), Chorus Systemes, et al.

Indeed! The list goes on and on.

 - Dan C.

Subject: Re: Unix
Posted by Anonymous on Thu, 30 Jan 2025 23:42:44 GMT
View Forum Message <> Reply to Message

Originally posted by: moi

On 30/01/2025 16:18, vallor wrote:

```
>  Oddly enough, MacOS is UNIX(R).  I have a correspondent
>  who claims otherwise, but they haven't described what UNIX(R)
>  offers that MacOS doesn't.  Nevertheless, I find it troublesome
>  to use because of its extra "security" features that hobble
>  ordinary usage.  Example:
>
>  (base) Mac:~ scott$ cd Desktop/
>  (base) Mac:Desktop scott$ ls
>  ls: .: Operation not permitted
```

I am not sure what you have done to your computer, but here:

```
/Users/wf: cd Desktop

/Users/wf/Desktop: arch
i386
/Users/wf/Desktop: uname -s
Darwin

/Users/wf/Desktop: ls
www.findlayw.plus.com    www.findlayw.plus.com.gz

/Users/wf/Desktop: ls -ld .
drwx------@ 8 wf  staff  256 30 Jan 23:33 .

/Users/wf/Desktop: ls -l
total 56
```

```
-rwxr-xr-x@ 1 wf  staff  22493 30 Jan 02:59 www.findlayw.plus.com
-rwxr-xr-x@ 1 wf  staff   2835 30 Jan 02:58 www.findlayw.plus.com.gz

/Users/wf/Desktop: id
uid=501(wf) gid=20(staff)
 groups=20(staff),12(everyone),61(localaccounts),100(_lpopera tor)
/Users/wf/Desktop:
```

--
Bill F.

Subject: Re: old pharts, Multics vs Unix
Posted by scott on Fri, 31 Jan 2025 00:22:47 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:
> On Thu, 30 Jan 2025 19:45:05 -0000 (UTC), John Levine wrote:
>
>> You'd use it for the same reason you'd use any other mainframe,
>> extremely high reliability with uptime measured in years and sometimes
>> decades. They can swap out entire hardware subsystems without
>> rebooting.
>
> That’s all a complete myth.
>
> There is an article from 1986 on Bitsavers, talking about maintaining
> correct time on IBM mainframes. It recommends rebooting to turn daylight
> saving on and off.

Half a century ago, even if what you say is true. Modern Z-series
cannot be compared to 70's vintage hardware.

There was also Tandem in that timeframe, and they were truly non-stop.

Although a former colleague was at Tandem a couple decades ago when
they got hit by date bug that started affecting systems at the
date line - it reached eastern europe before he pushed a microcode
patch to the rest of the globe.

Subject: Re: Unix (was: Re: old pharts, Multics vs Unix)
Posted by Anonymous on Fri, 31 Jan 2025 00:34:38 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 30 Jan 2025 19:19:40 GMT, Mister Johnson wrote:

```
>  On 2025-01-30, vallor <vallor@cultnix.org> wrote:
>
>  [...]
>>  (base) Mac:~ scott$ cd Desktop/
>>  (base) Mac:Desktop scott$ ls
>  ls: .: Operation not permitted (base)
>
>  that's odd!
>
>  % cd Desktop/
>  % ls [lots of stuff]
>  % ls -ld .
>  drwx------@ 129 foo  staff  4128 22 Jan 22:17 . % ls -lde .
>  drwx------@ 129 foo  staff  4128 22 Jan 22:17 .
```

As I recall, it was a common problem, at least in the early days of OS X, for various file/directory permissions to get screwed up and require the running of some utility to fix them up again.

Does that still occur?

Subject: Re: Unix (was: Re: old pharts, Multics vs Unix)
Posted by Anonymous on Fri, 31 Jan 2025 00:39:23 GMT

View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

“Unix” is the trademark. “Unix™” or “Unix®” is merely recognition of the trademark. You don’t evade the trademark by leaving those symbols out.

When people say “Unix”, they don’t usually think of the trademark, they think of some traditions on how the system is supposed to behave. But using the same term for both is confusing, even apart from the legal issues. That’s why some of us prefer to use a term like “*nix” to denote the traditional behaviour, as distinct from the trademark.

This is even more important given that macOS, the one system still on its feet that is legally entitled to call itself “Unix”, has departed so much from the traditions denoted by “*nix”.

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Fri, 31 Jan 2025 00:43:20 GMT

View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:

> On Thu, 30 Jan 2025 11:33:40 -0700, Peter Flass wrote:
>
>> You wouldn’t buy a mainframe to run Linux.
>
> You wouldn’t buy a mainframe for any rational reason. All the rational
> reasons for buying them disappeared during the 1980s.

You'd buy a mainframe if you accumulated billions of dollars worth of
software developed to only to run on mainframes.

IBM is still selling mainframe hardware.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Fri, 31 Jan 2025 00:55:57 GMT

View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:

> On Thu, 30 Jan 2025 10:12:15 -0500, Dan Espen wrote:
>
>> ... I wouldn't call z/OS an emulation layer. It looked and acted like a
>> Unix implementation.
>
> Not a very good one, by your own admission:
>
>> One a mainframe, there are a few issues to deal with to run Unix.
>> The common use terminal, a 3270 is not character at a time, data is
>> transferred in blocks with a pretty complex protocol. z/OS unix
>> couldn't do things like run Emacs on a 3270 but it did a reasonably good
>> job of providing a working stdin/stdout.
>
> Couldn’t even run Emacs?? What kind of “Unix” is this?

A Unix without A-sync terminals and no full duplex I/O.

>> Another issue is the native disk file system where data is read in
>> blocks with predefined block sizes. z/OS Unix also did a pretty good
>> job with this.
>
> What did “pretty good job” mean, exactly? Could it reliably and
> efficiently handle files with sizes that were not an exact multiple of
> some block size, or not?

Yes it could.
In C you had the choice of reading those files as binary or non-binary.
In binary mode you'd see fixed size blocks of data. In character mode
you'd see newlines and trailing spaces stripped.

z/OS also supported UNIX style files.

> How efficiently could it fork multiple processes?

Never saw a problem with that.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Fri, 31 Jan 2025 00:57:21 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:

> On Thu, 30 Jan 2025 15:56:22 +0000, Dennis Boone wrote:
>
>> Seems like it might be time to inject a reminder that "unix" sometimes
>> refers to kernel (AIX is not unix, etc.) and sometimes to API (POSIX
>> compliance might be thought of as "unix"). IIRC the z/OS unix stuff is
>> API compliance.
>
> In other words, it was an emulation layer. And an imperfect one, at that.

Whatever crap definition of "emulation" you want to tout.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by cross on Fri, 31 Jan 2025 01:10:18 GMT
View Forum Message <> Reply to Message

In article <vnh72t$36h20$2@dont-email.me>,
Dan Espen <dan1espen@gmail.com> wrote:
> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>
>> On Thu, 30 Jan 2025 10:12:15 -0500, Dan Espen wrote:
>>

>>> ... I wouldn't call z/OS an emulation layer. It looked and acted like a
>>> Unix implementation.
>>
>> Not a very good one, by your own admission:
>>
>>> One a mainframe, there are a few issues to deal with to run Unix.
>>> The common use terminal, a 3270 is not character at a time, data is
>>> transferred in blocks with a pretty complex protocol. z/OS unix
>>> couldn't do things like run Emacs on a 3270 but it did a reasonably good
>>> job of providing a working stdin/stdout.
>>
>> Couldn't even run Emacs?? What kind of "Unix" is this?
>
> A Unix without A-sync terminals and no full duplex I/O.

There was a lot of Unix that didn't run emacs before 1981.

I'd save your breath, though: Lawrence is a well-known troll who
will "argue" this to death, usually be repeating the same silly
points over and over again.

 - Dan C.

Subject: Re: Unix
Posted by Anonymous on Fri, 31 Jan 2025 03:17:22 GMT
View Forum Message <> Reply to Message

Originally posted by: vallor

On Thu, 30 Jan 2025 23:42:44 +0000, moi wrote:

> On 30/01/2025 16:18, vallor wrote:
>
>> Oddly enough, MacOS is UNIX(R). I have a correspondent
>> who claims otherwise, but they haven't described what UNIX(R)
>> offers that MacOS doesn't. Nevertheless, I find it troublesome
>> to use because of its extra "security" features that hobble
>> ordinary usage. Example:
>>
>> (base) Mac:~ scott$ cd Desktop/
>> (base) Mac:Desktop scott$ ls
>> ls: .: Operation not permitted
>
> I am not sure what you have done to your computer, but here:
>
> /Users/wf: cd Desktop
>

```
>  /Users/wf/Desktop: arch
>  i386
>  /Users/wf/Desktop: uname -s
>  Darwin
>
>  /Users/wf/Desktop: ls
>  www.findlayw.plus.com    www.findlayw.plus.com.gz
>
>  /Users/wf/Desktop: ls -ld .
>  drwx------@ 8 wf  staff  256 30 Jan 23:33 .
>
>  /Users/wf/Desktop: ls -l
>  total 56
>  -rwxr-xr-x@ 1 wf  staff  22493 30 Jan 02:59 www.findlayw.plus.com
>  -rwxr-xr-x@ 1 wf  staff   2835 30 Jan 02:58 www.findlayw.plus.com.gz
>
>  /Users/wf/Desktop: id
>  uid=501(wf) gid=20(staff)
>   groups=20(staff),12(everyone),61(localaccounts),100(_lpopera tor)
>  /Users/wf/Desktop:
```

I forgot to mention that my shell commands were used via an ssh session.

I regret the omission.

--
-Scott System76 Thelio Mega v1.1 x86_64 NVIDIA RTX 3090 Ti
   OS: Linux 6.13.0 Release: Mint 22.1 Mem: 258G
   "Useless Invention: Double-sided playing cards."

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Fri, 31 Jan 2025 06:30:53 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 30 Jan 2025 19:43:20 -0500, Dan Espen wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>
>> On Thu, 30 Jan 2025 11:33:40 -0700, Peter Flass wrote:
>>
>>> You wouldn’t buy a mainframe to run Linux.
>>
>> You wouldn’t buy a mainframe for any rational reason. All the rational
>> reasons for buying them disappeared during the 1980s.
>

> You'd buy a mainframe if you accumulated billions of dollars worth of
> software developed to only to run on mainframes.

If you haven't completely depreciated all that by now, then you're
probably out of business or heading that way.

> IBM is still selling mainframe hardware.

I doubt they're making any profit on it any more.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Fri, 31 Jan 2025 06:31:27 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 30 Jan 2025 19:57:21 -0500, Dan Espen wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>
>> On Thu, 30 Jan 2025 15:56:22 +0000, Dennis Boone wrote:
>>
>>> Seems like it might be time to inject a reminder that "unix" sometimes
>>> refers to kernel (AIX is not unix, etc.) and sometimes to API (POSIX
>>> compliance might be thought of as "unix"). IIRC the z/OS unix stuff
>>> is API compliance.
>>
>> In other words, it was an emulation layer. And an imperfect one, at
>> that.
>
> Whatever crap definition of "emulation" you want to tout.

The fact that the emulation was crap is all the info I need.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Fri, 31 Jan 2025 06:33:03 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Thu, 30 Jan 2025 19:55:57 -0500, Dan Espen wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>
>> Could it reliably and efficiently handle files with sizes that were not

>> an exact multiple of some block size, or not?
>
> Yes it could. In C you had the choice of reading those files as binary
> or non-binary. In binary mode you'd see fixed size blocks of data.

That's already wrong.

>> How efficiently could it fork multiple processes?
>
> Never saw a problem with that.

Somehow I doubt you tested it very heavily.

Subject: Re: Multics vs Unix
Posted by jgd on Fri, 31 Jan 2025 09:09:24 GMT
View Forum Message <> Reply to Message

In article <vn1apj$2fink$2@dont-email.me>, ldo@nz.invalid (Lawrence
D'Oliveiro) wrote:

> But note that the trend of falling memory prices was already
> becoming clear by the 1970s, if not earlier. The earliest batch
> systems only kept one program in memory at one time, and swapped it
> out for another one when it went into any kind of I/O wait, to keep
> the CPU busy;

Not so. That demands a plentiful supply of fast disk or drum space, and
the earliest batch systems often didn't have disks or drums.

The solutions to this problem included spooling to tape, and having
multiple programs resident in memory and switching between them.

https://en.wikipedia.org/wiki/OS/360_and_successors#OS/360_v ariants

On the IBM 360 family, working with multiple memory partitions required
256+KB RAM, which was a lot in the 1960s. Nowadays, that's L2 cache on a
mobile device processor, but it was a fairly big mainframe 55 years ago,
and the programs it ran were small enough - mostly written in assembler -
to fit several into memory.

John

Subject: Re: Multics vs Unix
Posted by jgd on Fri, 31 Jan 2025 09:09:24 GMT
View Forum Message <> Reply to Message

In article <87wmehmmqd.fsf@localhost>, lynn@garlic.com (Lynn Wheeler) wrote:

> IMS kept physical disk addresses internally in data records for
> related data

Ouch!

John

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Fri, 31 Jan 2025 15:24:01 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:

> On Thu, 30 Jan 2025 19:43:20 -0500, Dan Espen wrote:
>
>> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>>
>>> On Thu, 30 Jan 2025 11:33:40 -0700, Peter Flass wrote:
>>>
>>>> You wouldn’t buy a mainframe to run Linux.
>>>
>>> You wouldn’t buy a mainframe for any rational reason. All the rational
>>> reasons for buying them disappeared during the 1980s.
>>
>> You'd buy a mainframe if you accumulated billions of dollars worth of
>> software developed to only to run on mainframes.
>
> If you haven’t completely depreciated all that by now, then you’re
> probably out of business or heading that way.

I'm sorry, "depreciated"?
What are you talking about?
There are people running their businesses with their software.
It makes no difference how they are carrying their development cost on
their books.

>> IBM is still selling mainframe hardware.
>
> I doubt they’re making any profit on it any more.

That's something you could look up for yourself.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Fri, 31 Jan 2025 15:25:17 GMT
View Forum Message <> Reply to Message

cross@spitfire.i.gajendra.net (Dan Cross) writes:

> In article <vnh72t$36h20$2@dont-email.me>,
> Dan Espen <dan1espen@gmail.com> wrote:
>> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>>
>>> On Thu, 30 Jan 2025 10:12:15 -0500, Dan Espen wrote:
>>>
>>>> ... I wouldn't call z/OS an emulation layer. It looked and acted like a
>>>> Unix implementation.
>>>
>>> Not a very good one, by your own admission:
>>>
>>>> One a mainframe, there are a few issues to deal with to run Unix.
>>>> The common use terminal, a 3270 is not character at a time, data is
>>>> transferred in blocks with a pretty complex protocol. z/OS unix
>>>> couldn't do things like run Emacs on a 3270 but it did a reasonably good
>>>> job of providing a working stdin/stdout.
>>>
>>> Couldnâ€™t even run Emacs?? What kind of â€œUnixâ€ is this?
>>
>> A Unix without A-sync terminals and no full duplex I/O.
>
> There was a lot of Unix that didn't run emacs before 1981.
>
> I'd save your breath, though: Lawrence is a well-known troll who
> will "argue" this to death, usually be repeating the same silly
> points over and over again.

I'm getting Alan Connor flashbacks.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by scott on Fri, 31 Jan 2025 15:30:50 GMT
View Forum Message <> Reply to Message

Dan Espen <dan1espen@gmail.com> writes:
> Lawrence D'Oliveiro <ldo@nz.invalid> writes:

>> If you haven’t completely depreciated all that by now, then you’re
>> probably out of business or heading that way.

>
> I'm sorry, "depreciated"?
> What are you talking about?

Dan, Arguing with a troll is pointless.   Best to ignore it.

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Fri, 31 Jan 2025 16:29:47 GMT
View Forum Message <> Reply to Message

scott@slp53.sl.home (Scott Lurndal) writes:

> Dan Espen <dan1espen@gmail.com> writes:
>> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>
>>> If you havenâ€™t completely depreciated all that by now, then youâ€™re
>>> probably out of business or heading that way.
>>
>> I'm sorry, "depreciated"?
>> What are you talking about?
>
> Dan, Arguing with a troll is pointless.   Best to ignore it.

Done.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Fri, 31 Jan 2025 16:45:54 GMT
View Forum Message <> Reply to Message

Originally posted by: John Ames

On Fri, 31 Jan 2025 06:30:53 -0000 (UTC)
Lawrence D'Oliveiro <ldo@nz.invalid> wrote:

> I doubt they’re making any profit on it any more.

....do you think they're just doing it for fun?

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Fri, 31 Jan 2025 17:44:31 GMT

View Forum Message <> Reply to Message

Originally posted by: John Ames

On Fri, 31 Jan 2025 06:33:03 -0000 (UTC)
Lawrence D'Oliveiro <ldo@nz.invalid> wrote:

>>> How efficiently could it fork multiple processes?
>>
>> Never saw a problem with that.
>
> Somehow I doubt you tested it very heavily.

What's your basis for that assertion?

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Fri, 31 Jan 2025 17:58:17 GMT
View Forum Message <> Reply to Message

Originally posted by: Bob Eager

On Fri, 31 Jan 2025 10:24:01 -0500, Dan Espen wrote:

>>> IBM is still selling mainframe hardware.
>>
>> I doubt they’re making any profit on it any more.
>
> That's something you could look up for yourself.

I understand that all the profit has been in software, for many years.

--
Using UNIX since v6 (1975)...

Use the BIG mirror service in the UK:
 http://www.mirrorservice.org

Subject: Re: old pharts, Multics vs Unix
Posted by Anne &amp; Lynn Wheel on Fri, 31 Jan 2025 18:30:10 GMT
View Forum Message <> Reply to Message

Dan Espen <dan1espen@gmail.com> writes:
> You'd buy a mainframe if you accumulated billions of dollars worth of
> software developed to only to run on mainframes.

>
> IBM is still selling mainframe hardware.

turn of century, mainframe hardware was a few percent of IBM revenue and
dropping. a decade ago, mainframe hardware was a couple percent of
revenue and still dropping ... however the mainframe group accounted for
25% of revenue and 40% of profit ... nearly all software and services.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by John Levine on Fri, 31 Jan 2025 18:48:32 GMT
View Forum Message <> Reply to Message

According to Bob Eager <news0009@eager.cx>:
> On Fri, 31 Jan 2025 10:24:01 -0500, Dan Espen wrote:
>
>>>> IBM is still selling mainframe hardware.
>>>
>>> I doubt they're making any profit on it any more.
>>
>> That's something you could look up for yourself.
>
> I understand that all the profit has been in software, for many years.

IBM continues to put a lot of effort into their mainframe hardware. If
you look at their financial reports, they usually have a bullet for "IBM Z"
which is up some quarters, down others. The most recent slide deck has a
bullet that says:

z16 our most successful program in history

I agree that mainframes is a legacy business but it's one that still has
a long life aheade it it.

--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anne &amp; Lynn Wheel on Fri, 31 Jan 2025 18:53:26 GMT
View Forum Message <> Reply to Message

John Levine <johnl@taugh.com> writes:
> I agree that mainframes is a legacy business but it's one that still has
> a long life aheade it it.

IBM deliberately misclassified mainframe sales to enrich execs, lawsuit claims. Lawsuit accuses Big Blue of cheating investors by shifting systems revenue to trendy cloud, mobile tech
 https://www.theregister.com/2022/04/07/ibm_securities_lawsui t/
IBM has been sued by investors who claim the company under former CEO Ginni Rometty propped up its stock price and deceived shareholders by misclassifying revenues from its non-strategic mainframe business - and moving said sales to its strategic business segments - in violation of securities regulations.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by cross on Fri, 31 Jan 2025 19:10:38 GMT
View Forum Message <> Reply to Message

In article <vnj5u0$2vig$1@gal.iecc.com>, John Levine <johnl@taugh.com> wrote:
> According to Bob Eager <news0009@eager.cx>:
>> On Fri, 31 Jan 2025 10:24:01 -0500, Dan Espen wrote:
>>
>>>> > IBM is still selling mainframe hardware.
>>>>
>>>> I doubt they're making any profit on it any more.
>>>
>>> That's something you could look up for yourself.
>>
>> I understand that all the profit has been in software, for many years.
>
> IBM continues to put a lot of effort into their mainframe hardware. If
> you look at their financial reports, they usually have a bullet for "IBM Z"
> which is up some quarters, down others. The most recent slide deck has a
> bullet that says:
>
>   z16 our most successful program in history
>
> I agree that mainframes is a legacy business but it's one that still has
> a long life aheade it it.

z16 is an extremely impressive machine. I agree that if one is not already immersed in the ecosystem there's little reason to become so, but the hardware itself is very capable. I thknk our gear from Oxide gives it a run for the money, but I'm not going to knock IBM iron.

 - Dan C.

Subject: Re: old pharts, Multics vs Unix
Posted by cross on Fri, 31 Jan 2025 19:11:39 GMT
View Forum Message <> Reply to Message

In article <20250131084554.00003052@gmail.com>,
John Ames <commodorejohn@gmail.com> wrote:
> On Fri, 31 Jan 2025 06:30:53 -0000 (UTC)
> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>> I doubt they’re making any profit on it any more.
>
> ...do you think they're just doing it for fun?

That's the problem: he doesn't think, he just posts.

 - Dan C.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by John Levine on Fri, 31 Jan 2025 20:00:28 GMT
View Forum Message <> Reply to Message

According to Lynn Wheeler <lynn@garlic.com>:
> John Levine <johnl@taugh.com> writes:
>> I agree that mainframes is a legacy business but it's one that still has
>> a long life aheade it it.
>
> IBM deliberately misclassified mainframe sales to enrich execs, lawsuit
> claims. Lawsuit accuses Big Blue of cheating investors by shifting
> systems revenue to trendy cloud, mobile tech
> https://www.theregister.com/2022/04/07/ibm_securities_lawsui t/

The plaintiffs in that suit abanodoned it a few months later. It was
a putative class action which went nowhere.

x
--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Fri, 31 Jan 2025 21:25:14 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Fri, 31 Jan 2025 10:24:01 -0500, Dan Espen wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>
>> If you haven't completely depreciated all that by now, then you're
>> probably out of business or heading that way.
>
> I'm sorry, "depreciated"?
> What are you talking about?

You know, one of those amounts you offset against gross income to come up
with net profit (or loss).

Or, if you want it in simpler terms, "written off".

> There are people running their businesses with their software.

Computing needs have changed a lot. COBOL was designed for the batch era.
We no longer operate our businesses in the batch era.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Fri, 31 Jan 2025 21:26:07 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Fri, 31 Jan 2025 08:30:10 -1000, Lynn Wheeler wrote:

> turn of century, mainframe hardware was a few percent of IBM revenue and
> dropping. a decade ago, mainframe hardware was a couple percent of
> revenue and still dropping ... however the mainframe group accounted for
> 25% of revenue and 40% of profit ... nearly all software and services.

Perhaps even more than 100% software and services? So they could be making
0% or less on each mainframe unit still being sold?

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Fri, 31 Jan 2025 21:26:24 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Fri, 31 Jan 2025 08:45:54 -0800, John Ames wrote:

> On Fri, 31 Jan 2025 06:30:53 -0000 (UTC) Lawrence D'Oliveiro
> <ldo@nz.invalid> wrote:
>
>> I doubt they’re making any profit on it any more.
>
> ...do you think they're just doing it for fun?

I think Lynn Wheeler has already answered that question.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Fri, 31 Jan 2025 21:28:51 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Fri, 31 Jan 2025 18:48:32 -0000 (UTC), John Levine wrote:

> I agree that mainframes is a legacy business but it's one that still has
> a long life aheade it it.

If that were true, why is IBM the only one doing it?

Back in 1980, there were something like 10 different companies operating in the mainframe business. Now there is only one. And it’s a company which continues to shrink in most of its business divisions, with the exception of its Red Hat Linux business. Before acquiring Red Hat, it had been losing money for years.

Face it: mainframes are a dead-end business, with zero growth opportunities any more.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Fri, 31 Jan 2025 21:30:10 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Fri, 31 Jan 2025 09:44:31 -0800, John Ames wrote:

> On Fri, 31 Jan 2025 06:33:03 -0000 (UTC) Lawrence D'Oliveiro
> <ldo@nz.invalid> wrote:

>
>>>> How efficiently could it fork multiple processes?
>>>
>>> Never saw a problem with that.
>>
>> Somehow I doubt you tested it very heavily.
>
> What's your basis for that assertion?

It’s well known that the Unix creation-of-lots-of-processes model plays
poorly with every single proprietary OS out there.

The idea that a mainframe system, of all things, could handle process
creation efficiently, is laughable.

Subject: Re: old pharts, Multics vs Unix
Posted by scott on Fri, 31 Jan 2025 22:13:14 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:
> On Fri, 31 Jan 2025 09:44:31 -0800, John Ames wrote:
>
>> On Fri, 31 Jan 2025 06:33:03 -0000 (UTC) Lawrence D'Oliveiro
>> <ldo@nz.invalid> wrote:
>>
>>>> > How efficiently could it fork multiple processes?
>>>>
>>>> Never saw a problem with that.
>>>
>>> Somehow I doubt you tested it very heavily.
>>
>> What's your basis for that assertion?

>
> The idea that a mainframe system, of all things, could handle process
> creation efficiently, is laughable.

More ravings from a troll who has never even used a mainframe
operating system, much less actually written one.

Process creation on Burroughs mainframes was certainly
competitive with fork[*]. I speak from experience on both
ends (having written both mainframe operating systems
and an unix-compatible MPP operating system (specifically responsible
for the process create and management code). I have
patents related to the latter.

[*] In fact, the algol (MCP flavor) routine to create a
new process on the Burroughs Large systems was actually
named 'motherforker'.   On Medium systems, the early MCP
function that created a new context was called 'hiho'.   As
in Hi-Ho, Hi-Ho, it's off to work we go.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Fri, 31 Jan 2025 22:26:40 GMT
View Forum Message <> Reply to Message

Originally posted by: John Ames

On Fri, 31 Jan 2025 21:30:10 -0000 (UTC)
Lawrence D'Oliveiro <ldo@nz.invalid> wrote:

>> What's your basis for that assertion?
>
> It’s well known that the Unix creation-of-lots-of-processes model
> plays poorly with every single proprietary OS out there.
>
> The idea that a mainframe system, of all things, could handle process
> creation efficiently, is laughable.

So your basis for casting doubt on his specific attestation of personal
experience is "everybody knows?"

Hmm. Sure. Makes sense.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anne &amp; Lynn Wheel on Fri, 31 Jan 2025 22:59:08 GMT
View Forum Message <> Reply to Message

periodically reposted, including in afc a couple years ago

industry benchmark ... number of program iterations compared to
reference platform. Early mainframe actual benchmarks ... later
mainframe numbers derived from pubs with percent difference change from
previous generation (similar to most recent mainframe revenue, revenue
derived from percent change)

Early 90s:

eight processor ES/9000-982 : 408MIPS, 51MIPS/processor
RS6000/990 : 126MIPS; 16-way cluster: 2016MIPS, 128-way cluster:
  16,128MIPS (16.128BIPS)

Then somerset/AIM (apple, ibm, motorola), power/pc single chip as well as motorola 88k bus supporting cache consistency for multiprocessor. i86/Pentium new generation where i86 instructions are hardware translated to RISC micro-ops for actual execution (negating RISC system advantage compared to i86).

1999 single IBM PowerPC 440 hits 1,000MIPS (>six times each Dec2000 z900 processor)
1999 single Pentium3 hits 2,054MIPS (twice PowerPC and 13times each Dec2000 z900 processor).

Mainframe this century:

z900, 16 processors, 2.5BIPS (156MIPS/proc), Dec2000
z990, 32 processors, 9BIPS, (281MIPS/proc), 2003
z9, 54 processors, 18BIPS (333MIPS/proc), July2005
z10, 64 processors, 30BIPS (469MIPS/proc), Feb2008
z196, 80 processors, 50BIPS (625MIPS/proc), Jul2010
EC12, 101 processors, 75BIPS (743MIPS/proc), Aug2012
z13, 140 processors, 100BIPS (710MIPS/proc), Jan2015
z14, 170 processors, 150BIPS (862MIPS/proc), Aug2017
z15, 190 processors, 190BIPS (1000MIPS/proc), Sep2019
z16, 200 processors, 222BIPS (1111MIPS/proc), Sep2022

Note max configured z196 (& 50BIPS) went for $30M, at same time E5-2600 server blade (two 8core XEON & 500BIPS) had IBM base list price of $1815 (before industry press announced server chip vendors were shipping half the product directly to cloud megadatacenters for assembly at 1/3rd price of brand name systems, and IBM sells off server blade business)

Large cloud operations can have score of megadatacenters around the world, each with half million or more server blades (2010 era megadatacenter: processing equivalent around 5M max-configured z196 mainframes).

trivia, IBM early 80s, disk division hardware revunue slightly passed high end mainframe division hardware revenue. Late 80s, senior disk engineer got talk scheduled at annual, world-wide, internal communication group conference, supposedly on 3174 performance but opens the talk with statement that communication group was going to be responsible for the demise of the disk division. The disk division was seeing data fleeing mainframe datacenters to more distributed computing friendly platforms, with drop in disk sales. The had come up with a number of solutions but were constantly veoted by the communication group (had corporate strategic responsibility for everything that crossed datacenter walls and were fiercely fighting off client/server and distributed computing).

The disk division software executive partial work around was investing
in distributed computing startups (who would use IBM disks, as well as
sponsored POSIX implementation for MVS). He would sometimes ask us to
visit his investment to see if we could provide any help.

Turns out communication group affecting the whole mainframe revenue and
couple years later, IBM has one of the largest losses in the history of
US corporations ... and IBM was being reorged into the 13 "baby blues"
(take off on "baby bells" in breakup a decade area) in preperation for
breakup
https://web.archive.org/web/20101120231857/http://www.time.c
om/time/magazine/article/0,9171,977353,00.html
https://content.time.com/time/subscriber/article/0,33009,977 353-1,00.html

We had already left IBM but get a call from the bowels of Armonk (IBM
hdqtrs) asking if we could help with the breakup. Before we get started,
the board brings in the former AMEX president as CEO to try and save the
company, who (somewhat) reverses the breakup (although it wasn't long
before the disk division is gone).

Then IBM becomes a financial engineering company

Stockman; The Great Deformation: The Corruption of Capitalism in America
https://www.amazon.com/Great-Deformation-Corruption-Capitali
sm-America-ebook/dp/B00B3M3UK6/
pg464/loc9995-10000: IBM was not the born-again growth machine trumpeted
by the mob of Wall Street momo traders. It was actually a stock buyback
contraption on steroids. During the five years ending in fiscal 2011,
the company spent a staggering $67 billion repurchasing its own shares,
a figure that was equal to 100 percent of its net income.
pg465/loc10014-17: Total shareholder distributions, including dividends,
amounted to $82 billion, or 122 percent, of net income over this
five-year period. Likewise, during the last five years IBM spent less on
capital investment than its depreciation and amortization charges, and
also shrank its constant dollar spending for research and development by
nearly 2 percent annually.

(2013) New IBM Buyback Plan Is For Over 10 Percent Of Its Stock
http://247wallst.com/technology-3/2013/10/29/new-ibm-buyback
-plan-is-for-over-10-percent-of-its-stock/
(2014) IBM Asian Revenues Crash, Adjusted Earnings Beat On Tax Rate
Fudge; Debt Rises 20% To Fund Stock Buybacks (gone behind paywall)
https://web.archive.org/web/20140201174151/http://www.zerohe
dge.com/news/2014-01-21/ibm-asian-revenues-crash-adjusted-ea
rnings-beat-tax-rate-fudge-debt-rises-20-fund-st
The company has represented that its dividends and share repurchases
have come to a total of over $159 billion since 2000.

(2016) After Forking Out $110 Billion on Stock Buybacks, IBM Shifts Its
Spending Focus
 https://www.fool.com/investing/general/2016/04/25/after-fork
ing-out-110-billion-on-stock-buybacks-ib.aspx
(2018) ... still doing buybacks ... but will (now?, finally?, a little?)
shift focus needing it for redhat purchase.
 https://www.bloomberg.com/news/articles/2018-10-30/ibm-to-bu
y-back-up-to-4-billion-of-its-own-shares
(2019) IBM Tumbles After Reporting Worst Revenue In 17 Years As Cloud
Hits Air Pocket (gone behind paywall)
 https://web.archive.org/web/20190417002701/https://www.zeroh
edge.com/news/2019-04-16/ibm-tumbles-after-reporting-worst-r
evenue-17-years-cloud-hits-air-pocket

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by John Levine on Sat, 01 Feb 2025 01:31:37 GMT
View Forum Message <> Reply to Message

According to Lynn Wheeler <lynn@garlic.com>:
> z15, 190 processors, 190BIPS (1000MIPS/proc), Sep2019
> z16, 200 processors, 222BIPS (1111MIPS/proc), Sep2022
>
> Note max configured z196 (& 50BIPS) went for $30M, at same time E5-2600
> server blade (two 8core XEON & 500BIPS) had IBM base list price of $1815
> (before industry press announced server chip vendors were shipping half
> the product directly to cloud megadatacenters for assembly at 1/3rd
> price of brand name systems, and IBM sells off server blade business)

I gather that a major reason one still uses a mainframe is databases and
in particular database locking. Whi;e the aggregate throughput of a lot
of blades may be more than a single mainframe, when you need to do database
updates it's a lot faster if the updates are in one place rather than
trying to synchronize a lot of loosely coupled systems. On that 200
processor z16, you can do a CAS on one processor and the other 199
will see it consistently.

I realize there are ways around this, sharding and such, but there's good
reasons that the reading parts of database applications are widely
distributed while the writing parts are not.
--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anne &amp; Lynn Wheel on Sat, 01 Feb 2025 03:38:16 GMT
View Forum Message <> Reply to Message

John Levine <johnl@taugh.com> writes:
> I gather that a major reason one still uses a mainframe is databases and
> in particular database locking. Whi;e the aggregate throughput of a lot
> of blades may be more than a single mainframe, when you need to do database
> updates it's a lot faster if the updates are in one place rather than
> trying to synchronize a lot of loosely coupled systems. On that 200
> processor z16, you can do a CAS on one processor and the other 199
> will see it consistently.
>
> I realize there are ways around this, sharding and such, but there's good
> reasons that the reading parts of database applications are widely
> distributed while the writing parts are not.

Some of this left over from mid-90s where billions were spent on
rewritting mainframe financial systems that queued real time
transactions for processing during overnight batch settlement windows
(many from the 60s&70s) ... to straight-through processing using large
number of parallelized killer micros. Some of us pointed out that they
were using industry standard parallelization libraries that had hundred
times the overhead of mainframe cobol batch ... ignored until their
pilots went down in flames (retrenching to the existing status quo).

A major change was combination of 1) major RDBMS vendors (including IBM)
did significiant throughput performance work on cluster RDBMS, 2)
implementations done with fine-grain SQL statements that were highly
parallelized (rather than RYO implementation parallelization), and 3)
non-mainframe systems having significantly higher throughput.

2022 z16 200 processors shared memory multiprocessor and aggregate
222BIPS (1.11BIPS/processor) compared to (single blade) 2010 e5-2600 16
processors shared memory multiprocessor and aggregate 500BIPS
(31BIPS/processor) ... 2010 E5-2600 systen ten times 2022 z16 (2010
e5-2600 processor 30 times 2022 z16 processor). 2010 E5-2600 could have
up to eight processors (with aggregate throughput more than 2022 200
processor z16) sharing cache (w/o going all the way to memory).

When I has doing HA/CMP ... it originally started out HA/6000 for
NYTimes to move their newspaper system (ATEX) off DEC VAXCluster to
RS/6000. I rename it HA/CMP when start doing cluster scaleup with
national labs and RDBMS vendors (oracle, sybase, ingres, informix) that
had VAXCluster in same source base with UNIX.

I did distributed lock manager that emulated VAXCluster DLI semantics
.... but with a lot of enhancements. RDBMS had been doing cluster with
logging and lazy writes to home location ... however when write lock had

to move to a different system ... they had to force any pending associated write to RDBMS "home" location ... the system receiving the lock then would read the latest value from disk. Since I would be using gbit+ links, I did a DLI that could transmit both the lock ownership as well as the latest record(s) (avoiding the immediate disk write followed by immediate disk read on different system). Failures would recover from log records updating RDBMS "home" records.

However, still prefer to have transaction routing to processor (processor and/or cache affinity) holding current lock ... this is the observation that cache miss (all the main storage) when measured in count of processor cycles ... is similar to 60s disk latency when measured in 60s processor cycles (if purely for throughput).

I had also coined the terms "disaster survivability" and "geographically survivability" when out marketing HA/CMP ... so more processing for coordinate data at replicated distributed locations. Jim Gray's study of availability outages were increasingly shifting to human mistakes and environmental (as hardware was becoming increasingly reliable, I had worked with Jim Gray on System/R before he left IBM SJR for Tandem fall81).
https://www.garlic.com/~lynn/grayft84.pdf
'85 paper
 https://pages.cs.wisc.edu/~remzi/Classes/739/Fall2018/Papers /gray85-easy.pdf
 https://web.archive.org/web/20080724051051/http://www.cs.ber
keley.edu/~yelick/294-f00/papers/Gray85.txt

In 1988, the IBM branch asks if i could help LLNL (national lab) with standardization of some serial stuff they were working with, which quickly becomes fibre standard channel (FCS, initially 1gbit/sec transfer, full-duplex, aggregate 200Mbytes/sec, including some stuff I had done in 1980). POK ships their fiber stuff in 1990 with ES/9000 as ESCON (when it was already obsolete, 17mbytes/sec). Then some POK engineers become involved with FCS and define heavy weight protocol that significantly reduces throughput, ships as "FICON"

Latest public benchmark I could find was (2010) z196 "Peak I/O" getting 2M IOPS using 104 FICON. About the same time a "FCS" was announced for E5-2600 server blade claimining over million IOPS (two such FCS would have higher throughput than 104 FICON ... that run over FCS). Also, IBM pubs recommend that SAPs (system assist proceessors that actually do I/O) be kept to 70% CPU ... which would be more like 1.5M IOPS. Also no CKD DASD have been made for decades, all being simulated on industry standard fixed-block disks.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Sat, 01 Feb 2025 16:59:31 GMT
View Forum Message <> Reply to Message

Originally posted by: OrangeFish

On 2025-01-31 20:31, John Levine wrote (in part):
[...]
> I gather that a major reason one still uses a mainframe is databases and
> in particular database locking. Whi;e the aggregate throughput of a lot
> of blades may be more than a single mainframe, when you need to do database
> updates it's a lot faster if the updates are in one place rather than
> trying to synchronize a lot of loosely coupled systems. On that 200
> processor z16, you can do a CAS on one processor and the other 199
> will see it consistently.
>
> I realize there are ways around this, sharding and such, but there's good
> reasons that the reading parts of database applications are widely
> distributed while the writing parts are not.

Speaking of mainframes, Dave's Garage has an interesting video of the
IBM factory: https://www.youtube.com/watch?v=ouAG4vXFORc

OF.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by scott on Sat, 01 Feb 2025 18:35:30 GMT
View Forum Message <> Reply to Message

OrangeFish <OrangeFish@invalid.invalid> writes:
> On 2025-01-31 20:31, John Levine wrote (in part):
> [...]
>> I gather that a major reason one still uses a mainframe is databases and
>> in particular database locking. Whi;e the aggregate throughput of a lot
>> of blades may be more than a single mainframe, when you need to do database
>> updates it's a lot faster if the updates are in one place rather than
>> trying to synchronize a lot of loosely coupled systems. On that 200
>> processor z16, you can do a CAS on one processor and the other 199
>> will see it consistently.
>>
>> I realize there are ways around this, sharding and such, but there's good
>> reasons that the reading parts of database applications are widely
>> distributed while the writing parts are not.
>
> Speaking of mainframes, Dave's Garage has an interesting video of the
> IBM factory: https://www.youtube.com/watch?v=ouAG4vXFORc
>

Burroughs B6500 factory:
https://www.youtube.com/watch?v=rNBtjEBYFPk

Subject: Re: old pharts, Multics vs Unix
Posted by Peter Flass on Sat, 01 Feb 2025 19:08:41 GMT
View Forum Message <> Reply to Message

Dan Espen <dan1espen@gmail.com> wrote:
> Lynn Wheeler <lynn@garlic.com> writes:
>
>> however, all 3270s were half-duplex and if you were unfortunate to hit a
>> key same time system went to write to screen, it would lock the keyboard
>> and would have to stop and hit the reset key. YKT developed a FIFO box
>> for 3277, unplug the keyboard from the 3277 head, plug the FIFO box into
>> the head and plug the 3277 keyboard into the fifo box ... eliminating
>> the unfortunate keyboad lock.
>
> Back in the late 60s I finished implementing my first online system on a
> S/360-30 and IBM 2260s.
>
> Then my boss came in with the manuals for the IBM 3270s.
> He wanted my opinion on the terminals.
> I read the manual and told my boss they were unnecessary complicated
> crap.
>
> Over the years I did lots of stuff with 3270s. I never changed my mind.
>
> One project I did was using Bunker Ramo 3270 compatible terminals.
> By mistake I wrote some code that put more than 80 characters on a line.
> The Bunker Ramo CRT just squeezed the text a bit. If I recall you could
> put around 120 characters on a line and still read the display.
>

I loved 3270s. I never coded anything for 2260s, but the whole design was a
kludge.I did a lot with 3270 BMS and raw data stream, and i still miss
using them.

--
Pete

Subject: Re: old pharts, Multics vs Unix
Posted by Peter Flass on Sat, 01 Feb 2025 19:08:42 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Thu, 30 Jan 2025 19:45:05 -0000 (UTC), John Levine wrote:
>
>> You'd use it for the same reason you'd use any other mainframe,
>> extremely high reliability with uptime measured in years and sometimes
>> decades. They can swap out entire hardware subsystems without
>> rebooting.
>
> That's all a complete myth.
>
> There is an article from 1986 on Bitsavers, talking about maintaining
> correct time on IBM mainframes. It recommends rebooting to turn daylight
> saving on and off.
>

40 year old article.

--
Pete

Subject: Re: old pharts, Multics vs Unix
Posted by Peter Flass on Sat, 01 Feb 2025 19:08:43 GMT

View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Thu, 30 Jan 2025 10:12:15 -0500, Dan Espen wrote:
>
>> ... I wouldn't call z/OS an emulation layer. It looked and acted like a
>> Unix implementation.
>
> Not a very good one, by your own admission:
>
>> One a mainframe, there are a few issues to deal with to run Unix.
>> The common use terminal, a 3270 is not character at a time, data is
>> transferred in blocks with a pretty complex protocol. z/OS unix
>> couldn't do things like run Emacs on a 3270 but it did a reasonably good
>> job of providing a working stdin/stdout.
>
> Couldn't even run Emacs?? What kind of “Unix” is this?

What the frack is Emacs? I've been using forms for unix for decades, and
have managed to avoid it so far.

>
>> Another issue is the native disk file system where data is read in
>> blocks with predefined block sizes. z/OS Unix also did a pretty good
>> job with this.

>
> What did “pretty good job” mean, exactly? Could it reliably and
> efficiently handle files with sizes that were not an exact multiple of
> some block size, or not?
>
> How efficiently could it fork multiple processes?
>

--
Pete

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Peter Flass on Sat, 01 Feb 2025 19:08:44 GMT
View Forum Message <> Reply to Message

John Levine <johnl@taugh.com> wrote:
> According to Bob Eager <news0009@eager.cx>:
>> On Fri, 31 Jan 2025 10:24:01 -0500, Dan Espen wrote:
>>
>>>> > IBM is still selling mainframe hardware.
>>>>
>>>> I doubt they’re making any profit on it any more.
>>>
>>> That's something you could look up for yourself.
>>
>> I understand that all the profit has been in software, for many years.
>
> IBM continues to put a lot of effort into their mainframe hardware. If
> you look at their financial reports, they usually have a bullet for "IBM Z"
> which is up some quarters, down others. The most recent slide deck has a
> bullet that says:
>
> z16 our most successful program in history
>
> I agree that mainframes is a legacy business but it's one that still has
> a long life aheade it it.
>

No one is going to watch to convert 60 years of production software to
another platform. Actually, i take that back. It’s been tried, but never
successfully.

--
Pete

Subject: Re: old pharts, Multics vs Unix
Posted by Peter Flass on Sat, 01 Feb 2025 19:08:45 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Fri, 31 Jan 2025 10:24:01 -0500, Dan Espen wrote:
>
>> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>>
>>> If you haven’t completely depreciated all that by now, then you’re
>>> probably out of business or heading that way.
>>
>> I'm sorry, "depreciated"?
>> What are you talking about?
>
> You know, one of those amounts you offset against gross income to come up
> with net profit (or loss).
>
> Or, if you want it in simpler terms, “written off”.
>
>> There are people running their businesses with their software.
>
> Computing needs have changed a lot. COBOL was designed for the batch era.
> We no longer operate our businesses in the batch era.
>

Ever heard of CICS?

--
Pete

Subject: Re: old pharts, Multics vs Unix
Posted by Peter Flass on Sat, 01 Feb 2025 19:08:47 GMT
View Forum Message <> Reply to Message

Scott Lurndal <scott@slp53.sl.home> wrote:
> Lawrence D'Oliveiro <ldo@nz.invalid> writes:
>> On Fri, 31 Jan 2025 09:44:31 -0800, John Ames wrote:
>>
>>> On Fri, 31 Jan 2025 06:33:03 -0000 (UTC) Lawrence D'Oliveiro
>>> <ldo@nz.invalid> wrote:
>>>
>>>> >> How efficiently could it fork multiple processes?
>>>> >
>>>> > Never saw a problem with that.
>>>>
>>>> Somehow I doubt you tested it very heavily.

>>>
>>> What's your basis for that assertion?
>
>>
>> The idea that a mainframe system, of all things, could handle process
>> creation efficiently, is laughable.
>
> More ravings from a troll who has never even used a mainframe
> operating system, much less actually written one.
>
> Process creation on Burroughs mainframes was certainly
> competitive with fork[*]. I speak from experience on both
> ends (having written both mainframe operating systems
> and an unix-compatible MPP operating system (specifically responsible
> for the process create and management code). I have
> patents related to the latter.
>
> [*] In fact, the algol (MCP flavor) routine to create a
> new process on the Burroughs Large systems was actually
> named 'motherforker'.   On Medium systems, the early MCP
> function that created a new context was called 'hiho'.   As
> in Hi-Ho, Hi-Ho, it's off to work we go.
>

Burroughs naming was always creative, rivaled only by SDS/XDS.

--
Pete

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Sat, 01 Feb 2025 19:49:05 GMT
View Forum Message <> Reply to Message

Peter Flass <peter_flass@yahoo.com> writes:

> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>> On Thu, 30 Jan 2025 10:12:15 -0500, Dan Espen wrote:
>>
>>> ... I wouldn't call z/OS an emulation layer. It looked and acted like a
>>> Unix implementation.
>>
>> Not a very good one, by your own admission:
>>
>>> One a mainframe, there are a few issues to deal with to run Unix.
>>> The common use terminal, a 3270 is not character at a time, data is
>>> transferred in blocks with a pretty complex protocol. z/OS unix
>>> couldn't do things like run Emacs on a 3270 but it did a reasonably good

>>> job of providing a working stdin/stdout.
>>
>> Couldn’t even run Emacs?? What kind of “Unix” is this?
>
> What the frack is Emacs? I’ve been using forms for unix for decades, and
> have managed to avoid it so far.

Emacs was just an example. vi would also not run.
Anything that needed to react to a single character being typed would
not run.

A 3270 only transfers data when you hit enter, a PK key, hit attention,
just a few select keys.

The async terminals normally only react when you hit enter, (which
transmits a line), but they can be put in raw mode where the host
sees every character typed. The full screen editors rely on this.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by jgd on Sat, 01 Feb 2025 19:55:49 GMT

View Forum Message <> Reply to Message

In article <vnjfai$3mhvf$4@dont-email.me>, ldo@nz.invalid (Lawrence
D'Oliveiro) wrote:

> Back in 1980, there were something like 10 different companies
> operating in the mainframe business. Now there is only one.

Nope. IBM is the market leader, sure. In Japan, Fujitsu and Hitachi
continue to sell IBM-like 31-bit mainframes, and NEC sells a proprietary
mainframe. The ex-Univac OS 2200, ex-Burroughs MCP, Groupe Bull GCOS,
ex-Siemens BS2000 and ex-ICL VME are all still used, mostly under
emulation.

Yes, the mainframe business is shrinking, but it will be around for
decades yet.

John

Subject: Re: old pharts, Multics vs Unix
Posted by Peter Flass on Sat, 01 Feb 2025 21:49:51 GMT

View Forum Message <> Reply to Message

Dan Espen <dan1espen@gmail.com> wrote:
> Peter Flass <peter_flass@yahoo.com> writes:
>
>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>> On Thu, 30 Jan 2025 10:12:15 -0500, Dan Espen wrote:
>>>
>>>> ... I wouldn't call z/OS an emulation layer. It looked and acted like a
>>>> Unix implementation.
>>>
>>> Not a very good one, by your own admission:
>>>
>>>> One a mainframe, there are a few issues to deal with to run Unix.
>>>> The common use terminal, a 3270 is not character at a time, data is
>>>> transferred in blocks with a pretty complex protocol. z/OS unix
>>>> couldn't do things like run Emacs on a 3270 but it did a reasonably good
>>>> job of providing a working stdin/stdout.
>>>
>>> Couldn't even run Emacs?? What kind of "Unix" is this?
>>
>> What the frack is Emacs? I've been using forms for unix for decades, and
>> have managed to avoid it so far.
>
> Emacs was just an example. vi would also not run.
> Anything that needed to react to a single character being typed would
> not run.
>
> A 3270 only transfers data when you hit enter, a PK key, hit attention,
> just a few select keys.
>
> The async terminals normally only react when you hit enter, (which
> transmits a line), but they can be put in raw mode where the host
> sees every character typed. The full screen editors rely on this.
>

I always thought that this was a tremendously wasteful scheme, generating
tons of interrupts instead of only one or two. As I said, I liked the 3270,
I thought ISPF was one of the best editors I've ever used. Now my Linux
editor is the closest I could find to ISPF, but there are still features I
miss. I've thought of adding a Rexx interface, but it's a ways down in my
priority list.

--
Pete

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Sat, 01 Feb 2025 22:05:35 GMT

Peter Flass <peter_flass@yahoo.com> writes:

> Dan Espen <dan1espen@gmail.com> wrote:
>> Peter Flass <peter_flass@yahoo.com> writes:
>>
>>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>>> On Thu, 30 Jan 2025 10:12:15 -0500, Dan Espen wrote:
>>>>
>>>> > ... I wouldn't call z/OS an emulation layer. It looked and acted like a
>>>> > Unix implementation.
>>>>
>>>> Not a very good one, by your own admission:
>>>>
>>>> > One a mainframe, there are a few issues to deal with to run Unix.
>>>> > The common use terminal, a 3270 is not character at a time, data is
>>>> > transferred in blocks with a pretty complex protocol. z/OS unix
>>>> > couldn't do things like run Emacs on a 3270 but it did a reasonably good
>>>> > job of providing a working stdin/stdout.
>>>>
>>>> Couldn't even run Emacs?? What kind of "Unix" is this?
>>>
>>> What the frack is Emacs? I've been using forms for unix for decades, and
>>> have managed to avoid it so far.
>>
>> Emacs was just an example. vi would also not run.
>> Anything that needed to react to a single character being typed would
>> not run.
>>
>> A 3270 only transfers data when you hit enter, a PK key, hit attention,
>> just a few select keys.
>>
>> The async terminals normally only react when you hit enter, (which
>> transmits a line), but they can be put in raw mode where the host
>> sees every character typed. The full screen editors rely on this.
>
> I always thought that this was a tremendously wasteful scheme, generating
> tons of interrupts instead of only one or two. As I said, I liked the 3270,
> I thought ISPF was one of the best editors I've ever used. Now my Linux
> editor is the closest I could find to ISPF, but there are still features I
> miss. I've thought of adding a Rexx interface, but it's a ways down in my
> priority list.

IBM thought it wasteful too, that's why they developed all that block
mode stuff. It's still pretty cool to be able to hit one key and have
stuff happen.

No argument from me about ISPF edit. It's pretty neat.

Still, when I was given the choice, I did all my mainframe editing on my
Linux box.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by scott on Sat, 01 Feb 2025 22:15:50 GMT

View Forum Message <> Reply to Message

```
Dan Espen <dan1espen@gmail.com> writes:
> Peter Flass <peter_flass@yahoo.com> writes:
>
>>  Dan Espen <dan1espen@gmail.com> wrote:
>>>  Peter Flass <peter_flass@yahoo.com> writes:
>>>
>>>>  Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>>> > On Thu, 30 Jan 2025 10:12:15 -0500, Dan Espen wrote:
>>>> >
>>>> >> ... I wouldn't call z/OS an emulation layer. It looked and acted like a
>>>> >> Unix implementation.
>>>> >
>>>> > Not a very good one, by your own admission:
>>>> >
>>>> >> One a mainframe, there are a few issues to deal with to run Unix.
>>>> >> The common use terminal, a 3270 is not character at a time, data is
>>>> >> transferred in blocks with a pretty complex protocol. z/OS unix
>>>> >> couldn't do things like run Emacs on a 3270 but it did a reasonably good
>>>> >> job of providing a working stdin/stdout.
>>>> >
>>>> > Couldn’t even run Emacs?? What kind of “Unix” is this?
>>>>
>>>>  What the frack is Emacs? I’ve been using forms for unix for decades, and
>>>>  have managed to avoid it so far.
>>>
>>>  Emacs was just an example.  vi would also not run.
>>>  Anything that needed to react to a single character being typed would
>>>  not run.
>>>
>>>  A 3270 only transfers data when you hit enter, a PK key, hit attention,
>>>  just a few select keys.
>>>
>>>  The async terminals normally only react when you hit enter, (which
>>>  transmits a line), but they can be put in raw mode where the host
>>>  sees every character typed.  The full screen editors rely on this.
>>
>>  I always thought that this was a tremendously wasteful scheme, generating
```

>> tons of interrupts instead of only one or two. As I said, I liked the 3270,
>> I thought ISPF was one of the best editors I’ve ever used. Now my Linux
>> editor is the closest I could find to ISPF, but there are still features I
>> miss. I’ve thought of adding a Rexx interface, but it’s a ways down in my
>> priority list.
>
> IBM thought it wasteful too, that's why they developed all that block
> mode stuff. It's still pretty cool to be able to hit one key and have
> stuff happen.

Burroughs also used, primarily, block-mode terminals.   Teletypes were
supported (the I/O controller collected 'lines' and transferred them
to the host).   Data Communications processors had line control modules
that supported async (e.g. tty, modem) and sync (block-mode, multidrop)
terminals which would packet up the transmission and pass it to the host
MCS which would route the transmission to the appropriate program for
handling (using read-only screen fields to route).

None of those block mode editors can be compared with vim or emacs
from a usability or functionality standpoint.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Sun, 02 Feb 2025 00:58:36 GMT

View Forum Message <> Reply to Message

Originally posted by: antispam

Peter Flass <peter_flass@yahoo.com> wrote:
> John Levine <johnl@taugh.com> wrote:
>> According to Bob Eager <news0009@eager.cx>:
>>> On Fri, 31 Jan 2025 10:24:01 -0500, Dan Espen wrote:
>>>
>>>> >> IBM is still selling mainframe hardware.
>>>> >
>>>> > I doubt they’re making any profit on it any more.
>>>>
>>>> That's something you could look up for yourself.
>>>
>>> I understand that all the profit has been in software, for many years.
>>
>> IBM continues to put a lot of effort into their mainframe hardware. If
>> you look at their financial reports, they usually have a bullet for "IBM Z"
>> which is up some quarters, down others. The most recent slide deck has a
>> bullet that says:
>>
>>   z16 our most successful program in history
>>

>> I agree that mainframes is a legacy business but it's one that still has
>> a long life aheade it it.
>>
>
> No one is going to watch to convert 60 years of production software to
> another platform. Actually, i take that back. It's been tried, but never
> successfully.

I think that there were a lot of successful migrations from
mainframes to other systems. It is likely that most succesful
cases were gradual, taking substantial time.

In Poland during comunist times one of basic machines were
domesticaly produced ICL-1900 compatible machines, using ICL
operating system. It seems that our manufacturer of those
machines stopped production around 1990 (and during eighties
production was decreasing). For few years alternative
manufacturer offered compatible machines using bit-slice
microprocessors. Few years ago supposedly last machine
of this family was decomissioned (it was used by railway
as part of trafic control).

So, it is basically question of time and financial balance:
currently mainframe users pay higer prices to mainframe
vendors so that users can keep using their systems and
avoid migration. But as user population shrinks it may
become too small to adequatly support vendors. That may
force vendors to limit their offer leading to colapse:
limited vendor offer accelerates migration. That seem
to happened with several smaller vendors. IBM-compatible
world currently looks relatively healthy, but it is not
clear for how long. I expect at least 10 more years of use
of IBM-compatible mainframes, but it is hard to make
prediction for longer time. 10 years may look like
long time, but it is short compared to 60 year history.

--
Waldek Hebisch

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Sun, 02 Feb 2025 01:09:07 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sat, 1 Feb 2025 12:08:42 -0700, Peter Flass wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>
>> On Thu, 30 Jan 2025 19:45:05 -0000 (UTC), John Levine wrote:
>>
>>> You'd use it for the same reason you'd use any other mainframe,
>>> extremely high reliability with uptime measured in years and sometimes
>>> decades. They can swap out entire hardware subsystems without
>>> rebooting.
>>
>> That's all a complete myth.
>>
>> There is an article from 1986 on Bitsavers, talking about maintaining
>> correct time on IBM mainframes. It recommends rebooting to turn
>> daylight saving on and off.
>>
> 40 year old article.

That was from the heyday of mainframes, don't forget.

You think they've improved since then?

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Sun, 02 Feb 2025 01:11:19 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sat, 1 Feb 2025 12:08:44 -0700, Peter Flass wrote:

> No one is going to watch to convert 60 years of production software to
> another platform. Actually, i take that back. It's been tried, but never
> successfully.

More likely the company that is chained to that mainframe albatross goes
out of business, and its systems get junked instead of converted.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Sun, 02 Feb 2025 01:13:24 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Fri, 31 Jan 2025 14:26:40 -0800, John Ames wrote:

> On Fri, 31 Jan 2025 21:30:10 -0000 (UTC)

> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>
>>> What's your basis for that assertion?
>>
>> It’s well known that the Unix creation-of-lots-of-processes model plays
>> poorly with every single proprietary OS out there.
>>
>> The idea that a mainframe system, of all things, could handle process
>> creation efficiently, is laughable.
>
> So your basis for casting doubt on his specific attestation of personal
> experience is "everybody knows?"

In the absence of more detail being supplied, that is the logical
conclusion.

Here another example of something needing clarification: the POSIX fork(2)
call requires the child process to share the same open file descriptions
as the parent.

Name one proprietary (non-*nix) OS that manages to emulate that
successfully.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Sun, 02 Feb 2025 01:17:02 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sat, 1 Feb 2025 14:49:51 -0700, Peter Flass wrote:

> I always thought that this was a tremendously wasteful scheme,
> generating tons of interrupts instead of only one or two.

That’s the mainframe batch mentality speaking.

Yes, it was tremendously wasteful of computer time. But it was a key
factor in improving productivity of the human users. That’s why companies
like DEC, which sold hardware+software systems that were designed from the
ground up to operate in such a mode, were so successful.

IBM could never compete with them on the quality of its products.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Sun, 02 Feb 2025 01:20:46 GMT

View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sat, 1 Feb 2025 01:31:37 -0000 (UTC), John Levine wrote:

> I gather that a major reason one still uses a mainframe is databases and
> in particular database locking.

Ironic, actually, but no. You think a modern company like Facebook runs
its main system on a mainframe, using some proprietary mainframe DBMS? No,
it uses MySQL/MariaDB with other pieces like memcached, plus code written
in its home-grown PHP engine (open-sourced as HHVM), and also some back-
end Python (that we know of).

The irony is that COBOL, the “business-oriented” language traditionally
associated with mainframes, never quite got to grips with databases.
Accessing them was always a kludge.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Sun, 02 Feb 2025 03:04:25 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

On Sat, 1 Feb 2025 12:08:44 -0700, Peter Flass wrote:
>> No one is going to watch to convert 60 years of production software to
>> another platform. Actually, i take that back. It’s been tried, but never
>> successfully.

On 2025-02-02, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> More likely the company that is chained to that mainframe albatross goes
> out of business, and its systems get junked instead of converted.

I Denmark, the main users of mainframes are:
- all the banks
- state tax administration
- motor vehicle registry

They will not go out of business.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Sun, 02 Feb 2025 03:13:41 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 2 Feb 2025 03:04:25 -0000 (UTC), Lars Poulsen wrote:

> On 2025-02-02, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>
>> More likely the company that is chained to that mainframe albatross
>> goes out of business, and its systems get junked instead of converted.
>
> I Denmark, the main users of mainframes are:
> - all the banks
> - state tax administration
> - motor vehicle registry

I can imagine the last two could just about manage on batch operations,
but the banks cannot. Otherwise they cannot handle online transactions
which require pretty much instant updates.

Subject: Re: old pharts, Multics vs Unix
Posted by John Levine on Sun, 02 Feb 2025 03:21:21 GMT
View Forum Message <> Reply to Message

It appears that Dan Espen <dan1espen@gmail.com> said:
>>> A 3270 only transfers data when you hit enter, a PK key, hit attention,
>>> just a few select keys.
>>>
>>> The async terminals normally only react when you hit enter, (which
>>> transmits a line), but they can be put in raw mode where the host
>>> sees every character typed. The full screen editors rely on this.
>>
>> I always thought that this was a tremendously wasteful scheme, generating
>> tons of interrupts instead of only one or two. ...

It just changes where you put your hardware. On minicomputers with lightweight
interupts, it makes sense to make the controller simpler and do the character
buffering in the computer. As an extreme example, DEC's 680 TTY interface
took an interrupt not for each character, but for each bit on a serial line,
a total of 10 or 11 bits including start/stop. For 110 baud TTYs it worked
great, attaching a bunch of terminals with very little hardware. A PDP-8
was fast enough that it could do that and also run TSS-8, a nice little
timesharing system that gave everone a virtual PDP-8.

--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Sun, 02 Feb 2025 06:03:08 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sat, 01 Feb 2025 17:05:35 -0500, Dan Espen wrote:

> IBM thought it wasteful too, that's why they developed all that block
> mode stuff.

No matter how clever and complicated (and expensive) their terminals were,
it could never make up for the versatility of putting every keystroke
under software control.

> Still, when I was given the choice, I did all my mainframe editing on my
> Linux box.

Which proves my point.

Subject: Re: old pharts, Multics vs Unix
Posted by Charlie Gibbs on Sun, 02 Feb 2025 06:09:51 GMT
View Forum Message <> Reply to Message

On 2025-02-01, Scott Lurndal <scott@slp53.sl.home> wrote:

> Burroughs also used, primarily, block-mode terminals.

Univac did as well. A good editor would present you with
a screenful of data; you could edit it using the terminal's
built-in features (insert, delete, or overwrite characters
or lines) and send the updated screenful of data back to
the mainframe. It wasn't a full character-by-character
response, but it was tolerable.

I ported Adventure and Dungeon to OS/3; one of the most
difficult tasks was getting terminal I/O to work properly.

```
--
/~\  Charlie Gibbs                  |  Growth for the sake of
\ /  <cgibbs@kltpzyxm.invalid>      |  growth is the ideology
 X   I'm really at ac.dekanfrus     |  of the cancer cell.
/ \  if you read it the right way.  |    -- Edward Abbey
```

Subject: Re: old pharts, Multics vs Unix

Posted by Anonymous on Sun, 02 Feb 2025 12:24:04 GMT
View Forum Message <> Reply to Message

Originally posted by: Bob Eager

On Sun, 02 Feb 2025 06:09:51 +0000, Charlie Gibbs wrote:

> On 2025-02-01, Scott Lurndal <scott@slp53.sl.home> wrote:
>
>> Burroughs also used, primarily, block-mode terminals.
>
> Univac did as well. A good editor would present you with a screenful of
> data; you could edit it using the terminal's built-in features (insert,
> delete, or overwrite characters or lines) and send the updated screenful
> of data back to the mainframe. It wasn't a full character-by-character
> response, but it was tolerable.
>
> I ported Adventure and Dungeon to OS/3; one of the most difficult tasks
> was getting terminal I/O to work properly.

The operating system I helped to support for some years handled terminal
I/O in a PDP-11 used as a front end processor (sometimes more than one).
Characters were sent via an interface that used single characters or not,
as needed.

--
Using UNIX since v6 (1975)...

Use the BIG mirror service in the UK:
http://www.mirrorservice.org

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Sun, 02 Feb 2025 12:28:15 GMT
View Forum Message <> Reply to Message

Originally posted by: Bob Eager

On Sun, 02 Feb 2025 00:58:36 +0000, Waldek Hebisch wrote:

> In Poland during comunist times one of basic machines were domesticaly
> produced ICL-1900 compatible machines, using ICL operating system. It
> seems that our manufacturer of those machines stopped production around
> 1990 (and during eighties production was decreasing). For few years
> alternative manufacturer offered compatible machines using bit-slice
> microprocessors. Few years ago supposedly last machine of this family

> was decomissioned (it was used by railway as part of trafic control).

ICL were still selling 1900 compatibility (in two different forms) well
into the 1980s. Plus hardware that was essentially a 1900, but branded
differently.

This was on their 2900 series. One was Direct Machine Environment (DME)
where the machine was effectively a 1900. The other was mixed, where the
machine switched dynamically between 1900 and 2900 mode (it could also do
LEO and System 4 if required).

All of the machines were microcoded; I have seen quite a bit of the
microcode. It's the only time I have seen microcode overlaid (from main
memory).

--
Using UNIX since v6 (1975)...

Use the BIG mirror service in the UK:
 http://www.mirrorservice.org

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Sun, 02 Feb 2025 13:59:19 GMT

View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

On 2025-02-02, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Sun, 2 Feb 2025 03:04:25 -0000 (UTC), Lars Poulsen wrote:
>
>> On 2025-02-02, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>
>>> More likely the company that is chained to that mainframe albatross
>>> goes out of business, and its systems get junked instead of converted.
>>
>> I Denmark, the main users of mainframes are:
>> - all the banks
>> - state tax administration
>> - motor vehicle registry
>
> I can imagine the last two could just about manage on batch operations,
> but the banks cannot. Otherwise they cannot handle online transactions
> which require pretty much instant updates.
>

Denmark did away with checks a decade ago, and is headed towards
eliminating cash as well. Think 6-year olds with debit cards for their
allowances.

Here in the US, we still use checks. In fact, when I am moving money
between banks, I go into my branch of the big national bank, get a
cashier's check for some multiple of USD 50 000 and hand carry it to my
branch of the small regional bank to deposit it there.

If I were to do the transfer electronically, it would cost about USD 30,
and take about 36 hours before the money was available in the new
account. Doing it my way, it is free and the money is available when I
leave the branch.

The USA is stuck in the past, waxing nostalgic for the golden days of
the 1950s. And rapidly becoming a lawless 3rd world country.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Sun, 02 Feb 2025 14:07:13 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

It appears that Dan Espen <dan1espen@gmail.com> said:
```
>>>>  A 3270 only transfers data when you hit enter, a PK key, hit attention,
>>>>  just a few select keys.
>>>>
>>>>  The async terminals normally only react when you hit enter, (which
>>>>  transmits a line), but they can be put in raw mode where the host
>>>>  sees every character typed.  The full screen editors rely on this.
>>>
>>>  I always thought that this was a tremendously wasteful scheme, generating
>>>  tons of interrupts instead of only one or two. ...
```

The IBM 3270 scheme was very efficient for CICS transaction processing,
offloading memory for both multi-step transaction context and queueing
at peak load to the terminal network.

The context could be held in non-displaying data fields in the terminal
screen buffer, so that each incremental step in the conversation could
be processed independently. And since terminals only transmitted when
polled, any backlogged transactions never got into the mainframe memory,
but stayed in the terminal until it could be processed. So at peak
times, response times went up, buteverything kept running within
available capacity.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by jgd on Sun, 02 Feb 2025 15:43:55 GMT
View Forum Message <> Reply to Message

In article <vnmh9e$butt$6@dont-email.me>, ldo@nz.invalid (Lawrence D'Oliveiro) wrote:
> On Sat, 1 Feb 2025 01:31:37 -0000 (UTC), John Levine wrote:
>> I gather that a major reason one still uses a mainframe is
>> databases and in particular database locking.
>
> Ironic, actually, but no. You think a modern company like Facebook
> runs its main system on a mainframe, using some proprietary
> mainframe DBMS? No, it uses MySQL/MariaDB with other pieces like
> memcached, plus code written in its home-grown PHP engine
> (open-sourced as HHVM), and also some back-end Python (that we
> know of).

Facebook has quite different synchronisation requirements from a credit card provider. It doesn't matter to FB if updates to the page someone is looking at arrive take a few seconds to arrive.

Speed of synchronisation matters a lot to a credit card provider who is trying to enforce customer credit limits and avoid double-spends. They still use mainframes with z/TPF for that. z/TPF is a curious OS; it essentially makes a mainframe into a single real-time transaction processing system.

John

Subject: Re: old pharts, Multics vs Unix
Posted by Harry Vaderchi on Sun, 02 Feb 2025 17:56:02 GMT
View Forum Message <> Reply to Message

On Sat, 01 Feb 2025 17:05:35 -0500
Dan Espen <dan1espen@gmail.com> wrote:

> Peter Flass <peter_flass@yahoo.com> writes:
>
>> Dan Espen <dan1espen@gmail.com> wrote:
>>> Peter Flass <peter_flass@yahoo.com> writes:
>>>
>>>> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>>>> > On Thu, 30 Jan 2025 10:12:15 -0500, Dan Espen wrote:
>>>> >
>>>> >> ... I wouldn't call z/OS an emulation layer. It looked and acted like a
>>>> >> Unix implementation.
>>>> >

>>>> > Not a very good one, by your own admission:
>>>> >
>>>> >> One a mainframe, there are a few issues to deal with to run Unix.
>>>> >> The common use terminal, a 3270 is not character at a time, data is
>>>> >> transferred in blocks with a pretty complex protocol. z/OS unix
>>>> >> couldn't do things like run Emacs on a 3270 but it did a reasonably good
>>>> >> job of providing a working stdin/stdout.
>>>> >
>>>> > Couldn't even run Emacs?? What kind of "Unix" is this?
>>>>
>>>> What the frack is Emacs? I've been using forms for unix for decades, and
>>>> have managed to avoid it so far.
>>>
>>> Emacs was just an example. vi would also not run.
>>> Anything that needed to react to a single character being typed would
>>> not run.
>>>
>>> A 3270 only transfers data when you hit enter, a PK key, hit attention,
>>> just a few select keys.
>>>
>>> The async terminals normally only react when you hit enter, (which
>>> transmits a line), but they can be put in raw mode where the host
>>> sees every character typed. The full screen editors rely on this.
>>
>> I always thought that this was a tremendously wasteful scheme, generating
>> tons of interrupts instead of only one or two. As I said, I liked the 3270,
>> I thought ISPF was one of the best editors I've ever used. Now my Linux
>> editor is the closest I could find to ISPF, but there are still features I
>> miss. I've thought of adding a Rexx interface, but it's a ways down in my
>> priority list.
>
> IBM thought it wasteful too, that's why they developed all that block
> mode stuff. It's still pretty cool to be able to hit one key and have
> stuff happen.
>
> No argument from me about ISPF edit. It's pretty neat.
> Still, when I was given the choice, I did all my mainframe editing on my
> Linux box.

Back in the day, I was involved, (as a junior) in a project to "integrate"
Pcs and mainframes; - in the limited sense of a (Cobol) DevShop using PCs
for editing programs and testing (pseudo-transparently) via mainframe
compilers.

We didn't really foresee beyond offloading mainframe development load.

--

Bah, and indeed Humbug.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Harry Vaderchi on Sun, 02 Feb 2025 18:04:23 GMT
View Forum Message <> Reply to Message

On Sun, 2 Feb 2025 00:58:36 -0000 (UTC)
antispam@fricas.org (Waldek Hebisch) wrote:

```
>  Peter Flass <peter_flass@yahoo.com> wrote:
>>  John Levine <johnl@taugh.com> wrote:
>>>  According to Bob Eager  <news0009@eager.cx>:
>>>>  On Fri, 31 Jan 2025 10:24:01 -0500, Dan Espen wrote:
>>>>
>>>> >>> IBM is still selling mainframe hardware.
>>>> >>
>>>> >> I doubt they're making any profit on it any more.
>>>> >
>>>> > That's something you could look up for yourself.
>>>>
>>>>  I understand that all the profit has been in software, for many years.
>>>
>>>  IBM continues to put a lot of effort into their mainframe hardware.  If
>>>  you look at their financial reports, they usually have a bullet for "IBM Z"
>>>  which is up some quarters, down others.  The most recent slide deck has a
>>>  bullet that says:
>>>
>>>    z16 our most successful program in history
>>>
>>>  I agree that mainframes is a legacy business but it's one that still has
>>>  a long life aheade it it.
>>>
>>
>>  No one is going to watch to convert 60 years of production software to
>>  another platform. Actually, i take that back. It's been tried, but never
>>  successfully.
>
>  I think that there were a lot of successful migrations from
>  mainframes to other systems.  It is likely that most succesful
>  cases were gradual, taking substantial time.
>
>  In Poland during comunist times one of basic machines were
>  domesticaly produced ICL-1900 compatible machines, using ICL
>  operating system.  It seems that our manufacturer of those
>  machines stopped production around 1990 (and during eighties
>  production was decreasing).  For few years alternative
>  manufacturer offered compatible machines using bit-slice
```

> microprocessors. Few years ago supposedly last machine
> of this family was decomissioned (it was used by railway
> as part of trafic control).
>
> So, it is basically question of time and financial balance:
> currently mainframe users pay higer prices to mainframe
> vendors so that users can keep using their systems and
> avoid migration. But as user population shrinks it may
> become too small to adequatly support vendors. That may
> force vendors to limit their offer leading to colapse:
> limited vendor offer accelerates migration. That seem
> to happened with several smaller vendors. IBM-compatible
> world currently looks relatively healthy, but it is not
> clear for how long. I expect at least 10 more years of use
> of IBM-compatible mainframes, but it is hard to make
> prediction for longer time. 10 years may look like
> long time, but it is short compared to 60 year history.
>
No one expected 2 digit year field programs written in 1980 to
still be in production in 20 years time.

--
Bah, and indeed Humbug.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by scott on Sun, 02 Feb 2025 18:23:01 GMT
View Forum Message <> Reply to Message

"Kerr-Mudd, John" <admin@127.0.0.1> writes:
> On Sun, 2 Feb 2025 00:58:36 -0000 (UTC)
> antispam@fricas.org (Waldek Hebisch) wrote:
>
>> Peter Flass <peter_flass@yahoo.com> wrote:
>>> John Levine <johnl@taugh.com> wrote:
>>>> According to Bob Eager <news0009@eager.cx>:
>>>> > On Fri, 31 Jan 2025 10:24:01 -0500, Dan Espen wrote:
>>>> >
>>>> >>>> IBM is still selling mainframe hardware.

>> So, it is basically question of time and financial balance:
>> currently mainframe users pay higer prices to mainframe
>> vendors so that users can keep using their systems and
>> avoid migration. But as user population shrinks it may
>> become too small to adequatly support vendors. That may
>> force vendors to limit their offer leading to colapse:
>> limited vendor offer accelerates migration. That seem
>> to happened with several smaller vendors. IBM-compatible

>> world currently looks relatively healthy, but it is not
>> clear for how long. I expect at least 10 more years of use
>> of IBM-compatible mainframes, but it is hard to make
>> prediction for longer time. 10 years may look like
>> long time, but it is short compared to 60 year history.
>>
> No one expected 2 digit year field programs written in 1980 to
> still be in production in 20 years time.

At Burroughs, we fully expected 2-digit year field programs
written in 1968 to be still in production by 2000 when we
started the Y2K mitigation work in the mid 1980s.

1960 was used as the dividing line - any two digit year smaller
was assumed to be 21st century; any larger was assumed to be
20th century.

Subject: Re: old pharts, Multics vs Unix
Posted by cross on Sun, 02 Feb 2025 18:32:34 GMT

View Forum Message <> Reply to Message

In article <m096f4Fp4ruU1@mid.individual.net>,
Bob Eager <news0009@eager.cx> wrote:
> On Sun, 02 Feb 2025 06:09:51 +0000, Charlie Gibbs wrote:
>
>> On 2025-02-01, Scott Lurndal <scott@slp53.sl.home> wrote:
>>
>>> Burroughs also used, primarily, block-mode terminals.
>>
>> Univac did as well. A good editor would present you with a screenful of
>> data; you could edit it using the terminal's built-in features (insert,
>> delete, or overwrite characters or lines) and send the updated screenful
>> of data back to the mainframe. It wasn't a full character-by-character
>> response, but it was tolerable.
>>
>> I ported Adventure and Dungeon to OS/3; one of the most difficult tasks
>> was getting terminal I/O to work properly.
>
> The operating system I helped to support for some years handled terminal
> I/O in a PDP-11 used as a front end processor (sometimes more than one).
> Characters were sent via an interface that used single characters or not,
> as needed.

Multics did something similar; again, the theory was to protect
the main CPU(s) from a flurry of interrupts as users typed away;
instead, shuttle all of that overhead off to some satellite
processor (the General Input/Output Controller; GIOC, mentioned

here: https://multicians.org/rjf.html) and only send data when
the user's entered an entire line (they weren't exactly record
oriented; more line oriented).

Then Bernie Greenberg wrote an implementation of emacs for
Multics (in Lisp, no less!) and suddenly there was a need to
have the computer react to individual keystrokes; this led to
a notion of "Multics Echo Processing", described here:
https://multicians.org/mepap.html

 - Dan C.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Sun, 02 Feb 2025 19:04:00 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sat, 1 Feb 2025 19:16 +0000 (GMT Standard Time), John Dallman wrote:

> In article <vnjfai$3mhvf$4@dont-email.me>, ldo@nz.invalid (Lawrence
> D'Oliveiro) wrote:
>
>> Back in 1980, there were something like 10 different companies
>> operating in the mainframe business. Now there is only one.
>
> Nope. IBM is the market leader, sure. In Japan, Fujitsu and Hitachi
> continue to sell IBM-like 31-bit mainframes, and NEC sells a proprietary
> mainframe.

Don’t be fooled by continuity of branding. Remember, the big Japanese

Fujitsu, for example, among their many businesses sell high-performance
ARM chips (the A64FX and successors), which are used in supercomputers,
not mainframes.

> The ex-Univac OS 2200, ex-Burroughs MCP, Groupe Bull GCOS,
> ex-Siemens BS2000 and ex-ICL VME are all still used, mostly under
> emulation.

“Emulation” on what? What you’re admitting is mainframes as such are not
being sold, the legacy software, such as it is, is now running on
something else.

Subject: Re: banking for old pharts, Multics vs Unix vs mainframes

Posted by John Levine on Sun, 02 Feb 2025 20:51:48 GMT
View Forum Message <> Reply to Message

According to Lars Poulsen <lars@cleo.beagle-ears.com>:

>
> Denmark did away with checks a decade ago, and is headed towards
> eliminating cash as well. Think 6-year olds with debit cards for their
> allowances.
>
> Here in the US, we still use checks. In fact, when I am moving money
> between banks, I go into my branch of the big national bank, get a
> cashier's check for some multiple of USD 50 000 and hand carry it to my
> branch of the small regional bank to deposit it there.
>
> If I were to do the transfer electronically, it would cost about USD 30,
> and take about 36 hours before the money was available in the new
> account. Doing it my way, it is free and the money is available when I
> leave the branch.

You must have a terrible bank. At an account I have at my large US bank, wire
transfers are free and I consistently see them show up in the recipient account
in less than an hour. I need a large minimum balance for that account, but if
you are buying $50K cashier's checks, you surely qualify.

> The USA is stuck in the past, waxing nostalgic for the golden days of
> the 1950s. And rapidly becoming a lawless 3rd world country.

Somewhat. The US has two bank instant payment systems, Real Time Payments from
The Clearing House (ostentatiously abbreviated TCH RTP), and FedNow from the
Federal Reserve. The problem is that the US, unlike any other country, has a few
large banks and a long tail of 4,000 medium to tiny banks. The small banks know
their local communities but outsource all of their backend systems to a few
specialist providers like FISERV and Jack Henry. It is really frustrating that
the back end providers support RTP, but the banks are clueless. One time I made
an RTP payment to my account at my medium sized local bank, which showed up as
promised in a few seconds. Then I pointed it out to them and asked how do I do a
transfer the other way. Duh, I don't think we offer that.
--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

Subject: Re: how to run fast, old pharts, Multics vs Unix vs mainframes
Posted by John Levine on Sun, 02 Feb 2025 21:00:15 GMT
View Forum Message <> Reply to Message

According to John Dallman <jgd@cix.co.uk>:
> Speed of synchronisation matters a lot to a credit card provider who is
> trying to enforce customer credit limits and avoid double-spends. They
> still use mainframes with z/TPF for that. z/TPF is a curious OS; it
> essentially makes a mainframe into a single real-time transaction
> processing system.

TPF is an amazing fossil. Its design dates back to SABRE in the early 1960s
which borrowed the event loop from the SAGE system that ran on vacuum tube
computers in the 1950s. It still gets astonishing performance out of whatever
hardware it runs on.

The fundamental structure keeps being reinvented. Look inside node.js and it
looks oddly familiar.

--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: front ends, old pharts, Multics vs Unix
Posted by John Levine on Sun, 02 Feb 2025 21:05:01 GMT

View Forum Message <> Reply to Message

According to Dan Cross <cross@spitfire.i.gajendra.net>:
>> The operating system I helped to support for some years handled terminal
>> I/O in a PDP-11 used as a front end processor (sometimes more than one).
>> Characters were sent via an interface that used single characters or not,
>> as needed.
>
> Multics did something similar; again, the theory was to protect
> the main CPU(s) from a flurry of interrupts as users typed away;
> instead, shuttle all of that overhead off to some satellite
> processor (the General Input/Output Controller; GIOC, ...

DTSS, which ran on a GE 635, the much simpler predecessor to the Multics
645, did the same thing, The TTYs connected to a Datanet-30 which
buffered up all the input until the user typed RUN or something else
that needed a response, which it forwarded in a block to the 635.

The performance was great, 100 users on a machine the same size as the
original PDP-10.

The first version of DTSS ran the whole system on the DN-30 and just used
the larger 225 as a compute server to which it handed the users' programs.
You typed RUN, it usually responded WAIT since there were other jobs

ahead of yours.

--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Dan Espen on Sun, 02 Feb 2025 21:10:06 GMT

View Forum Message <> Reply to Message

"Kerr-Mudd, John" <admin@127.0.0.1> writes:

D> On Sun, 2 Feb 2025 00:58:36 -0000 (UTC)
> antispam@fricas.org (Waldek Hebisch) wrote:
>
>> Peter Flass <peter_flass@yahoo.com> wrote:
>>> John Levine <johnl@taugh.com> wrote:
>>>> According to Bob Eager <news0009@eager.cx>:
>>>> > On Fri, 31 Jan 2025 10:24:01 -0500, Dan Espen wrote:
>>>> >
>>>> >>>> IBM is still selling mainframe hardware.
>>>> >>>
>>>> >>> I doubt they're making any profit on it any more.
>>>> >>
>>>> >> That's something you could look up for yourself.
>>>> >
>>>> > I understand that all the profit has been in software, for many years.
>>>>
>>>> IBM continues to put a lot of effort into their mainframe hardware. If
>>>> you look at their financial reports, they usually have a bullet for "IBM Z"
>>>> which is up some quarters, down others. The most recent slide deck has a
>>>> bullet that says:
>>>>
>>>>   z16 our most successful program in history
>>>>
>>>> I agree that mainframes is a legacy business but it's one that still has
>>>> a long life aheade it it.
>>>>
>>>
>>> No one is going to watch to convert 60 years of production software to
>>> another platform. Actually, i take that back. It's been tried, but never
>>> successfully.
>>
>> I think that there were a lot of successful migrations from
>> mainframes to other systems. It is likely that most succesful
>> cases were gradual, taking substantial time.

>>
>> In Poland during comunist times one of basic machines were
>> domesticaly produced ICL-1900 compatible machines, using ICL
>> operating system. It seems that our manufacturer of those
>> machines stopped production around 1990 (and during eighties
>> production was decreasing). For few years alternative
>> manufacturer offered compatible machines using bit-slice
>> microprocessors. Few years ago supposedly last machine
>> of this family was decomissioned (it was used by railway
>> as part of trafic control).
>>
>> So, it is basically question of time and financial balance:
>> currently mainframe users pay higer prices to mainframe
>> vendors so that users can keep using their systems and
>> avoid migration. But as user population shrinks it may
>> become too small to adequatly support vendors. That may
>> force vendors to limit their offer leading to colapse:
>> limited vendor offer accelerates migration. That seem
>> to happened with several smaller vendors. IBM-compatible
>> world currently looks relatively healthy, but it is not
>> clear for how long. I expect at least 10 more years of use
>> of IBM-compatible mainframes, but it is hard to make
>> prediction for longer time. 10 years may look like
>> long time, but it is short compared to 60 year history.
>>
> No one expected 2 digit year field programs written in 1980 to
> still be in production in 20 years time.

Having started programming in 1964, I beg to differ.
We just expected someone to be around to deal with the problem.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Dan Espen on Sun, 02 Feb 2025 21:12:01 GMT
View Forum Message <> Reply to Message

scott@slp53.sl.home (Scott Lurndal) writes:

> "Kerr-Mudd, John" <admin@127.0.0.1> writes:
>> On Sun, 2 Feb 2025 00:58:36 -0000 (UTC)
>> antispam@fricas.org (Waldek Hebisch) wrote:
>>
>>> Peter Flass <peter_flass@yahoo.com> wrote:
>>>> John Levine <johnl@taugh.com> wrote:
>>>> > According to Bob Eager <news0009@eager.cx>:

>>>> >> On Fri, 31 Jan 2025 10:24:01 -0500, Dan Espen wrote:
>>>> >>
>>>> >>>>> IBM is still selling mainframe hardware.
>
>>> So, it is basically question of time and financial balance:
>>> currently mainframe users pay higer prices to mainframe
>>> vendors so that users can keep using their systems and
>>> avoid migration. But as user population shrinks it may
>>> become too small to adequatly support vendors. That may
>>> force vendors to limit their offer leading to colapse:
>>> limited vendor offer accelerates migration. That seem
>>> to happened with several smaller vendors. IBM-compatible
>>> world currently looks relatively healthy, but it is not
>>> clear for how long. I expect at least 10 more years of use
>>> of IBM-compatible mainframes, but it is hard to make
>>> prediction for longer time. 10 years may look like
>>> long time, but it is short compared to 60 year history.
>>>
>> No one expected 2 digit year field programs written in 1980 to
>> still be in production in 20 years time.
>
> At Burroughs, we fully expected 2-digit year field programs
> written in 1968 to be still in production by 2000 when we
> started the Y2K mitigation work in the mid 1980s.
>
> 1960 was used as the dividing line - any two digit year smaller
> was assumed to be 21st century; any larger was assumed to be
> 20th century.

I fixed one of my applications by looking at the current year, then
setting the window accordingly. Fixed forever.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Peter Flass on Sun, 02 Feb 2025 21:42:27 GMT
View Forum Message <> Reply to Message

Dan Espen <dan1espen@gmail.com> wrote:
> scott@slp53.sl.home (Scott Lurndal) writes:
>
>> "Kerr-Mudd, John" <admin@127.0.0.1> writes:
>>> On Sun, 2 Feb 2025 00:58:36 -0000 (UTC)
>>> antispam@fricas.org (Waldek Hebisch) wrote:
>>>
>>>> Peter Flass <peter_flass@yahoo.com> wrote:

>>>> > John Levine <johnl@taugh.com> wrote:
>>>> >> According to Bob Eager <news0009@eager.cx>:
>>>> >>> On Fri, 31 Jan 2025 10:24:01 -0500, Dan Espen wrote:
>>>> >>>
>>>> >>>>>> IBM is still selling mainframe hardware.
>>
>>>> So, it is basically question of time and financial balance:
>>>> currently mainframe users pay higer prices to mainframe
>>>> vendors so that users can keep using their systems and
>>>> avoid migration. But as user population shrinks it may
>>>> become too small to adequatly support vendors. That may
>>>> force vendors to limit their offer leading to colapse:
>>>> limited vendor offer accelerates migration. That seem
>>>> to happened with several smaller vendors. IBM-compatible
>>>> world currently looks relatively healthy, but it is not
>>>> clear for how long. I expect at least 10 more years of use
>>>> of IBM-compatible mainframes, but it is hard to make
>>>> prediction for longer time. 10 years may look like
>>>> long time, but it is short compared to 60 year history.
>>>>
>>> No one expected 2 digit year field programs written in 1980 to
>>> still be in production in 20 years time.
>>
>> At Burroughs, we fully expected 2-digit year field programs
>> written in 1968 to be still in production by 2000 when we
>> started the Y2K mitigation work in the mid 1980s.
>>
>> 1960 was used as the dividing line - any two digit year smaller
>> was assumed to be 21st century; any larger was assumed to be
>> 20th century.
>
> I fixed one of my applications by looking at the current year, then
> setting the window accordingly. Fixed forever.
>

Depends on the application. For something like Social Security you may have
records on someone born this year(parents applied for SSN) to this year
minus 100 or more. For a payments system this works fine, since all you
usually need is last year, this year, and next year.

--
Pete

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Charlie Gibbs on Sun, 02 Feb 2025 22:36:31 GMT
View Forum Message <> Reply to Message

On 2025-02-02, Dan Espen <dan1espen@gmail.com> wrote:

> scott@slp53.sl.home (Scott Lurndal) writes:
>
>> "Kerr-Mudd, John" <admin@127.0.0.1> writes:
>>
>>> No one expected 2 digit year field programs written in 1980 to
>>> still be in production in 20 years time.
>>
>> At Burroughs, we fully expected 2-digit year field programs
>> written in 1968 to be still in production by 2000 when we
>> started the Y2K mitigation work in the mid 1980s.
>>
>> 1960 was used as the dividing line - any two digit year smaller
>> was assumed to be 21st century; any larger was assumed to be
>> 20th century.
>
> I fixed one of my applications by looking at the current year,
> then setting the window accordingly. Fixed forever.

FSVO "forever". My first programming job was at a small
card-only shop in 1970. We did a lot of squeezing to fit
data onto an 80-column card. One way was to only store
the last digit of the year in date fields. My first
assignment was to go through all our programs and change
the digit they inserted from a 6 to a 7.

```
--
/~\  Charlie Gibbs                  |  Growth for the sake of
\ /  <cgibbs@kltpzyxm.invalid>      |  growth is the ideology
 X   I'm really at ac.dekanfrus     |  of the cancer cell.
/ \  if you read it the right way.  |    -- Edward Abbey
```

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anne &amp; Lynn Wheel on Sun, 02 Feb 2025 22:55:46 GMT

View Forum Message <> Reply to Message

Lynn Wheeler <lynn@garlic.com> writes:
> Some of this left over from mid-90s where billions were spent on
> rewritting mainframe financial systems that queued real time
> transactions for processing during overnight batch settlement windows
> (many from the 60s&70s) ... to straight-through processing using large
> number of parallelized killer micros. Some of us pointed out that they
> were using industry standard parallelization libraries that had hundred
> times the overhead of mainframe cobol batch ... ignored until their
> pilots went down in flames (retrenching to the existing status quo).
>

> A major change was combination of 1) major RDBMS vendors (including IBM)
> did significiant throughput performance work on cluster RDBMS, 2)
> implementations done with fine-grain SQL statements that were highly
> parallelized (rather than RYO implementation parallelization), and 3)
> non-mainframe systems having significantly higher throughput.

I got performance gig at turn of century w/financial outsourcer that handled half of all credit card accounts in the US ... datacenter had > 40 max configured mainframe systems (none older than 18months) all running the same 450K statement Cobol app (number needed to finish settlement in the overnight batch window i.e. real-time transactions were still being queued for settlement in overnight batch window).

after turn of century got involved with operation that had experience doing large scale financial production apps for a couple decades and had developed a fiancial application language that translated financial transaction specifications into fine-grain SQL statements. It significantly reduced development effort and showed that it was significantly more agile responding to things like regulatory changes.

they prepared a number of applications that simulated the 5 or 6 largest financial operations that then existed, for demos ... showing a six multiprocessor (four processors each) intel/microsoft system cluster having many times the throughput capacity of the existing production operations.

then demo'ed at multiple financial industry organizations that initially saw high acceptance ... then hit brick wall ... eventually told that most of their executives still bore the scars from the 90s "modernization" disasters.

Note the 90s disaster with large number of parallelized killer micros involved processors with throughput that were small fraction of mainframe and (effectively) toy parallelization. move to middle of the 1st decade after the turn of the century ... and those systems now had

1) higher throughput (1999 pentium3 single processor chip rated at 2BIPS while max. configured DEC2000 IBM Mainframe z900 (16 processors) was only 2.5BIPS aggregate) ... and Intel was still doubling chip throughput every 18-24months (aided in part with transition to multi-core).
July2005 IBM z9 with max-configured 54processor clocked at 18BIPS aggregate (333MIPS/processor) ... while Intel single multi-core chip was at least twice that.

2) significantly higher non-cluster and cluster RDBMS throughput

3) basically both mainframe & non-mainframe were using the same industry standard fixed-block disks ... giving non-mainframe higher IOPS

advantage with mainframe requiring extra layer providing CKD emulation.

trivia: late 70s an IBM SE on financial industry account with large
number of ATM machines, transactions running on 370/168 TPF machine. He
then re-implemented the ATM transactions on 370/158 VM370 machine
.... and was getting higher throughput (than TPF). He did some hacks to
get the ATM transcation pathlength close to TPF (ATM transaction
processing relatively trivial, typically swamped by disk I/O) ... and
then all sorts of very sophisticated transaction scheduling strategies
to optimize tansactions per disk arm sweep, including looking at account
records on the same arm, evaluating historical transaction patterns
(both by ATM machine and by account) that might include slight delays in
some transactions. This was cluster of virtual machines ... virtual
machines for transaction routing and sceduling along with dedicated
virtual machines for each disk arm.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Mon, 03 Feb 2025 02:50:16 GMT

View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 2 Feb 2025 18:04:23 +0000, Kerr-Mudd, John wrote:

> No one expected 2 digit year field programs written in 1980 to still be
> in production in 20 years time.

Already in the 1970s there were standards for representing dates and times
that specified a four-digit year. It’s not as though you could say nobody
saw this coming.

Subject: Re: banking for old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Mon, 03 Feb 2025 02:52:25 GMT

View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 2 Feb 2025 20:51:48 -0000 (UTC), John Levine wrote:

> One time I made an RTP payment to my account at my medium sized local
< bank, which showed up as promised in a few seconds.

Somehow I don’t think mainframe-based COBOL code was involved in that.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Mon, 03 Feb 2025 02:57:47 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 02 Feb 2025 06:09:51 GMT, Charlie Gibbs wrote:

> I ported Adventure and Dungeon to OS/3; one of the most difficult tasks
> was getting terminal I/O to work properly.

I remember at a software house where I was working for a while, there was
a Wang mini of some description, that some colleagues were using for
development work.

One of them showed me a “Space Invaders” game running on one of the
terminals. Then he pulled out the cable between the terminal and the main
machine ... and the game kept right on playing.

Effectively, the terminal was a fully programmable computer in its own
right. No “block mode” here -- the game was fully interactive, with all
the interaction happening locally.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Mon, 03 Feb 2025 02:59:38 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 2 Feb 2025 14:07:13 -0000 (UTC), Lars Poulsen wrote:

> So at peak times, response times went up, buteverything kept running
> within available capacity.

“Your transaction is important to us. Please hold.”

Subject: Wang Terminals (Re: old pharts, Multics vs Unix)
Posted by Anonymous on Mon, 03 Feb 2025 03:48:06 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

On Sun, 02 Feb 2025 06:09:51 GMT, Charlie Gibbs wrote:
>> I ported Adventure and Dungeon to OS/3; one of the most difficult tasks
>> was getting terminal I/O to work properly.

On 2025-02-03, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> I remember at a software house where I was working for a while, there was
> a Wang mini of some description, that some colleagues were using for
> development work.
>
> One of them showed me a “Space Invaders” game running on one of the
> terminals. Then he pulled out the cable between the terminal and the main
> machine ... and the game kept right on playing.
>
> Effectively, the terminal was a fully programmable computer in its own
> right. No “block mode” here -- the game was fully interactive, with all
> the interaction happening locally.

My second "real" employer was a small bespoke computer engineering shop.
In many ways similar to the one where I work now. This was in 1975.
We mostly helped university labs put specialty peripherals on their
various minicomputers. As part of that we had hooked things up to
som HP lab desktop minis using HP-IB interfaces. Then we came across a
lab that used Wang 2200 (?) minis that looked a lot like the HPs: A
terminal-like thing that had a Basic inerpreter built-in. Having chatted
with the Wang people, they talked us into representing their little
desktops in Denmark, and when they started making word processors, we
took on those as well, and soon that was most of our business.

There were several variations of the same 8-bit mini; the ones we used
for the wordprocessing system used a coax serial bus to talk to the
shared file server. With a different code load, it could be a terminal
for the Wang VS (370/130-like) as a forms-processor, not unlike the IBM
3278 from the perspective of the terminal user.

My first visit to Wang Labs in Lowell, MA, USA was to do a Danish
language localization for the OIS-140 (Office Information System) in
late April 1980.

I also had to battle Wang engineering to get proper font adjustments to
accommodate the extra Danish vowels: æøå/ÆØÅ and get them correctly
placed on the keyboard.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Charlie Gibbs on Mon, 03 Feb 2025 04:17:21 GMT

View Forum Message <> Reply to Message

On 2025-02-03, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:

> On Sun, 2 Feb 2025 18:04:23 +0000, Kerr-Mudd, John wrote:
>
>> No one expected 2 digit year field programs written in 1980 to still be
>> in production in 20 years time.
>
> Already in the 1970s there were standards for representing dates and times
> that specified a four-digit year. It’s not as though you could say nobody
> saw this coming.

If you were writing mortgage programs doing 20-year amortizations,
you had to have solved the problem by then.

```
--
/~\  Charlie Gibbs                  |  Growth for the sake of
\ /  <cgibbs@kltpzyxm.invalid>      |  growth is the ideology
 X   I'm really at ac.dekanfrus     |  of the cancer cell.
/ \  if you read it the right way.  |    -- Edward Abbey
```

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by scott on Mon, 03 Feb 2025 13:55:50 GMT
View Forum Message <> Reply to Message

Peter Flass <peter_flass@yahoo.com> writes:
> Dan Espen <dan1espen@gmail.com> wrote:
>> scott@slp53.sl.home (Scott Lurndal) writes:
>>
 <Y2K mitigations>

>>
>> I fixed one of my applications by looking at the current year, then
>> setting the window accordingly. Fixed forever.
>>
>
> Depends on the application. For something like Social Security you may have
> records on someone born this year(parents applied for SSN) to this year
> minus 100 or more. For a payments system this works fine, since all you
> usually need is last year, this year, and next year.

For SS, even in the 1960's, they'd have to be able to store dates
from the 19th century; two-digit years were never useful in that context.

Subject: Re: banking for old pharts, Multics vs Unix vs mainframes
Posted by scott on Mon, 03 Feb 2025 14:00:57 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> writes:
> On Sun, 2 Feb 2025 20:51:48 -0000 (UTC), John Levine wrote:
>
>> One time I made an RTP payment to my account at my medium sized local
> < bank, which showed up as promised in a few seconds.
>
> Somehow I don’think

Not surprising...

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Mon, 03 Feb 2025 15:03:10 GMT
View Forum Message <> Reply to Message

Originally posted by: Colin Macleod

Dan Espen <dan1espen@gmail.com> posted:

> Lynn Wheeler <lynn@garlic.com> writes:
>
>> however, all 3270s were half-duplex and if you were unfortunate to hit a
>> key same time system went to write to screen, it would lock the keyboard
>> and would have to stop and hit the reset key. YKT developed a FIFO box
>> for 3277, unplug the keyboard from the 3277 head, plug the FIFO box into
>> the head and plug the 3277 keyboard into the fifo box ... eliminating
>> the unfortunate keyboad lock.
>
> Back in the late 60s I finished implementing my first online system on a
> S/360-30 and IBM 2260s.

This reminds me of something when I was a student around 1975. My university's only computer was an IBM 360/44 and we could use some 2260 terminals. Digging through some manuals I got the idea that by munging the "carriage control character" from a Fortran program I might be able to break out of the restriction of block-mode operation and persuade a 2260 to do animated graphics.

When tried my proof-of-concept program I did get the terminal to keep redrawing a very flickery but changing few lines. But while I was watching this, the other users in the lab started complaining that their terminals were frozen. Then the operators started running around trying to find out why the whole mainframe had hung. It appeared that my hack had somehow elevated the priority of my animation such that nothing else was getting run at all. I couldn't even interrupt my own program. They had to reboot the machine to fix it and I was told in no uncertain terms to never do that again!

--

Colin Macleod ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ https://cmacleod.me.uk

Fermented grapes count towards my five-a-day, right?

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Dan Espen on Mon, 03 Feb 2025 16:39:25 GMT
View Forum Message <> Reply to Message

Charlie Gibbs <cgibbs@kltpzyxm.invalid> writes:

> On 2025-02-02, Dan Espen <dan1espen@gmail.com> wrote:
>
>> scott@slp53.sl.home (Scott Lurndal) writes:
>>
>>> "Kerr-Mudd, John" <admin@127.0.0.1> writes:
>>>
>>>> No one expected 2 digit year field programs written in 1980 to
>>>> still be in production in 20 years time.
>>>
>>> At Burroughs, we fully expected 2-digit year field programs
>>> written in 1968 to be still in production by 2000 when we
>>> started the Y2K mitigation work in the mid 1980s.
>>>
>>> 1960 was used as the dividing line - any two digit year smaller
>>> was assumed to be 21st century; any larger was assumed to be
>>> 20th century.
>>
>> I fixed one of my applications by looking at the current year,
>> then setting the window accordingly. Fixed forever.
>
> FSVO "forever". My first programming job was at a small
> card-only shop in 1970. We did a lot of squeezing to fit
> data onto an 80-column card. One way was to only store
> the last digit of the year in date fields. My first
> assignment was to go through all our programs and change
> the digit they inserted from a 6 to a 7.

Uggh!

Meanwhile, fixes using 4 digit years are setting us up for
the Y9K bug.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by scott on Mon, 03 Feb 2025 18:00:22 GMT
View Forum Message <> Reply to Message

Dan Espen <dan1espen@gmail.com> writes:
> Charlie Gibbs <cgibbs@kltpzyxm.invalid> writes:
>
>> On 2025-02-02, Dan Espen <dan1espen@gmail.com> wrote:
>>
>>> scott@slp53.sl.home (Scott Lurndal) writes:
>>>
>>>> "Kerr-Mudd, John" <admin@127.0.0.1> writes:
>>>>
>>>> > No one expected 2 digit year field programs written in 1980 to
>>>> > still be in production in 20 years time.
>>>>
>>>> At Burroughs, we fully expected 2-digit year field programs
>>>> written in 1968 to be still in production by 2000 when we
>>>> started the Y2K mitigation work in the mid 1980s.
>>>>
>>>> 1960 was used as the dividing line - any two digit year smaller
>>>> was assumed to be 21st century; any larger was assumed to be
>>>> 20th century.
>>>
>>> I fixed one of my applications by looking at the current year,
>>> then setting the window accordingly. Fixed forever.
>>
>> FSVO "forever". My first programming job was at a small
>> card-only shop in 1970. We did a lot of squeezing to fit
>> data onto an 80-column card. One way was to only store
>> the last digit of the year in date fields. My first
>> assignment was to go through all our programs and change
>> the digit they inserted from a 6 to a 7.
>
> Uggh!
>
> Meanwhile, fixes using 4 digit years are setting us up for
> the Y9K bug.

A signed 64-bit integer can represent any time value (in seconds)
for a few hundred million years BCE and CE.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Mon, 03 Feb 2025 18:21:15 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

"Kerr-Mudd, John" <admin@127.0.0.1> writes:
>>>> > No one expected 2 digit year field programs written in 1980 to
>>>> > still be in production in 20 years time.

scott@slp53.sl.home (Scott Lurndal) writes:
>>>> At Burroughs, we fully expected 2-digit year field programs
>>>> written in 1968 to be still in production by 2000 when we
>>>> started the Y2K mitigation work in the mid 1980s.
>>>>
>>>> 1960 was used as the dividing line - any two digit year smaller
>>>> was assumed to be 21st century; any larger was assumed to be
>>>> 20th century.

On 2025-02-02, Dan Espen <dan1espen@gmail.com> wrote:
>>> I fixed one of my applications by looking at the current year,
>>> then setting the window accordingly. Fixed forever.

Charlie Gibbs <cgibbs@kltpzyxm.invalid> writes:
>> FSVO "forever". My first programming job was at a small
>> card-only shop in 1970. We did a lot of squeezing to fit
>> data onto an 80-column card. One way was to only store
>> the last digit of the year in date fields. My first
>> assignment was to go through all our programs and change
>> the digit they inserted from a 6 to a 7.

On 2025-02-03, Dan Espen <dan1espen@gmail.com> wrote:
> Uggh!
> Meanwhile, fixes using 4 digit years are setting us up for
> the Y9K bug.

Problems are much closer. One of my customers asked us to explain how
out embedded firmware deals with the Y2038 situation.
    https://en.wikipedia.org/wiki/Year_2038_problem

I was pleased to be able to explain that our "calendar time" is an
unsigned 32-bit integer (since we never had to deal with pre-epoch
dates. They may have an issue 2106, however.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Rich Alderson on Mon, 03 Feb 2025 21:40:05 GMT

View Forum Message <> Reply to Message

scott@slp53.sl.home (Scott Lurndal) writes:

> Peter Flass <peter_flass@yahoo.com> writes:
>> Dan Espen <dan1espen@gmail.com> wrote:

>>> scott@slp53.sl.home (Scott Lurndal) writes:

> <Y2K mitigations>

>>> I fixed one of my applications by looking at the current year, then
>>> setting the window accordingly. Fixed forever.

>> Depends on the application. For something like Social Security you may have
>> records on someone born this year(parents applied for SSN) to this year
>> minus 100 or more. For a payments system this works fine, since all you
>> usually need is last year, this year, and next year.

> For SS, even in the 1960's, they'd have to be able to store dates
> from the 19th century; two-digit years were never useful in that context.

Indeed. My greatgrandfather Alderson was born in 1876, and died in 1962; my
greatgrandmother was born in 1885, and died 6 weeks short of her 90th birthday
in 1975.

--
Rich Alderson   news@alderson.users.panix.com
      Audendum est, et veritas investiganda; quam etiamsi non assequamur,
   omnino tamen proprius, quam nunc sumus, ad eam perveniemus.
         --Galen

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Dan Espen on Mon, 03 Feb 2025 22:21:50 GMT
View Forum Message <> Reply to Message

scott@slp53.sl.home (Scott Lurndal) writes:

> Dan Espen <dan1espen@gmail.com> writes:
>> Charlie Gibbs <cgibbs@kltpzyxm.invalid> writes:
>>
>>> On 2025-02-02, Dan Espen <dan1espen@gmail.com> wrote:
>>>
>>>> scott@slp53.sl.home (Scott Lurndal) writes:
>>>>
>>>> > "Kerr-Mudd, John" <admin@127.0.0.1> writes:
>>>> >
>>>> >> No one expected 2 digit year field programs written in 1980 to
>>>> >> still be in production in 20 years time.
>>>> >
>>>> > At Burroughs, we fully expected 2-digit year field programs
>>>> > written in 1968 to be still in production by 2000 when we
>>>> > started the Y2K mitigation work in the mid 1980s.
>>>> >

>>>> > 1960 was used as the dividing line - any two digit year smaller
>>>> > was assumed to be 21st century; any larger was assumed to be
>>>> > 20th century.
>>>>
>>>> I fixed one of my applications by looking at the current year,
>>>> then setting the window accordingly. Fixed forever.
>>>
>>> FSVO "forever". My first programming job was at a small
>>> card-only shop in 1970. We did a lot of squeezing to fit
>>> data onto an 80-column card. One way was to only store
>>> the last digit of the year in date fields. My first
>>> assignment was to go through all our programs and change
>>> the digit they inserted from a 6 to a 7.
>>
>> Uggh!
>>
>> Meanwhile, fixes using 4 digit years are setting us up for
>> the Y9K bug.
>
> A signed 64-bit integer can represent any time value (in seconds)
> for a few hundred million years BCE and CE.

Hopefully, those 64bit dates will remain an internal representation.
Anything converting a 4 digit year to a 64bit date still has
a Y9K problem.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by scott on Mon, 03 Feb 2025 22:53:24 GMT
View Forum Message <> Reply to Message

Rich Alderson <news@alderson.users.panix.com> writes:
> scott@slp53.sl.home (Scott Lurndal) writes:
>
>> Peter Flass <peter_flass@yahoo.com> writes:
>>> Dan Espen <dan1espen@gmail.com> wrote:
>>>> scott@slp53.sl.home (Scott Lurndal) writes:
>
>>   <Y2K mitigations>
>
>>>> I fixed one of my applications by looking at the current year, then
>>>> setting the window accordingly. Fixed forever.
>
>>> Depends on the application. For something like Social Security you may have

>>> records on someone born this year(parents applied for SSN) to this year
>>> minus 100 or more. For a payments system this works fine, since all you
>>> usually need is last year, this year, and next year.
>
>> For SS, even in the 1960's, they'd have to be able to store dates
>> from the 19th century; two-digit years were never useful in that context.
>
> Indeed. My greatgrandfather Alderson was born in 1876, and died in 1962; my
> greatgrandmother was born in 1885, and died 6 weeks short of her 90th birthday
> in 1975.

One of my greatgrandfathers was born in 1841, immigrated in 1850 and died in
1878 due to the effects of injuries sustained during the civil war.
His wife was born in 1850 and died in 1925. My great grandmother 1881 - 1966.

>
> --
> Rich Alderson   news@alderson.users.panix.com
>   Audendum est, et veritas investiganda; quam etiamsi non assequamur,
>   omnino tamen proprius, quam nunc sumus, ad eam perveniemus.
>   --Galen

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by scott on Mon, 03 Feb 2025 23:08:30 GMT

View Forum Message <> Reply to Message

Dan Espen <dan1espen@gmail.com> writes:
> scott@slp53.sl.home (Scott Lurndal) writes:
>
>> Dan Espen <dan1espen@gmail.com> writes:
>>> Charlie Gibbs <cgibbs@kltpzyxm.invalid> writes:
>>>
>>>> On 2025-02-02, Dan Espen <dan1espen@gmail.com> wrote:
>>>>
>>>> > scott@slp53.sl.home (Scott Lurndal) writes:
>>>> >
>>>> >> "Kerr-Mudd, John" <admin@127.0.0.1> writes:
>>>> >>
>>>> >>> No one expected 2 digit year field programs written in 1980 to
>>>> >>> still be in production in 20 years time.
>>>> >>
>>>> >> At Burroughs, we fully expected 2-digit year field programs
>>>> >> written in 1968 to be still in production by 2000 when we
>>>> >> started the Y2K mitigation work in the mid 1980s.
>>>> >>
>>>> >> 1960 was used as the dividing line - any two digit year smaller
>>>> >> was assumed to be 21st century; any larger was assumed to be

>>>> >> 20th century.
>>>> >
>>>> > I fixed one of my applications by looking at the current year,
>>>> > then setting the window accordingly. Fixed forever.
>>>>
>>>> FSVO "forever". My first programming job was at a small
>>>> card-only shop in 1970. We did a lot of squeezing to fit
>>>> data onto an 80-column card. One way was to only store
>>>> the last digit of the year in date fields. My first
>>>> assignment was to go through all our programs and change
>>>> the digit they inserted from a 6 to a 7.
>>>
>>> Uggh!
>>>
>>> Meanwhile, fixes using 4 digit years are setting us up for
>>> the Y9K bug.
>>
>> A signed 64-bit integer can represent any time value (in seconds)
>> for a few hundred million years BCE and CE.
>
> Hopefully, those 64bit dates will remain an internal representation.
> Anything converting a 4 digit year to a 64bit date still has
> a Y9K problem.

Does it? Let's take the year 10355, for example.

The 64-bit value will be 10335 * (num_seconds_in_year), or

```
$ printf "%'u\n" $(( 60 * 60 * 24 * 365 * 10335 ))
325,924,560,000
```

The maximum signed value is

```
$ printf "%'u\n" $(( (1 << 63) - 1 ))
9,223,372,036,854,775,807
```

which is almost eight orders of magnitude larger.   That supports
a huge number of years both sides of zero with one second
resolution.

As for converting between internal and external representation
taking locale into account:

https://pubs.opengroup.org/onlinepubs/9799919799/functions/s trftime.html
https://pubs.opengroup.org/onlinepubs/9799919799/functions/s trptime.html

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Dan Espen on Mon, 03 Feb 2025 23:50:50 GMT
View Forum Message <> Reply to Message

scott@slp53.sl.home (Scott Lurndal) writes:

> Dan Espen <dan1espen@gmail.com> writes:
>> scott@slp53.sl.home (Scott Lurndal) writes:
>>
>>> Dan Espen <dan1espen@gmail.com> writes:
>>>> Charlie Gibbs <cgibbs@kltpzyxm.invalid> writes:
>>>>
>>>> > On 2025-02-02, Dan Espen <dan1espen@gmail.com> wrote:
>>>> >
>>>> >> scott@slp53.sl.home (Scott Lurndal) writes:
>>>> >>
>>>> >>> "Kerr-Mudd, John" <admin@127.0.0.1> writes:
>>>> >>>
>>>> >>>> No one expected 2 digit year field programs written in 1980 to
>>>> >>>> still be in production in 20 years time.
>>>> >>>
>>>> >>> At Burroughs, we fully expected 2-digit year field programs
>>>> >>> written in 1968 to be still in production by 2000 when we
>>>> >>> started the Y2K mitigation work in the mid 1980s.
>>>> >>>
>>>> >>> 1960 was used as the dividing line - any two digit year smaller
>>>> >>> was assumed to be 21st century; any larger was assumed to be
>>>> >>> 20th century.
>>>> >>
>>>> >> I fixed one of my applications by looking at the current year,
>>>> >> then setting the window accordingly. Fixed forever.
>>>> >
>>>> > FSVO "forever". My first programming job was at a small
>>>> > card-only shop in 1970. We did a lot of squeezing to fit
>>>> > data onto an 80-column card. One way was to only store
>>>> > the last digit of the year in date fields. My first
>>>> > assignment was to go through all our programs and change
>>>> > the digit they inserted from a 6 to a 7.
>>>>
>>>> Uggh!
>>>>
>>>> Meanwhile, fixes using 4 digit years are setting us up for
>>>> the Y9K bug.
>>>
>>> A signed 64-bit integer can represent any time value (in seconds)
>>> for a few hundred million years BCE and CE.
>>
>> Hopefully, those 64bit dates will remain an internal representation.
>> Anything converting a 4 digit year to a 64bit date still has

>> a Y9K problem.
>
> Does it? Let's take the year 10355, for example.

I think you missed my point...It's going to be difficult to store
10355 in a 4 digit field.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by scott on Tue, 04 Feb 2025 00:09:04 GMT
View Forum Message <> Reply to Message

Dan Espen <dan1espen@gmail.com> writes:
> scott@slp53.sl.home (Scott Lurndal) writes:
>
>> Dan Espen <dan1espen@gmail.com> writes:
>>> scott@slp53.sl.home (Scott Lurndal) writes:
>>>
>>>> Dan Espen <dan1espen@gmail.com> writes:
>>>> >Charlie Gibbs <cgibbs@kltpzyxm.invalid> writes:
>>>> >
>>>> >> On 2025-02-02, Dan Espen <dan1espen@gmail.com> wrote:
>>>> >>
>>>> >>> scott@slp53.sl.home (Scott Lurndal) writes:
>>>> >>>
>>>> >>>> "Kerr-Mudd, John" <admin@127.0.0.1> writes:
>>>> >>>>
>>>> >>>>> No one expected 2 digit year field programs written in 1980 to
>>>> >>>>> still be in production in 20 years time.
>>>> >>>>
>>>> >>>> At Burroughs, we fully expected 2-digit year field programs
>>>> >>>> written in 1968 to be still in production by 2000 when we
>>>> >>>> started the Y2K mitigation work in the mid 1980s.
>>>> >>>>
>>>> >>>> 1960 was used as the dividing line - any two digit year smaller
>>>> >>>> was assumed to be 21st century; any larger was assumed to be
>>>> >>>> 20th century.
>>>> >>>
>>>> >>> I fixed one of my applications by looking at the current year,
>>>> >>> then setting the window accordingly. Fixed forever.
>>>> >>
>>>> >> FSVO "forever". My first programming job was at a small
>>>> >> card-only shop in 1970. We did a lot of squeezing to fit
>>>> >> data onto an 80-column card. One way was to only store
>>>> >> the last digit of the year in date fields. My first

>>>> >> assignment was to go through all our programs and change
>>>> >> the digit they inserted from a 6 to a 7.
>>>> >
>>>> >Uggh!
>>>> >
>>>> >Meanwhile, fixes using 4 digit years are setting us up for
>>>> >the Y9K bug.
>>>>
>>>> A signed 64-bit integer can represent any time value (in seconds)
>>>> for a few hundred million years BCE and CE.
>>>
>>> Hopefully, those 64bit dates will remain an internal representation.
>>> Anything converting a 4 digit year to a 64bit date still has
>>> a Y9K problem.
>>
>> Does it? Let's take the year 10355, for example.
>
> I think you missed my point...It's going to be difficult to store
> 10355 in a 4 digit field.

The problem will be in formatting and printing, and in COBOL PIC clauses.   Fortunately the latter are becoming more rare and hopefully won't still be used in 9999.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Charlie Gibbs on Tue, 04 Feb 2025 00:36:22 GMT
View Forum Message <> Reply to Message

On 2025-02-03, Dan Espen <dan1espen@gmail.com> wrote:

> scott@slp53.sl.home (Scott Lurndal) writes:
>
>> Dan Espen <dan1espen@gmail.com> writes:
>>
>>> Meanwhile, fixes using 4 digit years are setting us up for
>>> the Y9K bug.
>>
>> A signed 64-bit integer can represent any time value (in seconds)
>> for a few hundred million years BCE and CE.
>
> Hopefully, those 64bit dates will remain an internal representation.
> Anything converting a 4 digit year to a 64bit date still has
> a Y9K problem.

s/Y9K/Y10K/

At the year 9000 we'll still have a millennium left to fix it.

```
--
/~\  Charlie Gibbs                  |  Growth for the sake of
\ /  <cgibbs@kltpzyxm.invalid>      |  growth is the ideology
 X   I'm really at ac.dekanfrus     |  of the cancer cell.
/ \  if you read it the right way.  |    -- Edward Abbey
```

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Tue, 04 Feb 2025 00:55:59 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Mon, 03 Feb 2025 11:39:25 -0500, Dan Espen wrote:

> Meanwhile, fixes using 4 digit years are setting us up for the Y9K bug.

I'm expecting that the Gregorian calendar, with its zero point set to some arbitrary guess at the birth date of some figure that was mythological anyway, will not be in use for that long.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Tue, 04 Feb 2025 01:03:04 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Mon, 03 Feb 2025 17:21:50 -0500, Dan Espen wrote:

> Hopefully, those 64bit dates will remain an internal representation.
> Anything converting a 4 digit year to a 64bit date still has a Y9K
> problem.

The ultimate integer representation of time would be in units of the Planck interval (5.39e-44 seconds, according to Da Wiki). Representing the current age of the observable Universe would take about 200 bits, according to this scale.

So I figure a 256-bit integer in this scale (representing a maximum duration of about 10**16 years) ought to be enough to represent any duration of time we are ever likely to be physically concerned with. ;)

Subject: Re: old pharts, Multics vs Unix

Posted by Anonymous on Tue, 04 Feb 2025 01:05:33 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:

> I couldn't even interrupt my own program. They had to reboot the machine
> to fix it and I was told in no uncertain terms to never do that again!

Fragile things, mainframes. They were not battle-hardened by exposure to inquisitive students, the way interactive timesharing systems were.

Subject: Re: Wang Terminals (Re: old pharts, Multics vs Unix)
Posted by Anonymous on Tue, 04 Feb 2025 01:12:58 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Mon, 3 Feb 2025 03:48:06 -0000 (UTC), Lars Poulsen wrote:

> Having chatted with the Wang people, they talked us into representing
> their little desktops in Denmark, and when they started making word
> processors, we took on those as well, and soon that was most of our
> business.

I remember I got a glimpse at some Wang assembly code once, and it looked remarkably like my impressions of IBM 360/370 assembler. I think Dr Wang may have copied the IBM instruction set. Or at least large parts of it.

The Asianometry channel covered the history of Dr Wang and his company <https://www.youtube.com/watch?v=MgDZQy0nN-Y>. Obviously a brilliant engineer, who managed to be an astute businessman as well. But he insisted on passing control of the company to a son who was not nearly as astute.

> I also had to battle Wang engineering to get proper font adjustments to
> accommodate the extra Danish vowels: æøå/ÆØÅ and get them correctly
> placed on the keyboard.

Now that the technical obstacles to support of such alphabets have disappeared with the advent of Unicode, you still see a cultural reluctance among some to acknowledge their existence.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Tue, 04 Feb 2025 01:15:08 GMT

View Forum Message <> Reply to Message

Originally posted by: moi

On 04/02/2025 01:05, Lawrence D'Oliveiro wrote:
> On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>
>> I couldn't even interrupt my own program. They had to reboot the machine
>> to fix it and I was told in no uncertain terms to never do that again!
>
> Fragile things, mainframes. They were not battle-hardened by exposure to
> inquisitive students, the way interactive timesharing systems were.

Nonsense.

--
Bill F.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Charlie Gibbs on Tue, 04 Feb 2025 01:27:59 GMT

View Forum Message <> Reply to Message

On 2025-02-04, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:

> On Mon, 03 Feb 2025 17:21:50 -0500, Dan Espen wrote:
>
>> Hopefully, those 64bit dates will remain an internal representation.
>> Anything converting a 4 digit year to a 64bit date still has a Y9K
>> problem.
>
> The ultimate integer representation of time would be in units of the
> Planck interval (5.39e-44 seconds, according to Da Wiki). Representing the
> current age of the observable Universe would take about 200 bits,
> according to this scale.
>
> So I figure a 256-bit integer in this scale (representing a maximum
> duration of about 10**16 years) ought to be enough to represent any
> duration of time we are ever likely to be physically concerned with. ;)

Everybody sing along:

The whole universe was in a hot dense state
When nearly fourteen billion years ago expansion started... wait!
The earth began to cool
The autotrophs began to drool
Neanderthals developed tools
We built a wall (we built the pyramids)

    Math, science, history
    Unraveling the mystery
    That all started with a big bang (BANG!)

You're sitting in my spot.

```
--
/~\  Charlie Gibbs                  |  Growth for the sake of
\ /  <cgibbs@kltpzyxm.invalid>      |  growth is the ideology
 X   I'm really at ac.dekanfrus     |  of the cancer cell.
/ \  if you read it the right way.  |    -- Edward Abbey
```

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Charlie Gibbs on Tue, 04 Feb 2025 01:28:00 GMT
View Forum Message <> Reply to Message

On 2025-02-04, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:

> On Mon, 03 Feb 2025 11:39:25 -0500, Dan Espen wrote:
>
>> Meanwhile, fixes using 4 digit years are setting us up for the Y9K bug.
>
> I’m expecting that the Gregorian calendar, with its zero point set to some
> arbitrary guess at the birth date of some figure that was mythological
> anyway, will not be in use for that long.

We'll probably patch it with another rule. If, in addition to the
existing exceptions, any year divisible by 3200 is also not a leap
year, we should be getting pretty close. Of course, by then the
speed of the Earth's rotation might have changed enough to screw
things up. Or maybe the Sun will go supernova, rendering the
whole thing moot.

```
--
/~\  Charlie Gibbs                  |  Growth for the sake of
\ /  <cgibbs@kltpzyxm.invalid>      |  growth is the ideology
 X   I'm really at ac.dekanfrus     |  of the cancer cell.
/ \  if you read it the right way.  |    -- Edward Abbey
```

Subject: Re: old pharts, Multics vs Unix
Posted by Charlie Gibbs on Tue, 04 Feb 2025 01:28:01 GMT
View Forum Message <> Reply to Message

On 2025-02-04, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:

```
>  On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>
>>  I couldn't even interrupt my own program. They had to reboot the machine
>>  to fix it and I was told in no uncertain terms to never do that again!
>
>  Fragile things, mainframes. They were not battle-hardened by exposure to
>  inquisitive students, the way interactive timesharing systems were.
```

On the other hand, a lot of battle-hardening resulted soon afterwards.

```
--
/~\  Charlie Gibbs                  |  Growth for the sake of
\ /  <cgibbs@kltpzyxm.invalid>      |  growth is the ideology
 X   I'm really at ac.dekanfrus     |  of the cancer cell.
/ \  if you read it the right way.  |    -- Edward Abbey
```

Subject: Re: old pharts, Multics vs Unix
Posted by cross on Tue, 04 Feb 2025 01:45:32 GMT
View Forum Message <> Reply to Message

In article <m0d80sFllotU1@mid.individual.net>,
moi <findlaybill@blueyonder.co.uk> wrote:

```
> On 04/02/2025 01:05, Lawrence D'Oliveiro wrote:
>>  On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>>
>>>  I couldn't even interrupt my own program. They had to reboot the machine
>>>  to fix it and I was told in no uncertain terms to never do that again!
>>
>>  Fragile things, mainframes. They were not battle-hardened by exposure to
>>  inquisitive students, the way interactive timesharing systems were.
>
> Nonsense.
```

Lol. One of the major compute engines used by undergrads when I
was young was an ES/3090-600S running VM/ESA. It certainly got
tested by inquisitive students.

The troll continues to show his ignorance.

 - Dan C.

Subject: Re: Wang Terminals (Re: old pharts, Multics vs Unix)
Posted by Anonymous on Tue, 04 Feb 2025 02:55:08 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

On Mon, 3 Feb 2025 03:48:06 -0000 (UTC), Lars Poulsen wrote:
>> Having chatted with the Wang people, they talked us into representing
>> their little desktops in Denmark, and when they started making word
>> processors, we took on those as well, and soon that was most of our
>> business.

On 2025-02-04, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> I remember I got a glimpse at some Wang assembly code once, and it looked
> remarkably like my impressions of IBM 360/370 assembler. I think Dr Wang
> may have copied the IBM instruction set. Or at least large parts of it.
>
> The Asianometry channel covered the history of Dr Wang and his company
> <https://www.youtube.com/watch?v=MgDZQy0nN-Y>. Obviously a brilliant
> engineer, who managed to be an astute businessman as well. But he insisted
> on passing control of the company to a son who was not nearly as astute.

The Wang VS system really did copy most of the unprivileged IBM 360
instruction set. But they really did want the customers to program in
RPG-II and derivative "4th generation" programming languages.

On Mon, 3 Feb 2025 03:48:06 -0000 (UTC), Lars Poulsen wrote:
>> I also had to battle Wang engineering to get proper font adjustments to
>> accommodate the extra Danish vowels: æøå/ÆØÅ and get them correctly
>> placed on the keyboard.

On 2025-02-04, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> Now that the technical obstacles to support of such alphabets have
> disappeared with the advent of Unicode, you still see a cultural
> reluctance among some to acknowledge their existence.

Keyboards was the big problem. The Norvegian Wang subsidiary had copied
the DecWriter Norvegian keyboards, which were really badly screwed up.
(And DEC in Denmark had accepted those Norvegian keyboards.) I had some
fun researching /standards/ for office keyboards to get something that
would work for office typists.

Wang also did not understand the concept of /ergonomics/. We found that
the market required detached keyboards, but the only reason I got
anywhere, was because Germany had made it a workplace safety law in
order to reduce RSI and carpal tunnel disabilities.

When I visited the Wang HQ in early 1980, I got a glimpse of how
disconnected they were from my reality. In the lobby, 3-4 receptionist
secretaries in short skirts sat on bar stools in short skirts and high
heels at little round cocktail lounge tables with a terminal teetering
on top. Of course that would not have worked with a detached keyboard,

because the table was too small for that. The whole thing would not
have been tolerated, but have created instant lawsuits as a "hostile
work environment".

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Tue, 04 Feb 2025 04:05:54 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 4 Feb 2025 01:15:08 +0000, moi wrote:

> On 04/02/2025 01:05, Lawrence D'Oliveiro wrote:
>
>> On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>>
>>> I couldn't even interrupt my own program. They had to reboot the
machine
>>> to fix it and I was told in no uncertain terms to never do that again!
>>
>> Fragile things, mainframes.
>
> Nonsense.

The very post I was replying to gives the lie to your denial.

There is this persistent myth that mainframes are highly reliable, with
uptimes measurable in years.

Clearly not in this case.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Bob Martin on Tue, 04 Feb 2025 07:13:37 GMT
View Forum Message <> Reply to Message

On 3 Feb 2025 at 21:40:05, Rich Alderson <news@alderson.users.panix.com> wrote:
> scott@slp53.sl.home (Scott Lurndal) writes:
>
>> Peter Flass <peter_flass@yahoo.com> writes:
>>> Dan Espen <dan1espen@gmail.com> wrote:
>>>> scott@slp53.sl.home (Scott Lurndal) writes:
>
>> <Y2K mitigations>
>
>>>> I fixed one of my applications by looking at the current year, then

>>>> setting the window accordingly. Fixed forever.
>
>>> Depends on the application. For something like Social Security you may have
>>> records on someone born this year(parents applied for SSN) to this year
>>> minus 100 or more. For a payments system this works fine, since all you
>>> usually need is last year, this year, and next year.
>
>> For SS, even in the 1960's, they'd have to be able to store dates
>> from the 19th century; two-digit years were never useful in that context.
>
> Indeed. My greatgrandfather Alderson was born in 1876, and died in 1962; my
> greatgrandmother was born in 1885, and died 6 weeks short of her 90th birthday
> in 1975.

My grandfather was born in 1863.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Bob Martin on Tue, 04 Feb 2025 07:16:00 GMT
View Forum Message <> Reply to Message

On 4 Feb 2025 at 01:15:08, moi <findlaybill@blueyonder.co.uk> wrote:
> On 04/02/2025 01:05, Lawrence D'Oliveiro wrote:
>> On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>>
>>> I couldn't even interrupt my own program. They had to reboot the machine
>>> to fix it and I was told in no uncertain terms to never do that again!
>>
>> Fragile things, mainframes. They were not battle-hardened by exposure to
>> inquisitive students, the way interactive timesharing systems were.
>
> Nonsense.

Everything Lawrence says is nonsense.
If only people would stop responding to him.

Subject: Re: old pharts, Multics vs Unix
Posted by Peter Flass on Tue, 04 Feb 2025 13:09:27 GMT
View Forum Message <> Reply to Message

Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
> On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>
>> I couldn't even interrupt my own program. They had to reboot the machine
>> to fix it and I was told in no uncertain terms to never do that again!
>

> Fragile things, mainframes. They were not battle-hardened by exposure to
> inquisitive students, the way interactive timesharing systems were.
>

Might the problem have been the controller for the 2260s? I recall they
were somewhat kludge, using delay-lines as display buffers.

--
Pete

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Harry Vaderchi on Tue, 04 Feb 2025 13:45:43 GMT
View Forum Message <> Reply to Message

On Tue, 04 Feb 2025 01:27:59 GMT
Charlie Gibbs <cgibbs@kltpzyxm.invalid> wrote:

> On 2025-02-04, Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>
>> On Mon, 03 Feb 2025 17:21:50 -0500, Dan Espen wrote:
>>
>>> Hopefully, those 64bit dates will remain an internal representation.
>>> Anything converting a 4 digit year to a 64bit date still has a Y9K
>>> problem.
>>
>> The ultimate integer representation of time would be in units of the
>> Planck interval (5.39e-44 seconds, according to Da Wiki). Representing the
>> current age of the observable Universe would take about 200 bits,
>> according to this scale.
>>
>> So I figure a 256-bit integer in this scale (representing a maximum
>> duration of about 10**16 years) ought to be enough to represent any
>> duration of time we are ever likely to be physically concerned with. ;)
>
> Everybody sing along:
>
>   The whole universe was in a hot dense state
>   When nearly fourteen billion years ago expansion started... wait!
>   The earth began to cool
>   The autotrophs began to drool
>   Neanderthals developed tools
>   We built a wall (we built the pyramids)
>   Math, science, history
>   Unraveling the mystery
>   That all started with a big bang (BANG!)
>
> You're sitting in my spot.

>
I'll raise you the Galaxy song by Monty Py^w^w Eric Idle.

--
Bah, and indeed Humbug.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Harry Vaderchi on Tue, 04 Feb 2025 13:47:59 GMT
View Forum Message <> Reply to Message

On 4 Feb 2025 07:13:37 GMT
Bob Martin <bob.martin@excite.com> wrote:

```
>  On 3 Feb 2025 at 21:40:05, Rich Alderson <news@alderson.users.panix.com> wrote:
>>  scott@slp53.sl.home (Scott Lurndal) writes:
>>
>>>  Peter Flass <peter_flass@yahoo.com> writes:
>>>>  Dan Espen <dan1espen@gmail.com> wrote:
>>>> > scott@slp53.sl.home (Scott Lurndal) writes:
>>
>>>   <Y2K mitigations>
>>
>>>> > I fixed one of my applications by looking at the current year, then
>>>> > setting the window accordingly.  Fixed forever.
>>
>>>>  Depends on the application. For something like Social Security you may have
>>>>  records on someone born this year(parents applied for SSN) to this year
>>>>  minus 100 or more. For a payments system this works fine, since all you
>>>>  usually need is last year, this year, and next year.
>>
>>>  For SS, even in the 1960's, they'd have to be able to store dates
>>>  from the 19th century; two-digit years were never useful in that context.
>>
>>  Indeed.  My greatgrandfather Alderson was born in 1876, and died in 1962; my
>>  greatgrandmother was born in 1885, and died 6 weeks short of her 90th birthday
>>  in 1975.
>
>  My grandfather was born in 1863.
>
```

Can we presume that your other grandfather is no longer around?

--
Bah, and indeed Humbug.

Subject: Re: old pharts, Multics vs Unix

Posted by Anonymous on Tue, 04 Feb 2025 14:08:27 GMT
View Forum Message <> Reply to Message

Originally posted by: Colin Macleod

Peter Flass <peter_flass@yahoo.com> posted:

> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>> On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>>
>>> I couldn't even interrupt my own program. They had to reboot the machine
>>> to fix it and I was told in no uncertain terms to never do that again!
>>
>> Fragile things, mainframes. They were not battle-hardened by exposure to
>> inquisitive students, the way interactive timesharing systems were.
>>
>
> Might the problem have been the controller for the 2260s? I recall they
> were somewhat kludge, using delay-lines as display buffers.
>

--
Colin Macleod ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ https://cmacleod.me.uk

Fermented grapes count towards my five-a-day, right?

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Dan Espen on Tue, 04 Feb 2025 16:08:56 GMT
View Forum Message <> Reply to Message

Charlie Gibbs <cgibbs@kltpzyxm.invalid> writes:

> On 2025-02-03, Dan Espen <dan1espen@gmail.com> wrote:
>
>> scott@slp53.sl.home (Scott Lurndal) writes:
>>
>>> Dan Espen <dan1espen@gmail.com> writes:
>>>
>>>> Meanwhile, fixes using 4 digit years are setting us up for
>>>> the Y9K bug.
>>>
>>> A signed 64-bit integer can represent any time value (in seconds)
>>> for a few hundred million years BCE and CE.
>>
>> Hopefully, those 64bit dates will remain an internal representation.
>> Anything converting a 4 digit year to a 64bit date still has

>> a Y9K problem.
>
> s/Y9K/Y10K/

I was going to go with Y9999 but it just didn't seem right.

> At the year 9000 we'll still have a millennium left to fix it.

Y2K proved it's no fun unless you wait until you have 2 years left.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by Dan Espen on Tue, 04 Feb 2025 16:15:47 GMT
View Forum Message <> Reply to Message

Peter Flass <peter_flass@yahoo.com> writes:

> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>> On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>>
>>> I couldn't even interrupt my own program. They had to reboot the machine
>>> to fix it and I was told in no uncertain terms to never do that again!
>>
>> Fragile things, mainframes. They were not battle-hardened by exposure to
>> inquisitive students, the way interactive timesharing systems were.
>
> Might the problem have been the controller for the 2260s? I recall they
> were somewhat kludge, using delay-lines as display buffers.

I read about those delay lines long after my 2260 project.
They sure sound like they would be easy to overload but I don't know if
that would slow the whole system down.

My thoughts were, it could put a heavy load on the multiplexor channel
which the printers and card readers needed,
and any 2260 program was likely to be running at a high priority,
locally attached 2260's used the attention interrupt, possibly
interfering with trying to hit attention on a terminal.

--
Dan Espen

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Tue, 04 Feb 2025 16:21:46 GMT
View Forum Message <> Reply to Message

Originally posted by: John Ames

On Tue, 4 Feb 2025 01:45:32 -0000 (UTC)
cross@spitfire.i.gajendra.net (Dan Cross) wrote:

> Lol. One of the major compute engines used by undergrads when I
> was young was an ES/3090-600S running VM/ESA. It certainly got
> tested by inquisitive students.
>
> The troll continues to show his ignorance.

Lawrence's ability to make blanket assertions on things outside his own
window of experience with complete authority - in spite of the testimony
of those with firsthand knowledge of the domain in question - is fairly
breathtaking.

Subject: Re: old pharts, Multics vs Unix
Posted by Bill Findlay on Tue, 04 Feb 2025 17:33:17 GMT
View Forum Message <> Reply to Message

On 4 Feb 2025, Lawrence D'Oliveiro wrote
(in article <vns3n2$1n890$2@dont-email.me>):

> On Tue, 4 Feb 2025 01:15:08 +0000, moi wrote:
>
>> On 04/02/2025 01:05, Lawrence D'Oliveiro wrote:
>>
>>> On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>>>
>>>> I couldn't even interrupt my own program. They had to reboot the machine
>>>> to fix it and I was told in no uncertain terms to never do that again!
>>>
>>> Fragile things, mainframes.
>>
>> Nonsense.
>
> The very post I was replying to gives the lie to your denial.

I restore the context you omitted in bad faith:

> They were not battle-hardened by exposure to inquisitive students, the way
> interactive timesharing systems were.

I say again: nonsense.

--
Bill Findlay

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Bill Findlay on Tue, 04 Feb 2025 17:36:17 GMT
View Forum Message <> Reply to Message

On 4 Feb 2025, Bob Martin wrote
(in article <m0dt5gFpgnqU1@mid.individual.net>):

> On 4 Feb 2025 at 01:15:08, moi<findlaybill@blueyonder.co.uk> wrote:
>> On 04/02/2025 01:05, Lawrence D'Oliveiro wrote:
>>> On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>>>
>>>> I couldn't even interrupt my own program. They had to reboot the machine
>>>> to fix it and I was told in no uncertain terms to never do that again!
>>>
>>> Fragile things, mainframes. They were not battle-hardened by exposure to
>>> inquisitive students, the way interactive timesharing systems were.
>>
>> Nonsense.
>
> Everything Lawrence says is nonsense.
> If only people would stop responding to him.

Untruths need to be challenged.

--
Bill Findlay

Subject: Stress-testing of Mainframes (the HASP story)
Posted by Anonymous on Tue, 04 Feb 2025 20:26:30 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

Lynn Wheeler <lynn@garlic.com> writes:
>>> however, all 3270s were half-duplex and if you were unfortunate to hit a
>>> key same time system went to write to screen, it would lock the keyboard
>>> and would have to stop and hit the reset key. YKT developed a FIFO box
>>> for 3277, unplug the keyboard from the 3277 head, plug the FIFO box into
>>> the head and plug the 3277 keyboard into the fifo box ... eliminating
>>> the unfortunate keyboad lock.

On 2025-02-03, Colin Macleod <user7@newsgrouper.org.uk.invalid> wrote:
> This reminds me of something when I was a student around 1975. My university's
> only computer was an IBM 360/44 and we could use some 2260 terminals. Digging
> through some manuals I got the idea that by munging the "carriage control
> character" from a Fortran program I might be able to break out of the
> restriction of block-mode operation and persuade a 2260 to do animated
> graphics.
>
> When tried my proof-of-concept program I did get the terminal to keep
> redrawing a very flickery but changing few lines. But while I was watching
> this, the other users in the lab started complaining that their terminals were
> frozen. Then the operators started running around trying to find out why
> the whole mainframe had hung. It appeared that my hack had somehow elevated
> the priority of my animation such that nothing else was getting run at all.
> I couldn't even interrupt my own program. They had to reboot the machine to
> fix it and I was told in no uncertain terms to never do that again!

My own "NEVER do that again" stories:
1) 1971, I think:

I was one of three part-time operators of an IBM 1130 that had been
pressed into service as a HASP RJE terminal for the new IBM/360-65 MVT
system at NEUCC (DTU, Lyngby), but we still had a tray full of
pre-punched IBM 1130 DOS control cards:

```
  // JOB T
  // FOR
  // XQT
```

One of my coworkers pondered what would happen if you submitted a small
deck with an 1130 JOB card instead of an OS/360 JOB card. I explained
that the spool system would take it as a job with a sort-of valid job
card, but with invalid values in most fields, such as the billing
account, the job queue (class=) field etc. Then when the job scheduler
got to that point in the queue, it would fail all sorts of syntax and
validity checks, and a printout would be returned with all sorts of
error messages. He did not believe me, so I proved it by submitting it.
The one solitary JOB card came back as 4 or so pages of print:
- front separator page
- HASP console message log
- JCL processing log
- back separator page

He then pondered what would happen if we read in a whole stack of JOB
cards, and I said "same thing for each card in the stack". So we did
it.

It read the first couple of dozen cards quite fast, then slowed down a lot, and read one card at a time, a couple of seconds apart. And then it went REALLY slow, reading one card after printing the 4 pages of job output.

That was because the HASP job queue had about 100 slots, with a pre-allocated cylinder for the input cards for the job and I think also the log files. Once all the job queue slots were taken up by the BAD jobs, no new jobs could be read in FROM ANYWHERE in the RJE network, until a job slot had been finished and the slot made available for a new job.

But it was worse. The OS/360 job card had a job ID field at the beginning of the card (before the word JOB), and all the BAD jobs had the same invalid ID. So when a job came in with a blank ID field, it was a duplicate of the preceding BAD jobs; each new job had to be assigned a unique new ID, and this generated a console message announcing a duplicate ID, folowed by a console message announcing the new ID, followed by a message that the new job had been placed in HOLD status, so that only one of these jobs with the same ID could be in the OS job queue. After the first duplicate, the newly generated ID was also a duplicate, triggering more HASP console messages as the queue was searched trying to find a new unique name. As each job finished, HASP would release the jobs tht had been HELD, tgriggering more messages. Soon, the 1050 console was running 15 minutes behind, and the operator was unable to get a command in.

In the end, I think they had to re-IPL the system and FORMAT the HASP SPOOL disk to recover.

I was lucky to keep my job.

I think someone wrote a PTF for HASP to make sure that could not happen again!

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Tue, 04 Feb 2025 22:58:00 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 04 Feb 2025 17:33:17 +0000, Bill Findlay wrote:

> I restore the context you omitted ...

So you admit that your claim only applied to the subsidiary point, not the main one.

> ... in bad faith

Does not making your point clear count as “bad faith”?

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Tue, 04 Feb 2025 22:58:30 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 4 Feb 2025 08:21:46 -0800, John Ames wrote:

> Lawrence's ability to make blanket assertions on things outside his own
> window of experience with complete authority - in spite of the testimony
> of those with firsthand knowledge of the domain in question ...

In this case, it was very clearly because of it, not in spite of it.

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Tue, 04 Feb 2025 23:01:16 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 4 Feb 2025 06:09:27 -0700, Peter Flass wrote:

> Lawrence D'Oliveiro <ldo@nz.invalid> wrote:
>
>> On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>>
>>> I couldn't even interrupt my own program. They had to reboot the
>>> machine to fix it and I was told in no uncertain terms to never do
>>> that again!
>>
>> Fragile things, mainframes. They were not battle-hardened by exposure
>> to inquisitive students, the way interactive timesharing systems were.
>>
> Might the problem have been the controller for the 2260s? I recall they
> were somewhat kludge, using delay-lines as display buffers.

Remember, the whole point of a mainframe was to devolve as much I/O
processing load as possible to the peripheral controllers, bothering the
CPU as little as possible.

Obviously there was a loophole in this, if certain sequences of user actions could cause the controller to overload the CPU with interrupts.

Sometimes these bugs are not specifically in a particular component, but
in the way that different components interact.

Subject: Re: Stress-testing of Mainframes (the HASP story)
Posted by Anonymous on Tue, 04 Feb 2025 23:02:55 GMT

View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 4 Feb 2025 20:26:30 -0000 (UTC), Lars Poulsen wrote:

> In the end, I think they had to re-IPL the system and FORMAT the HASP
> SPOOL disk to recover.
>
> I was lucky to keep my job.

IBM was, of course blameless. It was easier to sack a hapless employee
than to switch to a supplier of better-quality products.

Subject: Re: Wang Terminals (Re: old pharts, Multics vs Unix)
Posted by Anonymous on Tue, 04 Feb 2025 23:05:21 GMT

View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 4 Feb 2025 02:55:08 -0000 (UTC), Lars Poulsen wrote:

> Keyboards was the big problem. The Norvegian Wang subsidiary had copied
> the DecWriter Norvegian keyboards, which were really badly screwed up.
> (And DEC in Denmark had accepted those Norvegian keyboards.) I had some
> fun researching /standards/ for office keyboards to get something that
> would work for office typists.

But wouldn’t they have been based on official national standards? Hard to
think of computer companies (ones smaller than IBM, anyway) making up
their own specs, if there is already one established in the target market.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Anonymous on Tue, 04 Feb 2025 23:09:26 GMT

View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 2 Feb 2025 15:17 +0000 (GMT Standard Time), John Dallman wrote:

> In article <vnmh9e$butt$6@dont-email.me>, ldo@nz.invalid (Lawrence
> D'Oliveiro) wrote:
>>
>> You think a modern company like Facebook runs
>> its main system on a mainframe, using some proprietary mainframe DBMS?
>> No, it uses MySQL/MariaDB with other pieces like memcached, plus code
>> written in its home-grown PHP engine (open-sourced as HHVM), and also
>> some back-end Python (that we know of).
>
> Facebook has quite different synchronisation requirements from a credit
> card provider. It doesn't matter to FB if updates to the page someone is
> looking at arrive take a few seconds to arrive.
>
> Speed of synchronisation matters a lot to a credit card provider who is
> trying to enforce customer credit limits and avoid double-spends. They
> still use mainframes with z/TPF for that. z/TPF is a curious OS; it
> essentially makes a mainframe into a single real-time transaction
> processing system.

Look at it this way: the article I got the info about Facebook from was
from some years ago, back when Facebook only had about a billion users who
were active at least once a month.

So that’s a bare minimum of about 380 real-time transactions per second,
on average, 24 hours a day, day in and day out, most likely much higher at
peak times.

Where do you have any IBM mainframe that can cope with that?

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Tue, 04 Feb 2025 23:35:43 GMT
View Forum Message <> Reply to Message

Originally posted by: moi

On 04/02/2025 22:58, Lawrence D'Oliveiro wrote:
> On Tue, 04 Feb 2025 17:33:17 +0000, Bill Findlay wrote:
>
>> I restore the context you omitted ...
>
> So you admit that your claim only applied to the subsidiary point, not the
> main one.
>
>> ... in bad faith

>
> Does not making your point clear count as “bad faith”?

I do not accept that you are as stupid as you seem to be.

Into the kill file with you.

--
Bill F.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Bob Martin on Wed, 05 Feb 2025 05:55:51 GMT
View Forum Message <> Reply to Message

On 4 Feb 2025 at 13:47:59, "Kerr-Mudd, John" <admin@127.0.0.1> wrote:
> On 4 Feb 2025 07:13:37 GMT
> Bob Martin <bob.martin@excite.com> wrote:
>
>> On 3 Feb 2025 at 21:40:05, Rich Alderson <news@alderson.users.panix.com> wrote:
>>> scott@slp53.sl.home (Scott Lurndal) writes:
>>>
>>>> Peter Flass <peter_flass@yahoo.com> writes:
>>>> > Dan Espen <dan1espen@gmail.com> wrote:
>>>> >> scott@slp53.sl.home (Scott Lurndal) writes:
>>>
>>>> <Y2K mitigations>
>>>
>>>> >> I fixed one of my applications by looking at the current year, then
>>>> >> setting the window accordingly. Fixed forever.
>>>
>>>> > Depends on the application. For something like Social Security you may have
>>>> > records on someone born this year(parents applied for SSN) to this year
>>>> > minus 100 or more. For a payments system this works fine, since all you
>>>> > usually need is last year, this year, and next year.
>>>
>>>> For SS, even in the 1960's, they'd have to be able to store dates
>>>> from the 19th century; two-digit years were never useful in that context.
>>>
>>> Indeed. My greatgrandfather Alderson was born in 1876, and died in 1962; my
>>> greatgrandmother was born in 1885, and died 6 weeks short of her 90th birthday
>>> in 1975.
>>
>> My grandfather was born in 1863.
>>
> Can we presume that your other grandfather is no longer around?

Sorry, full info:

Paternal grandfather : 1863 to 1952
Maternal grandfather : 1877 to 1940

I'm 83

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by cross on Wed, 05 Feb 2025 16:58:46 GMT
View Forum Message <> Reply to Message

In article <0001HW.2D528791002FB78C30DA3538F@news.individual.net>,
Bill Findlay <findlaybill@blueyonder.co.uk> wrote:
> On 4 Feb 2025, Bob Martin wrote
> (in article <m0dt5gFpgnqU1@mid.individual.net>):
>
>> On 4 Feb 2025 at 01:15:08, moi<findlaybill@blueyonder.co.uk> wrote:
>>> On 04/02/2025 01:05, Lawrence D'Oliveiro wrote:
>>>> On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>>>>
>>>> > I couldn't even interrupt my own program. They had to reboot the machine
>>>> > to fix it and I was told in no uncertain terms to never do that again!
>>>>
>>>> Fragile things, mainframes. They were not battle-hardened by exposure to
>>>> inquisitive students, the way interactive timesharing systems were.
>>>
>>> Nonsense.
>>
>> Everything Lawrence says is nonsense.
>> If only people would stop responding to him.
>
> Untruths need to be challenged.

My suggestion for handling this would be to have a periodically
posted FAQ that includes a section on cranks and bad-faith
posters that mentions Lawrence.

 - Dan C.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Harry Vaderchi on Wed, 05 Feb 2025 18:56:23 GMT
View Forum Message <> Reply to Message

On 5 Feb 2025 05:55:51 GMT
Bob Martin <bob.martin@excite.com> wrote:

> On 4 Feb 2025 at 13:47:59, "Kerr-Mudd, John" <admin@127.0.0.1> wrote:

>> On 4 Feb 2025 07:13:37 GMT
>> Bob Martin <bob.martin@excite.com> wrote:
>>
>>> On 3 Feb 2025 at 21:40:05, Rich Alderson <news@alderson.users.panix.com> wrote:
>>>> scott@slp53.sl.home (Scott Lurndal) writes:
>>>>
>>>> > Peter Flass <peter_flass@yahoo.com> writes:
>>>> >> Dan Espen <dan1espen@gmail.com> wrote:
>>>> >>> scott@slp53.sl.home (Scott Lurndal) writes:
>>>>
>>>> > <Y2K mitigations>
>>>>
>>>> >>> I fixed one of my applications by looking at the current year, then
>>>> >>> setting the window accordingly. Fixed forever.
>>>>
>>>> >> Depends on the application. For something like Social Security you may have
>>>> >> records on someone born this year(parents applied for SSN) to this year
>>>> >> minus 100 or more. For a payments system this works fine, since all you
>>>> >> usually need is last year, this year, and next year.
>>>>
>>>> > For SS, even in the 1960's, they'd have to be able to store dates
>>>> > from the 19th century; two-digit years were never useful in that context.
>>>>
>>>> Indeed. My greatgrandfather Alderson was born in 1876, and died in 1962; my
>>>> greatgrandmother was born in 1885, and died 6 weeks short of her 90th birthday
>>>> in 1975.
>>>
>>> My grandfather was born in 1863.
>>>
>> Can we presume that your other grandfather is no longer around?
>
> Sorry, full info:
> Paternal grandfather : 1863 to 1952
> Maternal grandfather : 1877 to 1940
>
> I'm 83
>
Wow! Sorry, I was just being a bit pernickity about only 1 grandfather.

I was lucky enough to have some IBM mainframe experience prior to being an
early PC adopter. I'm still reliving it - see my 8086 PC asm code in
a.l.a/c.o.m.p

--
Bah, and indeed Humbug.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by Harry Vaderchi on Wed, 05 Feb 2025 18:58:07 GMT
View Forum Message <> Reply to Message

On Wed, 5 Feb 2025 16:58:46 -0000 (UTC)
cross@spitfire.i.gajendra.net (Dan Cross) wrote:

> In article <0001HW.2D528791002FB78C30DA3538F@news.individual.net>,
> Bill Findlay <findlaybill@blueyonder.co.uk> wrote:
>> On 4 Feb 2025, Bob Martin wrote
>> (in article <m0dt5gFpgnqU1@mid.individual.net>):
>>
>>> On 4 Feb 2025 at 01:15:08, moi<findlaybill@blueyonder.co.uk> wrote:
>>>> On 04/02/2025 01:05, Lawrence D'Oliveiro wrote:
>>>> > On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>>>> >
>>>> > > I couldn't even interrupt my own program. They had to reboot the machine
>>>> > > to fix it and I was told in no uncertain terms to never do that again!
>>>> >
>>>> > Fragile things, mainframes. They were not battle-hardened by exposure to
>>>> > inquisitive students, the way interactive timesharing systems were.
>>>>
>>>> Nonsense.
>>>
>>> Everything Lawrence says is nonsense.
>>> If only people would stop responding to him.
>>
>> Untruths need to be challenged.
>
> My suggestion for handling this would be to have a periodically
> posted FAQ that includes a section on cranks and bad-faith
> posters that mentions Lawrence.
>

Challenging idi^w uninformed presid^w posters rarely gets you anywhere
good.

--
Bah, and indeed Humbug.

Subject: Re: old pharts, Multics vs Unix vs mainframes
Posted by cross on Wed, 05 Feb 2025 19:32:30 GMT
View Forum Message <> Reply to Message

In article <20250205185807.78ec4cec6c4b6d1f447cdd9a@127.0.0.1>,
Kerr-Mudd, John <admin@127.0.0.1> wrote:

> On Wed, 5 Feb 2025 16:58:46 -0000 (UTC)
> cross@spitfire.i.gajendra.net (Dan Cross) wrote:
>
>> In article <0001HW.2D528791002FB78C30DA3538F@news.individual.net>,
>> Bill Findlay <findlaybill@blueyonder.co.uk> wrote:
>>> On 4 Feb 2025, Bob Martin wrote
>>> (in article <m0dt5gFpgnqU1@mid.individual.net>):
>>>
>>>> On 4 Feb 2025 at 01:15:08, moi<findlaybill@blueyonder.co.uk> wrote:
>>>> > On 04/02/2025 01:05, Lawrence D'Oliveiro wrote:
>>>> > > On Mon, 03 Feb 2025 15:03:10 GMT, Colin Macleod wrote:
>>>> > >
>>>> > > > I couldn't even interrupt my own program. They had to reboot the machine
>>>> > > > to fix it and I was told in no uncertain terms to never do that again!
>>>> > >
>>>> > > Fragile things, mainframes. They were not battle-hardened by exposure to
>>>> > > inquisitive students, the way interactive timesharing systems were.
>>>> >
>>>> > Nonsense.
>>>>
>>>> Everything Lawrence says is nonsense.
>>>> If only people would stop responding to him.
>>>
>>> Untruths need to be challenged.
>>
>> My suggestion for handling this would be to have a periodically
>> posted FAQ that includes a section on cranks and bad-faith
>> posters that mentions Lawrence.
>
> Challenging idi^w uninformed presid^w posters rarely gets you anywhere
> good.

Truth.

Though in the case of this particular uninformed poster, he need
not be challenged; one can simply put a blurb about him the FAQ,
and post it on some regular cadence. Readers are free to use
that information or not, but it takes off any pressure to
respond directly to him.

 - Dan C.

Subject: Re: Stress-testing of Mainframes (the HASP story)
Posted by Peter Flass on Wed, 05 Feb 2025 22:59:32 GMT
View Forum Message <> Reply to Message

Lars Poulsen <lars@cleo.beagle-ears.com> wrote:

> Lynn Wheeler <lynn@garlic.com> writes:
>>>> however, all 3270s were half-duplex and if you were unfortunate to hit a
>>>> key same time system went to write to screen, it would lock the keyboard
>>>> and would have to stop and hit the reset key. YKT developed a FIFO box
>>>> for 3277, unplug the keyboard from the 3277 head, plug the FIFO box into
>>>> the head and plug the 3277 keyboard into the fifo box ... eliminating
>>>> the unfortunate keyboad lock.
>
> On 2025-02-03, Colin Macleod <user7@newsgrouper.org.uk.invalid> wrote:
>> This reminds me of something when I was a student around 1975. My university's
>> only computer was an IBM 360/44 and we could use some 2260 terminals. Digging
>> through some manuals I got the idea that by munging the "carriage control
>> character" from a Fortran program I might be able to break out of the
>> restriction of block-mode operation and persuade a 2260 to do animated
>> graphics.
>>
>> When tried my proof-of-concept program I did get the terminal to keep
>> redrawing a very flickery but changing few lines. But while I was watching
>> this, the other users in the lab started complaining that their terminals were
>> frozen. Then the operators started running around trying to find out why
>> the whole mainframe had hung. It appeared that my hack had somehow elevated
>> the priority of my animation such that nothing else was getting run at all.
>> I couldn't even interrupt my own program. They had to reboot the machine to
>> fix it and I was told in no uncertain terms to never do that again!
>
> My own "NEVER do that again" stories:
> 1) 1971, I think:
>
> I was one of three part-time operators of an IBM 1130 that had been
> pressed into service as a HASP RJE terminal for the new IBM/360-65 MVT
> system at NEUCC (DTU, Lyngby), but we still had a tray full of
> pre-punched IBM 1130 DOS control cards:
>   // JOB T
>   // FOR
>   // XQT
>
> One of my coworkers pondered what would happen if you submitted a small
> deck with an 1130 JOB card instead of an OS/360 JOB card. I explained
> that the spool system would take it as a job with a sort-of valid job
> card, but with invalid values in most fields, such as the billing
> account, the job queue (class=) field etc. Then when the job scheduler
> got to that point in the queue, it would fail all sorts of syntax and
> validity checks, and a printout would be returned with all sorts of
> error messages. He did not believe me, so I proved it by submitting it.
> The one solitary JOB card came back as 4 or so pages of print:
> - front separator page
> - HASP console message log
> - JCL processing log

> - back separator page
>
> He then pondered what would happen if we read in a whole stack of JOB
> cards, and I said "same thing for each card in the stack". So we did
> it.
>
> It read the first couple of dozen cards quite fast, then slowed down a
> lot, and read one card at a time, a couple of seconds apart. And then it
> went REALLY slow, reading one card after printing the 4 pages of job
> output.
>
> That was because the HASP job queue had about 100 slots, with a
> pre-allocated cylinder for the input cards for the job and I think also
> the log files. Once all the job queue slots were taken up by the BAD
> jobs, no new jobs could be read in FROM ANYWHERE in the RJE network,
> until a job slot had been finished and the slot made available for a new
> job.
>
> But it was worse. The OS/360 job card had a job ID field at the
> beginning of the card (before the word JOB), and all the BAD jobs had
> the same invalid ID. So when a job came in with a blank ID field, it was
> a duplicate of the preceding BAD jobs; each new job had to be assigned a
> unique new ID, and this generated a console message announcing a
> duplicate ID, folowed by a console message announcing the new ID,
> followed by a message that the new job had been placed in HOLD status,
> so that only one of these jobs with the same ID could be in the OS job
> queue. After the first duplicate, the newly generated ID was also a
> duplicate, triggering more HASP console messages as the queue was
> searched trying to find a new unique name. As each job finished, HASP
> would release the jobs tht had been HELD, tgriggering more messages.
> Soon, the 1050 console was running 15 minutes behind, and the operator
> was unable to get a command in.
>
> In the end, I think they had to re-IPL the system and FORMAT the HASP
> SPOOL disk to recover.
>
> I was lucky to keep my job.
>
> I think someone wrote a PTF for HASP to make sure that could not happen
> again!
>

There were only so many WTO (console) buffers, too, so writing lots of
stuff to the console would also bog the system down.

--
Pete

Subject: Re: Stress-testing of Mainframes (the HASP story)
Posted by Anonymous on Thu, 06 Feb 2025 00:07:11 GMT
View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

Lars Poulsen <lars@cleo.beagle-ears.com> wrote:
>> That was because the HASP job queue had about 100 slots, with a
>> pre-allocated cylinder for the input cards for the job and I think also
>> the log files. Once all the job queue slots were taken up by the BAD
>> jobs, no new jobs could be read in FROM ANYWHERE in the RJE network,
>> until a job slot had been finished and the slot made available for a new
>> job.
>>
>> But it was worse. The OS/360 job card had a job ID field at the
>> beginning of the card (before the word JOB), and all the BAD jobs had
>> the same invalid ID. So when a job came in with a blank ID field, it was
>> a duplicate of the preceding BAD jobs; each new job had to be assigned a
>> unique new ID, and this generated a console message announcing a
>> duplicate ID, folowed by a console message announcing the new ID,
>> followed by a message that the new job had been placed in HOLD status,
>> so that only one of these jobs with the same ID could be in the OS job
>> queue. After the first duplicate, the newly generated ID was also a
>> duplicate, triggering more HASP console messages as the queue was
>> searched trying to find a new unique name. As each job finished, HASP
>> would release the jobs that had been HELD, triggering more messages.
>> Soon, the 1050 console was running 15 minutes behind, and the operator
>> was unable to get a command in.
>>
>> In the end, I think they had to re-IPL the system and FORMAT the HASP
>> SPOOL disk to recover.
>>
>> I was lucky to keep my job.
>>
>> I think someone wrote a PTF for HASP to make sure that could not happen
>> again!

On 2025-02-05, Peter Flass <peter_flass@yahoo.com> wrote:
> There were only so many WTO (console) buffers, too, so writing lots of
> stuff to the console would also bog the system down.

The more I have learned later, the more I understand just how bad it
was. It was an impressive denial-of-service attack for its time. As the
phone calls flew from the machine room operator to the operations
manager, to the head systems programmer, to the IBM field support, and
on and on, red faces of embarrassment must have triggered explosive
anger.

And like any other system vulnerability then or later, it was a simple

case of insufficient input validation. In retrospect, it was bound to
happen sooner or later. ;-)

Subject: Re: Stress-testing of Mainframes (the HASP story)
Posted by cross on Thu, 06 Feb 2025 03:16:00 GMT
View Forum Message <> Reply to Message

In article <slrnvq7v9f.kpad.lars@cleo.beagle-ears.com>,
Lars Poulsen <lars@cleo.beagle-ears.com> wrote:
> [snip; great story]
> On 2025-02-05, Peter Flass <peter_flass@yahoo.com> wrote:
>> There were only so many WTO (console) buffers, too, so writing lots of
>> stuff to the console would also bog the system down.
>
> The more I have learned later, the more I understand just how bad it
> was. It was an impressive denial-of-service attack for its time. As the
> phone calls flew from the machine room operator to the operations
> manager, to the head systems programmer, to the IBM field support, and
> on and on, red faces of embarrassment must have triggered explosive
> anger.
>
> And like any other system vulnerability then or later, it was a simple
> case of insufficient input validation. In retrospect, it was bound to
> happen sooner or later. ;-)

Ah! So, what you're saying is that you added value by identify
the issue and its cause early on, allowing it to be corrected
before it appeared elsewhere. Well done! :-D

 - Dan C.

Subject: Re: Stress-testing of Mainframes (the HASP story)
Posted by John Levine on Thu, 06 Feb 2025 03:36:45 GMT
View Forum Message <> Reply to Message

It appears that Lars Poulsen <lars@cleo.beagle-ears.com> said:
> The more I have learned later, the more I understand just how bad it
> was. It was an impressive denial-of-service attack for its time. As the
> phone calls flew from the machine room operator to the operations
> manager, to the head systems programmer, to the IBM field support, and
> on and on, red faces of embarrassment must have triggered explosive
> anger.

As I've mentioned before, I crashed Princeton's 360/91 with this two line
Fortran program:

```
    CALL MAIN
    END
```

MAIN was the default name for a Fortran main program, so it recursively called itself. The Fortran initialization code told the system to save the existing floating point trap handlers, set up its own, and then restored them when the program exited.

Except that the space where it saved the existing trap handlers wasn't very big. Oops. Kaboom!
--
Regards,
John Levine, johnl@taugh.com, Primary Perpetrator of "The Internet for Dummies",
Please consider the environment before reading this e-mail. https://jl.ly

---

Subject: Re: Stress-testing of Mainframes (the HASP story)
Posted by Anne &amp; Lynn Wheel on Thu, 06 Feb 2025 19:06:08 GMT
View Forum Message <> Reply to Message

Lars Poulsen <lars@cleo.beagle-ears.com> writes:
> The more I have learned later, the more I understand just how bad it
> was. It was an impressive denial-of-service attack for its time. As the
> phone calls flew from the machine room operator to the operations
> manager, to the head systems programmer, to the IBM field support, and
> on and on, red faces of embarrassment must have triggered explosive
> anger.
>
> And like any other system vulnerability then or later, it was a simple
> case of insufficient input validation. In retrospect, it was bound to
> happen sooner or later. ;-)

As an undergraduate in the 60s, I had rewritten lots of CP67
.... including doing dynamic adpative resource management and scheduling.
After graduation I joined the science center and one of my hobbies was enhanced operating systems for internal datacenters.

In the decision to add virtual memory to all 370s, there was also the creation of the VM370 group and some of the people in the science split off from CSC, taking over the IBM Boston Programming Center on the 3rd flr (Multics was on 5th flr, CSC was on the 4th flr and CSC machine room was on the 2nd flr). In the morph of CP67->VM370, lots of stuff was simplified and/or dropped, no more multiprocessor support, in-queue scheduling time was only based on virtual problem CPU, and kernel integrity was really simplified, other stuff.

Now virtual machine could in get into top, interactive Q1 and execute some code that was almost supervisor CPU (and very little virtual problom CPU) ... resulting in run-away CPU use ... locking out much of the rest of the users. The simplification in kernel integrity resulted in "zombie" users. In 1974, I started migrated lots of CP67 to VM370R2-base for my internal CSC/VM ... which included curing the run-away CPU use and zombie users.

Another problem was CP67 would determine long wait state drop from queue and interactive Q1 based on real terminal type ... VM370 changed it to virtual terminal type. That worked OK as long as the virtual terminal type was the similar to the real terminal type ... which broke with CMS virtual terminal type 3215 but the real terminal was 3270. CMS would put up READ for 3215 and go into virtul wait (waiting for the enter interrupt indicating end of typing input) and would be dropped from queue. 3270 typing would be saved in local buffer, user hits enter and presents a ATTN to system, CMS does a read and goes into wait state and is dropped from queue, but end of read is almost immediately (rather than waiting for somebody typing).

CP67 increased the count of virtual machine active "high-speed" real device channel programs and at entry to virtual wait state and check "high-speed" channel program count ... if it was zero, virtual machine dropped from queue. VM370 at virtual machine entry to wait, would scan complete virtual device configuration looking for "high-speed" device active channel program, and virtual 3215 didn't qualify.

After transfer out to SJR on the west coast, ran into a different problem, SJR replaced it 370/195 MVT system with 370/168 MVS and 370/158 VM370. It included MVS 3830 disk controller and MVS 3330 string with VM370 3830 disk controller and VM370 3330 string. Both the MVS & VM370 3830 disk controller had dual channel connections to both systems, however there was strict rules that never would MVS 3330 on VM370 string .... but one morning operator mounted a MVS 3330 on VM370 drive ... and almost immediately operations started getting irate phone calls from all over the bldg.

Issue was OS/360 and descendents make exensive use of multi-track search CCW ... which can take 1/3rd sec elapsed time ... which locks up controller ... and locks out all devices on that controller .... interferring with trivial interactive response that involves any disk I/O (MVS/TSO users are use to it, but the CMS users were use to better than .25sec interactive response).

Demands that operations move the offending MVS disk to the MVS string was met with it would be done offshit. We get a VM370-tuned VS1 system and mount its sysem pack on a MVS 3330 drive and start a program. The highly tuned VM370 VS1, even running on loaded VM370 158 ... could bring the MVS 168 nearly to a halt ... minimizing its interference with CMS

workloads. Operations immediately agree to move the offending MVS 3330
if we move the VS1 3330.

--
virtualization experience starting Jan1968, online at home since Mar1970

Subject: Re: old pharts, Multics vs Unix
Posted by Anonymous on Thu, 06 Feb 2025 20:55:53 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Tue, 4 Feb 2025 23:35:43 +0000, moi wrote:

> I do not accept that ...

.... you quoted a part of my posting that you were not responding to, and
when I quoted the same part in my reply, you accused me of “bad faith”.

Subject: Re: Wang Terminals (Re: old pharts, Multics vs Unix)
Posted by Anonymous on Sun, 09 Feb 2025 18:04:13 GMT
View Forum Message <> Reply to Message

Originally posted by: David Lesher

I recall having to use Wang wordprocessing. Besides many
disadvantages vs. Wordstar &/or WordPerfect, the terminals
needed two coax cables, one BNC, another TNC.

Yet other times they only needed one. What was the story with
the two needed, sometimes?

--
A host is a host from coast to coast...............wb8foz@panix.com
& no one will talk to a host that's close.........................
Unless the host (that isn't close).........................pob 1433
is busy, hung or dead....................................20915-1433

Subject: Re: Wang Terminals (Re: old pharts, Multics vs Unix)
Posted by Anonymous on Sun, 09 Feb 2025 22:32:32 GMT

View Forum Message <> Reply to Message

Originally posted by: Lars Poulsen

On 2025-02-09, David Lesher <wb8foz@panix.com> wrote:
> I recall having to use Wang wordprocessing. Besides many
> disadvantages vs. Wordstar &/or WordPerfect, the terminals
> needed two coax cables, one BNC, another TNC.
>
> Yet other times they only needed one. What was the story with
> the two needed, sometimes?

Gee, it's been 45 years, so my memory is just a bit shaky.

First, the systems you are comparing them to required that you have a PC in the first place. The Wang 1200 WPS came out in 1976, the IBM 5150 did not come out until 1981. Wordstar for CP/M came out in 1978. WordPerfect first came out in 1980 (under the name SSI*WP) and became WordPerfect when it moved to MS-DOS in 1982.

When I was introduced to the WPS in 1978, I had been using documention scripting languages for about 8 years, mostly in the form of the Univac Exec-8 @DOC program, which was great for programmers, but much more suitable for technical documentation than for business correspondence. WPS was mostly WYSIWYG, which made it much easier to use for office people.

When Wang arrived, it owned this market by virtue of being first! Wikipedia says: "WordPerfect 1.0 represented a significant departure from the previous Wang standard for word processing."

As for the terminal wiring: I found a "maintenance manual"
 http://bitsavers.informatik.uni-stuttgart.de/pdf/wang/ois/74
2-0664-1_OIS140-145_Maintenance_19871120.pdf
.... but that seems only to cover installation procedures. But I think the coax wiring is described in
 http://bitsavers.informatik.uni-stuttgart.de/pdf/wang/vs/com
ms/742-1102_WangNet_Backbone_19850724.pdf

This describes an RF coax "loop" with branches. One cable is "outbound", the other is "inbound".

Subject: Re: Wang Terminals (Re: old pharts, Multics vs Unix)
Posted by Anonymous on Sun, 09 Feb 2025 23:49:12 GMT
View Forum Message <> Reply to Message

Originally posted by: Lawrence D'Oliveiro

On Sun, 9 Feb 2025 22:32:32 -0000 (UTC), Lars Poulsen wrote:

> When I was introduced to the WPS in 1978, I had been using documention
> scripting languages for about 8 years, mostly in the form of the Univac
> Exec-8 @DOC program, which was great for programmers, but much more
> suitable for technical documentation than for business correspondence.
> WPS was mostly WYSIWYG, which made it much easier to use for office
> people.

I wonder how office people collaborate on a document, though. How do you merge contributions from two or more contributors using a WYSIWYG app, without something equivalent to patch/diff?

> When Wang arrived, it owned this market by virtue of being first!
> Wikipedia says: "WordPerfect 1.0 represented a significant departure
> from the previous Wang standard for word processing."

Notice they say “Wang standard”, not “standard”. The world’s most popular word processor app, right into at least the late 1980s, was IBM’s DisplayWrite. This was because it was a close emulation of the DisplayWriter word-processing machine, which was very popular in IBM shops (i.e. most of the mainframe computing world). That was still a big enough market to outweigh the consumer/SME PC market at the time.

This didn’t make it a good word processor; reviews regularly found it pretty horrible to use. But people brainw^H^H^H^H^Hindoctrinated into the IBM culture/ecosystem seemed to think it was great.