How Much Has *NIX Changed?

Originally published on Advent of Computing [1]

UNIX-like systems have dominated computing for decades, and with the rise of the internet and mobile devices their reach has become even wider. Most computers nowadays use more modern versions and descendants, such as Linux. But exactly how different are these modern *NIXes from the early releases of AT&T's operating system?

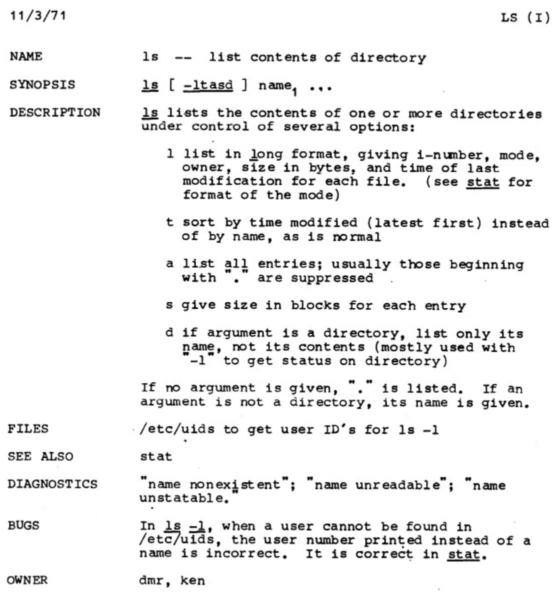
So, my question was this: how close is a modern *NIX userland to some of the earliest UNIX releases? To examine this I'm going to compare a few key points of a modern Linux system with the earliest UNIX documentation I can get my hands on. The doc I am going to be working off of (Via TUHS [2]) is from November 1971, predating v1 public release of the system.

I think the best place to start this comparison is to look at one of the highest-profile parts of the OS, that being the file system. Under the hood modern EXT file systems are completely different from the early UNIX FS. However, they are still presented in basically the same way, as a hierarchical structure of directories, files, and device files. So paths still look identical, and navigating the file system still functions almost the same. Often used commands like `ls`, `cp`, `mv`, `du`, and `df` all exist in the pre-v1 docs as well as modern distros, and largely function the same. So do `mount` and `umount`. But, there are some small differences. For instance, `cd` doesn't show up anywhere in the early docs, instead `chdir` fills its role. Also, `chmod` is somewhat different. Instead of the usual 3-digit octal codes for permissions we use today, this older version only uses 2 digits. That's due to the underlying file system using a different permission set than modern system. For the most part, all the file handling is actually pretty close to a Linux system from 2019.

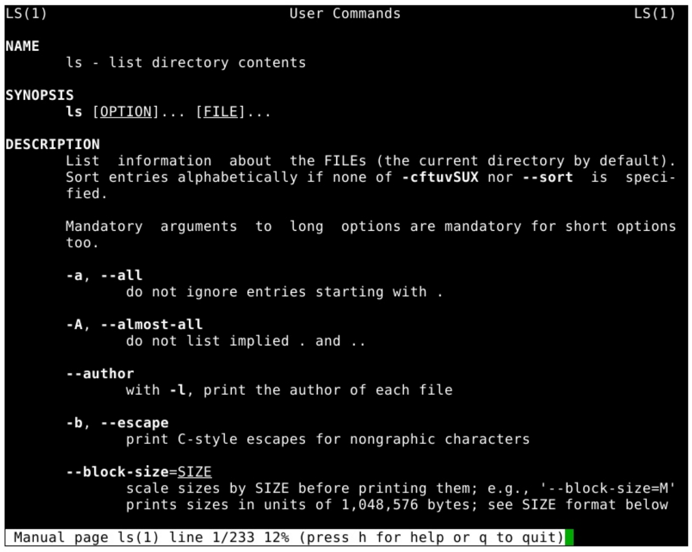
The other really high-profile part of any *NIX system in the shell. This '71 version of UNIX already had a userland shell: `sh`. A lot of Linux distros actually still default to using a much newer version of `sh`. But, the question is how much of that shell was already set in stone in the early 70s? Surprisingly, a lot. The basic layout of commands is totally unchanged: a program name followed by arguments and/or switches. Both `;` and `&` still function as command separators. File input and output redirects are still represented with $<$ and $>$ respectively. The biggest difference is there are no pipes, those won't appear on UNIX until at least 1973. Also, `sh` can already run script files in '71. Overall, I'm shocked by how similar the shell seems compared to today's version.

So superficially, this pre-release of UNIX looks remarkably close to a modern system. But what about programming utilities? This is where some big changes start to appear. First off, you won't find any C compiler here. Internally, UNIX wouldn't switch from assembly to C for another few years. To see what programming tools are still readily supplied I decided to compare the ones present in the '71 doc to the default Debian 9.9.0 install set (released April, 2019). The assembler, `as` still exists, but obviously for a different target than the PDP-11 used in '71. A linker, `ld`, is still present and accounted for today. However, the text editor, `ed`, is nowhere to be found in Debian's base install (but it is still available in aptitude). The same goes for the FORTRAN compiler `for` - nowadays `for` is used for loops instead of compiling mathematics programs. If you want to use FORTRAN you will need to install a compiler like `gfortran`. Something that I found surprising in the early UNIX docs was `bas`, a BASIC interpreter. Obviously, `bas` is not in the standard Debian install list today. Another relic is the `B` compiler described in the documentation (just as a side note, in the command index it shows a lowercase `b` but capitalizes it in the actual man page). B lived a short life, and would eventually be superseded by C. So seeing a listing for B is really a sign of the time this manual was released in.

Overall, it would appear that the core UNIX-like experience has changed little. A lot of the tech under the hood has been completely revamped many many times over, but the core way we interact with the system and most of the commands that come stock have remained the same since the 1970s. As a final note, I was blown away by just how much the very earliest man pages resemble current man pages. As an example, here is a side-by-side of `ls` from the 1971 docs on the left, and `man ls` from Debian 9.9.0 on the right.



```
11/3/71                                                    LS (I)

NAME            ls  --  list contents of directory

SYNOPSIS        ls [ -ltasd ] name1 ...

DESCRIPTION     ls lists the contents of one or more directories
                under control of several options:

                   l list in long format, giving i-number, mode,
                     owner, size in bytes, and time of last
                     modification for each file.  (see stat for
                     format of the mode)

                   t sort by time modified (latest first) instead
                     of by name, as is normal

                   a list all entries; usually those beginning
                     with "." are suppressed

                   s give size in blocks for each entry

                   d if argument is a directory, list only its
                     name, not its contents (mostly used with
                     "-l" to get status on directory)

                If no argument is given, "." is listed.  If an
                argument is not a directory, its name is given.

FILES           /etc/uids to get user ID's for ls -l

SEE ALSO        stat

DIAGNOSTICS     "name nonexistent"; "name unreadable"; "name
                unstatable."

BUGS            In ls -l, when a user cannot be found in
                /etc/uids, the user number printed instead of a
                name is incorrect.  It is correct in stat.

OWNER           dmr, ken
```



```
LS(1)                            User Commands                           LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about  the FILEs (the current directory by default).
       Sort entries alphabetically if none of -cftuvSUX nor --sort  is  speci-
       fied.

       Mandatory  arguments  to  long  options are mandatory for short options
       too.

       -a, --all
              do not ignore entries starting with .

       -A, --almost-all
              do not list implied . and ..

       --author
              with -l, print the author of each file

       -b, --escape
              print C-style escapes for nongraphic characters

       --block-size=SIZE
              scale sizes by SIZE before printing them; e.g., '--block-size=M'
              prints sizes in units of 1,048,576 bytes; see SIZE format below

 Manual page ls(1) line 1/233 12% (press h for help or q to quit)
```

[1] http://adventofcomputing.libsyn.com

[2] https://www.tuhs.org/Archive/Distributions/Research/Dennis_v1/UNIX_ProgrammersManual_Nov71.pdf