Scripting Language Report - CS 415 - BASH

Adam Stammer

(Dated: May 1, 2020)

Abstract

GNU Bash, herein referred to as Bash, was written by Brian Fox as a replacement for the Bourne Shell. It includes all Unix shell features, like wildcards, piping, variables, etc. Since it's devise in 1989 it has gained a considerable amount of popularity and can be found today in most Linux distributions, Solaris 11, Windows 10, among others.

HISTORY

In the beginning of Unix, users required some interaction with the existing systems outside of the kernel. Something that went beyond just preprogrammed files with hard coded input as this was tedious and slow. This problem was also a barrier in bringing computers outside of the research sector, as most people didn't know how to program. Shells were developed, like the Thompson shell to server not only as the middleman interaction between the user and the system, but also to facilitate development of programs that ran on top of Unix. These early shells quickly added support for things like piping, redirection, and asynchronous/sequential commands. Even wildcards were implemented by **glob**, a program made separate from the shell to keep the shell as simple and compact as possible. These early shells were unfortunately poor at automated scripting, and served as little more than the user interaction facilitators they were meant to be.

In 1977 the Bourne shell was released to fix that very problem. To accomplish this the shell added new control features like loops and variables. These greatly aided in scripting functionality, but also in interactive functionality. This shell was still proprietary though which limited access immensely.

That's where Brian Fox, working for the Free Software Foundation began development of the Bourne-Again Shell, frequently called bash. He sought to recreate the Bourne shell, with additional features, under an open GNU license so that it wasn't limited to just Bell Labs and a handful of universities. With increased wildcards available, brace expansion, startup scripts, more default commands, the ability to define functions, etc. this script launched in 1989. The shell quickly caught on, and today can be found as the default shell in almost all Linux distributions, among other places. Eventually Fox was laid off from the FSF and the project was passed on, but it remains one of the most popular shells in computing history.

ENVIRONMENT

Developed in a time before Graphical User Interfaces, bash can appear very alien to many computer users today. The shell relies entirely on typed commands, not requiring the use of a mouse at all, making it something of skill to use. Memorizing the commands can be tricky at first, especially the since the shell is usually only learned out of necessity. This is a strength and a weakness though. One can get by while only knowing a handful of commands. This however leaves a multitude of commands and features unused and unlearned by most of its users.

The shell is commonly used to peruse file systems, including searching, filtering, creating and deleting files, changing file/folder permissions, etc. It is also used to run programs, be them personalized bash scripts, or even some compiled code. In this aspect it also facilitates the data transfer between multiple programs either by redirection or piping. This is frequently a necessity in the automation of tasks. Today these scripts can even be called over http requests further supporting script based automation. This feature did eventually lead to the Shellshock bug found in 2014 if you're looking for a fun rabbit hole to jump in.

STATUS

As I've already said, Bash can be found in almost every Linux distribution today which accounts for roughly 95% of all public internet servers, and even more of all supercomputers, you can see that it is very common. Bash continues to see updates and security patches because of its popularity and continued use.

In September of 2014 it was revealed that Bash had a security vulnerability that allowed for the execution of privileged commands. The function export feature was added to bash in 1989 very shortly after the initial release of the shell. This feature allows users to share scripts between various instances of the shell. Since the new shell could not verify the permissions of the source of the script, it was possible for an unprivileged shell to pass a script containing privileged commands onto an elevated shell that could run those commands. Within hours of the announcement of this bug, there were attacks created to leverage this flaw. A security patch came with the announcement but it took quite some time for the patch to reach the majority of systems. The most common attacks involved creating botnets that would act as nodes in a Denial of Service attack, or a large scale scanning attack. It it believed the United States Department of Defense was even the target of some of these attacks.

There is considerable debate about whether or not bash should stick around. Many argue that something developed decades ago should not be considered the peak of computer shells. Direct complaints include poor error handling, syntax readability, user prompt security, among lack of data structures, among others. There are alternatives today like Fish Shell, Csh, sh, etc. but most of these alternatives bring in additional features either at the cost of of features Bash does have, or by increasing the computational and memory costs of the shell. With Bash being as popular as it is, there is also the issue of portability, as switching to any alternative shell, generally requires the porting of any preexisting scripts as well. Nothing lasts forever, but it seems that it may be some time before Bash is pushed off of its pedestal in the computing world.

---

[1] Florian Weimer (24 September 2014). "oss-sec: Re: CVE-2014-6271: remote code execution through bash". Seclists.org. Retrieved 1 November 2014.

[2] "Usage of OS for websites". W3Techs. 14 April 2020.

[3] https://ilya-sher.org/2016/05/26/fundamental-flaws-of-bash-and-its-alternatives/

[4] https://git.savannah.gnu.org/cgit/bash.git/