Sicherheitslücke auf Linux- und Unix-Systemen

Schwerenot

Christopher Dreher

Kurz nach dem Heartbleed-Desaster müssen Internetanwender erneut einen Tiefschlag hinnehmen: Die Schwachstelle „Shellshock" betrifft sämtliche GNU-Bash-Versionen von 1.14 bis 4.3 – und damit eine Vielzahl von Linux- und Unix-Systemen sowie Cygwin unter Windows.

Seit 1994 lauert die Bash-Schwachstelle im Quellcode, könnte also 20 Jahre lang unbemerkt ausgenutzt worden sein. Der Entdecker, Stéphane Chazelas, hat seine Erkenntnisse auf vorbildliche Weise zunächst ausgewählten Sicherheitsfachleuten und dem GNU-Bash-Projekt vorgelegt. Seit dem 24. September 2014 ist sie öffentlich. In der NIST-Bewertung des Schadenspotenzials rangiert Shellshock auf Maximalstufe 10 der „Common Vulnerabilities and Exposures" (CVE-2014-6271). Inzwischen haben Sicherheitsforscher weitere Schwachstellen in der Bash gefunden, die jedoch keine direkte Ausnutzung aus der Ferne oder das Ausführen von Code ermöglichen.

20 Jahre unentdeckt

Einige namhafte Internet- und IT-Konzerne sind bereits Shellshock-Angriffen zum Opfer gefallen. Der Sicherheitsforscher Jonathan Hall hat mithilfe von Google-Suchergebnissen anfällige Systeme bei Lycos, Yahoo und WinZip entdeckt, die bereits kompromittiert worden waren. Von rund 36 000 .de-Domains aus Alexas Top-1-Million-Liste erwiesen sich im Rahmen eines *iX*-Tests am Tag nach der Shellshock-Veröffentlichung 700 Server (2 %) als anfällig gegenüber einer Handvoll Tests auf bekannte CGI-Skripte. Als Testindikatoren dienten ICMP- und DNS-Anfragen an einen Server des Autors.

Mit der Bash sind prinzipiell viele Embedded-Systeme, Netz-Devices und Appliances anfällig. Die Ausnutzung der Schwachstelle auf entfernten Systemen bedarf aber eines erreichbaren Dienstes, der die Bash auf bestimmte Weise verwendet. Es gibt bereits Belege für die Ausnutzung von Shellshock in folgenden Netzwerkdiensten:

Quelle: twitter.com/pmg

Webserver: CGI-Skripte, die die Bash starten, könnten beliebigen Code ausführen.

Secure Shell: Nutzer, deren Rechte auf die Ausführung bestimmter Kommandos beschränkt sind, können diese Beschränkung umgehen.

DHCP: Bei Verbindung zu einem bösartigen DHCP-Server kann ein Angreifer einen beliebigen Code auf dem DHCP-Client ausführen.

Qmail, **Postifx**, **Pure-FTPd** und **OpenVPN** könnten je nach den eingesetzten Authentifizierungsmechanismen beliebigen Code ausführen.

Wie einfach sich die Schwachstelle auf einem Webserver ausnutzen lässt, zeigt die folgende per CGI-Skript bearbeitete HTTP-Anfrage:

```
GET /some/script.cgi HTTP/1.0
User-Agent: () { _; } >_[$($())] ⤷
        { id >/tmp/webserver_id; }
```

Sie enthält einen präparierten User-Agent-Header, den das CGI-Skript ausführt. Hier liest es lediglich die User-ID aus, unter der es läuft, und schreibt sie in eine Datei im Verzeichnis */tmp*.

Funktionsweise der Bash

Bash erlaubt es, Variablen und Funktionen zu definieren, die innerhalb der jeweiligen Instanz oder des aktuellen Skripts verwendet werden können. Darüber hinaus ist es beim Aufrufen einer neuen Bash-Instanz möglich, sowohl Variablen als auch Funktionen aus der aktuellen in die neue Bash-Instanz zu vererben. Zu diesem Zweck muss die entsprechende Variable oder Funktion zuvor per Schlüsselwort „exportiert" worden sein. Das Exportieren von Variablen und Funktionen erfolgt über Umgebungsvariablen. Da diese jedoch nur einfache Schlüssel-Wert-Paare enthalten können, müssen Funktionen beim Exportieren als einfache Zeichenkette kodiert sein. Bash verwendet für Funktionsdefinitionen spezielle Umgebungsvariablen, deren Inhalt mit der Zeichenfolge „()" beginnt. Bash überprüft daher unmittelbar nach dem Start alle Umgebungsvariablen nach solchen Funktionsdefinitionen. Für jede legt sie eine entsprechende Funktion in der aktuellen Instanz an.

Verwundbarkeitscheck

Der kritische Bug betrifft das Parsen der Funktionsdefinitionen. Hier können Angreifer zusätzlichen Code anfügen, den Bash beim Parsen der entsprechenden Umgebungsvariablen sofort und ungeprüft ausführt – selbst dann, wenn die entsprechende Funktion niemals aufgerufen wird.

Die Verwundbarkeit der Bash lässt sich auf einfache Weise durch die folgende Eingabe auf der Kommandozeile testen. Bei einer verwundbaren Shell führt die Sequenz

```
$ env x='() { :;}; echo ⤷
        verwundbar' bash -c ""
```

zur Ausgabe von „verwundbar", während ein geschütztes System nichts oder Fehlermeldungen ausgibt. Inzwischen gibt es diverse Online-Scanner, die unterschiedliche Netzwerkdienste auf die Shellshock-Schwachstellen hin untersuchen (siehe Kasten und „Alle Links").

Diverse Hersteller von Linux-Distributionen bieten bereits Patches an. Da viele der sofort bereitgestellten Patches keine Wirkung hatten, erschien ein neuer CVE-2014-7169, der sich den zwar gepatchten, aber dennoch verwundbaren Systemen widmet. Diese CVE-Nummer gilt es also ebenso im Auge zu behalten. Respekt gebührt dem derzeitigen Hauptentwickler des Bash-Projektes, Chet Ramey, der innerhalb weniger Wochen eine Vielzahl von Patches herausgebracht hat, die alle gemeldeten Schwachstellen beheben. Dank seines Einsatzes gibt es Patches für die Bash-Versionen von 2.5b bis 4.3.

Behebung und Notlösung

Das erfolgreiche Einspielen der Patches erfordert jedoch, dass das eingesetzte Betriebssystem noch Sicherheitsupdates von seinem Hersteller erhält oder dass sich der Anwender die Bash direkt aus den Quellen bauen kann, um aktuelle Patches mit einzubeziehen. Eine absolute Notlösung und eher ein Workaround für das Shellshock-Dilemma lässt sich mithilfe des sogenannten Binary-Patching erreichen. Das Starten des Einzeilers

```
perl -pe 's/\(\) {\0/(){\0\0/g' ⤷
                        << /bin/bash
```

modifiziert das Bash-Binary so, dass Angreifer die anfällige Funktionsdefinition nicht mehr nutzen können. (un)

Shellshock-Scanner

What is #shellshock?	shellshocker.net
ShellShock Tester	www.shellshocktest.com
Test your website for Shellshock	bashsmash.ccsir.org
Bash Vulnerability CVE-2014-6271 Test Tool	shellshock.brandonpotter.com

Alle Links: www.ix.de/ix1411018