Instituto Superior de Engenharia de Lisboa
MEIC, MEET, MEIM
Cibersegurança
Módulo 2: Primeiro exercício

1 Enunciado

Em 2014 foi identificada uma vulnerabilidade na aplicação *bash* de vários sistemas Linux, Unix e OS X, a qual ficou conhecida como *Shellshock*. Uma das formas de explorar esta vulnerabilidade consistia no atacante conseguir executar comandos ou programas na máquina vulnerável. Para as alíneas seguintes tenha em conta as informações de base sobre a vulnerabilidade e sua exploração presentes na Secção 2, e use a versão Ubuntu 16.04 dos laboratório SEED [1].

1. Realize as tarefas preparatórias 2.1 a 2.3 do guião SEED sobre a vulnerabilidade ShellShock [2]. Apresente um resumo das ações realizadas em cada tarefa e explique de que forma na 2.3 um atacante pode passar dados arbitrários para o contexto de execução de um *bash shell* através da *Common Gateway Interface* (CGI) do servidor Apache.

2. Lance duas máquinas virtuais (atacante e vítima) e realize o ataque pedido na tarefa 2.4, com o objectivo de obter da máquina vítima o ficheiro `/var/www/SQLInjection/safe_home.php`. Identifique no ficheiro o utilizador e palavra-chave usados pela aplicação *web* em causa para aceder à base de dados. Note que com a ferramenta `curl` é possível adicionar cabeçalhos aos pedidos HTTP (opções `-H` e `--header`).

3. Seria possível obter por esta via o conteúdo do ficheiro `/etc/shadow`, o qual contém o hash das *passwords* dos utilizadores? Justifique.

2 Introdução à vulnerabilidade Shellshock

Este anexo é uma adaptação do Capítulo 3 do livro *Computer Security - A Hands-on Approach* [3].

Uma das linhas de comandos mais utilizada em sistemas da família Unix/Linux é o *bash shell*, localizado em `/bin/bash`. Em *bash* podem ser definidas funções, como mostra o exemplo seguinte. Note que na VM dos laboratórios SEED o programa bash vulnerável tem o nome `bash_shellshock`.

```
$ foo() { echo "Inside␣function"; } # define a função 'foo'
$ declare -f foo # apresenta o código da função 'foo'
foo() {[...]}
$ foo           # executa a função 'foo'
Inside function
$ unset -f foo
$ declare -f foo
$
```

As funções definidas no processo pai podem ser exportas e assim utilizadas no processo filho, como mostra o exemplo:

```
$ foo() { echo "Inside␣function"; }
$ declare -f foo
$ export -f foo
$ bash  # executa /bin/bash num processo filho
(child)$ declare -f foo
(child)$foo
Inside function
```

A vulnerabilidade Shellshock está relacionada com a possibilidade de passar funções de um processo pai para um processo *bash* filho. Para além do método apresentado anteriormente (e que não é uma vulnerabilidade) a outra forma de passar funções é através da definição explícita de variáveis de ambiente. Quando o processo pai exporta uma variável de ambiente cujo valor é uma *string* definida pelo utilizador, com a definição de uma função, o valor da variável é interpretado e transformado numa função no processo filho, como mostra o exemplo:

```
$ foo='() { echo "Inside␣function"; }'
$ echo $foo
() { echo "Inside␣function"; }
$ declare -f foo
$ export foo
$ bash
(child)$ echo $foo
(child)$                  # no processo filho não existe variável 'foo' ...
(child)$ declare -f foo   # ... mas sim uma função de nome 'foo'
(child)$ foo ()
{
  echo "Inside␣function"
}
$ foo
Inside function
```

Quando no processo filho é executado `echo $foo` não é apresentado nada porque, durante a criação deste processo, se for encontrada uma variável de ambiente que começa por `()`, será analisado o seu valor para a transformar numa função com o mesmo nome. Durante esta análise, devido a um erro de programação, são executados os comandos depois da segunda chaveta, como mostra o exemplo seguinte:

```
$ foo='() { echo "Inside␣function"; }; echo "Hi!";'
$ echo $foo
$ () { echo "Inside␣function"; }; echo "extra";
$ export foo
$ bash
Hi!          # o comando extra é executado
$ echo $foo
$
$ delcare -f foo
foo ()
{
  echo "Inside␣function"
}
```

Note que no segundo método de passar funções, o processo pai não precisa de ser o *bash*. Qualquer processo que queira chamar o *bash shell* e passar-lhe uma função, apenas terá de a definir previamente numa variável de ambiente.

Referências

[1] `https://seedsecuritylabs.org/lab_env.html`

[2] `https://seedsecuritylabs.org/Labs_16.04/PDF/Shellshock.pdf`

[3] Wenliang Du, Computer Security: A Hands-on Approach, 1ª Edição, CreateSpace Independent Publishing Platform, 2017

ISEL, 6 de novembro de 2020.