Instituto Superior de Engenharia de Lisboa
MEIC, MEET, MEIM
Cibersegurança
Módulo 2: Exercício 1

1 Enunciado

Em 2014 foi descoberta uma vulnerabilidade no aplicativo *shell* de vários sistemas Linux, Unix e OS X, que ficou conhecido como *Shellshock*. Uma forma de explorar essa vulnerabilidade era o atacante executar comandos ou programas remotamente na máquina vulnerável. Os pontos seguintes são baseados no laboratório da vulnerabilidade *Shellshock* [3]. Tenha em conta as informações básicas sobre como explorar a vulnerabilidade (consulte a Seção 2) e use a versão Ubuntu 16.04 dos laboratórios SEED [1, 2].

1. Verifique que o ambiente do laboratório está correto seguindo o ponto 2.2 do guião SEED (*2.2 Task 2: Setting up CGI programs*).

2. Realize o ponto 2.3 e descreva resumidamente como um atacante com acesso remoto à máquina vulnerável pode passar dados arbitrários para o contexto de execução de um *bash shell* através da *Common Gateway Interface* (CGI) do servidor Apache. Explore a passagem de dados através de um *browser* e da ferramenta `curl`.

   Nota: Para editar o programa CGI tem disponível na VM vários editores de texto, nomeadamente o *gedit* e o `vi`.

3. Lance duas máquinas virtuais (atacante e vítima) e explique de que forma consegue obter da máquina vítima o seguinte 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.

4. 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* [4].

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"; } # defines function 'foo'
$ declare -f foo                    # shows the code of function 'foo'
foo() {[...]}
$ foo                               # executes function '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 $foo
(child)$                  # in the child process there is no variable 'foo' ...
(child)$                  # ... but there is a function with name 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!            # the extra command is executed
$ 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.

References

[1] `https://seedsecuritylabs.org/Labs_16.04/Documents/SEEDVM_VirtualBoxManual.pdf`

[2] `https://seedsecuritylabs.org/labsetup.html`

[3] `https://seedsecuritylabs.org/Labs_16.04/Software/Shellshock/`

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