Semana 4

Demonstração de Exploits Relativos a Controlo de Acesso

O objetivo do guião desta semana consiste em explorar *exploits* relacionados com más práticas no que diz respeito a controlo de acesso.

1. Capability Leaking

O programa seguinte tem como objetivo executar uma operação privilegiada (criar uma diretoria `ssi` no caminho `/root`), deixando cair os privilégios necessários para esta operação antes de prosseguir com o lançamento de uma `shell` para a execução posterior de comandos que não requerem privilégios de *root*.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

int main() {
    int dfd;
    char *argv[2];

    dfd = open("/root", O_RDONLY);
    if (dfd == -1) {
        perror("open /root");
        exit(1);
    }
    printf("Directory FD is %d\n", dfd);

    if (mkdir("/root/backupssi", 0700) == -1) {
        perror("mkdir /root/backupssi");
    }

    if (setuid(getuid()) == -1) {
        perror("setuid");
        exit(1);
    }

    argv[0] = "/bin/sh";
    argv[1] = NULL;
    execve(argv[0], argv, NULL);

    perror("execve");
    return 0;
}
```

Setup

1. Compile o programa e gere o executável: `gcc -o backupssi backupssi.c`
2. Defina *root* como o *owner* do programa: `sudo chown root:root backupssi`
3. Defina as permissões seguintes: `chmod 4755 backupssi`
    - 4 -> `setuid`
    - 7 -> owner: rwx
    - 5 -> group: r-x
    - 5 -> others: r-x

Exercício

1. Execute o programa com um utilizador normal (que não seja o *root*).
2. Analize o código (e respetivo output do programa) e identifique a vulnerabilidade relacionada com *capability leaking*.
3. Implemente um programa que demonstre como esta vulnerabilidade pode ser explorada por um utilizador normal (sem privilégios root) para aceder a diretorias protegidas (neste caso, `/root`).

4. Implemente uma correção para o excerto de código apresentado que mitigue a vulnerabilidade e explique em que medida o problema é resolvido.

2. Elevação de Privilégio

Atente no programa seguinte:

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>

int main() {
    int fd = open("/etc/passwd", O_WRONLY | O_APPEND);
    if (fd < 0) {
        perror("open /etc/passwd");
        exit(1);
    }

    printf("Passwd FD leaked: %d\n", fd);
    setuid(getuid());
    execl("/bin/sh", "sh", NULL);
}
```

Setup

1. Compile o programa e gere o executável: `gcc -o passwdleak passwdleak.c`
2. Defina *root* como o *owner* do programa: `sudo chown root:root passwdleak`
3. Defina as permissões seguintes: `chmod 4755 passwdleak`

Exercício

1. Execute o programa com um utilizador normal (que não seja o *root*).
2. Analize o código (e respetivo output do programa) e identifique a vulnerabilidade existente.
3. Identifique um possível exploit para a vulnerabilidade em questão. Pode recorrer exclusivamente à `bash` para o efeito.
    - Sugestão: tente adicionar a entrada `ssihacker::0:0::/root:/bin/sh\n` ao ficheiro `/etc/passwd`.
4. Quais as implicações práticas do exploit do ponto anterior? Quais as ações que o exploit possibilita?
    - Sugestão: tente efetuar login como `ssihacker`.
5. Implemente uma correção para o excerto de código apresentado que mitigue a vulnerabilidade e explique em que medida o problema é resolvido.