Pontifícia Universidade Católica do Rio Grande do Sul
Faculdade de Engenharia
Redes de Comunicação II

TCP Session Hijacking

Objetivo

O objetivo geral do trabalho é desenvolver uma aplicação usando *raw sockets* para sequestrar conexões TCP (TCP Session Hijacking):

- compreender de maneira prática o mecanismo de comunicação por *sockets*;
- entender o funcionamento dos protocolos da camada de rede e transporte e seus problemas de segurança em redes locais.

Descrição

O sequestro de conexões TCP implica em interceptar uma conexão já estabelecida entre duas partes comunicantes e, então, se passar por uma delas enviados pacotes para a outra parte. A ideia é dessincronizar a comunicação existente entre um cliente e um servidor (por exemplo, através do envio de uma mensagem de encerramento de conexão para o cliente) e assumir a identidade do cliente nessa conexão.

Uma conexão TCP estabelecida é unicamente identificada por uma quádrupla contendo endereço IP de origem, porta de origem, endereço IP de destino e porta de destino. Adicionalmente, cada pacote possui um número de sequencia (SEQ) que identifica o primeiro byte enviado no seguimento e, possivelmente, um número de reconhecimento (ACK) que informa à outra parte que os bytes anteriores àquele número foram recebidos com sucesso. Uma das maiores dificuldades em sequestrar uma conexão TCP é descobrir/adivinhar os números SEQ/ACK dos pacotes, tendo em vista que esses são valores de 32 bits atribuídos pelo sistema operacional (ver RFC 793 para mais detalhes).

Quando empregado em uma rede local, o sequestro de conexões pode ser combinado com alguma técnica conhecida de man-in-the-middle (MITM). Neste caso, pode-se utilizar um programa *sniffer* (analisador de pacotes) para monitorar o tráfego de rede entre o cliente e o servidor e descobrir os próximos número SEQ/ACK de uma dada conexão. Nesse trabalho, usaremos um ataque do tipo ARP *Spoofing* (ver ANEXO II) para interceptar o tráfego de rede do host alvo e descobrir as informações necessárias para realizar o sequestro de conexão. O funcionamento do ataque de sequestro de conexão está descrito no ANEXO I.

Todas as fazes do desenvolvimento do trabalho devem ser documentadas na forma de um relatório. Este relatório deve primeiramente descrever o funcionamento do protocolo TCP e descrever como foi explorado o problema de segurança usando digramas, trechos de códigos e/ou capturas de tela (sugestão: utilize capturas de telas do Wireshark para facilitar a explicação). Esse relatório deverá ser entregue juntamente com o código fonte utilizado.

O trabalho deve ser implementado na linguagem C. Exemplos utilizando *sockets* TCP e *sockets raw* foram disponibilizados no Moodle.

Resultados e Entrega

Grupos: Individual ou em dupla.
Entrega: Upload no Moodle de um arquivo zipado com o nome do(s) aluno(s), contendo:

1. Relatório descrevendo a implementação
2. Código da implementação

Prazo final para entrega: 20/06/2017 até as 19:00 (antes do início da aula)
Data da apresentação: 20/06/2017

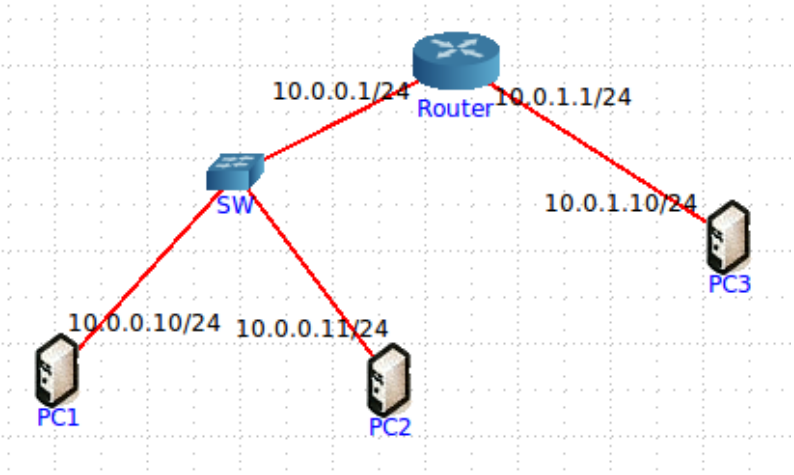
ANEXO I - Ataque TCP Session Hijaking

Uma vez que um ataque do tipo *man-in-the-middle* na rede local foi realizado com sucesso, toda comunicação realizada entre a vítima e a Internet passará pelo *host* atacante. Desta forma, é possível implementar um programa *sniffer* utilizando *sockets raw* que analisa os pacotes TCP trocados entre um cliente e um servidor a fim de descobrir as informações da conexão e realiza o sequestro.

O funcionamento básico do ataque é descrito nos passos a seguir:

- Passo 1: monitorar a comunicação entre o cliente e o servidor para descobrir as informações da conexão (as informações mais importantes são: porta de origem do cliente, número de sequência e número de reconhecimento);
- Passo 2: finalizar a conexão no lado do cliente, isto é, enviar um pacote de *reset* (RST) para o cliente;
- Passo 3: enviar dados para o servidor fingindo ser o cliente (enviar pelo menos um pacote para o servidor).

Foi disponibilizado no Moodle um programa cliente/servidor que utiliza TCP para ser usado como alvo desse ataque (simple-tcp.tar.gz). O programa pode ser executado da seguinte forma. Para facilitar o desenvolvimento e validação da técnica, sugere-se a utilização do emulador de redes CORE Emulator com a seguinte topologia.



Executar programa servidor no PC3 especificando um número de porta:

```
./simple-server 12124
```

Executar o programa cliente no PC1 especificando endereço IP e porta do servidor:

```
./simple-client 10.0.1.10 12124
```

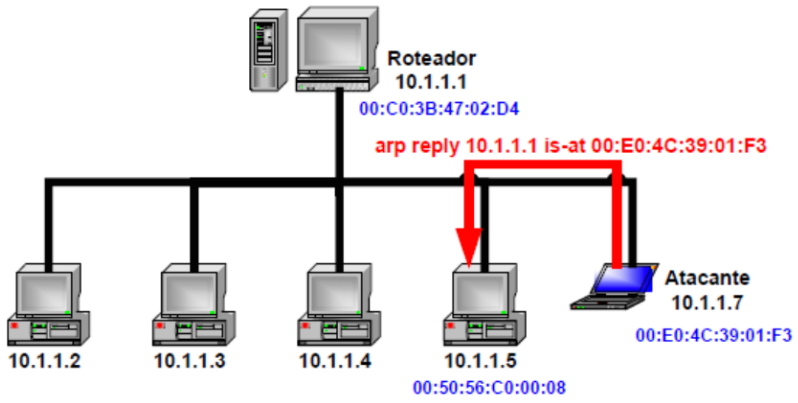
Monitorar a troca de pacotes entre os dois programas para entender o funcionamento.

O programa criado para efetuar o ataque deve receber como parâmetro o endereço IP do cliente, endereço IP do servidor e porta do servidor. As demais informações necessárias para o ataque podem ser obtidas através da monitoração de pacotes. Esse programa deve ser executado no PC2.

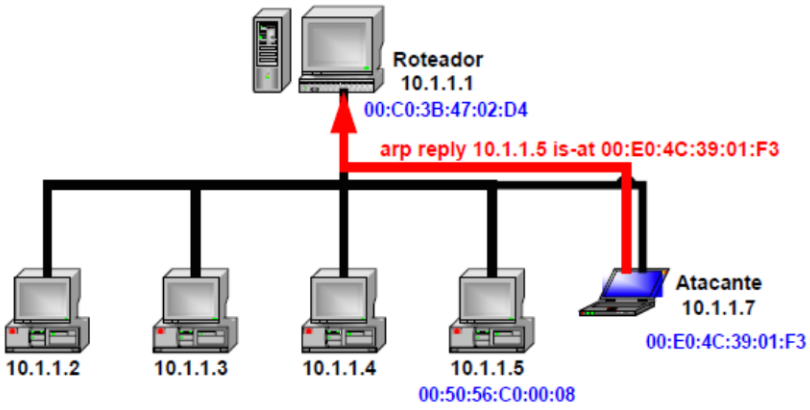
ANEXO II - Ataque ARP Spoofing

O ataque do tipo ARP *spoofing* consiste em enviar pacotes ARP *reply* não solicitados para os computadores alvo para modificar suas tabelas ARP locais. Utilizando o programa Wireshark é possível acompanhar o funcionamento do ataque em cada fase. Veja o exemplo abaixo.

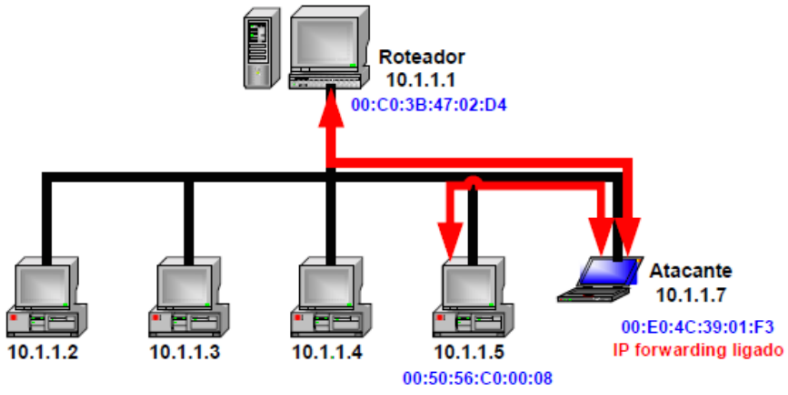
Passo 1:



Passo 2:



Resultado:



Alguns sistemas operacionais como Windows 7 (ou superiores) irão ignorar mensagens ARP *reply* não solicitadas e realizar uma nova consulta ARP para confirmar o

endereço físico de um computador. Neste caso, um método alternativo é enviar uma mensagem ARP *request* para o computador alvo usando endereços de IP/MAC de origem modificados.

Para que o sistema operacional não corrija a tabela ARP com as informações verdadeiras envias pelos computadores da rede, é necessário manter o envio constante de mensagens ARP modificadas (por exemplo, a cada 1 segundo).

Execução do ataque

Foi disponibilizado no Moodle um programa que implemente este ataque (arpspoof.tar.gz). O programa pode ser executado da seguinte forma na máquina atacante:

```
./arpspoof –v –g 10.0.0.1 10.0.0.10 –r 2
```

Obs: nesse caso o endereço IP do gateway é 10.0.0.1 e da máquina vítima é 10.0.0.10. O programa envia um ARP *reply* não solicitado a cada 2 segundos.

Verificação do funcionamento

Para verificar se o ataque funcionou, visualize as tabelas ARP de cada computador antes e depois do ataque e verifique se as mesmas foram alteradas com sucesso. O comando para verificar a tabela ARP no Linux é:

```
arp –n
```

Adicionalmente, é possível utilizar o programa Wireshark para acompanhar o envio/recebimento de mensagens ARP em cada computador.

Encaminhamento de pacotes

Por padrão, o Linux descarta pacotes que são destinados a outros computadores. Desta forma, para implementar um ataque do tipo *man-in-the-middle*, é necessário habilitar a funcionalidade de encaminhamento de pacotes do *kernel* do Linux (IP Forwarding). Isso fará com que o tráfego entre o computador alvo e o roteador não seja interrompido durante o ataque.

Para habilitar a funcionalidade de IP Forwarding, execute o seguinte comando no Linux:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Dependendo do ambiente de teste utilizado, pode ser necessário desabilitar o envio de mensagens ICMP *redirect* na máquina atacante. Este é o caso para testes realiados usando o CORE Emulator. Para desabilitar o envio de ICMP *redirect* no Linux, execute o seguinte comando:

```
for i in `ls –1 /proc/sys/net/ipv4/conf/*/send_redirects`; do
echo 0 > $i; done
```