Pontifícia Universidade Católica do Rio Grande do Sul
Escola Politécnica
Laboratório de Redes

Trabalho Final

Objetivo

Desenvolver um ataque man-in-the-middle para capturar o histórico de navegação web de um computador alvo remoto. O projeto será conduzido em três etapas principais:

- Descoberta de hosts: Desenvolver uma aplicação para identificar hosts ativos na rede, realizando uma varredura inicial semelhante a um *ping scan* para mapear os dispositivos conectados.
- Execução do ataque: Após identificar o host alvo (um dos hosts ativos na rede), realizar um ataque de ARP Spoofing com man-in-the-middle utilizando a ferramenta arpspoof, inserindo-se no fluxo de comunicação entre o alvo e o roteador.
- Monitoramento de tráfego: Criar uma aplicação para monitorar o tráfego de navegação web do host alvo, capturando pacotes HTTP e DNS para rastrear o histórico de navegação.

Descrição

Usando um programa sniffer como o Wireshark, é possível monitorar todo o tráfego de rede de um host e analisar seu conteúdo. Por exemplo, ao inspecionar pacotes dos protocolos DNS e HTTP, é possível reconstruir o histórico de navegação web de um dispositivo. Contudo, essa prática geralmente exige acesso físico ao host. Para realizar esse monitoramento remotamente em outros dispositivos de uma rede local, pode-se recorrer a um ataque man-in-the-middle, explorando vulnerabilidades comuns em redes locais.

Neste trabalho, utilizaremos um ataque de ARP Spoofing para interceptar o tráfego de rede e monitorar o histórico de navegação web de cada host alvo. A implementação será dividida em três etapas principais:

- Desenvolvimento da aplicação de varredura: Criação de uma ferramenta para identificar hosts ativos na rede local (ANEXO I).
- Execução de ARP Spoofing com man-in-the-middle: Configuração de um ataque que insere o atacante entre o host alvo e o roteador (ANEXO II).
- Desenvolvimento de aplicação de análise de tráfego: Criação de uma ferramenta para capturar e analisar o histórico de navegação web dos hosts atacados (ANEXO III).

Observações Gerais

As **aplicações implementadas** nas **etapas 1 e 3 deverão** utilizar a **API socket RAW**, devendo ser declaradas nos programas as **estruturas de dados** necessárias para a manipulação de pacotes **ICMP, IP e Ethernet** (arquivo header contendo as definições de tipo). Sua implementação deve ser modular, ou seja, deve ser organizada de maneira a ser estendida, prevendo a possibilidade de adicionar outros parâmetros, protocolos e modos de operação.

Os trabalhos não podem utilizar o scapy em sua implementação.

Resultado e Entrega

Grupo: grupos de até 3 alunos.

Data de entrega: 25/11 no Moodle

Apresentação: 25/11 e 02/12

Observações Gerais: É importante que **todos os integrantes dos grupos estejam aptos a apresentarem o trabalho** a partir do início da aula. Para a entrega, é esperado que apenas um dos integrantes envie pelo Moodle, até a data e hora especificadas, um **arquivo .zip com os nomes dos integrantes**, contendo **o código fonte completo do projeto** e **um relatório descrevendo a implementação e os testes realizados**. É importante que no **relatório** seja apresentada uma **análise, incluindo o uso de screenshots da ferramenta Wireshark**.

IMPORTANTE: Não serão aceitos trabalhos entregues fora do prazo. Trabalhos que não compilam ou que não executam não serão avaliados. Todos os trabalhos serão analisados e comparados. Caso seja identificada cópia de trabalhos, todos os trabalhos envolvidos receberão nota ZERO.

ANEXO I

Varredura de hosts ativos

A aplicação a ser desenvolvida deve realizar uma varredura na rede com a finalidade de descobrir os hosts ativos e o seu tempo de resposta. Para isso, essa aplicação deverá receber como argumento de linha de comando uma rede e máscara, como por exemplo: 192.168.1.128/25 (126 hosts) e o tempo limite de espera por uma resposta em milissegundos. Esse tipo de varredura é um modo de operação simples da ferramenta nmap[1].

Durante a varredura, a ferramenta deverá enviar mensagens do tipo *ICMP request* e aguardar pelo tempo limite uma resposta do tipo *ICMP reply*. O envio deve ser seguido de uma espera, definido pelo tempo limite especificado por linha de comando. Caso um host não responda dentro desse tempo, a aplicação deve considerar esse como inoperante e continuar com a busca. Caso um host responda dentro do tempo, a ferramenta deve anotar em uma lista o IP do host e o tempo de resposta. Ao final, deverá ser apresentado o número de máquinas ativas, o número total de máquinas na rede especificada (descartando-se o IP da rede e endereço de *broadcast*) e o tempo total de varredura, além da lista de hosts ativos e o seu tempo de resposta.

[1] https://nmap.org/

ANEXO II

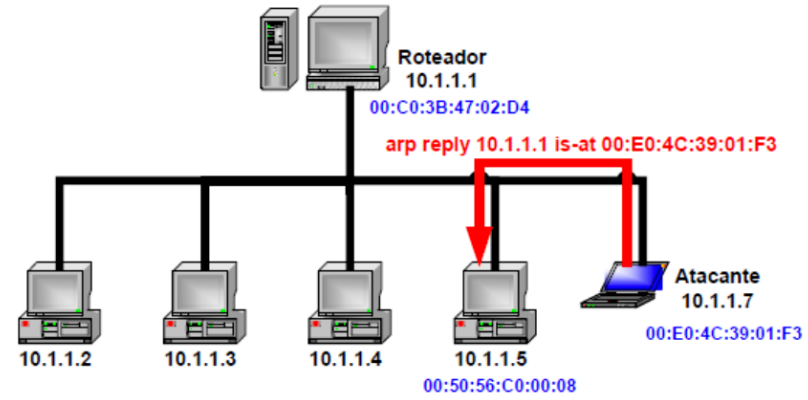
Ataque ARP Spoofing

Ataques do tipo ARP spoofing consistem basicamente em enviar ARP reply não solicitados ao computador alvo e ao roteador para modificar suas tabelas ARP locais, inserindo-se no fluxo de comunicação.

Utilize a ferramenta **arpspoof** do pacote dsniff para implementar o ataque. Veja o exemplo abaixo.

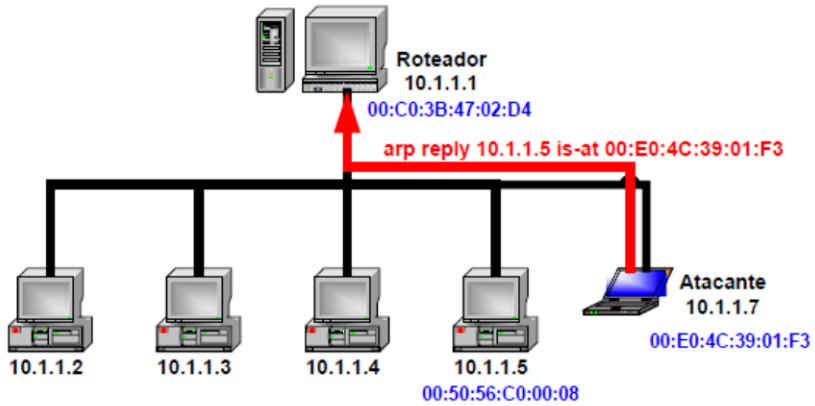
Passo 1: Enviar ARP reply não solicitado ao computador alvo

sudo arpspoof -i eth0 -t 10.1.1.5 10.1.1.1

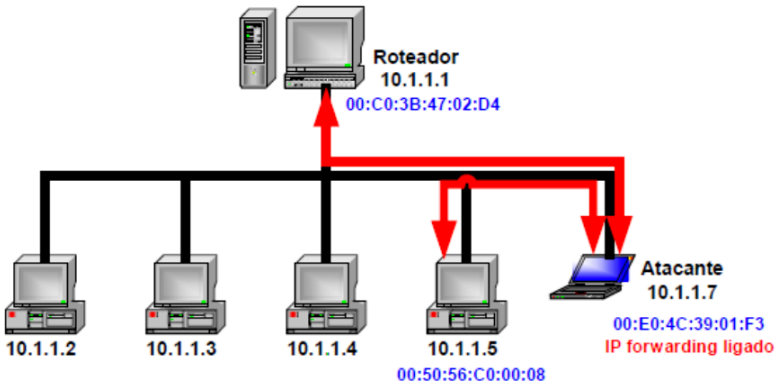


Passo 2:Enviar ARP reply não solicitado ao roteador

sudo arpspoof -i eth0 -t 10.1.1.1 10.1.1.5



Resultado: Todo tráfego entre o alvo e a Internet passará pelo atacante



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 hosts. 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 host 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
```

ANEXO III

Monitoração do Histórico de Navegação Web da Vítima

Uma vez que um ataque do tipo man-in-the-middle foi realizado com sucesso, toda a comunicação realizada entre a vítima e a Internet passará pelo host atacante. Desta forma, é possível implementar um programa sniffer utilizando **socket raw** que analisa as requisições DNS e HTTP e gera um arquivo contendo o histórico de navegação Web dos hosts alvo. É importante observar que ao utilizar **socket raw**, o sniffer deve ler os **campos de todos os cabeçalhos dos protocolos** necessários para identificar os protocolos de aplicação DNS e HTTP, isto é, **Ethernet, IPv4, TCP e UDP**.

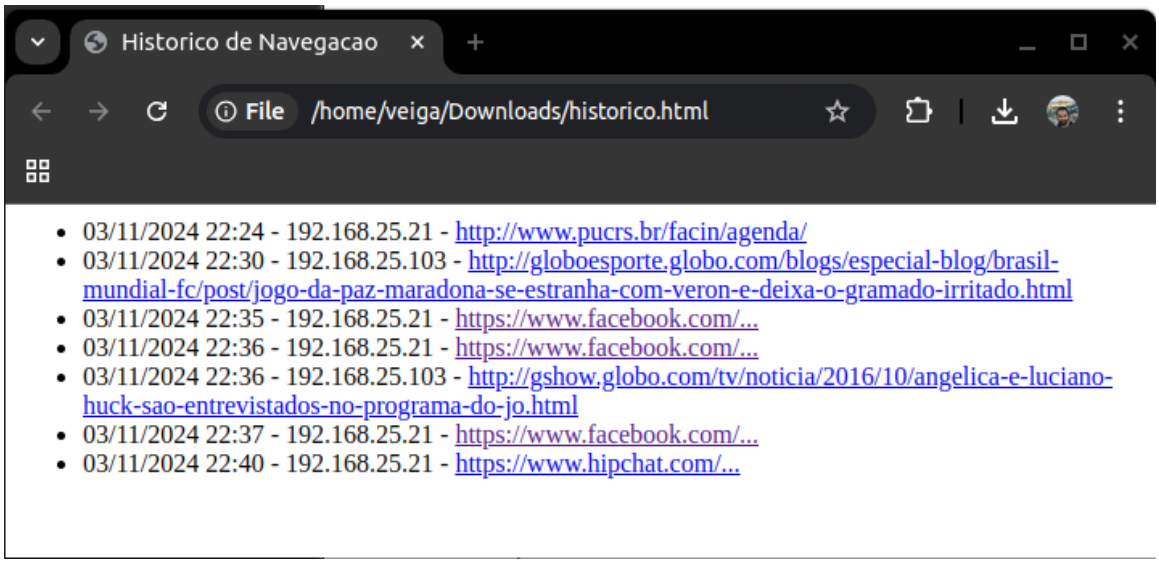
O arquivo de saída deve estar no formato HTML e cada entrada do histórico deve conter as seguintes informações:

- data e hora do acesso;
- endereço IP do host;
- nome do host;
- URL completa do endereço Web acessado (caso o endereço utilize HTTPS, fornecer apenas o nome do domínio).

Um exemplo de arquivo de saída é fornecido a seguir.

```
<html>
<header>
<title>Historico de Navegacao</title>
</header>
<body>
<ul>
<li>03/11/2024 22:24 - 192.168.25.21 - <a
href="http://www.pucrs.br/facin/agenda/">http://www.pucrs.br/facin
/agenda/</a></li>
<li>03/11/2024 22:30 - 192.168.25.103 - <a
href="http://globoesporte.globo.com/blogs/especial-blog/brasil-
mundial-fc/post/jogo-da-paz-maradona-se-estranha-com-veron-e-
deixa-o-gramado-
irritado.html">http://globoesporte.globo.com/blogs/especial-
blog/brasil-mundial-fc/post/jogo-da-paz-maradona-se-estranha-com-
veron-e-deixa-o-gramado-irritado.html</a></li>
<li>03/11/2024 22:35 - 192.168.25.21 - <a
href="https://www.facebook.com/">https://www.facebook.com/...</a><
/li>
<li>03/11/2024 22:36 - 192.168.25.21 - <a
href="https://www.facebook.com/">https://www.facebook.com/...</a><
/li>
<li>03/11/2024 22:36 - 192.168.25.103 - <a
href="http://gshow.globo.com/tv/noticia/2024/10/angelica-e-
luciano-huck-sao-entrevistados-no-programa-do-
jo.html">http://gshow.globo.com/tv/noticia/2016/10/angelica-e-
luciano-huck-sao-entrevistados-no-programa-do-jo.html</a></li>
<li>03/11/2024 22:37 - 192.168.25.21 - <a
href="https://www.facebook.com/">https://www.facebook.com/...</a><
/li>
<li>03/11/2024 22:40 - 192.168.25.21 - <a
href="https://www.hipchat.com/">https://www.hipchat.com/...</a></l
i>
</ul>
</body>
</html>
```

A escolha pelo formato HTML foi realizada para permitir a visualização do histórico de navegação de forma similar a fornecida pelos navegadores Web (ex: Google Chrome). Um exemplo de visualização do arquivo criado é apresentado a seguir.



Observação: Para obter todas as informações necessárias para gerar um histórico de monitoração como o demonstrado acima, será necessário combinar informações extraídas de pacotes DNS e HTTP. Note que apesar de simples, essa tarefa não é trivial! Recomenda-se um estudo detalhado desses protocolos, através de suas RFCs, e a realização de experimentos em laboratório utilizando Wireshark para monitorar o conteúdo dos pacotes desses protocolos. Também será necessário filtrar as requisições para não poluir o arquivo gerado. Por exemplo, o acesso a uma página Web pode gerar centenas de requisições HTTP adicionais para obter todos os objetos ligados à página, tais como arquivos de imagens, folhas de estilo (CSS), java scripts, etc.