Relatório do Desenvolvimento

Emanoel Aurélio Vianna Fabiano, Gabriell Alves de Araujo

Faculdade de Informática – Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Av. Ipiranga, 6681 - Partenon, RS, 90619-900 – Porto Alegre – RS – Brazil

{emanoel.fabiano, gabriell.araujo}@acad.pucrs.br

1. Introdução

Dentro do escopo da disciplina de Redes de Computadores, o presente relatório apresenta o desenvolvimento do trabalho final, que tem como objetivo mostrar e explorar alguns problemas de segurança que são encontrados no protocolo ARP. Com base na fundamentação previamente estabelecida, é apresentado uma solução para o problema proposto.

2. Compreendendo o protocolo ARP

O ARP é um protocolo utilizado para realizar a resolução de endereços da camada de rede e camada enlace, seu objetivo principal é realizar traduções de endereços IPs(endereços lógicos) em endereços MAC(endereços físicos). Para realizar esse mapeamento é utilizado uma tabela chamada ARP Table, onde para cada novo host, ou seja computador ou dispositivo em rede é realizado um mapeamento interrogando-o com o objetivo de saber o seu endereço e então posteriormente adicionando na tabela a correspondência. Está primeira parte pode ser verificada se utilizando de um exemplo simples, como abaixo onde temos dois computadores em rede, estando ligados a um switch.

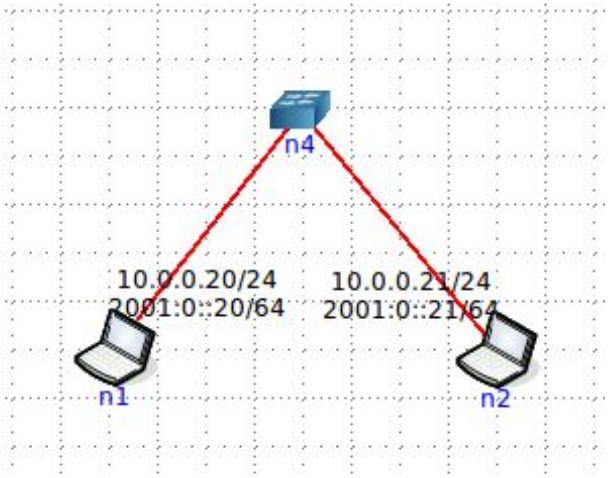

Figura 1. Exemplo de uma rede ligando dois computadores em uma rede.

Quando realizamos o comando *arp -n* que realiza a verificação na máquina para saber os endereços guardados em cache na tabela, inicialmente não irá existir nenhum endereço conhecido. Quando uma máquina deseja realizar uma comunicação com uma outra, por exemplo realizando um *ping* passando como argumento o endereço IP, ela então consulta sua tabela de correspondência, e caso o endereço solicitado não existir, o protocolo ARP então

realiza um pedido na rede, desta forma, o conjunto de máquinas ligadas em rede vai realizar a comparação de endereços lógicos ao seu, caso uma delas se identificar como sendo aquele o seu endereço, ela então irá responder a solicitação, nisso então a máquina que realizou essa solicitação acaba por conhecer o endereço e armazenando em sua tabela, possibilitado assim continuar a comunicação.

Seguindo o nosso exemplo de rede, podemos observar que a máquina n2 está com a tabela arp ainda sem nenhum endereço. Logo em seguida então realizamos um ping para o IP da máquina vizinha. Se verificarmos novamente sua tabela, observamos que agora existe um endereço registrado.

Figura 2. Exemplo de uma verificação da tabela ARP na máquina n2.

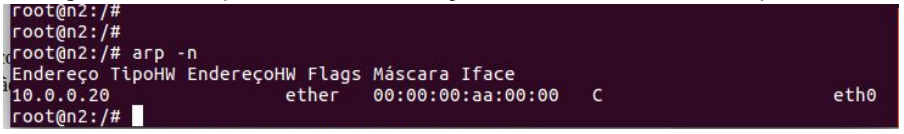

Figura 3. Tabela ARP após realizar um ping para a máquina n1.

Para entendermos melhor como a tabela é preenchida, vamos analisar o que o protocolo ARP realiza para conseguir mapear os endereços na rede. Quando a máquina n2 tentou realizar um ping sobre a máquina n1 e verificou que não reconhece o seu endereço, o protocolo ARP então envia um broadcast(endereço FF:FF:FF:FF:FF:FF) a "questionar" (ARP Request) a quem pertence o endereço IP (neste caso o endereço IP da máquina n1). Podemos observar esse funcionamento na imagem abaixo:

Figura 4. Cabeçalho Ethernet informando que a máquina n1 realizou um broadcast.

A máquina então n1 então irá responder a solicitação (ARP Response) mando os eu endereço físico, como podemos observar na imagem abaixo do pacote ethernet.

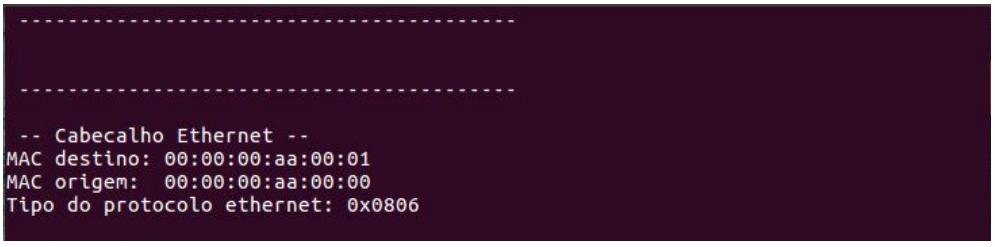

Figura 5. Cabeçalho Ethernet realizando um response para a máquina n1.

O protocolo ARP é encapsulado no padrão Ethernet e possui um padrão de pacote que pode ser observado abaixo, nele estão informações como IP de origem e destino, tipo de

operação, endereço físico de origem e destino entre outros dados.

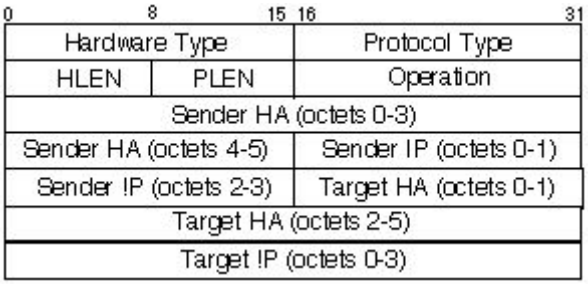

Figura 6.Formato do protocolo ARP.

Para a realização deste experimento foi utilizado o utilitário arpsniffer, desenvolvido na primeira etapa do trabalho, nele é possível observar pacotes ARP que estão sendo enviados na rede local. O utilitário informa todas os campos informações do pacote em específico, como pode ser observado abaixo:

```
 ----------------------------------------

 -- Cabecalho Ethernet --
MAC destino: 00:00:00:aa:00:00
MAC origem:  00:00:00:aa:00:02
Tipo do protocolo ethernet: 0x0806

 -- Pacote ARP --
Tipo de hardware: 0001
Tipo de protocolo: 0800
Comprimento do endereço arp: 06
Comprimento do protocolo arp: 04
Tipo da operação: 0002
Endereço fisico de origem:  00:00:00:aa:00:02
Endereço logico origem:  10.0.0.22
Endereço fisico de destino: 00:00:00:aa:00:00
Endereço logico destino:  10.0.0.20
Dados do pacote: 00 00 00 aa 00 00 0a 00 00 14 00 00 00 00 00 00 10 00

 ----------------------------------------
```

Figura 7.Exibição de um pacote arp trafegando em rede.

3. Descobrindo os endereços lógicos com arpdiscover

O segundo utilitário realiza uma busca na rede com o objetivo de descobrir todo os endereços de IP na rede local, nele utilizamos threads com o objetivo de realizar uma consulta sobre os endereços de uma forma paralela, obtendo assim uma melhor performance na busca. Assim como no arpsniffer trabalhamos com a estrutura de declaração do pacote ARP, mas agora realizando o replay na rede local, ou seja na declaração *arp_options* declaramos o valor de 1.

Para a realização dos teste sobre o código utilizamos a declaração de uma rede como abaixo, onde temos alguns computadores estando conectados em um switch.

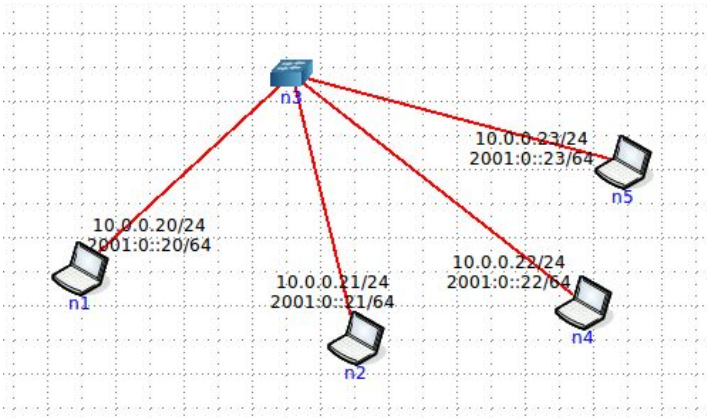

Figura 8. Configuração da rede para a realização dos teste do projeto.

Realizando a execução do código então no computador n1 temos como saída os endereços de cada host em rede, como pode ser observado na imagem abaixo:

```
<lvimento/repositorio/trabalho-redes# ./arpdiscover eth0

--Eviando o pacote--

--Recebendo os pacotes--
MAC: 00:00:00:aa:00:01
IP: 10.0.0.21
MAC: 00:00:00:aa:00:02
IP: 10.0.0.22
MAC: 00:00:00:aa:00:03
IP: 10.0.0.23
```

Figura 9. Resultado da execução na busca de endereços.

Em resumo o funcionamento segue a premissa já conhecida utilizada na troca de informações, primeiro monta uma mensagem ARP realiza um broadcast variando os endereços com o objetivo de encontrar todos os host na rede e então recebe as mensagens de resposta (reply) de cada host em rede, capturando os seus endereços e os imprimindo em tela.

4. Compilando o código e os executando

Para compilar o código existe na raiz um arquivo *Makefile* com o objetivo de facilitar a compilação, executando então o comando *make* na pasta do projeto os arquivos executáveis iram ser criados, para então executar então os utilitários devemos passar a interface de rede.

```
emanoel@pc:~/desenvolvimento/repositorio/trabalho-redes$ make
gcc -o arpsniffer arpsniffer.c -Wall
emanoel@pc:~/desenvolvimento/repositorio/trabalho-redes$ ./arpsniffer eth0
```

Figura 10. Compilando e executando o código.

Ainda assim é possível realizar a compilação individualmente, se utilizando do exemplo abaixo:

```
gcc -o arpsniffer arpsniffer.c -Wall
./arpsniffer <interface_de_rede>
```

5. Referências

Gustavo Pantuza. (2017) “O PROTOCOLO ARP - ADDRESS RESOLUTION PROTOCOL”, https://blog.pantuza.com/artigos/o-protocolo-arp-address-resolution-protocol