Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Ataque Distribuído de Negação de Serviço por Reflexão Amplificada usando o protocolo Simple Service Discovery Protocol

Eduardo S. Duarte

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Orientador
Prof. Dr. João José Costa Gondim

Brasília
2018

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Ataque Distribuído de Negação de Serviço por Reflexão Amplificada usando o protocolo Simple Service Discovery Protocol

Eduardo S. Duarte

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Prof. Dr. João José Costa Gondim (Orientador)
CIC/UnB

Prof. Dr. Ricardo Pezzuol Jacobi
Universidade de Brasília

Prof. Dr. Eduardo Adilio Pelinson Alchieri
Universidade de Brasília

Prof. Dr. José Edil Guimarães de Medeiros
Coordenador do Curso de Engenharia da Computação

Brasília, 10 de dezembro de 2018

Agradecimentos

Agradeço aos meus pais e a minha família que sempre me apoiaram nas minhas escolhas durante todos esses anos da minha vida. Sou grato a todos os momentos em que a minha pouca experiência foi substituita pelos sábios conselhos que foram ditos por eles. Obrigado pelo suporte, pois sem vocês, minha vida seria completamente mais difícil e isso deve ser observado e nunca esquecido.

Agradeço aos meus amigos de curso que dividiram experiências boas e ruins. Obrigado pelo apoio e suporte, eu cheguei nessa posição que estou hoje também graças ao esforço de vocês.

Agradeço os professores que me ajudaram durante a minha formação. Foram vocês que tornaram tudo isso possível. Agradeço e ao mesmo tempo que vocês continuem a realizar esse trabalho, pois é por meio desse trabalho que pessoas ignorantes se tornam humildes e os humildes se tornam sábios.

Por fim, agradeço a Universidade de Brasília por me proporcionar toda a experiência e momentos que obtive durantes esses anos da graduação.

Resumo

Ataques de Negação de Serviços são conhecidos desde a década de 90, sendo que sempre foram uma das ferramentas mais comuns para se realizar ataques. Dessa forma, a comunidade de segurança de dados conhecia muito bem esse tipo de ataque, porém, com o desenvolvimento de novos protocolos e tecnologias, esses ataques se diversificaram, tornando-os mais complexos, dificultando a sua mitigação e aumentando o número de ações de defesa para combatê-los.

Um desses protocolos, objeto desse estudo, é o *Simple Service Discovery Protocol (SSDP)*. Esse protocolo é responsável por descobrir novos dispositivos em uma rede, sendo base do protocolo *Universal Plug and Play (UPnP)* e, ainda, utilizado por outros protocolos.

Considerando que cada novo dispositivo em uma rede de comunicação pode se tornar uma nova ferramenta de amplificação e replicação de ataques de negação de serviço, o crescimento mundial das redes de comunicações residenciais com a adição de novos dispositivos como celulares, tablets, TVs, impressoras, câmeras, estações multimídias e até mesmo dispositivos de automação residencial, os quais utilizam o protocolo SSDP para se comunicarem, aumentaram consideravelmente a capacidade deste tipo de ataque.

Esse estudo possui como objetivo realizar um ataque DDoS explorando o protocolo SSDP. Isso será realizado por meio de uma ferramenta, desenvolvida em linguagem de programação C, conectada a uma rede local que utiliza o protocolo SSDP para comunicação entre os dispositivos. Nessa rede, um dispositivo realiza um ataque que é posteriormente analisado, considerando variáveis como amplificação, saturação e efetividade do ataque.

Palavras-chave: SSDP, Ataque de negação de serviço, IP spoofing, Segurança de rede

Abstract

Denial of Service attacks have been known since the 1990s, and have always been one of the most common tools for attacking. The data security community knew this kind of attack very well, but with the development of new protocols and technologies, these attacks diversified, making the DoS more complex, more difficult to mitigate and making the defense be harder with time.

One of these new protocols, which is object of this study, is the Simple Service Discovery Protocol (SSDP). This protocol is responsible for discovering new devices in a network, and is used by the Universal Plug and Play (UPnP) protocol, which is a another new protocol used by a larger number of devices.

Considering that every new device in a communication network can become a new tool for amplification and replication of denial of service attacks, the global growth of residential communications networks with the addition of new devices such as mobile phones, tablets, TVs, printers, cameras, multimedia stations, and even home automation devices, which use the SSDP protocol to communicate, have greatly increased the capacity of this type of attack.

This study aims to perform a DDoS attack by exploiting the SSDP protocol. This will be done through a tool, developed in programming language C, connected to a local network that uses the SSDP protocol for communication between the devices. In this network, a device performs an attack that is later analyzed, considering variables such as amplification, saturation and effectiveness.

Keywords: SSDP, DDoS, IP spoofing, network security

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

DDoS *Distributed Denial of Service.*

DNS *Domain Name System.*

DoS *Denial of Service.*

HTTP *Hypertext Transfer Protocol.*

IoT *Internet of Things.*

IP *Internet Protocol.*

NTP *Network Time Protocol.*

OSI *Open System Interconnection.*

RFC *Request for Comments.*

SNMP *Simple Network Management Protocol.*

SSDP *Simple Service Discovery Protocol.*

TCP *Transmission Control Protocol.*

UDP *User Datagram Protocol.*

USN *Unique Service Name.*

UUID *Universal Unique Identifier.*

Capítulo 1

Introdução

Em busca de novos clientes, as empresas investem fortemente no desenvolvimento de tecnologias de fácil manipulação. Dentre essas tecnologias, o desenvolvimento de novos dispositivos autoconfiguráveis se tornou prática comum na indústria tecnológica.

Assim, não é difícil de encontrar dispositivos pessoais como celulares, tablets, câmeras e impressoras conectadas em redes privadas ou públicas, sendo que esses dispositivos, ao se autoconfigurarem em uma rede, utilizam protocolos que podem ser vulneráveis a ataques, principalmente, os de negação de serviço.

Essa vulnerabilidade faz com que os clientes possam ser alvos de ataques sem que estejam cientes da exposição em que eles se encontram.

1.1 Motivação

Existem diversos tipos de ataques de negação de serviço (*DoS - Denial of Service Attack*). O princípio básico desse ataque é o de exaurir os recursos de um equipamento que fornece algum determinado serviço por meio do envio de diversos pacotes direcionados a ele [4]. Para que isso ocorra, o ataque pode utilizar uma única fonte de origem (*DoS*) ou pode utilizar várias fontes diferentes, distribuindo a origem do ataque (*DDoS - Distributed Denial of Service Attack*).

Um dos ataques mais comuns do *DDoS* é o ataque de negação de serviço por reflexão. Nesse tipo de ataque, um dispositivo utiliza outros como refletores com o objetivo de amplificar o ataque. Essa amplificação acontece porque os dispositivos refletores recebem um pacote e amplificam o tráfego da rede por meio do envio de um pacote maior ou de vários pacotes que, quando somados, totalizam um tamanho maior de dados que aquele pacote inicial tinha [5].

Atualmente, observa-se ataques de *DDoS* por meio de protocolos como: *Domain Name System* (*DNS*), *Network Time Protocol* (*NTP*), *Simple Network Management Protocol*

(*SNMP*) e outros. Outro protocolo que está ganhando espaço na utilização para esse tipo de ataque é o protocolo *Simple Service Discovery Protocol* (*SSDP*). Esse protocolo, objeto de estudo desse trabalho, possui algumas peculiaridades que têm chamado muita atenção. Aspectos como a amplificação do ataque e a capacidade de ataque a diversas portas são características que facilitam a escolha desse protocolo para a realização de um ataque.

1.2 Justificativa

O constante avanço tecnológico é acompanhado em paralelo pela necessidade de facilitar o uso dessas tecnologias pelo usuário final. A inviabilidade de capacitar o usuário final para tratar questões sobre segurança tecnológica, principalmente quanto a configuração dos dispositivos conforme requisitos mínimos exigidos, exigiu que o protocolo SSDP fosse amplamente utilizado nesses dispositivos. Esse cenário facilita ataques de DDoS que utilizam esse protocolo, aumentando a quantidade de possíveis refletores em uma rede.

O fato da Internet das Coisas (*IoT - Internet of Things*) estar cada vez mais próxima da população gera um questionamento sobre segurança dos serviços que estarão na internet. *IoT* prega a ideia de que dispositivos como carros, celulares, câmeras, máquinas de café, máquinas de lavar e outros possuirão a conectividade com a internet, aumentando a interconectividade dos dispositivos.

Esse avanço técnológico e o aparecimento da *IoT* gera questionamentos sobre como garantir a segurança desses dispositivos conectados a internet. Como garantir que esses dispositivos não sejam utilizados como refletores em um ataque de *DDoS* a um sistema bancário ou a uma usina nuclear, é uma questão que deve ser levantada por exemplo.

1.3 Objetivos

O objetivo dessa monografia é desenvolver um programa que possa realizar um ataque de DDoS, de forma que possamos analisar todas as fases de um ataque. O intuito desse estudo é a prevenção a possíveis ataques de DDoS utilizando o protocolo SSDP.

Esse programa deve procurar por refletores na rede, caso não seja informado nenhum refletor, e ainda deve calcular a amplificação máxima obtida no ataque, com base na saturação do dispositivo.

Vale ressaltar que o trabalho desenvolvido possui fins acadêmicos para demonstração e análise de um ataque em um ambiente controlado. O autor e seu orientador não se responsabilizam por qualquer uso inapropriado, inadequado ou ilegal que estão fora do contexto e dos objetivos desse trabalho. O código desenvolvido está sob a custódia do autor e do orientador para a continuidade futura desse estudo.

1.4 Organização do texto

Esse trabalho está organizado da seguinte forma:

- No Capítulo 2 serão descritos os fundamentos teóricos necessários para compreensão do desenvolvimento desse trabalho.
- No Capítulo 3 será discutida e apresentada a ferramenta de ataque desenvolvida nesse trabalho.
- No Capítulo 4 será apresentada a análise dos resultados obtidos.
- No Capítulo 5 será apresentada a conclusão e os trabalhos futuros.

Capítulo 2

Fundamentação Teórica

Nesse capítulo serão apresentados os conceitos fundamentais para se compreender o ataque DDoS e o protocolo *SSDP*. Além disso, será apresentada ainda a relação existente entre o ataque DDoS e o protocolo *SSDP*.

2.1 Ataques de Negação de Serviço

2.1.1 *DoS - Denial of Service*

Ataques de Negação de Serviço, normalmente representados pela sigla DoS (*denial-of-service*), são ataques que tem por objetivo esgotar os recursos de um dispositivo por meio do envio de várias requisições, sobrecarregando-o, esgotando sua capacidade de resposta, deixando o serviço indisponível para qualquer outra requisição enviada a ele.

De acordo com a *cloudflare*[4], ataques DoS possuem duas categorias:

- *Buffer overflow attack* - Um ataque que faz com que o dispositivo alvo consuma toda a memória RAM, a memória do disco rígido ou o processamento de CPU, fazendo com que ocorra uma falha de funcionamento do sistema.
- *Flood attack* - Um ataque que faz com que o dispositivo alvo receba uma quantidade de requisições maior do que ela pode atender. Dessa forma, o dispositivo alvo fica indisponível por estar ocupado tentando tratar a grande quantidade de requisições feitas. Vale lembrar que, para esse tipo de ataque funcionar, é necessário que a capacidade de geração de requisição do dispositivo atacante, seja maior que maior que a capacidade de atender as requisições do dispositivo alvo.

Um ataque DoS é caracterizado pela existência de apenas um único dispositivo de origem. Quando da existência de mais de um dispositivo realizando o mesmo ataque, dá-se o nome de Ataque de Negação de Serviço Distribuido, representado pela sigla DDoS

(*Distributed Denial of Service*). A grande diferença dessa mudança está relacionada a utilização de múltiplas conexões realizando o ataque e quanto mais conexões, mais pacotes podem ser enviados a rede.

O ataque proposto nesse trabalho é um *Flood attack*, pois será utilizada a mesma filosofia de interrupção de serviço por meio do envio de uma grande quantidade de pacotes na rede. Sendo o envio dos pacotes será feito de forma distribuíd.

2.1.2 *DDoS - Distributed Denial of Service*

O *DDoS* possui duas classificações[6]:

- Ataque volumétrico. Nesse tipo de ataque é enviado uma quantidade de pacotes que esgota os recursos de um dispositivo, sendo esse recurso, por exemplo a quantidade de banda disponível para esse recurso.
- Ataque por abuso de protocolo. Nesse tipo de ataque, é enviado uma quantidadee de pacotes que possuem como características alguma falha de algum protocolo de forma que essa a exploração esgote os recursos do dispositivo alvo.
- Ataque por abuso da camada de aplicação. Nesse tipo de ataque, o atacante busca por programas que estão sendo executados e utilizam a internet para tratar requisições, dessa forma, o atacante visa abusar dessas aplicações de forma que busque e explore falhas.

Alguns ataques comuns de *DDoS*, de acordo com a *cloudflare*[5], são o *HTTP Flood* e o *SYN Flood*, sendo que cada um explora camadas diferentes para realizar o ataque: o *HTTP Flood* explora a camada de aplicação e o *SYN Flood* explora a camada de transporte do modelo de camadas OSI[7]. Dessa forma, o ataque *HTTP Flood* é classificado como ataque por abuso da camada de aplicação e o ataque *SYN Flood* é classificado como ataque por abuso de protocolo.

Já no caso do *DDoS* utilizando o *SSDP*, o ataque é classificado como um ataque volumétrico, pois seu foco não está na exploração de uma falha do protocolo ou de uma aplicação, mas na geração de uma grande quantidade de pacotes a partir de um único pacote utilizando o protocolo.

A estrutura de um ataque volumétrico de *DDoS* pode ser observado na Figura 2.1. Nela pode ser identificado o dispositivo *Attacker*, que controlará o ataque e ainda os dispositivos *Slaves*, que receberão as informações e realizarão o ataque diretamente ao dispositivo (*Victim*).

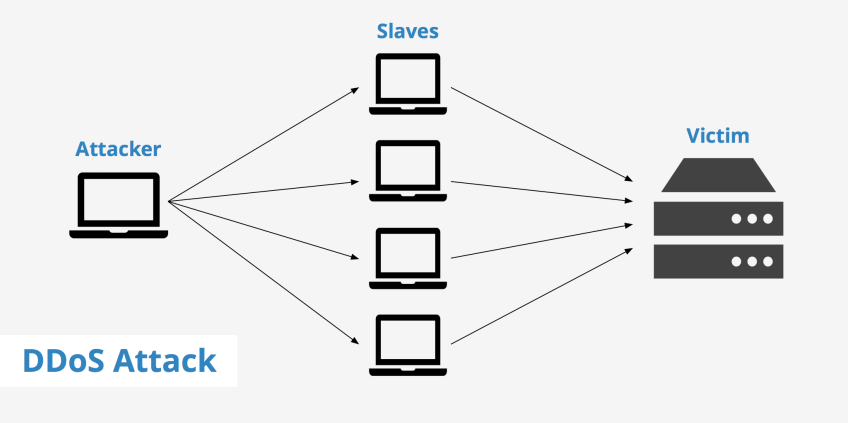

Figura 2.1: Modelo de ataque *DDoS*[1].

No caso do ataque *DDoS* utilizando o protocolo *SSDP*, o dispositivo responsável por iniciar o ataque envia uma mensagem para os refletores, que, na Figura 2.1, são chamados por *Slaves*. Essa mensagem possui um endereço de *IP* (*Internet Protocol*) trocado, utilizando a técnica do *IP Spoofing*[8]. Essa mensagem inicial tem o tamanho menor do que a mensagem que cada refletor irá enviar para a vítima. A diferença de tamanho dessas mensagens mensura a amplificação do ataque.

A amplificação é umas das características desse ataque. Quanto maior for sua amplificação, maior é o volume de pacotes que o ataque gera na rede. A amplificação pode ser calculada da seguinte forma: seja a a amplificação do ataque, p_{req} o pacote da requisição, p_{res} o pacote de resposta e T a função que obtém o tamanho do pacote, o calculo da amplificação será:

$$a = \frac{T(p_{res})}{T(p_{req})} \tag{2.1}$$

Vale lembrar que o tamanho da amplificação é influenciado com base na mensagem que é enviada. Isso ocorre porque todos os pacotes possuem o mesmo tamanho de cabeçalho, ou seja, ambos possuem o cabeçalho ethernet (14 bytes), o cabeçalho *IP* (20 bytes) e o cabeçalho *UDP* (8 bytes). Dessa forma, quando se calcula a amplificação, a diferença entre os pacotes está localizada no tamanho da mensagem de requisição e resposta. Porém, nesse trabalho, não será excluido o cabeçalho *IP* e *UDP* do cálculo da amplificação, mesmo que o resultado da amplificação seja o mesmo, pois nesse casso o intuíto é de evidenciar que os cabeçalhos *IP* e *UDP* também influenciam no processamento das informações, o que pode causar impactos no cálculo da amplificação do ataque.

A Figura 2.2 mostra uma tabela contendo a informação sobre amplificação dos ataques *DDoS* de diversos prococolos:

TYPE	AMPLIFICATION FACTOR
Memcached	10,000 to 51,000
NTP	556.9
CharGEN	358.8
QOTD	140.3
RIPv1	131.24
CLDAP	56 to 70
LDAP	46 to 70
DNS	28 to 54
Quake Network Protocol	63.9
TFTP	60
SSDP	30.8
MSSQL	25
Kad (P2P)	16.3
Portmap (RPCbind)	7 to 28
SNMP	6.3
Steam Protocol	5.5
NetBIOS	3.8
BitTorrent	3.8
Multicast DNS (mDNS)	2 to 10

Figura 2.2: Tabela com amplificação de cada ataque *DDoS* por protocolo.

Conforme descrito na Figura 2.2, o ataque *DDoS* que utiliza o protocolo *SSDP* possui um fator de amplificação máximo igual a 30,8 de acordo com o artigo de *Donald Shin*[9]. Porém esse valor não pode se tornar como valor absoluto, já que não existem informações de como foi obtido esse valor. Esse valor de amplificação será utilizado apenas para comparação com os valores de amplificação que serão obtidos pela ferramenta utilizada no tabalho.

2.2 Protocolo *UPnP*

O protocolo *SSDP* é utilizado como parte do protocolo *UPnP* (*Universal Plug and Play*), dessa forma, é importante a apresentação do protocolo *UPnP* antes de descrever o funcionamento do protocolo *SSDP*.

De acordo com a documentação do protocolo [2], a tecnologia *UPnP* define uma arquitetura de conexão inteligente à internet para dispositivos *wireless*. Essa tecnologia possui como objetivo o aumento da facilidade em seu uso e, ainda, o funcionamento sem configuração de rede para o dispositivo.

Essa configuração automática de conexão à rede permite que o dispositivo possa se juntar a uma rede obtendo um endereço *IP*, aprendendo sobre a presença de outros dispositivos na rede, descobrindo os serviços que os outros dispositivos na rede prestam e, ainda, apresentando os serviços que ele próprio realiza. Todo esse processo ocorre sem que o usuário necessite saber qualquer tipo de informação de configuração.

A palavra *"Universal"* do *UPnP* significa que esse protocolo não utiliza nenhum driver de configuração, apenas protocolos comuns, de acesso universal. Essa característica ressalta que o *UPnP* é um tipo de protocolo independente de arquitetura, podendo ser desenvolvido em qualquer linguagem de programação. A única restrição está relacionada a implementação adequada dos protocolos que o *UPnP* utiliza.

A Figura 2.3 mostra a pilha de protocolos e informações usados no *UPnP*. Como o foco desse estudo é o protocolo *SSDP*, os demais protocolos utilizados *UPnP* UPnP não serão objetos de análise.

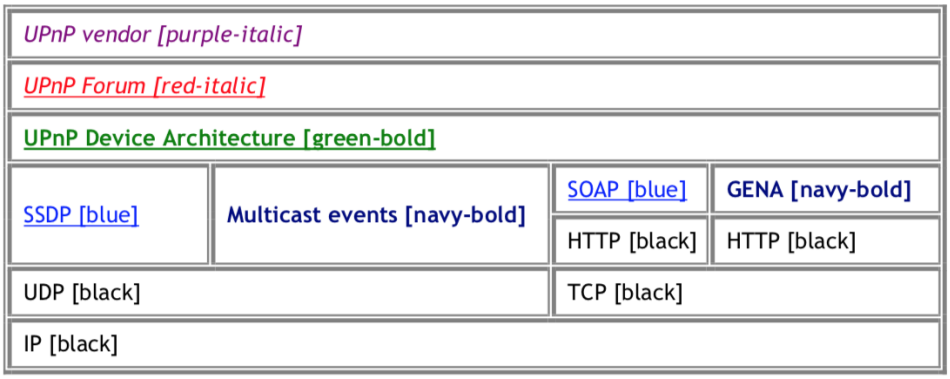

Figura 2.3: Pilha de protocolos usados no UPnP[2].

Os dispositivos que utilizam o protocolo *UPnP* são divididos em duas categorias:

- Dispositivos controlados: São dispositivos que funcionam como servidores, ou seja, respondem a requisições que são enviadas dos pontos de controle. Por exemplo: Impressoras, roteadores, câmeras de segurança, sensores e, dependendo da situação, alguns computadores.
- Dispositivos de controle: São dispositivos que criam uma requisição de serviço para um determinado dispositivo controlado. Normalmente são dispositivos utilizados diretamente por alguém.

A Figura 2.4 apresenta os passos que o protocolo *UPnP* utiliza em seu funcionamento. Esse trabalho foca nos passos 1 e 2, pois assim como comentado no parágrafo anterior, apenas esses passos utilizam o protocolo *SSDP*.

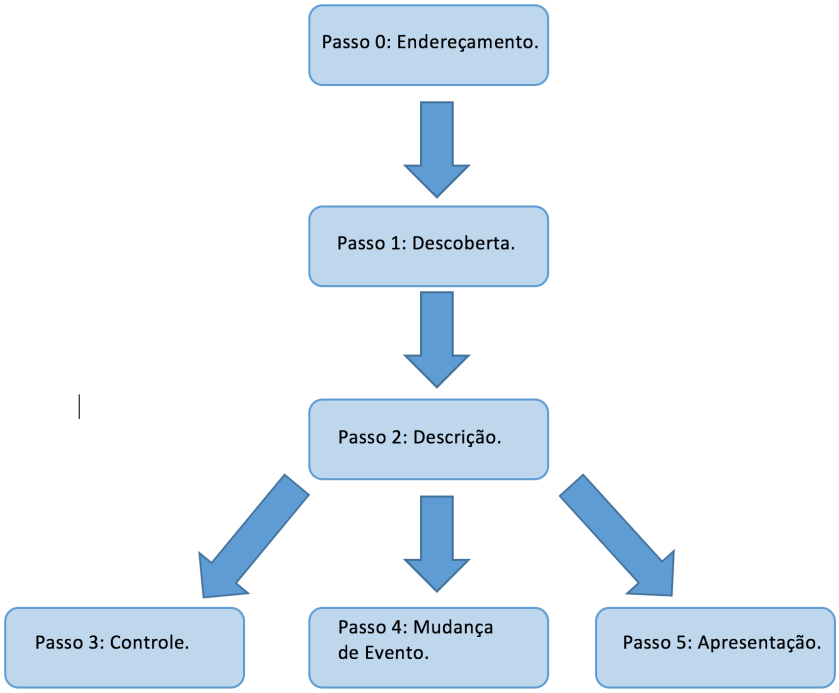

Figura 2.4: Passos utilizados pelo UPnP.

O passo zero (passo inicial) do protocolo possui o objetivo de descobrir um endereço *IP* válido para se conectar na rede. Após esse passo, o *UPnP* inicia o primeiro passo com a descoberta dos dispositivos na rede. Já no segundo passo, após mapeados os dispositivos existentes na rede, o protocolo procura descobrir, de forma mais detalhada, os serviços disponíveis dos dispositivos controlados que estão conectados a rede.

Como o protocolo *SSDP* é responsável pela comunicação entre os dispositivos *UPnP* nos passos 1 e 2, é importante detalhadar esses passos para compreender como é o comportamento do protocolo.

2.2.1 Passo 1: Descoberta

No passo 1 do protocolo *UPnP*, que é a descoberta de dispositivo, utiliza-se mensagens *SSDP* para mapear os atuais ou novos dispositivos conectados à rede. Métodos diferentes de identificação são realizados conforme o tipo do dispositivo.

Quando um dispositivo controlado é adicionado à rede, ele envia uma mensagem *multicast* anunciando seus serviços aos demais dispositivos da rede. Todos os pontos de controle da rede escutam essas mensagens e registram a notificação do novo dispositivo. Como esse anuncio ocorre de forma *multicast*, então o endereço e a porta utilizado para essa comunicação são 239.255.255.250 e 1900 respectivamente[10].

Já quando um dispositivo de controle é adicionado à rede, ele pode mandar uma mensagem para descobrir quais são os dispositivos existentes. Todos os dispositivos ficam escutando o endereço *multicast* e respondem à mensagem quando enviada por um dispositivo de controle. O dispositivo de controle pode enviar uma mensagem *unicast* caso desejado, sendo apenas necessário informar o endereço de IP do dispositivo a ser controlado e aguardar uma resposta.

As informações e as propriedades das mensagens que são trocadas nesse passo serão abordadas na seção de discussão sobre o protocolo *SSDP*.

2.2.2 Passo 2: Descrição

Um dispositivo de controle pode aprender mais sobre os serviços que os outos dispositivos possuem. Caso ocorra esse interesse, o dispositivo de controle utiliza a informação da mensagem de descoberta de dispositivo, que foi enviada no passo anterior, para montar uma mensagem de descrição.

O envio dessa mensagem de descrição ocorre de forma *unicast* e a resposta a essa mensagem deve conter informações completas de como consumir os serviços conforme o dispositivo de controle requisitou.

As informações e as propriedades das mensagens que são trocadas nesse passo serão abordadas na seção de discussão sobre o protocolo SSDP.

2.3 Protocolo *SSDP*

Como foi mencionado anteriormente, o protocolo SSDP é utilizado para a troca de mensagens entre os passos 1 e 2 do protocolo *UPnP*[2]. A necessidade da utilização de uma formatação de troca de mensagens fez com que esse protocolo fosse adotado.

As mensagens do protocolo *SSDP* utilizam parte do campo de cabeçalho do protocolo HTTP (*Hypertext Transfer Protocol*), como definido na RFC(*Request for Comments*) do protocolo *SSDP*[10], porém diferentemente do HTTP, o *SSDP* utiliza os serviços de envio de pacotes ofertados pelo protocolo UDP (*User Datagram Protocol*) e não pelo protocolo TCP (*Transmission Control Protocol*).

Todas as mensagens *SSDP* possuem um formato inicial que indica que tipo de mensagem ela contém. No escopo do protocolo, existem três tipos gerais de mensagens, e para que uma mensagem esteja no padrão do protocolo, a primeira linha de mensagem deve estar formatada de acordo com uma das três opções abaixo:

- NOTIFY * HTTP/1.1\r\n
- M-SEARCH * HTTP/1.1\r\n

- HTTP/1.1 200 OK\r\n

Na mensagem *SSDP*, os campos do cabeçalho podem ser escritos com letras maiúsculas ou minúsculas, seguidos por dois pontos (:) e seguidos pelo valor do campo. Um exemplo de um campo do cabeçalho seria:

"HOST: 239.255.255.250:1900"

Caso a mensagem não esteja no padrão de formatação do protocolo *SSDP*, então os dispositivos podem ignorar a mensagem. Se a mensagem estiver no padrão, porém com algum campo de cabeçalho que o dispositivo desconheça, o dispositivo pode ignorar as informações daquele campo.

Em vários momentos os dispositivos se comunicam para descobrirem serviços que estão disponíveis e os seus respectivos status. Para uma melhor análise, o estudo será separado por tipos de mensagens enviadas e quando cada mensagem será necessária ser transmitida.

2.3.1 Mensagens NOTIFY

Mensagens NOTIFY aparecem sempre que o dispositivo precisa notificar algo, ou seja, sempre que ocorrer alguma mudança. Situações como o surgimento de novos dispositivos, novos serviços, desligamento de algum dispositivo da rede e atualização das informações de serviços podem exigir que os dispositivos de controle sejam notificados.

Para o dispositivo se anunciar de forma completa, deve-se utilizar o formato de mensagem NOTIFY do protocolo *SSDP* e será enviada uma mensagem por serviço que aquele dispositivo oferece. Cada mensagem contem informações sobre o dispositivo e o tipo de serviço que é ofertado, lembrando que se um dispositivo possui mais de um serviço, então será enviado mais de uma mensagem ao se anunciar.

A estrutura das mensagens do tipo NOTIFY, utilizadas para anunciar novos serviços, pode ser observada na Figura 2.5[2].

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age = seconds until advertisement expires
LOCATION: URL for UPnP description for root device
NT: notification type
NTS: ssdp:alive
SERVER: OS/version UPnP/1.1 product/version
USN: composite identifier for the advertisement
BOOTID.UPNP.ORG: number increased each time device sends an initial announce or an update
message
CONFIGID.UPNP.ORG: number used for caching description information
SEARCHPORT.UPNP.ORG: number identifies port on which device responds to unicast M-SEARCH
```

Figura 2.5: Estrutura da mensagem NOTIFY para informar novos serviços.

Na Figura 2.5 podemos perceber que o cabeçalho da mensagem possui diversos campos e informações. Dessa forma temos:

- **NOTIFY * HTTP/1.1**: Linha de requisição. Essa linha é de uso obrigatório. Nela é informado o método de notificação e a versão do protocolo HTTP que é utilizado para envio da mensagem.
- **HOST: 239.255.255.250:1900**: Esse campo é obrigatório contendo o valor do endereço *multicast*, já que ela é utilizada para anunciar serviços de novos dispositivos.
- **CACHE-CONTROL**: Esse campo é obrigatório e deve conter a palavra "max-age = ", seguido por um número inteiro que indica o tempo, em segundos, que o serviço estará disponível no dispositivo. Esse número deve ser maior ou igual a 1800 segundos (30 minutos), porém é permitido exceções, caso exista.
- **LOCATION**: Esse campo é obrigatório. Esse campo contem o endereço para a descrição do dispositivo *UPnP*.
- **NT**: Esse campo é obrigatório, podendo assumir diferentes valores, mas todos trazem informações mais detalhadas sobre as notificações. Pode-se mostrar informações como o UUID (*Universal Unique Identifier*), que é um identificador único para cada dispositivo, informações sobre o nome e a versão do serviço.
- **NTS**: Esse campo é obrigatório, que informa que esse método é uma notificação de surgimento de novos dispositivos. Nesse caso, esse campo deve sempre possuir o valor "ssdp:alive".
- **SERVER**: Esse campo é obrigatório, contendo o valor de um token, no qual é dividido em três partes, a primeira indica o nome do sistema operacinal e a versão, o segundo indica que utiliza o protocolo *UPnP* e a sua versão, a terceira parte indica o produto e a sua versão.
- **USN**: Esse campo é obrigatório. O USN (*Unique Service Name*) indica o identificador único do serviço. Em alguns casos esse campo possui o mesmo valor do campo NT e em outros esse campo possui informações que englobam aquelas que estão no campo NT.
- **BOOTID.UPNP.ORG, CONFIGID.UPNP.ORG, SEARCHPORT.UPNP.ORG**: São campos obrigatórios para o protocolo *UPnP*, porém não são campos obrigatórios para o *SSDP*, dessa forma, eles não influenciam nesse estudo.

A Figura 2.6 mostra um exemplo de mensagem NOTIFY do protocolo *SSDP*. Nela é possível observar os campos e seus valores contidos no cabeçalho da mensagem.

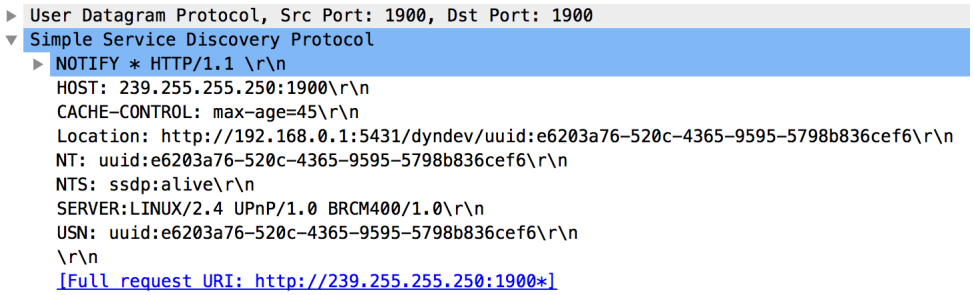

Figura 2.6: Exemplo de mensagem NOTIFY para informar novos serviços.

Quando um dispositivo é removido da rede, ele pode anunciar a sua saída por meio de uma mensagem NOTIFY, utilizando o valor *"ssdp:byebye"* no campo **NTS**. Porém, caso não seja possível enviar essa mensagem antes do dispositivo ser removido, os dispositivos de controle irão apenas remover as informações sobre os serviços que aquele dispositivo realiza quando as mensagens NOTIFY, que anunciaram aqueles serviços, expirarem.

A Figura 2.7[2] mostra a estrutura da mensagem NOTIFY responsável pelo desligamento do dispositivo e dos serviços que ele fornece. Pode-se observar que a estrutura da mensagem é mais simples que a estrutura da mensagem NOTIFY para anunciar serviços. Não entraremos novamente nos detalhes dos campos pois são os mesmos da mensagem de NOTIFY para anunciar os serviços.

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
NT: notification type
NTS: ssdp:byebye
USN: composite identifier for the advertisement
BOOTID.UPNP.ORG: number increased each time device sends an initial announce or an update
message
CONFIGID.UPNP.ORG: number used for caching description information
```

Figura 2.7: Exemplo de mensagem NOTIFY para informar desligamento de dispositivos.

Assim como existe a mensagem para desligamento do dispositivo, também existe a mensagem para atualização de um serviço do dispositivo controlado que já está conectado à rede.

Para que uma atualização de serviços de um dispositivo controlado ocorra e todos os outros dispositivos de controle passem a conhecer esse novo serviço, é necessário uma mensagem NOTIFY com o valor *"ssdp:update"* no campo **NTS**. A Figura 2.8[2] mostra a estrutura de uma mensagem para atualizar serviços de um dispositivo controlado. Pode-se observar que a estrutura da mensagem é mais simples que a estrutura da mensagem NOTIFY para anunciar serviços. Novamente não entraremos nos detalhes dos campos pois são os mesmos da mensagem de NOTIFY para anunciar os serviços.

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
LOCATION: URL for UPnP description for root device
NT: notification type
NTS: ssdp:update
USN: composite identifier for the advertisement
BOOTID.UPNP.ORG: BOOTID value that the device has used in its previous announcements
CONFIGID.UPNP.ORG: number used for caching description information
NEXTBOOTID.UPNP.ORG: new BOOTID value that the device will use in subsequent announcements
SEARCHPORT.UPNP.ORG: number identifies port on which device responds to unicast M-SEARCH
```

Figura 2.8: Exemplo de mensagem NOTIFY para informar a disponibilidade de serviço para dispositivo controlado.

2.3.2 Mensagens M-SEARCH

Quando um dispositivo de controle é adicionado a uma rede, esse utilizará mensagens M-SEARCH do protocolo *SSDP* para descobrir dispositivos controlados que já estão na rede. O envio dessa mensagem também ocorre por meio do endereço e porta *multicast* do protocolo *SSDP*, ou seja, utiliza o endereço 239.255.255.250 e porta 1900.

Os dispositivos controlados estarão escutando nesse endereço e, quando receberem essa mensagem de busca por dispositivos, eles responderão essa mensagem por meio de uma outro tipo *unicast*, com uma mensagem de notificação informando os seus serviços.

A Figura 2.9[2] mostra a estrutura da mensagem M-SEARCH que é enviada para todos os dispositivos da rede.

```
M-SEARCH * HTTP/1.1
HOST: 239.255.255.250:1900
MAN: "ssdp:discover"
MX: seconds to delay response
ST: search target
USER-AGENT: OS/version UPnP/1.1 product/version
```

Figura 2.9: Estrutura da mensagem M-SEARCH para buscar por serviços dos dispositivos controlados..

Na Figura 2.9 existem campos que divergem da mensagem NOTIFY, dessa forma, temos que:

- **M-SEARCH * HTTP/1.1**: Linha de requisição. Essa linha é de uso obrigatório. Nela é informado o método de busca do serviço e a versão do protocolo HTTP que é utilizado para envio da mensagem.

- **HOST: 239.255.255.250:1900**: Esse campo é obrigatório, no caso esse campo sempre terá o valor do endereço *multicast* pois essa mensagem é utilizada para anunciar serviços de novos dispositivos.
- **MAN: "ssdp:discover"**: Esse campo é obrigatório e define o escopo da mensagem, que é uma mensagem de busca por novos serviços. Deve possuir a mensagem *"ssdo:discover"* dentro de aspas duplas.
- **MX**: Esse campo é obrigatório. Esse campo contém o tempo máximo, em segundos, para esperar uma resposta da mensagem de busca por serviços que acabou de ser enviada. Esse campo deve ser maior ou igual a 1 segundo e normalmente não é maior que 5 segundos. Esse valor pode aumentar dependendo da quantidade de respostas a se obter em uma rede, porém o valor desse campo não pode ser influenciado por características como latência ou demora no fluxo de informações na rede.
- **ST**: Esse campo é obrigatório e indica o alvo da mensagem. Para mensagens *multicast* como é o caso, ele deve possuir o valor *"ssdp:all"* e, dessa forma, o dispositivo de controle deve obter todos os serviços de todos os dispositivos controlados que estão na rede.
- **USER-AGENT**: Esse campo é opcional. O valor esperado nesse campo é um token de identificação com as características do dispositivo. Esse token possui vários campos, o primeiro dele indica o sistema operacional, o segundo indica a versão do protocolo *UPnP*, o terceiro indica o produto(software) e a sua versão.

Mensagens M-SEARCH também podem ser direcionadas a um único dispositivo. A estrutura da mensagem é mostrada na Figura 2.10[2]. A diferença entre as duas estruturas é que no campo *HOST* e *ST*, o valor deles serão iguais e devem ser o endereço e a porta que aquele dispositivo que irá responder a requisição. Também é importante observar que não existe o campo *MX* já que, por definição, a resposta para mensagens M-SEARCH *unicast* deve ser enviada imediatamente e o dispositivo de controle deve esperar por, no máximo, 1 segundo.

```
M-SEARCH * HTTP/1.1
HOST: hostname:portNumber
MAN: "ssdp:discover"
ST: search target
USER-AGENT: OS/version UPnP/1.1 product/version
```

Figura 2.10: Estrutura da mensagem M-SEARCH para buscar por serviços de um único dispositivo controlado..

A Figura 2.11 mostra um exemplo de um pacote de busca enviado pela aplicação Chrome em um sistema operacional Windows.

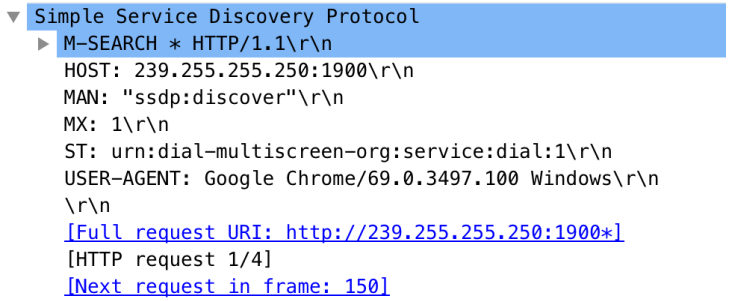

Figura 2.11: Exemplo de uma mensagem M-SEARCH para buscar por serviços de vários dispositivos em uma rede..

2.3.3 Mensagens 200 OK

Após o dispositivo de controle buscar por serviços por meio de uma mensagem M-SEARCH, uma resposta deve ser enviada pelo dispositivo controlado. Essa resposta possui o formato das "Mensagens 200 OK"do protocolo *SSDP*. A estrutura pode ser observada na Figura 2.12[2]:

```
HTTP/1.1 200 OK
CACHE-CONTROL: max-age = seconds until advertisement expires
DATE: when response was generated
EXT:
LOCATION: URL for UPnP description for root device
SERVER: OS/version UPnP/1.1 product/version
ST: search target
USN: composite identifier for the advertisement
BOOTID.UPNP.ORG: number increased each time device sends an initial announce or an update
message
CONFIGID.UPNP.ORG: number used for caching description information
SEARCHPORT.UPNP.ORG: number identifies port on which device responds to unicast M-SEARCH
```

Figura 2.12: Estrutura da mensagem 200 OK que é utilizada como resposta a mensagem M-SEARCH.

Essa estrutura tenta seguir a estrutura da mensagem NOTIFY, dessa forma é possível observar alguns campos similares entre as duas mensagens. Os seguintes campos são diferentes:

- **DATE**: Esse campo não é obrigatório e possui a data de envio da mensagem de resposta.
- **EXT**: Esse campo é obrigatório e necessário para compatibilidade com o UPnP 1.0.

- **ST**: Esse campo é obrigatório e possui a mesma funcionalidade na mensagem M-SEARCH.

A Figura 2.13 mostra uma mensagem de resposta de um dispositivo controlado. Nela pode-se observar os campos descritos anteriormente. É importante salientar que a quantidade de respostas respeita a seguinte regra[2].:

$$M = 3 + 2 \times d + k$$

onde:
M = Quantidade de mensagens de resposta.
d = Quantidade de dispositivos embarcados.
k = Quantidade de serviços distintos oferecidos.

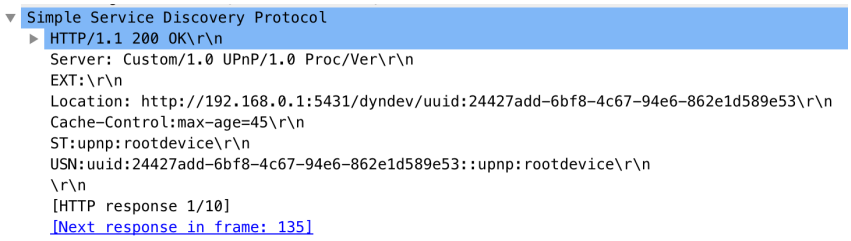

Figura 2.13: Exemplo da mensagem 200 OK que é utilizada como resposta a mensagem M-SEARCH..

2.3.4 Visão geral do protocolo SSDP

A Figura 2.14[3] mostra uma visão geral da troca de mensagens do protocolo *SSDP*. Nela pode-se observar que o envio de mensagens NOTIFY são enviadas dos dispositivos controlados para os dispositivos de controle por meio do envio *multicast*. Essas mensagens são enviadas de forma independente, ou seja, nenhum dispositivo de controle solicitou a mensagem. Dessa forma, percebe-se que o dispositivo a ser controlado sempre informa quando seus serviços estão disponíveis ou não, por meio das mensagens com os valores *Alive* ou *ByeBye*.

A Figura 2.14 também mostra a troca de informações entre os dispositivos por meio de mensagens *request* e *response*. Dessa forma, observa-se que o dispositivo de controle envia uma mensagem de busca de forma *multicast* e os dispositivos controlados respondem de forma *unicast*, informando seus serviços.

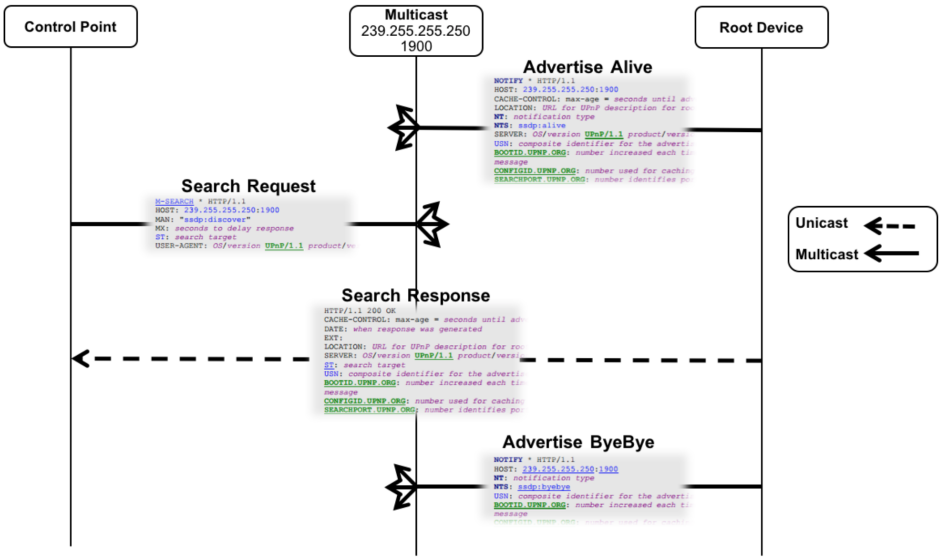

Figura 2.14: Visão geral do envio das mensagens do protocolo *SSDP*[3].

Essa interação entre os dispositivos são a base para o desenvolvimento do protocolo *SSDP*. Agora que entendemos o funcionamento do protocolo, será possível entender como explorar o protocolo e realizar um ataque *DDoS* por reflexão amplificada utilizando esse protocolo.

2.4 *DDoS* utilizando o protocolo *SSDP*

A utilização do protocolo *SSDP* em ataques *DDoS* não é antigo como o do ataque *DDoS* que utiliza o protocolo *SNMP*, porém é um tipo de ataque que se tornou famoso pela quantidade de novos dispositivos que surgem e que podem ser explorados por esse ataque.

A ideia base da utilização do protocolo *SSDP* para a realização do ataque *DDoS* é a utilização de mensagens M-SEARCH como mensagens de requisição. Um dispositivo que envia essa mensagem, buscando por serviços de outros dispositivos, receberá várias mensagens com informações sobre os serviços dos dispositivos existentes. Dessa forma, se ocorrer a realização da troca de endereços IP, ou seja, o *IP Spoofing*, no pacote, temos que ocorrerá o ataque. A Figura 2.15[11] ilustra esse tipo de ataque:

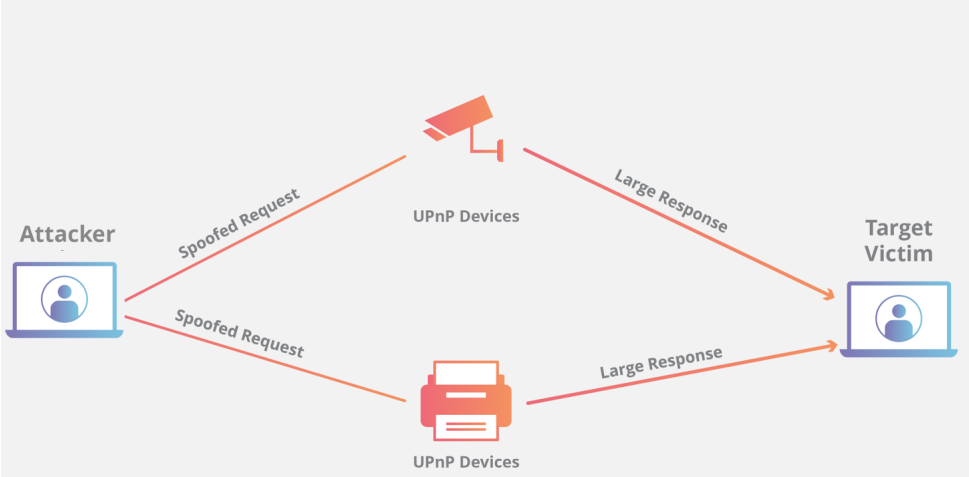

Figura 2.15: Ilustração de um ataque DDoS utilizando o protocolo SSDP.

O envio da mensagem de busca por serviços dos dispositivos podem ocorrer de duas maneiras: a primeira está relacionada com o envio de mensagens *multicast* e a segunda está relacionada com o envio de mensagens *unicast*.

As mensagens *multicast*, como a mensagem descrita na Figura 2.9, são enviadas para todos os dispositivos que possuem serviços disponíveis. Caso ocorra a troca de endereço *IP*, colocando o endereço *IP* do dispositivo alvo na mensagem, então todos os dispositivos da rede responderão a mensagem enviando as informações dos seus serviços para o dispositivo alvo. Dessa forma, todos os dispositivos a serem controlados serão dispositivos refletores.

Já as mensagens *unicast*, como a mensagem descrita na Figura 2.10, são enviadas para apenas um dispositivo que possui algum serviço. Com a troca de endereços *IP*, apenas esse dispositivo participará do ataque, ou seja, apenas esse dispositivo será o dispositivo refletor.

Capítulo 3

Ferramenta de Ataque SSDP

Nesse capítulo será mostrada a ferramenta desenvolvida para realizar o ataque DDoS por reflexão amplificada utilizando o protocolo SSDP. Porém, antes de detalhar a ferramenta utilizada, é necessário entender a rede de dispositivos que foi utilizada, juntamente com a sua organização.

3.1 Rede de dispositivos

A Figura 3.3 e a Figura 3.3 mostram os dispositivos que foram utilizados na montagem da rede.

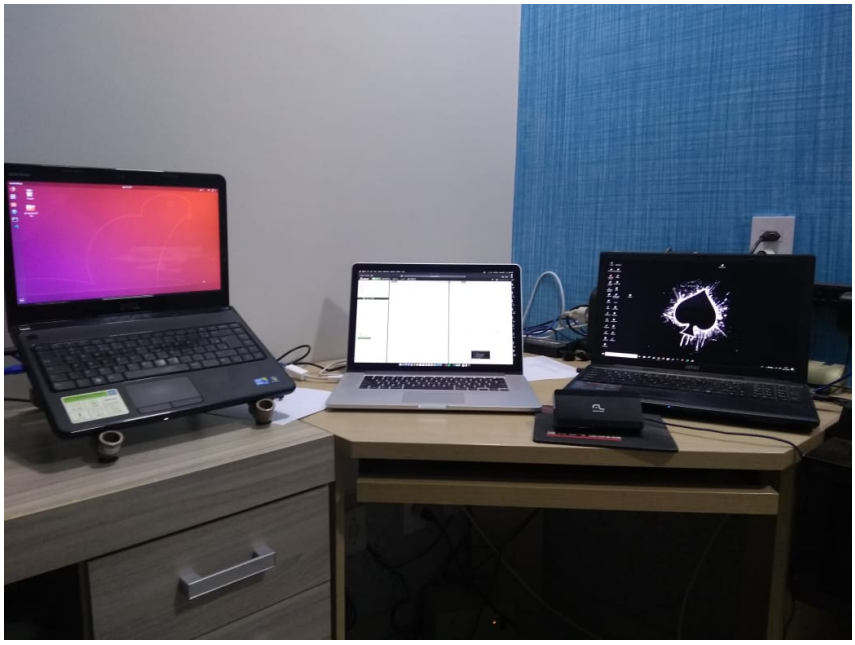

Figura 3.1: Dispositivos utilizados na rede montada para analisar o ataque.

Figura 3.2: Os dois Modems utilizados na rede montada.

Como mostrado nas imagens, são utilizados três computadores, um switch e um modem. Sendo que para o estudo do ataque, esses dispositivos são utilizados com o seguinte propósito:

- Computador com MacOS: Computador que gera os ataques.
- Computador com Linux: Computador que simula um dispositivo refletor.
- Computador com Windows: Computador que é o alvo do ataque.
- Modem da provedora NET: Dispositivo refletor real.
- Switch: Utilizado para conectar todos os dispositivos em uma rede cabeada e isolada.
- Modem D-Link: Utilizado para isolar o dispositivo alvo da rede montada.

O computador responsável por gerar os ataques executa um programa em linguagem C responsável por realizar efetivamente o ataque. Já o dispositivo que simula um refletor executa um outro programa em linguagem C que simula o comportamento de um dispositivo refletor. Já o modem e o computador alvo não executam nenhum programa, sendo apenas explorados na rede.

Para um melhor entendimento da participação dos dispositivos envolvidos na rede, a descrição do procedimento executado está dividido conforme o nível de influência de cada dispositivo no experimento: primeiro será apresentado o código executado no computador que gera o ataque, depois a partipação dos dispositivos refletores e por fim as ocorrências no dispositivo alvo.

3.2 Gerando o ataque

O programa em linguegem C responsável por realizar o ataque é executado no computador que possui o sistema operacional MacOS. O programa está dividido em módulos e, dessa forma, a explicação do programa também será dividida nos mesmos módulos, facilitando o seu entendimento. A partir disso, temos:

1. Pesquisa de dispositivos na rede.
2. Configuração do ataque.
3. Montagem do pacote.
4. Realização do ataque.

3.2.1 Pesquisa de dispositivos na rede

A pesquisa de dispositivos na rede é o primeiro passo que o programa realiza. É nessa pesquisa que se monta a lista de dispositivos que estão na rede, além de obter outras informações importantes que serão utilizadas para se realizar o ataque.

A pesquisa por dispositivos é dividida em duas partes:

- Pesquisa por dispositivos controlados.
- Pesquisa por dispositivos de controle.

Pesquisa por dispositivos controlados

Para que o programa seja informado sobre os dispositivos que utilizam o protocolo SSDP que estão conectados na rede, existem duas formas a realizar: a primeira seria esperar por mensagens do tipo *NOTIFY* desses dispositivos, já a segunda forma seria enviar uma mensagem *M-SEARCH multicast* de forma que todos os dispositivos a serem controlados teriam que responderiam essa mensagem. No programa foi utilizado essa segunda forma.

Então o programa envia uma mensagem nos padrões da Figura 2.9 e aguarda a respostas dos dispositivos. A Figura 3.3 mostra por meio do programa *Wireshark*, que é

um programa de captura de pacotes de rede, o envio dessa mensagem e as respostas obtidas imediatamente após o envio. Sendo que nela é possível observar que a mensagem enviada pelo computador atacante, possui o endereço *IP* igual a 192.168.0.14, possui um tamanho de 180 *bytes*. O modem, que possui o endereço *IP* igual a 192.168.0.1, possui 10 mensagens de retorno, totalizando um tamanho de 3411 *bytes* de resposta. Já o computador que executa um programa que simula um refletor, que possui o endereço *IP* igual a 192.168.0.36, possui 8 mensagens de retorno, totalizando um tamanho de 2136 *bytes* de resposta.

2	4.277518	192.168.0.14	239.255.255.250	SSDP	180 M-SEARCH * HTTP/1.1
3	4.278784	192.168.0.1	192.168.0.14	SSDP	299 HTTP/1.1 200 OK
4	4.278826	192.168.0.1	192.168.0.14	SSDP	308 HTTP/1.1 200 OK
5	4.278913	192.168.0.1	192.168.0.14	SSDP	371 HTTP/1.1 200 OK
6	4.278988	192.168.0.1	192.168.0.14	SSDP	363 HTTP/1.1 200 OK
7	4.279069	192.168.0.1	192.168.0.14	SSDP	308 HTTP/1.1 200 OK
8	4.279150	192.168.0.1	192.168.0.14	SSDP	347 HTTP/1.1 200 OK
9	4.279223	192.168.0.1	192.168.0.14	SSDP	379 HTTP/1.1 200 OK
10	4.279288	192.168.0.1	192.168.0.14	SSDP	308 HTTP/1.1 200 OK
11	4.279376	192.168.0.1	192.168.0.14	SSDP	367 HTTP/1.1 200 OK
12	4.279447	192.168.0.1	192.168.0.14	SSDP	361 HTTP/1.1 200 OK
15	5.278861	192.168.0.36	192.168.0.14	SSDP	267 HTTP/1.1 200 OK
16	5.278865	192.168.0.36	192.168.0.14	SSDP	267 HTTP/1.1 200 OK
17	5.278868	192.168.0.36	192.168.0.14	SSDP	267 HTTP/1.1 200 OK
18	5.278871	192.168.0.36	192.168.0.14	SSDP	267 HTTP/1.1 200 OK
19	5.278874	192.168.0.36	192.168.0.14	SSDP	267 HTTP/1.1 200 OK
20	5.278876	192.168.0.36	192.168.0.14	SSDP	267 HTTP/1.1 200 OK
21	5.278879	192.168.0.36	192.168.0.14	SSDP	267 HTTP/1.1 200 OK
22	5.278882	192.168.0.36	192.168.0.14	SSDP	267 HTTP/1.1 200 OK

Figura 3.3: Respostas à mensagem M-SEARCH utilizada no programa obtidas pelo *Wireshark*.

Já a Figura 3.4 mostra de forma esquemática o que ocorre quando a mensagem *multicast M-SEARCH* é enviada na rede.

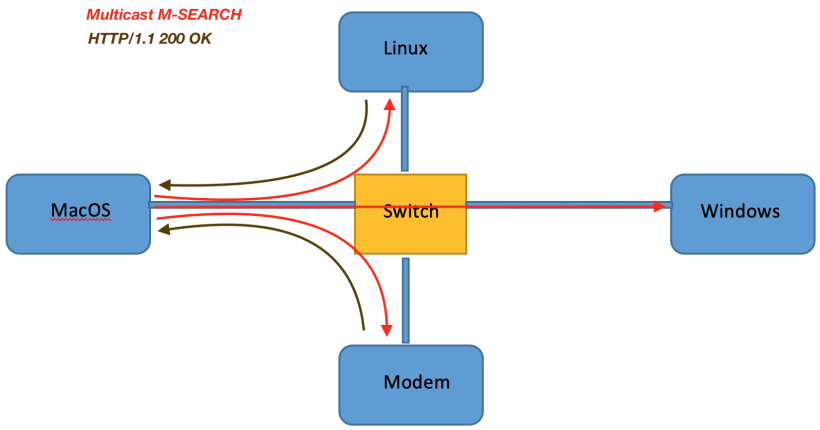

Figura 3.4: Representação da mensagem de busca por dispositivos controlados na rede montada..

No esquema, os dispositivos a serem controlados respondem a mensagem por meio da mensagem *200 OK* do protocolo SSDP. No teste realizado, o Modem é responsável por enviar 10 mensagens e o Linux envia 8 mensagens. O computador com o sistema operacional Windows também recebe a mensagem, porém como ele é um dispositivo de controle, ele não responde.

Dessa forma, os dispositivos são adicionados em uma lista, que será utilizada para informar quais e que tipos de dispositivos existem na rede.

Pesquisa por dispositivos de controle

A pesquisa por dispositivos de controle ocorre de forma um pouco diferente, porém os dispositivos acabam sendo salvos na mesma lista criada anteriormente, identificando sua diferença devido ao tipo do dispositivo.

Para obter esses dispositivos, o programa começa a escutar por mensagens que são direcionadas ao endereço *multicast* (239.255.255.250) na porta 1900. As mensagens que chegam nesse endereço podem ser mensagens de dois tipos: Se for uma mensagem *NOTIFY*, então o dispositivo que mandou a mensagem é do tipo controlado e, possivelmente já, deve estar adicionado na lista de dispositivos; Se for uma mensagem do tipo *M-SEARCH*, então o dispositivo é do tipo controle, logo esse dispositivo é salvo na lista. A Figura 3.4 mostra de forma esquemática como essa captura de dispositivos ocorre.

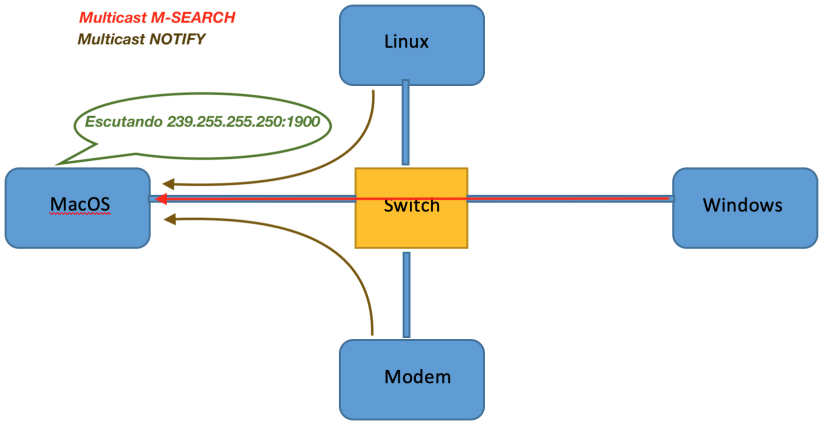

Figura 3.5: Representação da busca por dispositivos de controle na rede montada..

Após essa pesquisa de dispositivos, o programa já possui conhecimento de todos os dispositivos que estão presentes na rede. Dessa forma, esse passo é necessário para que ocorra o ataque, já que a partir dessa lista que será escolhido o dispositivo alvo do ataque.

3.2.2 Configuração do ataque

Na configuração do ataque, algumas definições deverão ser informadas para o programa entenda como deverá ser o ataque. Para isso é necessário definir e informar os seguintes dados:

- Dispositivo alvo.
- Itensidade do ataque(valor de 1 a 8).
- Tempo que ocorrerá o ataque (em segundos).

A escolha do dispositivo alvo ocorrerá na configuração do ataque, sendo realizada após a verificação dos dispositivos que estão na rede, conforme a lista de dispositivos que foi criada na fase anterior.

Uma vez que o dispositivo alvo já escolhido, será necessário escolher a intensidade do ataque. Esse parâmetro indicará quantos pacotes por segundo serão enviados no ataque. Essa quantidade de pacotes por segundo respeita a seguinte fórmula:

$$10^{L-1}$$

Onde L é a intensidade do ataque escolhida. Dessa forma, temos a seguinte tabela:

Tabela 3.1: Intensidade por Pacotes/s.

L	**Pacotes/s**
1	1
2	10
3	100
4	1000
5	10000
6	100000
7	1000000
8	10000000

Vale lembrar que esse parâmetro indica o fluxo de pacotes a serem enviados aos refletores da rede, ou seja, esse parâmetro não indica a intensidade final do ataque, apenas a taxa de criação e envio dos pacotes iniciais. Sendo que essa taxa é um valor teórico, o qual não pode ser garantido que o programa conseguirá atingir esses valores durante o processo de ataque.

A última parte da configuração é a definição do tempo de ataque, em segundos.

3.2.3 Montagem do pacote

Após a configuração do ataque, o proximo passo é a montagem do pacote que será utilizado para enviar mensagens aos refletores. A montagem ocorre a partir do cabeçalho *IP*, depois do cabeçalho *UDP* e por último a mensagem *SSDP* é anexada.

Durante a montagem do pacote, é realizado o *IP Spoofing*, ou seja, troca-se o endereço *IP*, colocando o endereço *IP* do computador alvo. Dessa forma, as mensagens que chegarem aos refletores serão enviadas ao computador alvo.

A montagem do pacote ficou da seguinte forma:

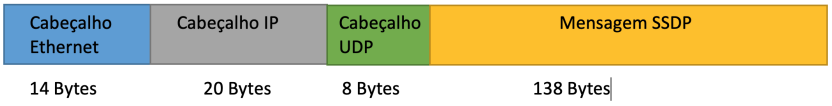

Figura 3.6: Organização do pacote utilizado para realização do ataque.

Importante observar que o programa apenas monta os cabeçalhos *IP* e *UDP*, não mudando o cabeçalho *Ethernet*. Também é importante saber que o pacote montado ficou com um tamanho de 180 Bytes e essa informação será importante para analisar a cálculo do ganho de amplificação do ataque e que a mensagem SSDP mostrada na imagem é a mensagem *multicast M-SEARCH*.

Uma vez que o pacote está montado e o ataque está configurado, o programa está pronto para realizar o ataque.

3.2.4 Realização do ataque

No computador responsável pelo ataque

A realização do ataque se resume apenas em enviar os pacotes que foram gerados no passo da montagem, conforme configuração realizada. Para que o ataque ocorresse conforme a configuração inicial, foi necessário levantar alguns pontos importantes: relacionar a intensidade definida, o tempo de espera entre o envio dos pacotes e quantas *threads* seriam necessárias para realizar o ataque.

O tempo de espera entre o envio dos pacotes é definido como sendo o inverso da quantidade de envio de pacotes por segundo, ou seja:

$$T = \frac{1}{10^{L-1}}$$

Onde T é o tempo de espera entre o envio dos pacotes. Dessa forma, temos os seguintes casos:

Tabela 3.2: Pacotes/s e tempo/s.

L	**Pacotes/s**	**T/s**
1	1	1
2	10	0,1
3	100	0,01
4	1000	0,001
5	10000	0,0001
6	100000	0,00001
7	1000000	0,000001
8	10000000	0,0000001

O questionamento sobre a quantidade de *threads* a se utilizar para enviar os pacotes se justifica por meio do propósito de retirar o possível gargalo do tempo de envio de pacotes. A ideia é utilizar mais de uma *thread* para enviar pacotes de forma que o tempo de espera entre o envio dos pacotes seja maior, possibilitando que não ocorra a saturação na geração dos pacotes.

Dessa forma, o calculo do tempo de espera mostrado anteriormente funciona para a realização do ataque utilizando apenas uma thread. Para adaptar o cálculo do tempo de espera para esse novo cenário, foi necessário acrescentar a quantidade de *threads* utilizada no programa. Dessa forma, temos:

$$T = N \times \frac{1}{10^{L-1}}$$

Onde T é o tempo de espera entre o envio dos pacotes e N o número de *threads* que estão enviando os pacotes.

No programa é estudado o uso de no máximo 8 *threads* para a geração dos pacotes, dessa forma, a tabela do tempo de espera entre o envio dos pacotes seria a seguinte:

	Numero de threads:	1	2	3	4	5	6	7	8
L	pacotes/s	T/s	T/s	T/s	T/s	T/s	T/s	T/s	T/s
1	1	1	2	3	4	5	6	7	8
2	10	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8
3	100	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08
4	1000	0,001	0,002	0,003	0,004	0,005	0,006	0,007	0,008
5	10000	0,0001	0,0002	0,0003	0,0004	0,0005	0,0006	0,0007	0,0008
6	100000	0,00001	0,00002	0,00003	0,00004	0,00005	0,00006	0,00007	0,00008
7	1000000	0,000001	0,000002	0,000003	0,000004	0,000005	0,000006	0,000007	0,000008
8	10000000	0,0000001	0,0000002	0,0000003	0,0000004	0,0000005	0,0000006	0,0000007	0,0000008

Figura 3.7: Utilização de tempos entre envios dos pacotes por threads.

Na análise de resultados serão discutidas as diferenças na utilização de diferentes *threads* para o envio dos pacotes para os refletores, dessa forma, a discussão sobre a quantidade de *threads* utilizada no programa retornará com outra visão.

Nos refletores

O envio de pacotes *multicast M-SEARCH* é responsável por promover respostas *200 OK* nos refletores, dessa forma, a Figura 3.8 mostra de forma esquemática o comportamento da rede enquanto o ataque está ocorrendo. Vale lembrar que o Modem responde a mensagem de requisição com 10 mensagens de resposta e o Lunix com 8 mensagens de resposta.

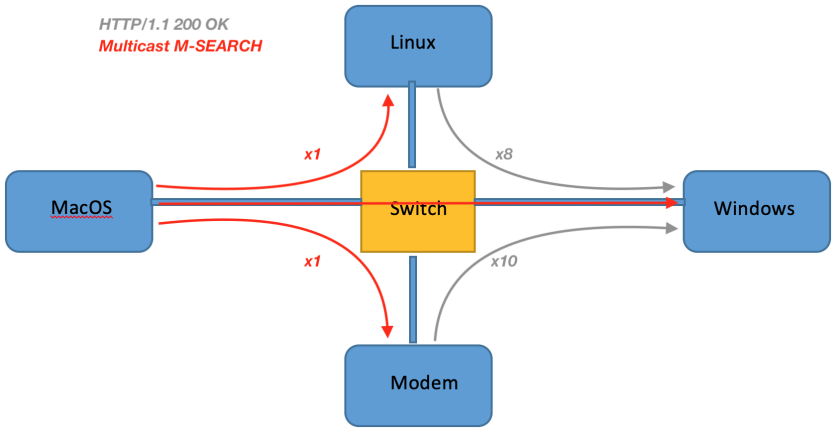

Figura 3.8: Representação geral dos envios dos pacotes durante o ataque.

O modem utilizado possui três dispositivos embarcados e um serviço distinto. Dessa forma, respeitando a regra do protocolo sobre a quantidade de mensagens a serem respondidas, temos que:

$$M = 3 + 2 \times 3 + 1$$

Assim, o modem responde com 10 mensagens a requisição de *M-SEARCH*, sendo cada mensagem com um tamanho diferente, porém ao todo, as 10 mensagens possui um tamanho igual a 3411 Bytes. Vale lembrar que a Figura 3.3 mostra as mensagens de resposta do Modem, que possui o endereço *IP* 192.168.0.1.

Já no Linux, que foi utilizado como um simulador de refletor, foi necessário escrever um código em linguagem C que respondesse às mensagens *M-SEARCH* de forma que simulasse o comportamento de um refletor.

O código inicialmente escuta mensagens que são enviadas para o endereço *multicast* do protocolo SSDP e responde 8 mensagens para quem enviou a mensagem. O simulador de refletor responde como se possuísse dois dispositivos embarcados e um serviço distinto, porém, essa simulação respeita apenas a quantidade de mensagens e não o conteúdo delas, já que todas elas são iguais e possuem o mesmo tamanho. Assim, o tamanho de todas as mensagens é de 2136 bytes. A Figura 3.3 mostra as mensagens de resposta do simulador de refletor, que possui o endereço *IP* 192.168.0.36.

Capítulo 4

Testes e Resultados

Esse capítulo é responsável por apresentar os testes e os resultados obtidos por meio do uso da ferramenta de ataque baseado no protocolo SSDP. Esse capítulo está dividido nas seguintes partes:

- Especificação do equipamento utilizado.
- Análise de desempenho na criação de pacotes.
- Análise de desempenho entre os dispositivos.

Um fato importante e que deve ser mencionado é que nas partes de Análise de desempenho, tanto da criação de pacotes, como no desempenho entere os dispositivos, foi utilizado o programa *Wireshark* que é um *Sniffer* de rede, ou seja, é um programa desenvolvido para escutar a rede e capturar os pacotes que foram transmitidos em uma rede.

4.1 Especificação do equipamento utilizado

Para a realização desse trabalho, foi necessário montar uma rede para realizar o ataque. Os dispositivos utilizados foram:
Macbook Pro.

- Função: Computador atacante. Executa o programa em linguagem C que gera os pacotes para o ataque.
- Sistema Operacional: macOS Mojave versão 10.14.1
- Processador: 2,30 GHz Intel Core i7
- Memória RAM: 16 GB 1600 MHz DDR3

- HD: SSD 512 GB
- Apple 57762-A0. Velocidade 100Mbits.

Computador Dell Inspiron 14 620.

- Função: Utilzado como dispositivo refletor. Executa um programa em linguagem C que simula um dispositivo refletor, ou seja, recebe as mensagens geradas pelo Macbook Pro e retorna as várias mensagens de resposta, amplificando o ataque.
- Sistema Operacional: Ubuntu 18.04 LTS
- Processador: 2,53 GHz Intel Core i3
- Memória RAM: 4 GB 1333 MHz DDR3
- HD: 80 GB
- Adaptador de rede: AR8152 v2.0 Fast Ethernet. Velocidade 100Mbits.

Computador MSI Leopard.

- Função: Computador alvo. Recebe todos os pacotes gerados pelos refletores.
- Sistema Operacional: Windows 10
- Processador: Intel Core i5 2,90 GHz
- Memória RAM: 8 GB 1600 MHz DDR3
- HD: 1 TB
- Adaptador de rede: Killer E2200 Gigabit Ethernet Controller.

Roteador NET.

- Função: Dispositivo controlado. É utilizado com um dispositivo refletor do ataque.

Switch Multilaser 8 portas 100MB.

- Função: Utilizado apenas para montar a rede.

Roteador D-Link Dir 900L 100MB.

- Função: Utilizado para montar a rede apenas no terceiro teste de ataque.

4.2 Análise de desempenho na criação de pacotes

A realização do ataque depende de três variáveis de configuração, sendo elas a intensidade do ataque, o tempo ataque e o dispositivo alvo do ataque.

A intensidade do ataque é a variável responsável pela quantidade de pacotes que serão criados pelo dispositivo atacante, dessa forma, foi mencionado na seção 3.2.4 a utilização de threads no programa para melhorar o envio dos pacotes para os refletores.

A Figura 4.1 mostra a quantidade de pacotes gerados a cada nível de intensidade por quantidade de threads utilizadas no programa. Sendo que o tempo total de ataque foi de 30 segundos.

		Quantidade de pacotes enviados em todo o ataque							
L	Quantidade teórica de pacotes	1 Thread	2 Threads	3 Threads	4 Threads	5 Threads	6 Threads	7 Threads	8 Threads
1	30	30	30	30	32	30	30	35	32
2	300	290	296	297	300	300	300	301	304
3	3.000	2.458	2.524	2.689	2.740	2.800	2.826	2.849	2.869
4	30.000	22.804	22.817	23.343	22.790	23.862	23.878	24.449	24.319
5	300.000	204.509	222.597	159.354	218.845	236.088	225.186	166.896	209.770
6	3.000.000	205.013	384.551	199.411	179.189	186.241	247.479	493.320	384.871
7	30.000.000	255.425	307.230	342.486	298.920	442.948	463.623	320.190	166.607
8	300.000.000	385.444	299.977	178.338	189.468	228.188	359.859	364.502	363.480

Figura 4.1: Dados sobre a quantidade de pacotes gerados por threads.

Nessa tabela é possível perceber que a partir do L(nível de intensidade) igual e maior a seis, a geração de pacotes saturou para todos os casos de uso de threads e que a maior variação da geração dos pacotes no nível de intensidade seis, por exemplo, foi de 493.320, o que corresponde apenas a 16,44% dos pacotes que deveriam ser teóricamente enviados.

A Figura 4.2 mostra a quantidade média de pacotes gerados em comparação com a quantidade de pacotes que deveriam ser teoricamente criados, confirmando que a saturação ocorreu a partir do nível de intensidade igual a seis.

L	Quantidade teórica de pacotes	Quantidade média de pacotes criados	Porcentagem
1	30	31	104%
2	300	299	100%
3	3.000	2.719	91%
4	30.000	23.533	78%
5	300.000	205.406	68%
6	3.000.000	285.009	10%
7	30.000.000	324.679	1%
8	300.000.000	296.157	0%

Figura 4.2: Dados sobre a quantidade média de pacotes gerados e a relação com a quantidade que teóricamente deveria ser criada.

Também percebe-se que o computador atacante conseguiu gerar os pacotes com uma taxa próxima ou acima dos 70% para os níveis de intensidade de 1 a 5. A queda na taxa

de criação dos pacotes ocorre devido ao grande número de pacotes a serem gerados, o que indica que o potencial do ataque não está na quantidade de pacotes gerados pelo computador atacante.

A Figura 4.2 mostra uma tabela com os dados médios. Esses dados foram obtidos após o a realização de ataques testes com a mudança de várias threads na geração dos pacotes. As tabelas abaixo mostram os dados obtidos nos testes realizados para que se obtesse esses valores de média.

Tabela 4.1: Teste com refletor 1 e 1 thread.

L	**Pacotes Recebidos**	**Pacotes Enviados**	**Bytes Enviados**
1	30	240	64.000
2	290	2.320	619.000
3	2.458	19.664	5.250.000
4	22.804	167.104	48.000.000
5	204.509	22.513	6.010.000
6	205.013	109.810	29.000.000
7	255.425	25.092	6.699.000
8	385.444	26.334	7.031.000

Tabela 4.2: Teste com refletor 1 e 2 threads.

L	**Pacotes Recebidos**	**Pacotes Enviados**	**Bytes Enviados**
1	30	240	64.000
2	296	2.368	632.000
3	2.524	20.192	5.591.000
4	22.817	168.144	44.000.000
5	222.597	17.020	4.544.000
6	384.551	25.120	6.707.000
7	307.230	20.964	5.597.000
8	299.977	27.292	7.286.600

Tabela 4.3: Teste com refletor 1 e 3 threads.

L	Pacotes Recebidos	Pacotes Enviados	Bytes Enviados
1	30	240	64.000
2	297	2.368	632.000
3	2.689	21.472	5.733.000
4	23.343	185.664	49.000.000
5	159.354	145.127	38.000.000
6	199.411	149.910	40.000.000
7	342.486	7.611	2.032.000
8	178.338	137.597	36.000.000

Tabela 4.4: Teste com refletor 1 e 4 threads.

L	Pacotes Recebidos	Pacotes Enviados	Bytes Enviados
1	32	256	68.000
2	300	2.392	638.000
3	2.740	14.976	3.998.000
4	22.790	188.040	50.000.000
5	218.845	44.269	11.000.000
6	179.189	147.765	38.000.000
7	298.920	13.613	3.634.000
8	189.468	156.609	41.000.000

Tabela 4.5: Teste com refletor 1 e 5 threads.

L	Pacotes Recebidos	Pacotes Enviados	Bytes Enviados
1	30	240	64.000
2	300	2.368	632.000
3	2.800	22.296	5.953.000
4	23.862	190.024	50.000.000
5	236.088	24.870	6.640.000
6	186.241	145.118	38.000.000
7	442.948	191.554	51.000.000
8	228.188	166.543	44.000.000

Tabela 4.6: Teste com refletor 1 e 6 threads.

L	Pacotes Recebidos	Pacotes Enviados	Bytes Enviados
1	30	241	64.000
2	300	2.376	634.000
3	2.826	22.480	6.002.000
4	23.878	190.080	50.000.000
5	225.186	24.794	6.619.000
6	247.479	148.779	39.000.000
7	463.623	15.338	4.095.000
8	359.859	176.740	47.000.000

Tabela 4.7: Teste com refletor 1 e 7 threads.

L	Pacotes Recebidos	Pacotes Enviados	Bytes Enviados
1	35	299	76.000
2	301	2.709	697.000
3	2.849	25.521	6.566.000
4	24.449	193.776	51.000.000
5	166.896	154.753	41.000.000
6	493.320	7.302	1.949.000
7	320.190	166.334	44.000.000
8	364.502	2.382	635.000

Tabela 4.8: Teste com refletor 1 e 8 threads.

L	Pacotes Recebidos	Pacotes Enviados	Bytes Enviados
1	32	288	74.000
2	304	2.712	697.000
3	2.869	22.744	6.072.000
4	24.319	193.536	51.000.000
5	209.770	20.414	5.450.000
6	384.871	175.225	46.000.000
7	166.607	129.902	34.000.000
8	363.480	190.592	50.000.000

4.3 Análise de desempenho entre o computador atacante e os refletores

Essa análise visa a entender o comportamento dos refletores utilizados na rede, tentando identificar se ocorreu a amplificação esperada, quando que eles saturaram e se o ataque foi bem sucedido.

A Figura 4.3 mostra as informações relacionadas aos pacotes de entrada e saída dos refletores. Esses dados foram obtidos a partir de um segundo teste com os dois refletores, sendo que o ataque permaneceu com o tempo de 30 segundos.

L	Total de pacotes recebidos	Total pacotes enviados refletor 1	Total pacotes enviados refletor 2	Relação saída/entrada refletor 1	Relação saída/entrada refletor 2
1	31	244	306	98,39%	98,71%
2	299	2378	2.989	99,41%	99,97%
3	2.728	22.293	26.601	102,15%	97,51%
4	23.551	167.652	145.114	88,98%	61,62%
5	149.845	36.493	111.765	3,04%	7,46%
6	173.181	119.614	100.951	8,63%	5,83%
7	203.727	122.313	105.902	7,50%	5,20%
8	181.177	104.495	96.713	7,21%	5,34%

Figura 4.3: Dados sobre a quantidade de pacotes entrando e saidno dos refletores.

Pode-se perceber que para os níveis de um a três, a resposta de ambos os refletores foram próxima a 100%, ou seja, os refletores não tiveram problemas em gerar os pacotes e enviá-los ao dispositivo alvo. Já para o nível quatro, houve uma pequena queda no refletor 1, que é o computador Linux que simula um dispositivo refletor, e houve uma expressiva queda no refletor 2, que é o Modem da NET.

Verifica-se ainda que a partir do nível cinco iniciou a saturação. Pode-se afirmar isso porque a quantidade de pacotes que foram enviados para o refletor foi maior que a quantidade de pacotes que sairam do refletor e que, ainda, não houve crescimento na quantidade de pacotes enviados pelos refletores quando foi aumentado o nível de intensidade do ataque.

A quinta e a sexta coluna da tabela mostram justamente a quantidade de pacotes refletidos em porcentagem, dessa forma, por exemplo, para o nível de intensidade igual a 4, o refletor 1 amplificou cerca de 88,98% dos pacotes e o refletor 2 amplificou cerca de 61,62% dos pacotes.

A Figura 4.3 mostra dados médios que foram obtidos por meio da realização de vários ataques testes com variação do número de *threads*. Esses dados podem ser observados nas tabelas abaixo.

Tabela 4.9: Teste com os 2 refletores e 1 thread.

		Refletor 1		**Refletor 2**	
L	**Pacotes Recebidos**	**Pacotes Enviados**	**Bytes Enviados**	**Pacotes Enviados**	**Bytes Enviados**
1	30	240	64000	300	103.000
2	291	2.321	619.000	2.911	993.000
3	2.466	19.633	5.241.000	24.541	8.370.000
4	22.709	172.977	46.000.000	131.814	44.000.000
5	126.764	82.517	22.000.000	96.934	33.000.000
6	255.931	176.834	47.000.000	116.078	39.000.000
7	304.426	175.341	46.000.000	118.663	40.000.000
8	136.404	116.473	31.000.000	98.242	33.000.000

Tabela 4.10: Teste com os 2 refletores e 2 threads.

		Refletor 1		**Refletor 2**	
L	**Pacotes Recebidos**	**Pacotes Enviados**	**Bytes Enviados**	**Pacotes Enviados**	**Bytes Enviados**
1	30	240	64.000	300	103.000
2	297	2.368	632.000	2.970	1.013.000
3	2.526	25.260	8.616.000	20.192	5.391.000
4	22.761	167.221	44.000.000	145.424	49.000.000
5	159.211	19.776	5.280.000	109.076	37.000.000
6	196.265	146.038	38.000.000	103.281	35.000.000
7	155.114	124.572	33.000.000	97.653	33.000.000
8	117.592	101.714	27.000.000	90.012	30.000.000

Tabela 4.11: Teste com os 2 refletores e 3 threads.

		Refletor 1		**Refletor 2**	
L	**Pacotes Recebidos**	**Pacotes Enviados**	**Bytes Enviados**	**Pacotes Enviados**	**Bytes Enviados**
1	30	241	64.000	301	103.000
2	297	2.376	634.000	2.980	1.016.000
3	2.688	21.456	5.728.000	26.890	9.172.000
4	23.364	167.366	44.000.000	145.603	49.000.000
5	156.000	19.121	5.105.000	116.502	39.000.000
6	172.322	141.441	37.000.000	10.0040	34.000.000
7	148.544	50.887	13.000.000	99.213	33.000.000
8	256.092	159.519	42.000.000	113.765	38.000.000

Tabela 4.12: Teste com os 2 refletores e 4 threads.

		Refletor 1		**Refletor 2**	
L	**Pacotes Recebidos**	**Pacotes Enviados**	**Bytes Enviados**	**Pacotes Enviados**	**Bytes Enviados**
1	32	256	68.000	330	113.000
2	300	2.392	638.000	3.000	1.027.000
3	2.773	21.812	5.823.000	27.641	9.428.000
4	23.691	171.308	45.000.000	142.277	48.000.000
5	141.775	82.729	22.000.000	102.978	35.000.000
6	127.311	73.889	19.000.000	90.323	30.000.000
7	242.905	149.596	39.000.000	115.877	39.000.000
8	151.361	94.244	25.000.000	100.431	34.000.000

Tabela 4.13: Teste com os 2 refletores e 5 threads.

		Refletor 1		**Refletor 2**	
L	**Pacotes Recebidos**	**Pacotes Enviados**	**Bytes Enviados**	**Pacotes Enviados**	**Bytes Enviados**
1	30	240	64.000	300	103.000
2	300	2.390	638.000	3.000	1.027.000
3	2.804	22.232	5.935.000	27.950	9.534.000
4	23.765	153.723	41.000.000	134.162	45.000.000
5	156.053	24.609	6.570.000	117.674	40.000.000
6	136.574	94.441	25.000.000	95.680	32.000.000
7	301.379	174.316	46.000.000	118.623	40.000.000
8	187.099	102.542	27.000.000	101.599	34.000.000

Tabela 4.14: Teste com os 2 refletores e 6 threads.

		Refletor 1		**Refletor 2**	
L	**Pacotes Recebidos**	**Pacotes Enviados**	**Bytes Enviados**	**Pacotes Enviados**	**Bytes Enviados**
1	30	240	64.000	300	103.000
2	298	2.390	638.000	3.000	1.027.000
3	2.845	22.560	6.023.000	28.470	9.712.000
4	23.950	168.466	44.000.000	157.649	53.000.000
5	160.837	22.341	5.964.000	115.129	39.000.000
6	176.273	128.552	34.000.000	103.444	35.000.000
7	217.620	142.399	38.000.000	108.044	36.000.000
8	342.786	112.837	30.000.000	78.199	26.000.000

Tabela 4.15: Teste com os 2 refletores e 7 threads.

		Refletor 1		**Refletor 2**	
L	**Pacotes Recebidos**	**Pacotes Enviados**	**Bytes Enviados**	**Pacotes Enviados**	**Bytes Enviados**
1	30	240	64.000	300	103.000
2	300	2.390	638.000	3.000	1.027.000
3	2.862	22.792	6.085.000	28.630	9.766.000
4	24.223	170.586	45.000.000	144.471	49.000.000
5	157.538	23.255	6.208.000	117.911	40.000.000
6	153.393	102.242	27.000.000	96.047	32.000.000
7	126.540	84.717	22.000.000	92.071	31.000.000
8	120.487	61.783	16.000.000	95.752	32.000.000

Tabela 4.16: Teste com os 2 refletores e 8 threads.

		Refletor 1		**Refletor 2**	
L	**Pacotes Recebidos**	**Pacotes Enviados**	**Bytes Enviados**	**Pacotes Enviados**	**Bytes Enviados**
1	32	256	68.000	320	110.000
2	306	2.400	640.000	3.050	1.040.000
3	2.858	22.600	6.034.000	28.490	9.718.000
4	23.943	169.572	45.000.000	159.508	54.000.000
5	140.583	17.596	4.697.000	117.913	40.000.000
6	167.382	93.471	24.000.000	102.711	35.000.000
7	133.284	76.678	20.000.000	97.072	33.000.000
8	137.594	86.844	23.000.000	95.704	32.000.000

Já em relação a amplificação, os dois refletores refletores possuiram resultados de amplificação diferentes, a Figura 4.4 mostra a quantidade de bytes recebidos e enviados para cada refletor, além de apresentar a amplificação que ocorre para cada refletor nos oito níveis de intensidade.

L	Total de bytes recebidos	Total de bytes enviados Refletor 1	Total de bytes enviados Refletor 2	Ampl. Refletor 1	Ampl. Refletor 2	Ampl. Total
1	5.056	61.584	100.841	12,18	19,94	32,13
2	49.567	601.333	979.404	12,13	19,76	31,89
3	452.803	5.873.523	8.513.961	12,97	18,80	31,77
4	3.909.421	41.902.872	46.843.404	10,72	11,98	22,70
5	24.874.293	9.217.098	36.310.290	0,37	1,46	1,83
6	28.748.114	29.700.404	32.586.686	1,03	1,13	2,17
7	33.818.592	30.412.618	34.142.372	0,90	1,01	1,91
8	30.075.360	26.162.070	31.021.018	0,87	1,03	1,90

Figura 4.4: Dados sobre a quantidade de pacotes entrando e saindo nos refletores.

Importante notar que para o cálculo da amplificação, foram retirados os bytes relacionados ao cabeçalho ethernet, dessa forma, o cálculo da amplificação foi feito da seguinte forma:

$$Amp = (Benv - Penv * 14)/(Brec - Prec * 14)$$

Onde:
Amp é a amplificação do refletor
Benv é a quantidade de bytes enviados
Penv é a quantidade de pacotes enviados pelo refletor
Brec é a quantidade de bytes recebidos pelo refletor
Prec é a quantidade de pacotes recebidos pelo refletor
Por exemplo, para o refletor 1 e para o nível de intensidade igual a um, temos:

$$Amp = (65000 - 244 * 14)/(5490 - 31 * 14)$$

$$Amp = (65000 - 244 * 14)/(5490 - 31 * 14)$$

$$Amp = 61584/5056$$

$$Amp = 12,18$$

Dessa forma, pode-se observar a amplificação de cada nível na Figura 4.4. É importante notar que as melhores amplificações foram nos níveis de intensidade iniciais, porém o volume de ataque é baixo. No nível 4, ocorre o ataque na melhor configuração possível, pois foi gerado aproximadamente 4 MB de pacotes para os refletores e eles geraram um ataque de quase 90 MB.

O gráfico que é mostrado na Figura 4.5 apresenta essas mesmas informações de forma gráfica, no qual é fácil perceber a diferença entre o tamanho dos pacotes enviados e recebidos pelos refletores.

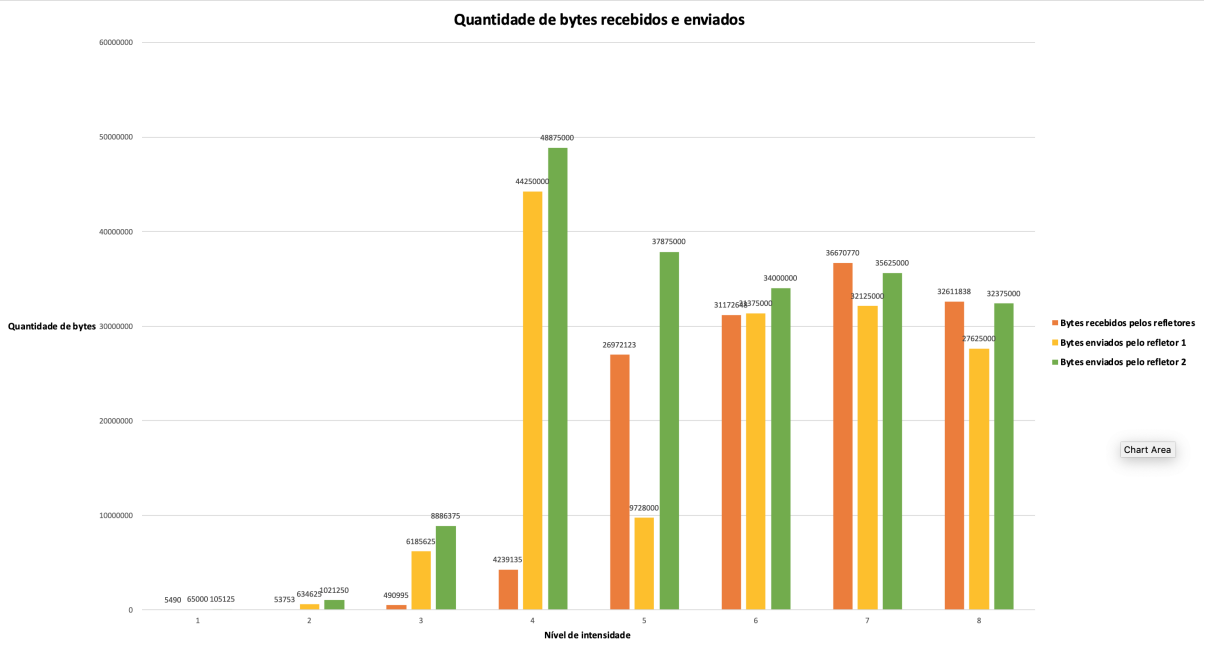

Figura 4.5: Gráfico mostrando quantidade de bytes recebidos e enviados pelos refletores.

Já o gráfico mostrado na Figura 4.6 apresenta a saturação no envio dos pacotes para os dispositivos em estudo.

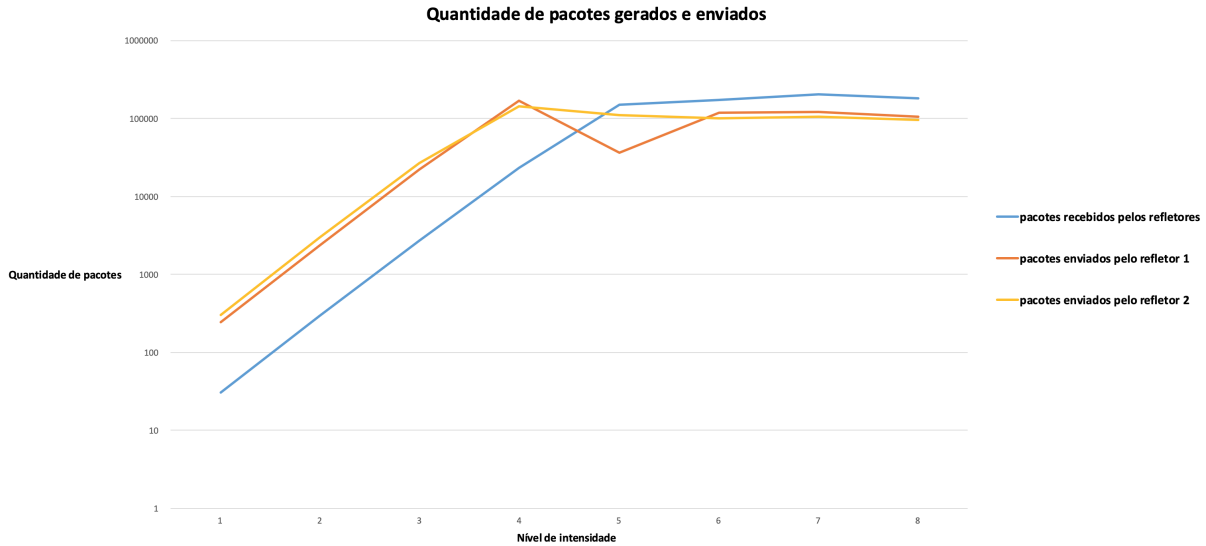

Figura 4.6: Gráfico da saturação dos dispositivos envolvidos no ataque.

Um outro fato importante de ser analisado é o comportamento da rede em relação a transmissão dos pacotes. A Figura 4.7 mostra o total de bytes que foram transmitidos na rede por meio dos dispositivos utilizados.

L	Total de bytes recebidos	Total de bytes enviados Refletor 1	Total de bytes enviados Refletor 2	Total de bytes na rede.
1	5.490	65.000	105.125	175.615
2	53.753	634.625	1.021.250	1.709.623
3	490.995	6.185.625	8.886.375	15.562.995
4	4.239.135	44.250.000	48.875.000	97.364.135
5	26.972.123	9.728.000	37.875.000	74.575.123
6	31.172.648	31.375.000	34.000.000	96.547.648
7	36.670.770	32.125.000	35.625.000	104.420.770
8	32.611.838	27.625.000	32.375.000	92.611.838

Figura 4.7: Tabela com o total de bytes transmitidos no ataque durante 30 segundos.

Um fato importante de se observar é que o total de bytes transmitidos na rede ficou limitado a aproximadamente 100 MB, fato que pode ser justificado pela limitação do switch utilizado. Com essa limitação, os pacotes enviados pelo dispositivo atacante ocupou a largura de banda da rede, já que o hardware do dispositivo atacante é melhor e consegue gerar mais pacotes para o ataque.

Já para realizar a análise da saturação do dispositivo alvo, houve uma mudança na estrutura da rede, foi necessário adicionar um novo roteador para que o dispositivo alvo ficasse em uma rede separada dos outros dispositivos. Isso ocorreu porque o dispositivo atacante envia mensagens *multicast* para os refletores e essas mensagens também estavam sendo enviadas para o dispositivo alvo, dessa forma, foi necessario separá-lo. Dessa forma, foi utilizado um roteador D-link dir 900L.

Para que as mensagens de ataque fossem direcionadas ao dispositivo alvo, foi necessário realizar o *port foward* no roteador D-Link, dessa forma, todos as mensagens que o roteador receber na porta 1900, o roteador enviará para o dispositivo alvo.

Agora com essa configuração, foi possível obter as informações do ataque para entender a saturação do dispositivo alvo. A Figura 4.8 possui as informações sobre a transmissão dos pacotes dos refletores para o dispositivo alvo.

	Teste de ataque 3 - Análise sobre dispositivo alvo. Ataque de 60 segundos.					
L	Total de pacotes enviados Refletor 1	Total de bytes enviados Refletor 1	Total de pacotes enviados Refletor 2	Total de bytes enviados Refletor 2	Total de pacotes recebidos alvo.	Total de bytes recebidos alvo.
1	480	128.000	600	204.000	1.229	364.000
2	4.600	1.228.000	5.750	1.961.000	10.566	3.262.000
3	38.216	10.000.000	47.630	16.000.000	85.952	26.000.000
4	134.751	35.000.000	245.310	83.000.000	385.700	124.000.000
5	49.089	13.000.000	160.381	54.000.000	209.766	67.000.000
6	46.148	12.000.000	71.619	24.000.000	117.988	36.000.000
7	50.070	13.000.000	80.548	27.000.000	132.594	42.000.000
8	43.713	11.000.000	75.650	25.000.000	120.115	37.000.000

Figura 4.8: Tabela com o total de bytes transmitidos no ataque durante 60 segundos.

A Figura 4.9 mostra a um gráfico com a quantidade de bytes que foram enviados no ataque. Pode-se perceber que houve uma saturação na quantidade de bytes enviados para os níveis de intensidade iguais e maiores que cinco, sendo que a partir do nível 4 o

dispositivo alvo saturou. Dessa forma, mesmo com a saturação dos refletores, o dispositivo alvo continuou sem acesso a internet.

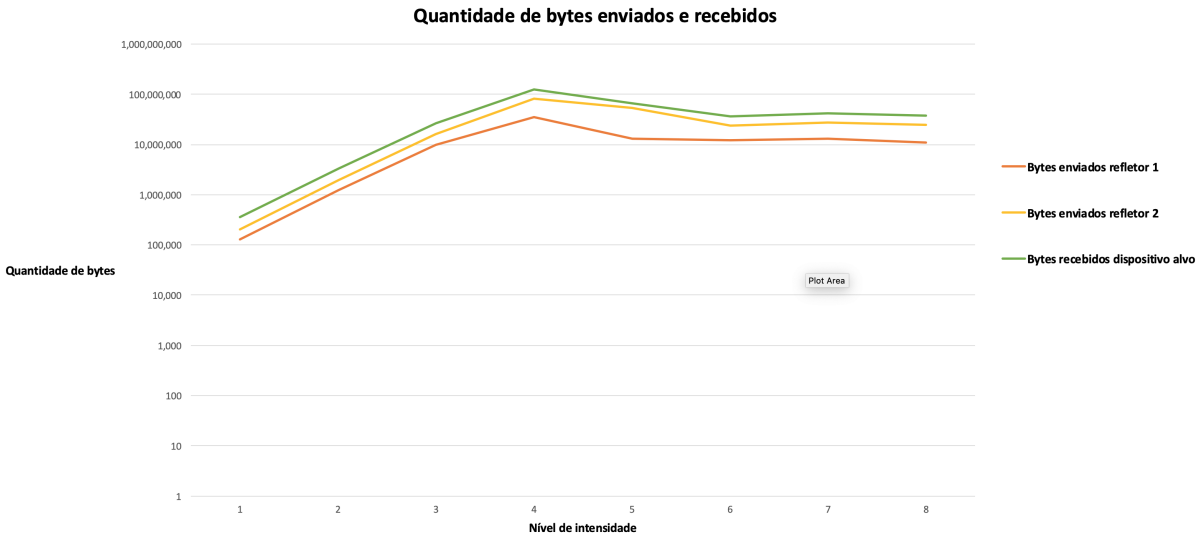

Figura 4.9: Tabela com o total de bytes transmitidos no ataque durante 60 segundos.

Capítulo 5

Conclusão e Trabalhos futuros

5.1 Conclusão

Essa monografia mostrou as características de um ataque de negação de serviço por reflexão explorando o protocolo SSDP, que é um protocolo utilizado para descoberta de serviços de dispositivos que estão presentes na rede.

Na parte inicial, foi apresentado o funcionamento de um ataque *DDoS*, o conceito de amplificação de ataque, as informações necessárias para entender o protocolo SSDP e colo explorar o protocolo de forma que fosse realizado o ataque proposto.

Em seguida, foi explicado a ferramenta de ataque que foi criada. A ferramenta possui módulos, sendo eles a pesquisa na rede por dispositivos que utilizam o protocolo SSDP, a configuração do ataque, a montagem do pacote para o ataque e o último módulo que é responsável por realizar o ataque.

Na discussão da ferramenta que foi desenvolvida em linguagem de programação C, foi questionado quantas threads seriam criadas para o envio de pacotes de forma que deixaria o ataque o mais eficaz possível. Após a análise da utilização de até oito threads para o envio dos pacotes pela ferramenta, foi concluido que o número de threads utilizado para o envio dos pacotes não influenciou de forma efetiva, sendo que a maior diferença no envio de pacotes ficou próximo de 10%, o que não impactou no estudo do ataque como um tudo.

Já com a ferramenta em funcionamento, foi necessário entender o comportamento do refletor que seria utilizado no ataque. Inicialmente, foi utilizado apenas um refletor para a amplificação do ataque, sendo que esse refletor era o Modem da NET. Dessa forma, foram realizados vários testes para entender o comportamento do refletor. Esses testes mostraram a quantidade de amplificação do refletor e o ponto de saturação desse refletor, porém a realização do teste com apenas um refletor não foi o suficiente para indicar todas as características do ataque *DDoS*. Lembrando que as informações sobre esse teste de ataque foram obtidas pelo *Wireshark* e podem ser observadas nas Tabelas 4.1 a 4.8.

A partir desse questionamento, foi criado um outro programa feito em linguagem C, no qual simula um dispositivo refletor. Dessa forma, foi utilizado dois dispositivos refletores para a realização do ataque. Já com dois refletores, foi realizado outra séries de ataques, para obter o comportamento deles, a Figura 4.3 mostra os dados obtidos por meio dos testes realizados. Lembrando que as informações completas sobre esse teste de ataque foram obtidas pelo *Wireshark* e podem ser observadas nas Tabelas 4.9 a 4.16.

A Figura 4.4 mostra a amplificação desses dois refletores. Se comparar com a amplificação do protocolo SSDP que foi mostrada na Figura 2.2, temos que a amplificação foi muito menor que o mostrado, porém não se sabe quais foram as condições do ataque que foi realizado para se obter uma amplificação de 30,8.

Tendo em vista a participação de dois refletores para o ataque e uma amplificação total do ataque com o valor entre 22,7 e 32,13 para valores de intensidade de no máximo 4, foi concluído que toda a dinâmica do ataque estava sendo estudada.

Foi observado que os refletores saturaram antes do computador atacante, fato que poder ser justificado pelo hardware utilizado. O computador atacante foi saturar a geração de pacotes para níveis de intensidade iguais a 6 ou maiores, já os refletores saturaram com níveis de itensidade iguais a 5 ou maiores.

Já o melhor comportamento do ataque foi quando o nível de intensidade de envio dos pacotes foi igual a 4. A Figura 4.5 mostra a quantidade de bytes que cada refletor recebei e a quantidade de bytes que cada refletor enviou e foi possível obter um ataque de aproximadamente 97 MB por meio do envio de 4 MB para os dois refletores da rede.

Por fim, para uma análise da saturação do computador alvo, foi necessário realizar algumas mudanças na estrutura da rede, já que as mensagens *multicast* que eram enviadas para os refletores também estavam chegando no dispositivo alvo e consumindo processamento. A mudança estrutural na rede foi feita por meio do acréscimo de um roteador de forma que o dispositivo alvo ficasse isolado e não escutasse as mensagens que o computador atacante gerava.

As informações desse novo ataque podem ser observadas na Figura 4.8 e Figura 4.9, sendo que a quantidade de bytes que chegaram no dispositivo alvo é a soma dos bytes que foram enviados pelos refletores e pelo novo roteador utilizado para isolar o computador alvo, porém a quantidade de bytes que o novo roteador envia para o dispositivo alvo não influencia o ataque, muito menos a saturação do computador alvo.

5.2 Trabalhos Futuros

Os seguintes trabalhos futuros são propostos:

- Integração dessa ferramenta com o projeto *Linderhof* do professor João Gondim.
- Melhorias no código de forma a otimizar a criação de pacotes.
- Testes com uma maior quantidade de refletores para compreender o ataque em uma escala maior.
- Realizar o ataque incremental.
- Realizar análise da rede para entender quantos refletores possuem e possível fator e amplificação.

Referências

[1] Shaw, Keith: *Keycdn*. 2018. https://www.keycdn.com/support/ddos-attack. viii, 6

[2] Forum, UPnP: *Upnptm device architecture 1.1*. 2008. http://www.upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf. viii, 7, 8, 10, 11, 13, 14, 15, 16, 17

[3] Foltz, Mark: *Open screen protocol*. 2018. https://webscreens.github.io/openscreenprotocol/ssdp.html. viii, 17, 18

[4] CloudFlare: *Denial-of-service (dos)*. https://www.cloudflare.com/learning/ddos/glossary/denial-of-service/. 1, 4

[5] CloudFlare: *What is a ddos attack?* https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/. 1, 5

[6] Imperva: *Ddos attack*. 2018. https://www.incapsula.com/ddos/ddos-attacks.html. 5

[7] Shaw, Keith: *The osi model explained: How to understand (and remember) the 7 layer network model*. 2018. https://www.networkworld.com/article/3239677/lan-wan/the-osi-model-explained-how-to-understand-and-remember-the-7-layer-network-model.html. 5

[8] CloudFlare: *Ip spoofing*. https://www.cloudflare.com/learning/ddos/glossary/ip-spoofing/. 6

[9] Shin, Donald: *How to defend against amplified reflection ddos attacks*. 2018. https://www.a10networks.com/resources/articles/how-defend-against-amplified-reflection-ddos-attacks. 7

[10] Force, Internet Engineering Task: *Simple service discovery protocol/1.0 operating without an arbiter*. 1999. https://tools.ietf.org/html/draft-cai-ssdp-v1-03. 9, 10

[11] CloudFlare: *Ssdp ddos attack*. https://www.cloudflare.com/learning/ddos/ssdp-ddos-attack/. 18