DISSERTAÇÃO DE MESTRADO PROFISSIONAL

Análise de Saturação de Dispositivos IoT Atuando como Refletores em Ataques Distribuídos de Negação de Serviço por Reflexão Amplificada

Alan Tamer Vasques

Brasília, 08 de outubro de 2020

Programa de Pós-Graduação Profissional em Engenharia Elétrica

DEPARTAMENTO DE ENGENHARIA ELÉTRICA
FACULDADE DE TECNOLOGIA
UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia
Departamento de Engenharia Elétrica

Saturation Analysis of IoT Devices Acting as Reflectors on Amplified Reflection Distributed Denial of Service Attacks

Análise de Saturação de Dispositivos IoT Atuando como Refletores em Ataques Distribuídos de Negação de Serviço por Reflexão Amplificada

ALAN TAMER VASQUES

ORIENTADOR: DR. JOÃO JOSÉ COSTA GONDIM

DISSERTAÇÃO DE MESTRADO PROFISSIONAL EM
ENGENHARIA ELÉTRICA

PUBLICAÇÃO: PPEE.MP.005
BRASÍLIA/DF: OUTUBRO - 2020

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia
Departamento de Engenharia Elétrica

DISSERTAÇÃO DE MESTRADO PROFISSIONAL

Análise de Saturação de Dispositivos IoT Atuando como Refletores em Ataques Distribuídos de Negação de Serviço por Reflexão Amplificada

Alan Tamer Vasques

Dissertação de Mestrado Profissional submetida ao Departamento de Engenharia Elétrica como requisito parcial para obtenção do grau de Mestre em Engenharia Elétrica

Banca Examinadora

Prof. João José Costa Gondim, Dr., CIC/UnB
Orientador

Prof. Robson de Oliveira Albuquerque, Dr., FT/UnB
Examinador Interno

Profa. Ana Lucila Sandoval Orozco, Dra., Universidad Complutense de Madrid
Examinador Externo

Prof. Georges Daniel Amvame Nze, Dr., FT/UnB
Suplente

FICHA CATALOGRÁFICA

VASQUES, ALAN TAMER
Análise de Saturação de Dispositivos IoT Atuando como Refletores em Ataques Distribuídos de Negação de Serviço por Reflexão Amplificada [Distrito Federal] 2020.
xvi, 100 p., 210 x 297 mm (ENE/FT/UnB, Mestre, Engenharia Elétrica, 2020).
Dissertação de Mestrado Profissional - Universidade de Brasília, Faculdade de Tecnologia.
Departamento de Engenharia Elétrica

1. Negação de Serviço	2. Internet das Coisas
3. Amplificação	4. Reflexão
I. ENE/FT/UnB	II. Mestre

REFERÊNCIA BIBLIOGRÁFICA

VASQUES, A.T. (2020). *Análise de Saturação de Dispositivos IoT Atuando como Refletores em Ataques Distribuídos de Negação de Serviço por Reflexão Amplificada*. Dissertação de Mestrado Profissional, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 100 p.

CESSÃO DE DIREITOS

AUTOR: Alan Tamer Vasques
TÍTULO: Análise de Saturação de Dispositivos IoT Atuando como Refletores em Ataques Distribuídos de Negação de Serviço por Reflexão Amplificada.
GRAU: Mestre em Engenharia Elétrica ANO: 2020

Alan Tamer Vasques
Depto. de Engenharia Elétrica (ENE) - FT
Universidade de Brasília (UnB)
Campus Darcy Ribeiro
CEP 70919-970 - Brasília - DF - Brasil

DEDICATÓRIA

Dedico este trabalho aos meus amados filhos, Bruna e Daniel, que são a razão pela qual me fazem ser cada vez mais uma pessoa melhor.

AGRADECIMENTOS

Gostaria de agradecer a todos os que contribuíram de alguma forma para a realização desta dissertação, em especial:

Ao meu orientador, Prof. Dr. João Gondim, por ter aceitado e acreditado acompanhar-me nesta pesquisa. A sua orientação e suporte foram essenciais para a minha motivação à medida que os obstáculos surgiam ao longo da jornada.

À minha família e à Cyntia Adames, por todas as palavras de incentivo e motivação nos momentos em que precisei.

À Tayana Neves, pelo incentivo, compreensão e ajuda com as crianças nas inúmeras horas que foram necessárias para que este objetivo fosse concretizado.

Aos estimados docentes do PPEE, com os quais tive a oportunidade de obter valiosos ensinamentos para a elaboração deste trabalho.

Aos meus colegas de curso, pela parceria e conhecimentos compartilhados ao longo das diversas discussões durante as aulas e fora delas.

Agradeço ainda aos professores Dr. Antônio Jorge Gomes Abelém e Dr. Luiz Fernando Sirotheau Serique Júnior pelas cartas de recomendação durante o processo seletivo, que abriram as portas para que essa pesquisa fosse realizada.

RESUMO

Título: Análise de Saturação de Dispositivos IoT Atuando como Refletores em Ataques Distribuídos de Negação de Serviço por Reflexão Amplificada
Autor: Alan Tamer Vasques
Orientador: Dr. João José Costa Gondim
Programa de Pós-Graduação Profissional em Engenharia Elétrica – Área de Concentração em Segurança Cibernética
Brasília, 08 de outubro de 2020.

No contexto dos ataques distribuídos de negação de serviço (DDoS), os ataques por reflexão amplificada (AR-DDoS) representam uma tendência que vem se intensificando ao longo dos últimos anos, com volumes de tráfego cada vez maiores. Isso se deve, em parte, à crescente utilização de dispositivos da Internet das Coisas (IoT) nestes ataques, principalmente devido à ampla superfície de ataque que tais dispositivos proporcionam. Com esta motivação, foram executados diversos ataques AR-DDoS com três dispositivos IoT típicos (*gateway* ADSL, câmera IP e Raspberry Pi) atuando como refletores, em ambiente controlado, explorando três protocolos comumente encontrados na Internet das Coisas – SSDP, SNMP e CoAP – , sendo o último uma tendência recente, sobre IPv4 e IPv6 (quando possível), de forma a se avaliar a saturação desses equipamentos e as taxas máximas de amplificação dos ataques em curso. Os resultados obtidos são consistentes com estudos anteriores envolvendo equipamentos convencionais e caracterizam a saturação dos refletores para baixas taxas de injeção de pacotes.

Palavras-chave: Negação de Serviço, Internet das Coisas, Amplificação, Reflexão.

ABSTRACT

Title: Saturation Analysis of IoT Devices Acting as Reflectors on Amplified Reflection Distributed Denial of Service Attacks
Author: Alan Tamer Vasques
Supervisor: Dr. João José Costa Gondim
Professional Post-Graduate Program in Electrical Engineering – Cybersecurity Concentration Area
Brasília, October 8th, 2020.

In the context of distributed denial of service (DDoS) attacks, those which use amplified reflection (AR-DDoS) represent a trend that has been intensifying over the past few years, with increasing volumes of traffic. This happens, in part, due to the increasing use of Internet of Things (IoT) devices in those attacks, mainly because of the extensive attack surface that IoT devices provide. With this motivation, several AR-DDoS attacks were carried out with three typical IoT devices (ADSL gateway, IP camera and Raspberry Pi) acting as reflectors, in a controlled environment, abusing three protocols commonly found on IoT devices - SSDP , SNMP and CoAP -, the latter one a recent trend, over IPv4 and IPv6 (when possible), in order to assess their saturation behavior and the maximum amplification rates of the ongoing attacks. The results achieved are consistent with previous studies involving conventional equipment and characterize reflector saturation at low probe injection rates.

Keywords: Denial of Service, Internet of Things, Amplification, Reflection.

SUMÁRIO

Lista de Figuras ..... **x**

Lista de Tabelas ..... **xiii**

Lista de Acrônimos ..... **xv**

1 INTRODUÇÃO **1**
1.1 Justificativa 2
1.2 Objetivos 2
1.2.1 Objetivos Específicos 3
1.3 Principais Contribuições 3
1.4 Organização do Trabalho 4

2 REVISÃO BIBLIOGRÁFICA **5**
2.1 Falsificação de Endereços IP 5
2.2 Ataques de Negação de Serviço (DoS) 6
2.2.1 Ataques Distribuídos de Negação de Serviço (DDoS) 7
2.2.2 Ataques Distribuídos de Negação de Serviço por Reflexão Amplificada (AR-DDoS) 10
2.3 Anatomia dos Ataques 11
2.3.1 Simple Service Discovery Protocol (SSDP) 12
2.3.2 Simple Network Management Protocol (SNMP) 16
2.3.3 Constrained Application Protocol (CoAP) 22
2.4 Internet das Coisas (IoT) em Ataques DDoS 28
2.5 Trabalhos Correlatos 30
2.5.1 Segurança em IoT 30
2.5.2 Ataques DDoS 31
2.5.3 Ataques AR-DDoS 31
2.5.4 Ataques DDoS/AR-DDoS em Ambientes IoT 32
2.5.5 Resumo dos Trabalhos Correlatos 34
2.6 Resumo do Capítulo 37

3 MATERIAIS E MÉTODOS **38**
3.1 Ferramenta Linderhof 38
3.1.1 Arquitetura 38
3.1.2 Melhorias Desenvolvidas 44
3.2 Metodologia dos Testes 46
3.2.1 Equipamentos Utilizados 47

3.2.2 Cenários 48
3.2.3 Procedimentos de Testes 49
3.3 Resumo do Capítulo 52

4 RESULTADOS DOS TESTES 53
4.1 Resultados 53
4.1.1 Cenário 1 (*Gateway* ADSL) 53
4.1.2 Cenário 2 (Câmera IP) 60
4.1.3 Cenário 3 (Raspberry Pi) 65
4.1.4 Fatores de Amplificação 80
4.2 Resumo dos Resultados 87
4.3 Discussão 90
4.4 Resumo do Capítulo 93

5 CONCLUSÃO 94
5.1 Trabalhos Futuros 95

REFERÊNCIAS BIBLIOGRÁFICAS 96

LISTA DE FIGURAS

2.1 Quantidade de Sistemas Autônomos Suscetíveis à Falsificação de Endereços IPv4 . 6
2.2 Quantidade de Sistemas Autônomos Suscetíveis à Falsificação de Endereços IPv6 . 6
2.3 Exemplo de ataque DoS 7
2.4 Exemplo de ataque DDoS 8
2.5 Taxonomia de ataques DDoS adaptada de [1] 8
2.6 Exemplo de ataque AR-DDoS 10
2.7 Passos do UPnP 12
2.8 Ataque AR-DDoS com SSDP 13
2.9 Formato da mensagem M-SEARCH em multicast especificada na UDA 1.0..... 14
2.10 Formato da mensagem M-SEARCH em multicast especificada na UDA 1.1..... 14
2.11 Formato da mensagem M-SEARCH em multicast especificada na UDA 2.0..... 15
2.12 Formato da mensagem M-SEARCH em unicast especificada na UDA 1.1 15
2.13 Formato da mensagem M-SEARCH em unicast especificada na UDA 2.0 15
2.14 Exemplo de resposta 200 OK..... 15
2.15 Funcionamento normal do SNMP 17
2.16 Estrutura da MIB do SNMP 18
2.17 Ataque AR-DDoS com SNMP 19
2.18 Formato da mensagem GetBulkRequest 20
2.19 Exemplo de requisição GetBulkRequest..... 21
2.20 Exemplo de resposta 21
2.21 Ataque AR-DDoS com CoAP 23
2.22 Formato das mensagens do CoAP 24
2.23 Exemplo de requisição Confirmable (GET)..... 25
2.24 Exemplo de resposta Acknowledgment (2.05 Content) 26
2.25 Conteúdo do *payload* da resposta da figura 2.24 26
2.26 Exemplo de resposta Acknowledgment (2.05 Content) para bloco de 2048 bytes ... 27
2.27 Botnet de dispositivos IoT..... 28
2.28 Dispositivos IoT atuando como refletores em ataques AR-DDoS 29
2.29 Dispositivos IoT em botnets e atuando como refletores em ataques AR-DDoS..... 29

3.1 Arquitetura do Linderhof 39
3.2 Saída em tela do Linderhof 42
3.3 Pacote SNMP gerado pelo Linderhof..... 43
3.4 Cenário 1 49
3.5 Cenário 2 49
3.6 Cenário 3 49

4.1 Cenário 1 (SSDP) - IPv4 - Kbps 55

4.2 Cenário 1 (SSDP) - IPv4 - Pacotes/s 55
4.3 Cenário 1 (SSDP) - IPv4 - Kbps - Metodologia Alternativa 56
4.4 Cenário 1 (SSDP) - IPv4 - Pacotes/s - Metodologia Alternativa 57
4.5 Cenário 1 (SNMP) - IPv4 - Kbps 58
4.6 Cenário 1 (SNMP) - IPv4 - Pacotes/s 58
4.7 Cenário 1 (SNMP) - IPv4 - Kbps - Metodologia Alternativa 59
4.8 Cenário 1 (SNMP) - IPv4 - Pacotes/s - Metodologia Alternativa 60
4.9 Cenário 2 (SSDP) - IPv4 - Kbps 61
4.10 Cenário 2 (SSDP) - IPv4 - Pacotes/s 62
4.11 Cenário 2 (SNMP) - IPv4 - Kbps 63
4.12 Cenário 2 (SNMP) - IPv4 - Pacotes/s 64
4.13 Cenário 2 (SNMP) - IPv6 - Kbps 65
4.14 Cenário 2 (SNMP) - IPv6 - Pacotes/s 65
4.15 Cenário 3 (SSDP) - IPv4 - Kbps 67
4.16 Cenário 3 (SSDP) - IPv4 - Pacotes/s 67
4.17 Cenário 3 (SSDP) - IPv6 - Kbps 68
4.18 Cenário 3 (SSDP) - IPv6 - Pacotes/s 69
4.19 Cenário 3 (SNMP) - IPv4 - Kbps 71
4.20 Cenário 3 (SNMP) - IPv4 - Pacotes/s 71
4.21 Cenário 3 (SNMP) - IPv6 - Kbps 72
4.22 Cenário 3 (SNMP) - IPv6 - Pacotes/s 73
4.23 Cenário 3 (SNMP) - IPv4 - Kbps - Metodologia Alternativa 75
4.24 Cenário 3 (SNMP) - IPv4 - Pacotes/s - Metodologia Alternativa 75
4.25 Cenário 3 (SNMP) - IPv6 - Kbps - Metodologia Alternativa 76
4.26 Cenário 3 (SNMP) - IPv6 - Pacotes/s - Metodologia Alternativa 77
4.27 Cenário 3 (CoAP) - IPv4 - Kbps 78
4.28 Cenário 3 (CoAP) - IPv4 - Pacotes/s 79
4.29 Cenário 3 (CoAP) - IPv6 - Kbps 80
4.30 Cenário 3 (CoAP) - IPv6 - Pacotes/s 80
4.31 Fator de Amplificação *Bytes* (Cenário 1 - SSDP) 81
4.32 Fator de Amplificação Pacotes (Cenário 1 - SSDP) 81
4.33 Fator de Amplificação *Bytes* (Cenário 1 - SNMP) 82
4.34 Fator de Amplificação Pacotes (Cenário 1 - SNMP) 82
4.35 Fator de Amplificação *Bytes* (Cenário 1 - SSDP) - Metodologia Alternativa 83
4.36 Fator de Amplificação Pacotes (Cenário 1 - SSDP) - Metodologia Alternativa 83
4.37 Fator de Amplificação *Bytes* (Cenário 1 - SNMP) - Metodologia Alternativa 83
4.38 Fator de Amplificação Pacotes (Cenário 1 - SNMP) - Metodologia Alternativa 83
4.39 Fator de Amplificação *Bytes* (Cenário 2 - SSDP) 84
4.40 Fator de Amplificação Pacotes (Cenário 2 - SSDP) 84
4.41 Fator de Amplificação *Bytes* (Cenário 2 - SNMP) 84
4.42 Fator de Amplificação Pacotes (Cenário 2 - SNMP) 84

4.43 Fator de Amplificação *Bytes* (Cenário 3 - SSDP)..... 85
4.44 Fator de Amplificação Pacotes (Cenário 3 - SSDP)..... 85
4.45 Fator de Amplificação *Bytes* (Cenário 3 - SNMP) 86
4.46 Fator de Amplificação Pacotes (Cenário 3 - SNMP)..... 86
4.47 Fator de Amplificação *Bytes* (Cenário 3 - CoAP) 86
4.48 Fator de Amplificação Pacotes (Cenário 3 - CoAP)..... 86
4.49 Fator de Amplificação *Bytes* (Cenário 3 - SNMP) - Metodologia Alternativa..... 87
4.50 Fator de Amplificação Pacotes (Cenário 3 - SNMP) - Metodologia Alternativa..... 87
4.51 Fator de Amplificação *Bytes* - SSDP (Todos os Cenários) 88
4.52 Fator de Amplificação Pacotes - SSDP (Todos os Cenários) 88
4.53 Fator de Amplificação *Bytes* - SNMP (Todos os Cenários) 89
4.54 Fator de Amplificação Pacotes - SNMP (Todos os Cenários) 89
4.55 Fator de Amplificação *Bytes* - CoAP (Todos os Cenários) 89
4.56 Fator de Amplificação Pacotes - CoAP (Todos os Cenários) 89
4.57 Fator de Amplificação *Bytes* - SSDP/SNMP/CoAP (Todos os Cenários)..... 90
4.58 Fator de Amplificação Pacotes - SSDP/SNMP/CoAP (Todos os Cenários)..... 90

LISTA DE TABELAS

2.1 Resumo dos parâmetros personalizáveis na mensagem M-SEARCH 16
2.2 Resumo dos parâmetros personalizáveis na mensagem GetBulkRequest 22
2.3 Resumo dos parâmetros personalizáveis na mensagem Confirmable 28
2.4 Resumo dos Principais Trabalhos Correlatos 35
2.5 Comparação entre os Principais Trabalhos Correlatos com Foco em AR-DDoS 37

3.1 Parâmetros globais do Linderhof 40
3.2 Parâmetros dos espelhos do Linderhof 40
3.3 Exemplo de arquivo de configuração do Linderhof 41
3.4 Pacotes por segundo gerados em cada nível de ataque do Linderhof 44
3.5 Resumo dos equipamentos utilizados e protocolos suportados 49

4.1 Cenário 1 (SSDP) - IPv4 54
4.2 Cenário 1 (SSDP) - IPv4 - ICMP 54
4.3 Cenário 1 (SSDP) - IPv4 - Metodologia Alternativa 56
4.4 Cenário 1 (SSDP) - IPv4 - ICMP - Metodologia Alternativa 56
4.5 Cenário 1 (SNMP) - IPv4 57
4.6 Cenário 1 (SNMP) - IPv4 - ICMP 58
4.7 Cenário 1 (SNMP) - IPv4 - Metodologia Alternativa 59
4.8 Cenário 1 (SNMP) - IPv4 - ICMP - Metodologia Alternativa 59
4.9 Cenário 2 (SSDP) - IPv4 61
4.10 Cenário 2 (SSDP) - IPv4 - ICMP 61
4.11 Cenário 2 (SNMP) - IPv4 63
4.12 Cenário 2 (SNMP) - IPv4 - ICMP 63
4.13 Cenário 2 (SNMP) - IPv6 64
4.14 Cenário 2 (SNMP) - IPv6 - ICMP 64
4.15 Cenário 3 (SSDP) - IPv4 - Kbps 66
4.16 Cenário 3 (SSDP) - IPv4 - Pacotes/s 66
4.17 Cenário 3 (SSDP) - IPv4 - ICMP 66
4.18 Cenário 3 (SSDP) - IPv6 - Kbps 67
4.19 Cenário 3 (SSDP) - IPv6 - Pacotes/s 68
4.20 Cenário 3 (SSDP) - IPv6 - ICMP 68
4.21 Cenário 3 (SNMP) - IPv4 - Kbps 70
4.22 Cenário 3 (SNMP) - IPv4 - Pacotes/s 70
4.23 Cenário 3 (SNMP) - IPv4 - ICMP 70
4.24 Cenário 3 (SNMP) - IPv6 - Kbps 71
4.25 Cenário 3 (SNMP) - IPv6 - Pacotes/s 72
4.26 Cenário 3 (SNMP) - IPv6 - ICMP 72

4.27 Cenário 3 (SNMP) - IPv4 - Kbps - Metodologia Alternativa 74
4.28 Cenário 3 (SNMP) - IPv4 - Pacotes/s - Metodologia Alternativa 74
4.29 Cenário 3 (SNMP) - IPv4 - ICMP - Metodologia Alternativa 74
4.30 Cenário 3 (SNMP) - IPv6 - Kbps - Metodologia Alternativa 75
4.31 Cenário 3 (SNMP) - IPv6 - Pacotes/s - Metodologia Alternativa 76
4.32 Cenário 3 (SNMP) - IPv6 - ICMP - Metodologia Alternativa 76
4.33 Cenário 3 (CoAP) - IPv4 - Kbps 77
4.34 Cenário 3 (CoAP) - IPv4 - Pacotes/s 78
4.35 Cenário 3 (CoAP) - IPv4 - ICMP 78
4.36 Cenário 3 (CoAP) - IPv6 - Kbps 79
4.37 Cenário 3 (CoAP) - IPv6 - Pacotes/s 79
4.38 Cenário 3 (CoAP) - IPv6 - ICMP 79
4.39 Fatores de Amplificação - Cenário 1 81
4.40 Fatores de Amplificação - Cenário 1 - Metodologia Alternativa 82
4.41 Fatores de Amplificação - Cenário 2 84
4.42 Fatores de Amplificação - Cenário 3 85
4.43 Fatores de Amplificação - Cenário 3 - Metodologia Alternativa 86
4.44 Resumo da Saturação dos Refletores (Bytes) 87
4.45 Resumo da Saturação dos Refletores (Pacotes) 88

LISTA DE ACRÔNIMOS

AR-DDoS	Amplified Reflection Distributed Denial of Service
CoAP	Constrained Application Protocol
DoS	Denial of Service
DDoS	Distributed Denial of Service
ICMP	Internet Control Message Protocol
IoT	Internet of Things
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
MTU	Maximum Transmission Unit
RFC	Request for Comments
SNMP	Simple Network Management Protocol
SSDP	Simple Service Discovery Protocol
TCP	Transmission Control Protocol
UDA	UPnP Device Architecture
UDP	User Datagram Protocol
UPnP	Universal Plug and Play

1 INTRODUÇÃO

Os ataques distribuídos de negação de serviço (DDoS, na sigla em inglês) vêm crescendo consideravelmente em volume de tráfego nos últimos anos [2]. Um dos mais significativos ocorreu em outubro de 2016 ao provedor de DNS Dyn [3], que atendia empresas como Amazon, Spotify, Netflix, Twitter, entre outras, e chegou à magnitude de 1,2 Tbps de tráfego, causando algumas horas de indisponibilidade e trazendo prejuízos financeiros às empresas afetadas, inclusive com a perda de vários clientes [4].

Em fevereiro de 2018, um ataque ao GitHub sobrecarregou os enlaces da empresa e indisponibilizou o acesso ao sítio por aproximadamente 9 minutos, alcançando a marca de 1,35 Tbps de tráfego [5]. De acordo com [6], menos de uma semana depois desse ataque, um novo foi lançado a um provedor norte-americano, cujo nome não foi revelado, com picos de tráfego de aproximadamente 1,7 Tbps, apesar de se ter pouca informação a respeito dele. Em um relatório do primeiro trimestre de 2020 [7], a *Amazon Web Services* (AWS) reportou que sofreu o maior ataque DDoS que se tem registro até hoje [8], chegando à marca de 2,3 Tbps de tráfego.

O incidente ao provedor Dyn foi relevante não apenas pelo volume de tráfego gerado, mas também por envolver centenas de milhares de dispositivos com menor poder de processamento. Eles fazem parte da chamada Internet das Coisas (IoT, do termo em inglês), uma tendência onde não apenas computadores estão conectados à rede, mas uma infinidade de dispositivos, como câmeras, telefones, eletrodomésticos, carros e equipamentos de monitoramento de cidades [9].

Este ataque também trouxe à tona o *malware* Mirai [10], que permite a criação, com uma certa facilidade, de grandes botnets contendo tais dispositivos, assim como a coordenação de ataques DDoS de forma centralizada por parte do atacante. Com a divulgação do seu código-fonte, diversas novas variantes surgiram [11], aumentando a preocupação existente em torno dele.

Outra tendência [12] observada nos ataques DDoS é o uso de uma técnica baseada em reflexão amplificada (AR-DDoS, em inglês), que consiste em gerar grandes respostas para requisições bem pequenas e direcioná-las (ou refletí-las) a uma vítima específica, sobrecarregando a largura de banda da sua rede.

Essa modalidade de ataque demanda uma preparação bem menor por parte do atacante, pois não há a necessidade de se infectar ou instalar *malwares* nos refletores, que são equipamentos que rodam algum serviço baseado principalmente no protocolo *User Datagram Protocol* (UDP). A principal tarefa do atacante é localizar *hosts* que possam servir como refletores e disparar requisições a eles. Os ataques ao GitHub e ao provedor americano utilizaram essa técnica, explorando o Memcached [13], um sistema distribuído de *cache* em memória, que roda sobre UDP, desenvolvido para acelerar o acesso a páginas e aplicações *web* dinâmicas.

Os estudos [14] e [15] apontam que a falta de segurança em dispositivos IoT é um dos motivos que os tornam tão atraentes para a realização de ataques DDoS. Isso se deve principalmente

à limitação de recursos desses equipamentos, que nem sempre permitem a utilização de mecanismos de segurança adequados para protegê-los, sobretudo os que envolvem criptografia. Há uma preocupação legítima em relação a estes dispositivos, dado o alto número de equipamentos suscetíveis a se tornarem refletores, e os possíveis impactos que o seu uso em ataques dessa natureza podem gerar na Internet. Se combinados com o Mirai ou outros *malwares*, os danos podem ser devastadores.

De acordo com [16], alguns protocolos tipicamente utilizados em dispositivos IoT estão vulneráveis a ataques AR-DDoS, entre os quais destacam-se o *Simple Service Discovery Protocol* (SSDP), o *Simple Network Management Protocol* (SNMP) e o *Constrained Application Protocol* (CoAP). Todos eles rodam sobre UDP, sendo o último uma tendência mais recente [17].

1.1 JUSTIFICATIVA

Apesar de os ataques DDoS terem surgido no final da década de 90 [18], até hoje eles representam uma ameaça à disponibilidade de serviços na Internet. Ao longo dos anos, embora as técnicas de mitigação e prevenção tenham evoluído, os ataques também sofreram melhorias e estão cada vez mais poderosos e sofisticados.

Considerando o grande número de dispositivos IoT na Internet [19], é importante compreender o seu possível emprego e comportamento como refletores, ou seja, se são capazes de gerar altos volumes de tráfego, de aumentar a intensidade do ataque sem sofrer consequências ou ainda de suportar o ataque por longos períodos.

Enquanto [20] analisou a saturação de computadores de uso geral - e não dispositivos IoT - atuando como refletores durante ataques simulados de DDoS, [21] fez uma comparação com diversos dispositivos IoT, mas a maioria era bem limitada computacionalmente e não suportava todos os protocolos testados.

1.2 OBJETIVOS

O objetivo deste trabalho é analisar o comportamento da saturação de diferentes tipos de dispositivos IoT utilizados como refletores em ataques DDoS por reflexão amplificada, em ambiente controlado, explorando os protocolos SSDP, SNMP e CoAP, a fim de avaliar a viabilidade de sua utilização em ataques reais, assim como sua capacidade de sustentar o ataque à medida em que a quantidade de pacotes vai aumentando ao longo do tempo.

Para atingir esse objetivo, o trabalho foi encaminhado de forma analítica e experimental e foi elaborado através de ampla pesquisa bibliográfica e testes práticos, utilizando dispositivos IoT diversos e uma ferramenta de ataque desenvolvida para esta finalidade. Vários cenários de testes foram montados e os resultados foram posteriormente analisados e discutidos.

1.2.1 Objetivos Específicos

- Descrever como ocorrem os ataques DDoS por reflexão amplificada explorando os protocolos SSDP, SNMP e CoAP e como os dispositivos IoT podem ser utilizados como vetores de propagação desses ataques;
- Realizar ataques DDoS por reflexão amplificada, em ambiente controlado, explorando os protocolos SSDP, SNMP e CoAP utilizando a ferramenta desenvolvida, com vários cenários e situações distintas;
- Analisar as diferenças quando versões diferentes do *Internet Protocol* (IP) – IPv4 e IPv6 – são utilizadas nos ataques DDoS por reflexão amplificada;
- Analisar o momento em que os dispositivos IoT utilizados como refletores começam a não ser mais efetivos para os ataques, identificando a taxa de amplificação máxima para cada cenário testado;
- Comparar a saturação dos diversos tipos de equipamentos IoT testados.

1.3 PRINCIPAIS CONTRIBUIÇÕES

Esta pesquisa trouxe como principais contribuições e resultados:

- Compreensão de como alguns dispositivos IoT típicos se comportam atuando como refletores em ataques AR-DDoS abusando os protocolos SSDP, SNMP e CoAP;
- Aprimoramento e ampliação da capacidade da ferramenta de avaliação de ataques AR-DDoS utilizada neste trabalho.

Estes resultados levaram às seguintes ações de disseminação:

- Publicação do artigo "*Amplified Reflection DDoS Attacks over IoT Mirrors: A Saturation Analysis*" [22], no *Workshop on Communication Networks and Power Systems* (WCNPS) 2019, que ocorreu em Brasília, Brasil, em outubro de 2019;
- Publicação do artigo "*Amplified Reflection DDoS Attacks over IoT Reflector Running CoAP*" [23], na 15ª Conferência Ibérica de Sistemas e Tecnologias de Informação (CISTI) 2020, que ocorreu em Sevilha, Espanha, em junho de 2020;
- Submissão e aceitação do artigo "Linderhof: uma ferramenta para avaliação de sistemas de mitigação de ataques reflexivos volumétricos (DDoS)" [24], a ser apresentado no Salão de Ferramentas do XXXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC), que ocorrerá no Rio de Janeiro, Brasil, em dezembro de 2020;

- Registro de Programa de Computador (RPC), junto ao Instituto Nacional de Propriedade Industrial (INPI), referente à ferramenta Linderhof, sob o processo "BR512020000389-3", solicitado em 15/04/2020 e expedido em 01/09/2020.

1.4 ORGANIZAÇÃO DO TRABALHO

A partir dos próximos capítulos, essa dissertação se divide em três partes principais: revisão bibliográfica, desenvolvimento e conclusão. O Capítulo 2 discute os conceitos abordados neste trabalho, esmiúçando como acontecem os ataques AR-DDoS com cada um dos protocolos utilizados nessa dissertação, assim como realiza comparações com os principais trabalhos correlatos encontrados na literatura. O Capítulo 3 descreve o funcionamento da ferramenta utilizada para gerar os ataques e detalha como os testes foram realizados. O Capítulo 4 analisa e discute os resultados obtidos. O Capítulo 5 apresenta as conclusões deste trabalho e aborda possíveis temas de trabalhos futuros, ou porque não puderam ser abordados ou porque fugiam ao escopo desta dissertação.

2 REVISÃO BIBLIOGRÁFICA

Este capítulo tem como objetivo apresentar uma revisão dos principais conceitos que permeiam esta pesquisa. Primeiramente, será introduzido o conceito de falsificação de endereços IPs, uma técnica bastante utilizada em ataques diversos. Posteriormente, os tipos de ataques de negação de serviço serão examinados, com ênfase nos ataques distribuídos, que serão explanados a partir de taxonomias encontradas na literatura. Em seguida, os três protocolos de camada de aplicação utilizados nos testes dessa dissertação – SSDP, SNMP e CoAP –, assim como as suas características que são exploradas nos ataques distribuídos de negação de serviço por reflexão amplificada, serão detalhados. Adiante, as formas de utilização de dispostivos IoT em ataques DDoS serão apresentadas. Por fim, serão realizadas comparações deste com outros trabalhos correlatos.

2.1 FALSIFICAÇÃO DE ENDEREÇOS IP

Antes de detalhar como os ataques de Negação de Serviço funcionam, é importante conhecer a técnica de falsificação de endereços IP, ou *IP Spoofing* em inglês, pois muitos ataques a utilizam com frequência. [25] afirma que a falsificação de endereços IP consiste em alterar o campo "endereço de origem" dos cabeçalhos dos pacotes IP, de forma a ocultar o verdadeiro rementente dos pacotes ou personificar outro dispositivo, dificultando a sua localização real.

Essa falsificação é possível em razão do roteamento na Internet ser baseado apenas no endereço IP de destino dos pacotes. Com isso, o atacante pode facilmente modificar o endereço de origem para outro valor sintaticamente correto e encaminhá-lo ao seu destinatário. Entretanto, uma vez que o endereço de origem não corresponda ao do verdadeiro emissor dos pacotes, as respostas e eventuais mensagens de erro serão encaminhadas ao endereço que foi forjado, o que pode ser o objetivo do atacante em alguns casos.

Existem algumas formas de se evitar a falsificação de endereços IP, sendo as mais comuns a utilização de filtros de ingresso, descritos nas *Best Current Practices* (BCP) 38 [26] e 84 [27]. Enquanto a primeira recomenda que os pacotes na interface de entrada da rede de um sistema autônomo sejam filtrados, para permitir somente aqueles cujo endereço de origem seja parte da rede conectada àquela interface, a segunda é voltada para redes de maior complexidade, geralmente conectadas a mais de um sistema autônomo de trânsito e com um número maior de roteadores e usuários conectados. Nesse caso, a indicação é de uma técnica denominada *Reverse Path Forwarding* (RPF), que faz com o roteador filtre qualquer pacote em uma interface cuja rota de retorno do pacote não seja através dessa mesma interface.

O problema é que nem todos os provedores e empresas utilizam essas técnicas, conforme se observa na figura 2.1, onde aproximadamente 27,6% dos sistemas autônomos testados estão

suscetíveis a aceitarem pacotes com endereços IPv4 de origem forjados [28]. Em IPv6 (figura 2.2), esse número é ainda maior (34,3%).

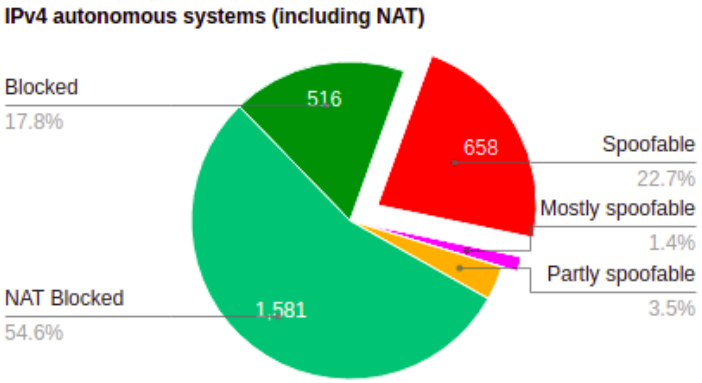

Figura 2.1: Quantidade de Sistemas Autônomos Suscetíveis à Falsificação de Endereços IPv4

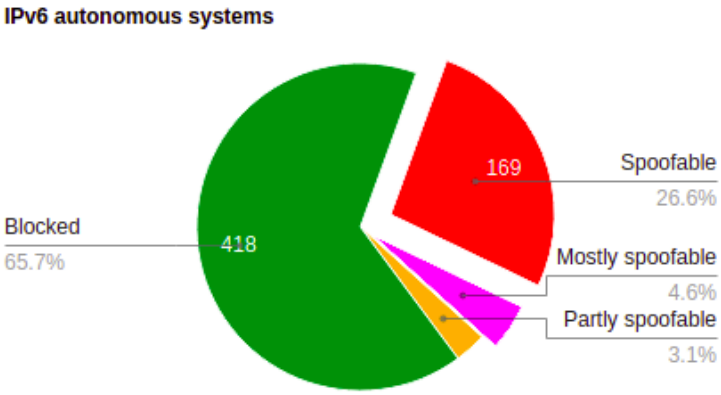

Figura 2.2: Quantidade de Sistemas Autônomos Suscetíveis à Falsificação de Endereços IPv6

2.2 ATAQUES DE NEGAÇÃO DE SERVIÇO (DOS)

Os ataques de Negação de Serviço (*Denial of Service*, ou DoS, em inglês) têm como objetivo indisponibilizar o acesso de usuários legítimos a um determinado sistema, dispositivo ou recurso de rede, em função de alguma ação de um atacante [29]. Geralmente essa ação consiste em inundar a vítima com uma quantidade de tráfego tão grande que ela não é capaz de responder novas requisições, conforme a figura 2.3, fazendo com que a organização atacada perca tempo e dinheiro mitigando o ataque, até que o recurso seja restabelecido.

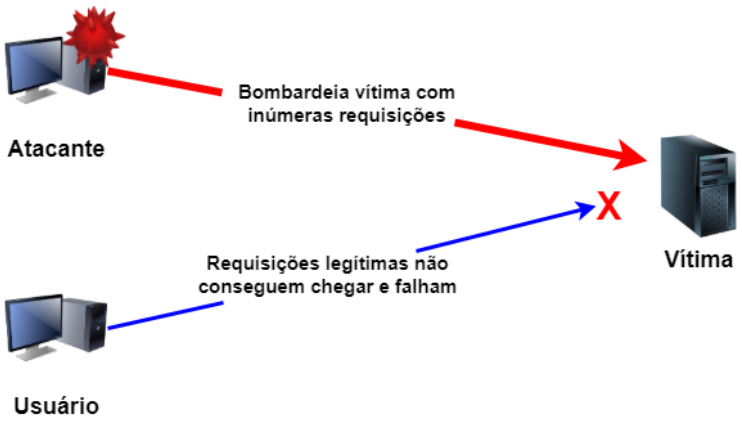

Figura 2.3: Exemplo de ataque DoS

O primeiro ataque DoS que se tem notícia ocorreu em 1974 [30], quando David Dennis, um garoto de 13 anos, desenvolveu um programa que forçou o desligamento de 31 terminais PLATO, uma espécie de computador utilizado para ensino, de um laboratório de pesquisa da Universidade de Illinois, em Urbana-Champaign, impedindo a utilização dos terminais até que estes fossem religados manualmente.

Com o passar dos anos, à medida que os equipamentos foram ficando mais robustos e os enlaces com maior capacidade de tráfego, novos métodos de ataques foram desenvolvidos, explorando-se peculiaridades dos protocolos utilizados nas comunicações ou ainda técnicas mais engenhosas.

Provavelmente a técnica que causou maior impacto tenha sido a que distribuiu o ataque DoS entre diversos equipamentos atacantes, surgida em 1999 [31], o que foi chamado de ataque Distribuído de Negação de Serviço (*Distributed Denial of Service*, ou DDoS, em inglês), que será explicado com mais detalhes a seguir.

2.2.1 Ataques Distribuídos de Negação de Serviço (DDoS)

Os ataques DDoS possuem o mesmo objetivo dos ataques DoS, mas o fazem de forma coordenada entre múltiplos atacantes, intensificando o poder do ataque, de acordo com o exemplo da figura 2.4. O ataque requer um planejamento maior e exige coordenação entre os equipamentos, geralmente feito através de uma ferramenta de Comando e Controle (C&C).

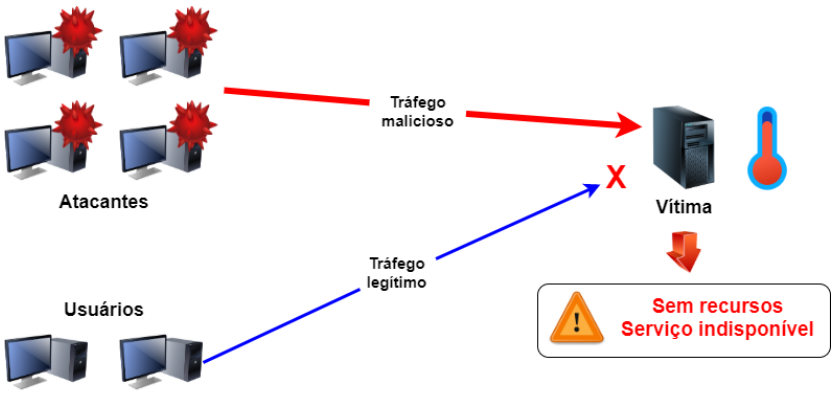

Figura 2.4: Exemplo de ataque DDoS

O trabalho [1] dividiu os ataques DDoS em dois grandes tipos: esgotamento de recursos e saturação de largura de banda. Enquanto o primeiro tipo explora vulnerabilidades ou falhas em protocolos ou implementações utilizadas nas vítimas a fim de indisponibilizá-las, o segundo se preocupa em preencher toda a largura de banda dos enlaces da vítima com uma quantidade muito alta de tráfego para evitar que novas requisições cheguem ao recurso atacado.

Além desses, o estudo [2] acrescentou duas novas categorias: ataques de infraestrutura e de dia zero. O primeiro se refere a ataques a alvos relacionados a elementos críticos da Internet, como os servidores DNS raiz, enquanto que o segundo está associado àqueles ataques desconhecidos, como os que exploram novas vulnerabilidades em protocolos.

A taxonomia adotada neste trabalho (figura 2.5) foi adaptada e simplificada em relação à que foi apresentada em [1]. Os ataques baseados em esgotamento de recursos dividem-se em ataques de exploração de protocolos e de exploração de implementações, enquanto que os de saturação de largura de banda dividem-se em ataques por inundação e por reflexão/amplificação, todos eles explicados a seguir.

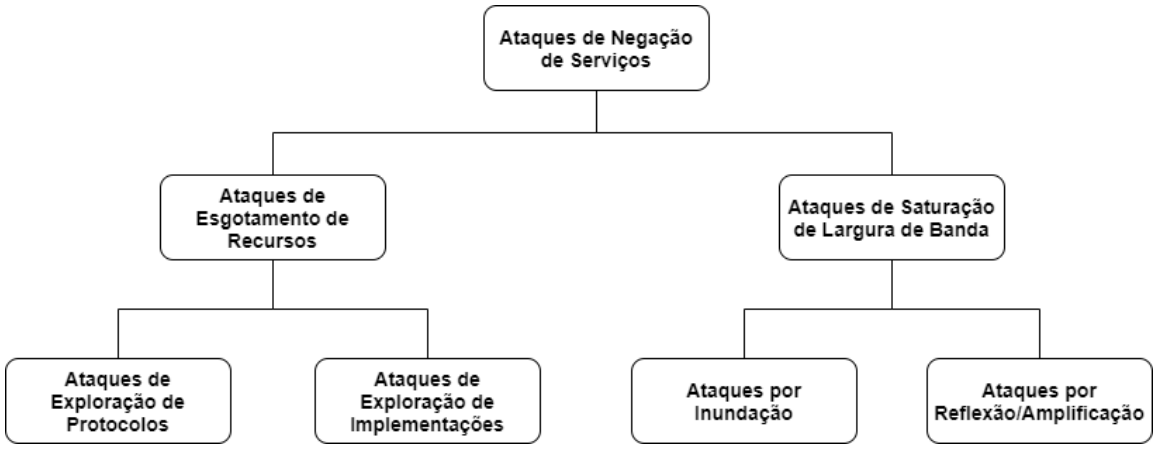

Figura 2.5: Taxonomia de ataques DDoS adaptada de [1]

2.2.1.1 Ataques de Esgotamento de Recursos

O objetivo principal dessa categoria de ataques é fazer com que os recursos de uma vítima, como o uso de CPU ou memória ou a quantidade de *sockets* disponíveis, sejam transbordados, de forma que ela fique impedida de atender novas requisições de usuários legítimos. Basicamente, existem duas formas de se atingir essa meta:

- **Exploração de Protocolos:** nesse método, os atacantes se aproveitam de fraquezas existentes em diferentes protocolos utilizados em diversas camadas. Alguns dos ataques existentes são: TCP SYN, TCP PUSH + ACK, inundação HTTP, inundação SIP, requisição/resposta lenta, entre outros.
- **Exploração de Implementações:** o foco, nesse caso, é explorar implementações inadequadas de protocolos, utilizando principalmente pacotes mal formados, com o intuito de confundir a vítima, resultando em quebras do sistema. Um típico ataque desses consiste em utilizar o endereço IP da vítima tanto no campo de origem quanto no campo de destino do cabeçalho IP, fazendo com que a vítima responda pra si, em um loop infinito, até que seus recursos sejam exauridos. Outros ataques conhecidos são o ping da morte, *teardrop* e o que manipula o campo "Opções" do cabeçalho IPv4.

Como esses ataques não são o foco desse trabalho, detalhes sobre cada um deles podem ser encontrados em [2].

2.2.1.2 Ataques de Saturação de Largura de Banda

Por outro lado, a intenção de ataques que saturam a largura de banda dos enlaces de uma vítima é impedir que requisições válidas consigam passar pelos enlaces congestionados da vítima, causando lentidão no acesso ou até mesmo o seu descarte. Geralmente isso é possível utilizando-se um exército muito grande de atacantes e eles podem ser de dois tipos:

- **Inundação:** o objetivo desse método é enviar o máximo de pacotes possíveis a uma determinada vítima, de forma direta. Dois ataques muito comuns são a inundação de pacotes UDP e a de pacotes *echo request* do protocolo *Internet Control Message Protocol* (ICMP). No primeiro, são direcionados pacotes UDP a uma porta específica ou a portas aleatórias. Caso não haja algum serviço rodando nessas portas, a vítima ainda é sobrecarregada pela geração de respostas ICMP do tipo "Destino inalcançável" com código "Porta inalcançável". No segundo, são enviadas mensagens *echo request* (ping), que são respondidas pela vítima com mensagens *echo reply*.
- **Reflexão/Amplificação:** essa categoria possui o maior potencial de danos, uma vez que o atacante se aproveita de características de alguns protocolos de aplicação para amplificar o poder do ataque, utilizando para isso um equipamento intermediário, chamado de refletor. O

ataque consiste em enviar requisições muito pequenas ao refletor, que por sua vez direciona as respostas, com tamanhos geralmente bem superiores, à vítima. Os maiores ataques DDoS registrados até hoje utilizaram essa técnica, que será detalhada nas seções seguintes, por ser a base desse trabalho.

2.2.2 Ataques Distribuídos de Negação de Serviço por Reflexão Amplificada (AR-DDoS)

Os ataques AR-DDoS possuem como principal objetivo o esgotamento da largura de banda de uma vítima, através do envio de uma quantidade muito alta, e geralmente amplamente distribuída, de requisições a equipamentos que rodam algum tipo de serviço, geralmente em cima do protocolo UDP, mas também podendo ser por abuso ao *Transmission Control Protocol* (TCP) [32], apesar de ser menos comum. Por não serem o foco deste trabalho, os ataques baseados em TCP não serão abordados nessa dissertação.

O ataque baseado no protocolo UDP se aproveita de uma característica dele, que por não ser orientado a conexão, não requer o estabelecimento de uma sessão entre as partes envolvidas na comunicação. Com isso, o atacante pode forjar o endereço IP de origem dos pacotes e, caso não haja algum mecanismo que detecte endereços de origem forjados em uma rede, eles irão trafegar normalmente sem que sejam descartados, dificultando a sua identificação e tornando a mitigação do ataque mais complexa.

O atacante, portanto, gera pacotes muito pequenos de requisição com o endereço de origem modificado para o endereço da vítima, e os envia a *hosts* que estejam rodando algum serviço baseado em UDP. Estes dispositivos intermediários atuam como refletores, ou espelhos, pois aceitam e processam os pacotes, mas em seguida envia as respostas, geralmente de tamanho bem maior, para a vítima, amplificando assim o poder do ataque, uma vez que, se as requisições fossem enviadas diretamente à vítima, não teriam o mesmo tamanho. Um exemplo de como funciona o ataque está exibido na figura 2.6.

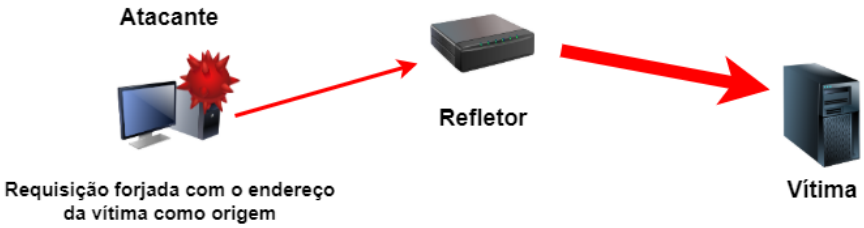

Figura 2.6: Exemplo de ataque AR-DDoS

O principal desafio do atacante, nesse caso, é explorar características dos protocolos de aplicação que rodam sobre UDP nos refletores, de forma que possam obter o máximo fator de amplificação de largura de banda possível. Segundo [12], o fator de amplificação de largura de banda de um ataque é obtido pela razão entre o tamanho dos dados da camada de aplicação recebidos pela vítima e o tamanho dos dados da camada de aplicação enviados pelo atacante, ambos em bytes. O autor do estudo ignora o tamanho dos cabeçalhos Ethernet, IP e UDP para as medidas

permanecerem válidas na eventualidade de os protocolos superiores mudarem, como no caso do IPv4 e IPv6, que possuem tamanhos de cabeçalhos distintos.

Entretanto, [33] pondera que, durante um ataque, há um esforço computacional em relação ao volume de tráfego gerado tanto no atacante quanto no refletor que deveria ser considerado e opta por incluir os cabeçalhos dos protocolos IP e UDP no cálculo do fator de amplificação de largura de banda. Apesar de a tecnologia de interconexão de redes na Internet ser predominantemente a Ethernet, o seu cabeçalho não é considerado, justamente por não ser obrigada a sua utilização, diferentemente do IP e do UDP, que sempre estarão presentes nos ataques AR-DDoS realizados nessa pesquisa.

O cálculo adotado neste trabalho segue a linha de [33] e é exibido na equação (2.1), onde o tamanho da informação recebida pela vítima é dividida pela enviada pelo atacante, ambas em bytes e sem considerar o tamanho do cabeçalho Ethernet (ou da tecnologia que esteja sendo utilizada). Dessa forma, apesar de os fatores de amplificação serem diferentes para ataques com IPv4 e IPv6, eles refletirão de forma mais fidedgina a realidade para cada protocolo.

$$FA_{Bytes} = \frac{(Cab_{IP} + Cab_{UDP} + Dados)\ resposta}{(Cab_{IP} + Cab_{UDP} + Dados)\ requisição} \tag{2.1}$$

De modo similar, também é possível obter o fator de amplificação de pacotes de um ataque, exibido na equação (2.2), que é obtido pela razão entre a quantidade de pacotes recebidos pela vítima e a quantidade de pacotes enviados pelo atacante.

$$FA_{Pacotes} = \frac{quantidade\ de\ pacotes\ recebidos\ pela\ vítima}{quantidade\ de\ pacotes\ enviados\ pelo\ atacante} \tag{2.2}$$

Se o datagrama UDP for muito grande, sendo maior do que a *Maximum Transmission Unit* (MTU) do enlace, e ocorrer fragmentação IP durante a transmissão, os fragmentos também são considerados nos cálculos, uma vez que cabeçalhos adicionais são adicionados, assim como o processamento de todos os fragmentos nos equipamentos envolvidos será necessário.

2.3 ANATOMIA DOS ATAQUES

Nas subseções seguintes, o funcionamento normal dos protocolos SSDP, SNMP e CoAP serão analisados, assim como a dinâmica dos ataques de amplificação baseados neles, de forma a se conhecer as características que são exploradas em cada protocolo para se obter as maiores respostas para as menores perguntas, ou seja, para se alcançar o maior fator de amplificação de largura de banda possível. Em geral, a anatomia desses ataques não é complexa e requer apenas a manipulação do endereço IP de origem e de alguns parâmetros das mensagens de requisição.

2.3.1 Simple Service Discovery Protocol (SSDP)

O protocolo SSDP é utilizado como parte da tecnologia *Universal Plug and Play* (UPnP), que define uma arquitetura para conectividade de rede pervasiva entre os mais variados dispositivos de diversos tamanhos e utilidades [34], permitindo que esses equipamentos se autoconfigurem e aprendam sobre a presença de outros dispositivos na rede, descobrindo os serviços que eles disponibilizam, sem qualquer interação do usuário, facilitando a sua interconexão em uma rede.

As normas e os protocolos para a comunicação entre os dispositivos são estabelecidas em um documento chamado *UPnP Device Architecture* (UDA), que também define os tipos e formatos das mensagens que são trocadas em todos os passos dessa comunicação, ilustrados na figura 2.7 e explicados a seguir, sendo que existe uma UDA para cada versão do UPnP lançada até o momento (1.0 [34], 1.1 [35] e 2.0 [36]).

Para o UPnP, existem dois tipos de dispositivos, os controlados (simplesmente chamados de dispositivos) e os pontos de controle. O primeiro atua como um servidor, esperando e respondendo as requisições dos pontos de controle, enquanto o segundo cria as requisições de serviço e as envia aos dispositivos controlados. O protocolo SSDP é o responsável pela comunicação entre esses dispositivos.

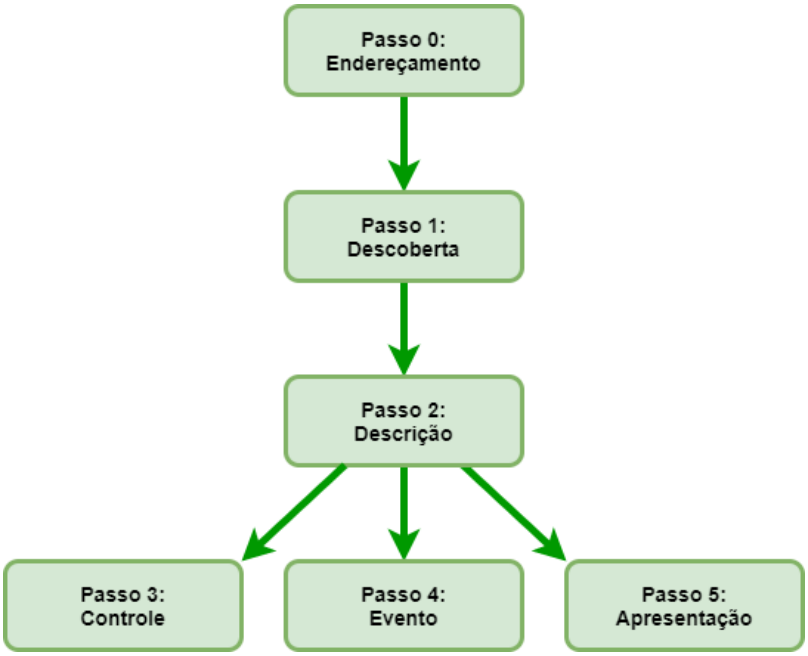

Figura 2.7: Passos do UPnP

- **Passo 0 (Endereçamento):** é o passo inicial, onde o dispositivo obterá um endereço IP para se conectar à rede, geralmente através do *Dynamic Host Configuration Protocol* (DHCP);
- **Passo 1 (Descoberta):** é onde ocorre a descoberta de dispositivos controlados na rede, assim como os serviços que estes proveem, através do protocolo SSDP;
- **Passo 2 (Descrição):** após descobrir o dispositivo, nesse passo obtem-se mais informações

acerca dele e de suas capacidades, como a sua descrição e os *Uniform Resource Locators* (URLs) para as descrições dos seus serviços;

- **Passo 3 (Controle):** uma vez obtida a descrição do dispositivo, ações podem ser invocadas aos seus serviços ou os valores de suas variáveis podem ser consultados;
- **Passo 4 (Evento):** é onde as mudanças nas variáveis realizadas no passo anterior são informadas através de mensagens de eventos;
- **Passo 5 (Apresentação):** faz a interface com o usuário, permitindo que este controle o dispositivo ou veja o seu estado.

É muito comum encontrar o SSDP rodando em equipamentos de redes domésticas, como computadores, impressoras, pontos de acesso sem fio, roteadores de Internet, entre outros, possuindo, portanto, uma superfície de ataque muito grande. Em muitos desses dispositivos, o protocolo já vem habilitado por padrão, inclusive nas interfaces voltadas para a Internet, e não costumam possuir mecanismos de segurança habilitados que impeçam a sua exploração por parte dos atacantes.

O SSDP roda, por padrão, na porta UDP 1900 para trocar informações e, durante os ataques, dois tipos de mensagens da etapa de Descoberta do UPnP são trocadas entre atacante, refletor e vítima. O atacante envia mensagens do tipo **M-SEARCH** ao refletor, que por sua vez, responde à vítima com várias mensagens do tipo **200 OK**, conforme a figura 2.8. O número exato de mensagens de resposta é calculado pela equação (2.3), onde 3 equivale à quantidade de mensagens correspondentes ao dispositivo raiz, $2d$ corresponde a duas vezes a quantidade de dispositivos embarcados que o dispositivo raiz possui e k é o número de tipos de serviços distintos de cada dispositivo.

$$N_{resp} = 3 + 2d + k \tag{2.3}$$

Figura 2.8: Ataque AR-DDoS com SSDP

Dependendo da versão do UPnP utilizada, as mensagens de requisição **M-SEARCH** sofrem uma pequena variação. Enquanto a UDA 1.0 define que essas mensagens são enviadas apenas em *multicast*, para o grupo 239.255.255.250, as UDAs 1.1 e 2.0 aceitam mensagens tanto em *multicast*, para o mesmo grupo, como em *unicast*, para o endereço do dispositivo.

O conteúdo da mensagem **M-SEARCH** em cada tipo de transmissão também sofre alterações. A UDA 1.0 especifica que os campos dessa mensagem em *multicast* são os constantes na figura 2.9, sendo que todos eles são obrigatórios, mas apenas o **MX** e o **ST** são personalizáveis.

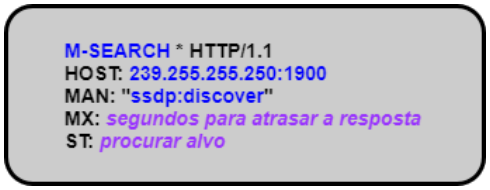

Figura 2.9: Formato da mensagem M-SEARCH em multicast especificada na UDA 1.0

O campo **MX** diz respeito ao tempo máximo de espera por uma resposta, podendo variar entre 1 e 120 segundos, sendo que o dispositivo responderá em um período aleatório entre 0 e o valor existente nesse campo. Para garantir que o dispositivo consultado responda o mais rápido possível, o valor do **MX** deve ser ajustado para "1".

O campo **ST** se refere ao que deve ser buscado no dispositivo que está sendo consultado, se todos os dispositivos e serviços que ele possui ou apenas uma parte deles. Para se obter a maior amplificação possível, o valor do **ST** deve ser ajustado para "ssdp:all". Dessa forma, o dispositivo enviará a quantidade de mensagens descrita na equação (2.3).

Cabe ressaltar que, apesar de a UDA 1.0 especificar que as mensagens **M-SEARCH** devem ser enviadas via *multicast*, é comum encontrar dispositivos com implementações UPnP 1.0 que aceitem essas mensagens via *unicast* [37], também, o que os torna vulneráveis a serem explorados a partir da Internet.

O conteúdo da mensagem **M-SEARCH** em *multicast* definido pela UDA 1.1 é apresentado na figura 2.10 e pela UDA 2.0 é exibido na figura 2.11. Os campos obrigatórios continuam sendo os mesmos, mas dessa vez o valor do **MX** deve variar entre 1 e 5 segundos. Alguns campos foram acrescentados em ambas as versões, mas o único obrigatório é o **CPFN.UPNP.ORG** no UPnP 2.0, que especifica um nome amigável para o dispositivo. Para se obter uma amplificação maior, os campos opcionais devem ser removidos das mensagens de requisição para que elas tenham um tamanho menor em bytes.

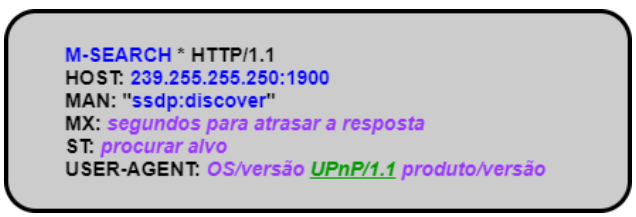

Figura 2.10: Formato da mensagem M-SEARCH em multicast especificada na UDA 1.1

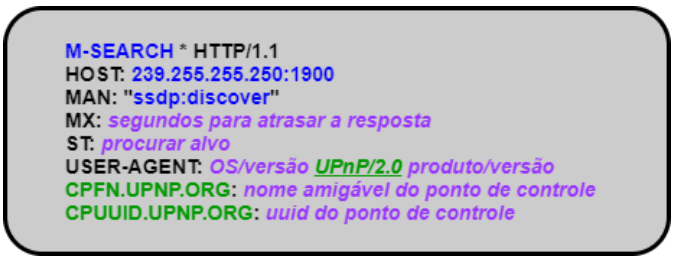

Figura 2.11: Formato da mensagem M-SEARCH em multicast especificada na UDA 2.0

O conteúdo da mensagem **M-SEARCH** em *unicast* definido pelas UDAs 1.1 e 2.0 é mostrado nas figuras 2.12 e 2.13, respectivamente. As principais diferenças estão no campo **MX**, que deixa de existir, e no campo **HOST**, que agora deve possuir o endereço IP (IPv4 ou IPv6) do dispositivo consultado e o número da porta que ele escuta por mensagens. O campo **USER-AGENT** em ambas as versões é opcional e deve ser removido para se obter uma amplificação maior do ataque.

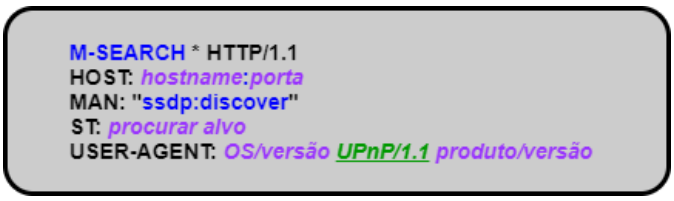

Figura 2.12: Formato da mensagem M-SEARCH em unicast especificada na UDA 1.1

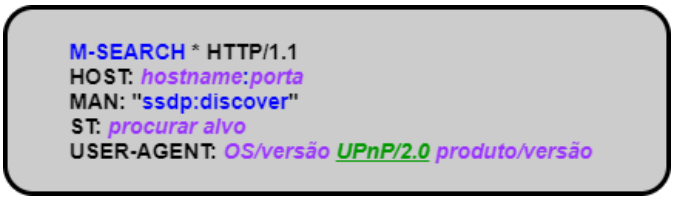

Figura 2.13: Formato da mensagem M-SEARCH em unicast especificada na UDA 2.0

Em resposta à requisição **M-SEARCH**, diversas mensagens do tipo **200 OK** são geradas pelo refletor. A figura 2.14 exibe um exemplo dessa mensagem, capturada via Wireshark em um dos testes realizados. É possível observar que o conteúdo é relativamente maior do que a mensagem de requisição e, por serem enviadas "3 + 2.d + k" mensagens desse tipo, o ataque acaba resultando em uma boa amplificação de largura de banda, além de uma amplificação de pacotes.

Figura 2.14: Exemplo de resposta 200 OK

2.3.1.1 Resumo do ataque via SSDP

Para se conseguir melhores taxas de amplificação em ataques AR-DDoS com o SSDP, segue um resumo dos passos a serem seguidos pelo atacante. A tabela 2.1 traz uma síntese dos parâmetros que podem ser modificados na mensagem M-SEARCH.

1. Descobrir a versão do UPnP que está rodando no refletor. Caso seja a 1.0, só há um tipo de mensagem **M-SEARCH** a ser utilizada, mas se for a 1.1 ou 2.0 (pouco comum, por ser mais recente), a mensagem via *unicast*, por possuir menos campos obrigatórios, deve ser a escolhida;

2. Gerar uma mensagem **M-SEARCH** com os campos **ST** ajustado para "ssdp:all" e **MX** (caso a versão do UPnP seja a 1.0) para "1". No caso do UPnP 1.1 ou 2.0, o campo **HOST** deve conter o endereço IPv4/IPv6 do refletor e a porta em que ele esteja aceitando requisições (geralmente a 1900);

3. Gerar um datagrama UDP com a mensagem **M-SEARCH** criada no passo 2, com uma porta de origem aleatória e a porta de destino 1900 (ou a que o refletor esteja utilizando para o SSDP);

4. Adicionar o cabeçalho IPv4/IPv6 ao datagrama UDP, com o endereço de origem forjado para o IP da vítima e o endereço de destino ajustado para o IP do refletor. Mesmo que o refletor rode apenas o UPnP 1.0, que teoricamente só aceita mensagens em *multicast*, deve-se preferir que o endereço de destino do pacote seja o endereço do refletor e não o do grupo *multicast* **239.255.255.250**;

5. Encaminhar o pacote ao refletor, que deve estar apto a receber e responder mensagens do tipo **M-SEARCH**;

6. O refletor recebe a mensagem, a processa e responde desatentamente com várias mensagens **200 OK** ao endereço da vítima, que estava forjado no pacote.

Tabela 2.1: Resumo dos parâmetros personalizáveis na mensagem M-SEARCH

Parâmetro	**Valor**	**Versão do UPnP**
ST	"ssdp:all"	1.0 / 1.1 / 2.0
MX	1	1.0
HOST	<ip-refletor>:1900	1.1 / 2.0

2.3.2 Simple Network Management Protocol (SNMP)

O protocolo SNMP é utilizado para gerenciar remotamente dispositivos em uma rede, tanto para coletar como para alterar informações, chamadas de variáveis. Apesar de a sua utilização

ser mais comum em ambientes corporativos, diversos dispositivos em redes domésticas, como roteadores de Internet e impressoras, vêm com ele habilitado por padrão.

O SNMP roda tipicamente sobre UDP, em duas portas distintas: 161 e 162. A porta 161 é utilizada para buscar/alterar variáveis no dispositivo gerenciado, que se comunica através de um agente SNMP, enquanto a 162 é usada pelo dispositivo para enviar mensagens de eventos ou notificações, chamadas de *traps*, ao servidor ou estação de gerenciamento SNMP, conforme exemplificado na figura 2.15.

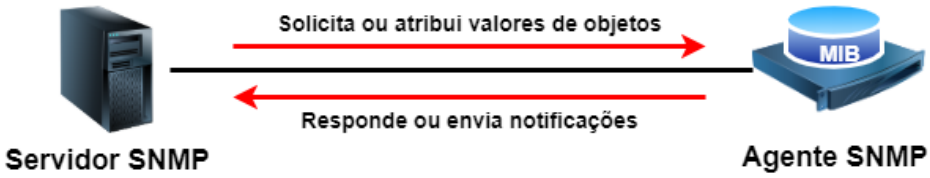

Figura 2.15: Funcionamento normal do SNMP

O SNMP foi primeiramente definido na RFC 1157 [38] e foi chamado de SNMPv1. Em conjunto com outras RFCs, foram definidos ainda dois componentes que permitem o correto funcionamento do protocolo: a *Management Information Base* (MIB), uma espécie de base de dados de objetos armazenada no dispositivo gerenciado, que contêm valores que são consultados ou alterados remotamente, e a *Structure of Management Information* (SMI), que é responsável por definir os tipos, padrões e regras dos objetos existentes em uma MIB.

A SMI define que a MIB é estruturada em um formato de árvore hierárquica com tabelas conceituais, como ilustra a figura 2.16, e que cada recurso gerenciado é representado por um objeto, também chamado de variável, que recebe um identificador, denominado *Object Identifier* (OID). O OID ".1.3.6.1.2.1.2", por exemplo, contem informações acerca das interfaces de rede de um determinado dispositivo. Outra forma de representar esse OID seria através da sintaxe ".iso.org.dod.internet.mgmt.mib-2.interfaces".

No SNMPv1, para que ocorra a comunicação entre o servidor de gerência e o dispositivo gerenciado, uma autenticação simples é feita utilizando-se uma *string* de comunidade definida neste último. Durante a configuração da comunidade no dispositivo, um dos dois níveis de acesso disponíveis podem ser escolhidos: somente leitura, onde apenas informações podem ser consultadas no dispositivo gerenciado, e leitura/gravação, que permite que dados sejam lidos ou alterados no dispositivo. A utilização de comunidades para gerenciar dispositivos em uma rede acaba se tornando um risco, uma vez que essa informação trafega em texto claro. Caso o valor dessa *string* seja descoberto por terceiros mal intencionados, dados podem ser coletados ou até alterados.

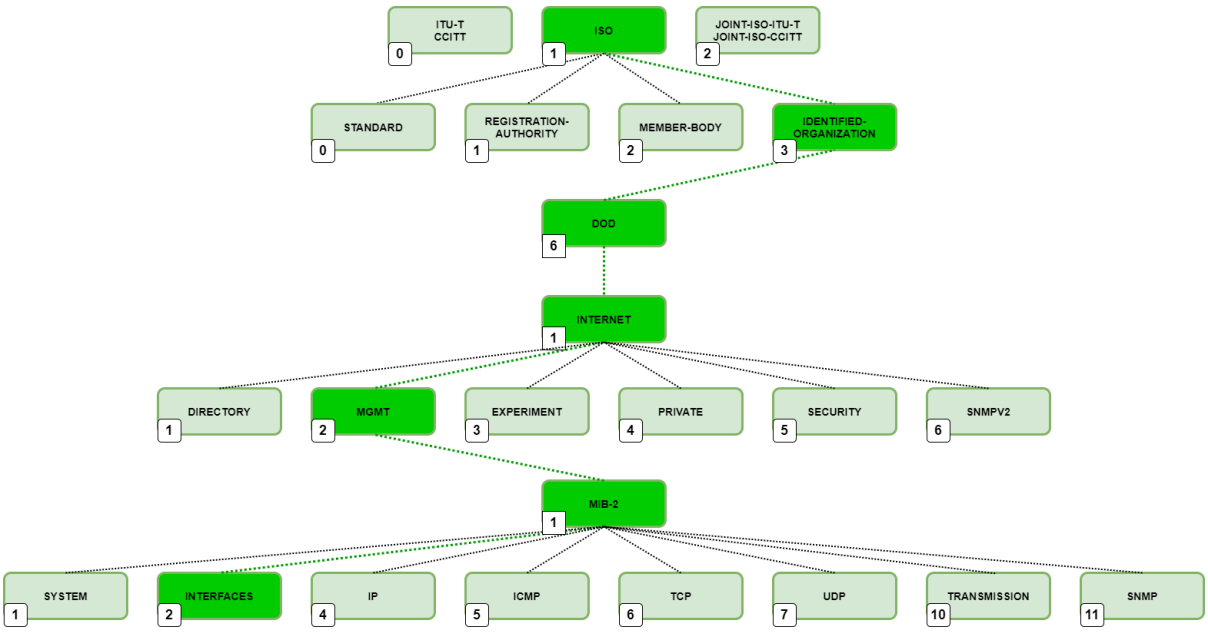

Figura 2.16: Estrutura da MIB do SNMP

O SNMPv1 prevê cinco tipos de operações ou mensagens, também chamadas de *Protocol Data Units* (PDUs):

- **GetRequest:** permite que o servidor de gerência consulte os valores dos objetos no dispositivo gerenciado;
- **GetNextRequest:** permite que o servidor de gerência consulte o valor do próximo objeto na MIB do dispositivo gerenciado. Caso ele queira consultar múltiplos valores, ele deve informar o OID da base da tabela a ser consultada. Nesse caso, uma mensagem GetNextRequest será gerada para cada elemento da tabela, até que se chegue ao último;
- **SetRequest:** permite que o servidor de gerência altere valores dos objetos no dispositivo gerenciado;
- **Trap:** usada pelo dispositivo gerenciado para notificar o servidor de gerência de algum evento que ocorreu nele;
- **Response:** usado como resposta para as mensagens **Get** e **Set**.

Alguns anos depois, a SMIv2 e a MIB-II foram lançadas, trazendo melhorias na sintaxe e acrescentando novos objetos para a MIB. Uma nova versão para o SNMP, chamada de SNMPv2, também foi lançada, apresentando um modelo de segurança baseado em partes, com a adição de mecanismos de autenticação e privacidade para prover mais confiabilidade e segurança ao protocolo. Entretanto, essa versão, chamada de SNMPv2p, não teve uma receptividade muito boa e foi movida para o estado de histórica. Novas variantes foram propostas, sendo a SNMPv2c [39] a mais amplamente aceita. Ela fornece segurança baseada em comunidades, no mesmo modelo

de segurança do SNMPv1. Além disso, dois novos PDUs foram acrescentados aos já existentes e um foi modificado:

- **InformRequest:** permite que o servidor de gerência envie informações para outro servidor;
- **GetBulkRequest:** permite que o servidor de gerência consulte os valores de uma tabela inteira em uma única operação de consulta, ao invés de múltiplas do tipo GetNextRequest;
- **Trapv2:** tem a mesma funcionalidade do Trap do SNMPv1, mas foi redesenhado para possuir o mesmo formato das demais mensagens, otimizando o processamento no servidor de gerência;

Por fim, uma terceira versão do protocolo SNMP foi lançada, chamada de SNMPv3 [40]. A principal contribuição dessa versão está relacionada aos mecanismos de segurança oferecidos, que proveem autenticação e confidencialidade, através de um modelo baseado em visões, que determina se um objeto na MIB pode ser consultado por um sistema remoto. Além disso, a versão 3 definiu uma arquitetura de referência para o padrão SNMP. Ademais, o uso de comunidades, por ser mais inseguro, foi extinto.

Durante a realização dos ataques AR-DDoS com SNMP, dois tipos de mensagens são trocadas entre atacante, refletor e vítima. Para se obter a maior taxa de amplicação possível, o atacante envia mensagens do tipo **GetBulkRequest** ao refletor, que responde à vítima com mensagens do tipo **Response**, conforme a figura 2.17. Para o ataque funcionar corretamente, é imprescindível que o SNMPv2c esteja rodando no dispositivo, pois a mensagem **GetBulkRequest** não existe no SNMPv1 e, como o SNMPv3 não utiliza mais comunidades e implementa mecanismos de segurança mais robustos, torna-se inviável realizar o ataque nessa versão.

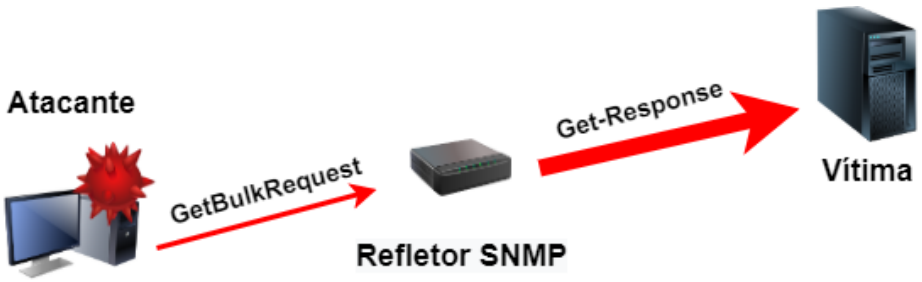

Figura 2.17: Ataque AR-DDoS com SNMP

A mensagem **GetBulkRequest** permite que, em uma única consulta, um conjunto de variáveis e/ou uma tabela inteira da MIB do dispositivo seja consultada. Na versão 1 do SNMP, quando essa mensagem não existia, era necessário enviar múltiplas mensagens do tipo **GetRequest** ou **GetNextRequest**. Esse tipo de mensagem deve seguir o formato especificado na figura 2.18.

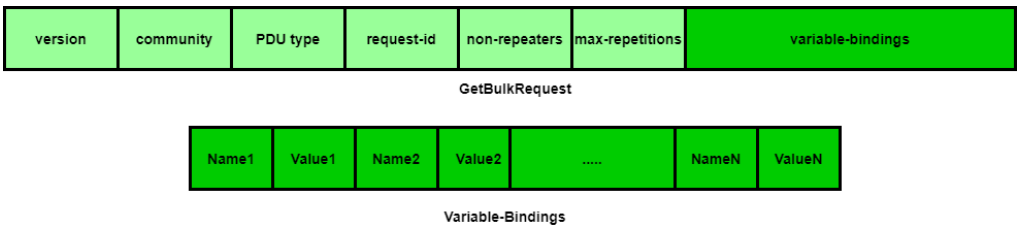

Figura 2.18: Formato da mensagem GetBulkRequest

O campo **version** deve ser preenchido com o valor "0x01", que representa a versão 2c do SNMP. O campo **community** deve ser preenchido com o valor da comunidade utilizada para gerenciar o dispositivo. A mais comum é a "public", que vem configurada por padrão em uma quantidade significativa de dispositivos, tornando-se essa a primeira opção a ser testada pelo atacante. Cabe ressaltar que, para a realização do ataque, somente consultas são realizadas. Portanto, a comunidade utilizada no equipamento pode ser apenas de leitura, não havendo necessidade de possuir permissão de escrita no equipamento.

O valor de **PDU Type** deve ser ajustado para "0xA5", que equivale ao tipo de mensagem **GetBulkRequest**. O campo **request-id** é um identificador para a requisição. Se o valor for gerado de forma aleatória e cada requisição for enviada com um identificador diferente, dificulta-se a mitigação do ataque. A resposta de uma requisição deve possuir o mesmo identificador dela.

O campo **non-repeaters** define a quantidade de variáveis que serão consultadas de forma única do dispositivo, como se várias mensagens **GetRequest** fossem utilizadas. Os OIDs das variáveis buscadas devem ser passados no campo **variable-bindings**. Por exemplo, se o objetivo da mensagem é buscar o conteúdo das variáveis "1.3.6.1.2.1.1.1.0" e "1.3.6.1.2.1.7.1.0", o campo **non-repeaters** deve ser ajustado para "2" e esses dois OIDs devem ser informados no campo **variable-bindings**. Além dessas 2 variáveis, caso se quisesse buscar de forma iterativa os valores de um conjunto de variáveis sequenciais iniciando a partir da variável "1.3.6.1.2", por exemplo, esse OID também deveria ser incluído no campo **variable-bindings**. Isso seria o equivalente a enviar múltiplas mensagens **GetNextRequest**. Para a realização do ataque, entretanto, o campo **non-repeaters** é ajustado para "0", informando que apenas consultas iterativas serão realizadas.

O que vai determinar quantas variáveis serão buscadas de forma iterativa por essa mensagem é o valor inserido no campo **max-repetitions**. Quanto maior for o valor preenchido, maior será a resposta gerada pelo refletor. Contudo, deve-se ter cuidado para que a resposta não seja maior do que 65535 bytes, pois ela seria descartada por extrapolar o limite máximo de tamanho para um datagrama IP. Os campos **non-repeaters** e **max-repetitions** controlam, portanto, o comportamento da resposta enviada à vítima.

Apesar de as MIBs dos dispositivos geralmente começarem pelo objeto "1.3.6.1.2.1", para maximizar o fator de amplificação do ataque, o objeto inicial a ser buscado na MIB do refletor será o "1.3", economizando alguns bytes na consulta. Esse valor será inserido no campo **variable-bindings**. Mesmo que não tenha nada a ser buscado nessa variável, a próxima disponível no equipamento será consultada na sequência.

A figura 2.19 mostra um exemplo de mensagem **GetBulkRequest**, capturada via Wireshark em um dos testes realizados, com todos os campos acima preenchidos.

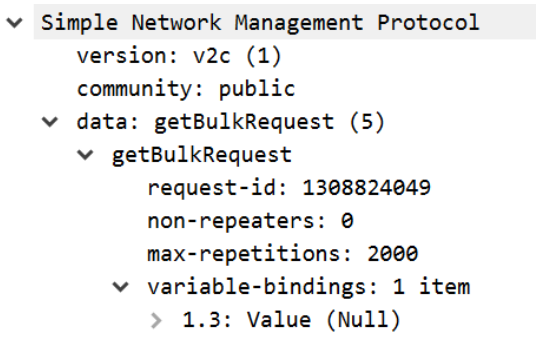

Figura 2.19: Exemplo de requisição GetBulkRequest

Como resposta, o refletor envia apenas uma mensagem do tipo **Response**. Dependendo do valor utilizado no campo **max-repetitions** da requisição, a resposta gerada poderá ser muito maior do que a requisição, ocasionando na divisão da mensagem em múltiplos fragmentos IP, fazendo com que tanto a amplificação de largura de banda como a de pacotes seja alta. A figura 2.20 ilustra um exemplo dessa mensagem, capturada via Wireshark em um dos testes realizados.

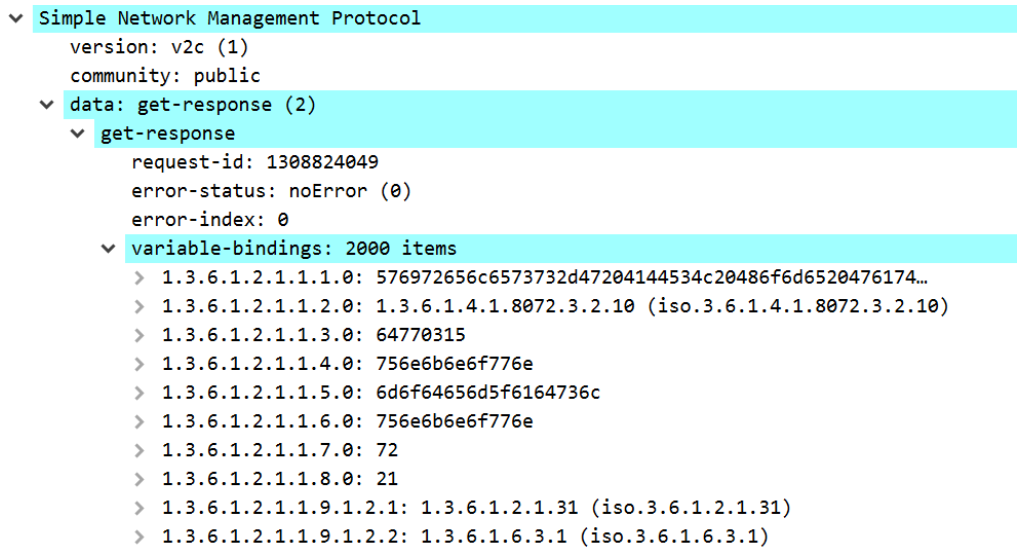

Figura 2.20: Exemplo de resposta

2.3.2.1 Resumo do ataque via SNMP

Para se conseguir melhores taxas de amplificação em ataques AR-DDoS com o SNMP, segue um resumo dos passos a serem seguidos pelo atacante. A tabela 2.2 traz uma síntese dos parâmetros que podem ser modificados na mensagem GetBulkRequest.

1. Descobrir se o refletor está rodando a versão 2c do SNMP e a comunidade que está sendo utilizada (começar os testes pela "public");

2. Gerar uma mensagem **GetBulkRequest** com um valor aleatório no campo **request-id**. Em seguida, preencher o campo **non-repeaters** com o valor "0", para permitir apenas consultas iterativas. No campo **max-repetitions**, para cada refletor utilizado nos ataques, deve-se testar vários valores, a fim de se determinar qual deles irá gerar a resposta com tamanho mais próximo de 65535 bytes, que é o limite para a mensagem não ser descartada. Por fim, o campo **variable-bindings** deve ser preenchido com o valor "1.3", pois o tamanho da mensagem será menor e, consequentemente, a amplificação será maior;

3. Gerar um datagrama UDP com a mensagem **GetBulkRequest** criada no passo 2, com uma porta de origem aleatória e a porta de destino 161 (ou a que o refletor esteja utilizando para o SNMP);

4. Adicionar o cabeçalho IPv4/IPv6 ao datagrama UDP, com o endereço de origem forjado para o IP da vítima e o endereço de destino ajustado para o IP do refletor. Em seguida, encaminhar o pacote ao refletor;

5. O refletor recebe a mensagem, a processa e responde desavisadamente com uma mensagem **Response** muito longa, provavelmente dividida em diversos fragmentos IP, ao endereço da vítima, que estava forjado no pacote.

Tabela 2.2: Resumo dos parâmetros personalizáveis na mensagem GetBulkRequest

Parâmetro	**Valor**
request-id	Aleatório
non-repeaters	0
max-repetitions	Testar até resposta ficar mais próxima de 65535 bytes
variable-bindings	1.3

2.3.3 Constrained Application Protocol (CoAP)

O protocolo CoAP [41] é utilizado para a troca de informações entre equipamentos com poder de processamento mais limitado, como *smartphones*, pontos de acesso sem fio, terminais *Gigabit Passive Optical Network* (GPON) e outros dispositivos IoT. O seu funcionamento é similar ao HTTP, porém ele é mais leve e o intercâmbio de mensagens ocorre de forma assíncrona sobre UDP na porta 5683.

O CoAP foi projetado por um grupo especializado em *Constrained RESTful Environments* (CoRE), com foco em possibilitar comunicação genérica entre dispositivos com capacidades restritas e dispositivos da Internet em geral. Ele possibilita, então, uma comunicação eficiente e simples entre nós heterogêneos na rede, otimizada especificamente para comunicação *Machine-to-Machine* (M2M).

Seguindo a estrutura de comunicação máquina a máquina, foram definidos quatro tipos de mensagens possíveis: **Confirmable - CON** (confirmável), **Non-confirmable - NON** (não confir-

mável), **Acknowledgement - ACK** (confirmada) e **Reset - RST** (redefinida). Requisições podem ser realizadas por mensagens do tipo CON, onde a confiabilidade entre a troca de informações é requerida ou do tipo NON, quando ela não é necessária. A resposta de uma requisição pode ser do tipo CON, NON ou ACK, esse último sendo utilizado para confirmar o recebimento de uma mensagem do tipo CON. E, quando um destinatário não consegue processar a requisição, ele responde com uma mensagem do tipo RST.

De forma parecida ao HTTP, o CoAP implementa quatro métodos de requisição, que são responsáveis por indicar a ação a ser executada para um dado recurso, que é identificado por uma *Uniform Resource Identifier* (URI):

- **GET:** solicita as informações de um recurso específico identificado pela URI informada. Se a requisição for bem sucedida, uma resposta do tipo "2.05 (Content)" ou "2.03 (Valid)" devem ser enviadas na resposta;
- **POST:** solicita que as informações incluídas na requisição sejam processadas;
- **PUT:** solicita que o recurso informado seja atualizado ou criado com as informações incluídas na requisição;
- **DELETE:** solicita a remoção do recurso especificado.

O ataque baseado no protocolo CoAP utliza requisições do tipo **Confirmable (CON)**, com método **GET**, pois elas requerem que uma mensagem de resposta seja enviada. A resposta é enviada à vítima no formato de uma mensagem do tipo **Acknowledgment (ACK)**, com código **2.05 Content**. A figura 2.21 ilustra o funcionamento de um ataque AR-DDoS usando CoAP.

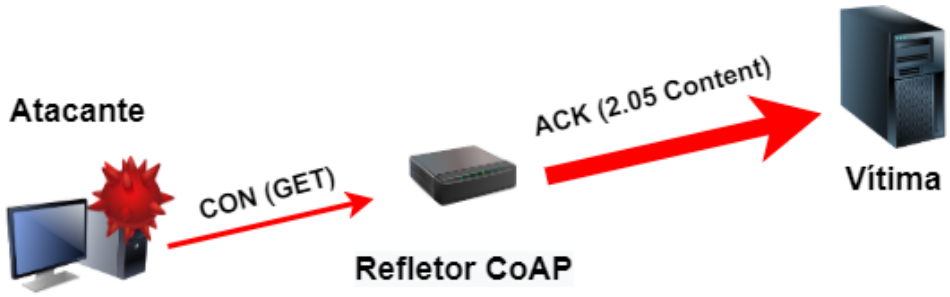

Figura 2.21: Ataque AR-DDoS com CoAP

O protocolo CoAP tem uma particularidade que permite que recursos, que são os caminhos das "URIs" disponíveis no servidor CoAP, sejam descobertos sem que se conheça como o servidor está estruturado ou configurado. Para isso, basta enviar uma mensagem com método **GET** para o recurso "/.well-known/core", que o servidor retorna uma lista com todos os recursos que possui. Com isso, o atacante pode previamente testar qual recurso do servidor irá gerar respostas com o maior tamanho possível, ou seja, com maiores taxas de amplificação.

O formato das mensagens do CoAP, tanto para as requisições quanto para as respostas, é especificado na figura 2.22.

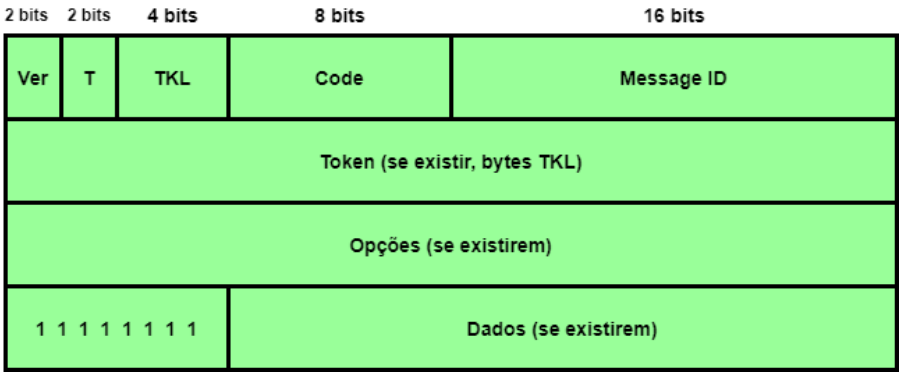

Figura 2.22: Formato das mensagens do CoAP

O campo **Version (Ver)** é preenchido com o valor "0x01 (1)", que é a única versão do CoAP disponível atualmente. **Type (T)** identifica o tipo de mensagem CoAP e deve ser ajustado com o valor "0x00 (Confirmable)".

Para o ataque ser realizado, os campos **Token Length (TKL)** e **Token** não são necessários. Desse modo, o campo **TKL** é preenchido com o valor "0x0000 (0)", indicando que não haverá nenhuma informação sobre *tokens* na mensagem.

Em requisições do tipo **Confirmable**, o campo **Code** informa o método que será utilizado na consulta e deve ser preenchido com o valor "0x01 (1)", que informa que o método será o **GET**. O valor do **Message ID** deve ser preenchido aleatoriamente, sendo que a resposta deve possuir o mesmo valor da requisição.

Para personalizar o ataque e maximizar a sua taxa de amplificação, 2 opções são utilizadas nas mensagens de requisição do ataque. A primeira opção é a **Uri-Path**, que informa qual será o recurso (URI) a ser consultado e a segunda é a **Block2**, que manipula alguns parâmetros da resposta, como o tamanho que ela deve possuir.

A opção **Uri-Path** é identificada pelo tipo 11 e o seu tamanho é variável, de acordo com o tamanho da URI. Quanto menor a URI, menor será a requisição e, consequentemente, maior será o fator de amplificação do ataque. Caso a URI possua subdiretórios, como em "/.well-known/core", cada diretório será exibido em uma opção diferente, mas todas sendo do mesmo tipo.

A opção **Block2** foi introduzida na RFC 7959 [42] e é identificada pelo tipo 23, possuindo 3 subcampos: **NUM**, **M** e **SZX**. **NUM** representa o número do bloco a ser buscado e é preenchido com "0" para indicar que seria o primeiro. [42] especifica que, em mensagens de requisição com o código **GET**, o parâmetro **M (More)** não tem função e deve ser preenchida com "0". O subcampo **SZX** indica o tamanho do bloco a ser buscado, ou seja, o tamanho que o *payload* da mensagem de resposta terá. Como o subcampo é representado em potência de 2 e está limitado a 3 bits (variando de 0 a 7 em decimal), é necessário utilizar a fórmula (2.4) para determinar o valor do bloco (**SZX**) em bytes.

$$Tam_{Bloco} = 2^{(SZX + 4)} \tag{2.4}$$

Segundo [42], o valor de **SZX** varia de 0 (bloco de 16 bytes) a 6 (bloco de 1024 bytes). O valor 7 (teoricamente de 2048 bytes) é reservado e deve retornar com uma resposta de erro com código "4.00 Bad Request". Contudo, isso depende da implementação do servidor CoAP e seria necessário testar em cada refletor o comportamento da resposta quando o valor de **SZX** for preenchido com "7".

A figura 2.23 exemplifica uma requisição com todos os campos explicados anteriormente preenchidos, capturada via Wireshark em um dos testes realizados. A mensagem é do tipo **CON** (código **"GET"**) e serve para consultar a URI "/.well-known/core", a fim de se obter uma lista com todos os recursos que o servidor possui. Nesse caso, o tamanho do bloco (resposta) foi ajustado para 2048 bytes (SZX = 7), em uma tentativa de determinar o comportamento do servidor, uma vez que este valor é reservado.

```
Constrained Application Protocol, Confirmable, GET, MID:62794
  01.. .... = Version: 1
  ..00 .... = Type: Confirmable (0)
  .... 0000 = Token Length: 0
  Code: GET (1)
  Message ID: 62794
▼ Opt Name: #1: Uri-Path: .well-known
    Opt Desc: Type 11, Critical, Unsafe
    1011 .... = Opt Delta: 11
    .... 1011 = Opt Length: 11
    Uri-Path: .well-known
▼ Opt Name: #2: Uri-Path: core
    Opt Desc: Type 11, Critical, Unsafe
    0000 .... = Opt Delta: 0
    .... 0100 = Opt Length: 4
    Uri-Path: core
▼ Opt Name: #3: Block2: NUM:0, M:0, SZX:2048
    Opt Desc: Type 23, Critical, Unsafe
    1100 .... = Opt Delta: 12
    .... 0001 = Opt Length: 1
    Block Number: 0
    .... 0... = More Flag: 0
    Block Size: 2048 (7 encoded)
  [Uri-Path: /.well-known/core]
```

Figura 2.23: Exemplo de requisição Confirmable (GET)

Como resposta a uma requisição do tipo **CON** com método **GET**, o refletor deve enviar uma mensagem do tipo **ACK** (código **"2.05 Content"**). O conteúdo do recurso consultado será enviado dentro do campo **Payload**, que terá o tamanho máximo definido pelo campo **SZX** da requisição. A figura 2.24 ilustra a resposta da requisição da figura 2.23, capturada via Wireshark. Apesar de a requisição ter sido enviada com um tamanho de bloco inválido, não foi retornada nenhuma mensagem de erro. Entretanto, como o conteúdo do recurso era de apenas 207 bytes, não

é possível afirmar, nesse caso, qual seria o comportamento se o recurso possuísse um tamanho maior do que 1024 bytes.

```
Constrained Application Protocol, Acknowledgement, 2.05 Content, MID:62794
   01.. .... = Version: 1
   ..10 .... = Type: Acknowledgement (2)
   .... 0000 = Token Length: 0
   Code: 2.05 Content (69)
   Message ID: 62794
 Opt Name: #1: Content-Format: application/link-format
      Opt Desc: Type 12, Elective, Safe
      1100 .... = Opt Delta: 12
      .... 0001 = Opt Length: 1
      Content-type: application/link-format
 Opt Name: #2: Block2: NUM:0, M:0, SZX:1024
      Opt Desc: Type 23, Critical, Unsafe
      1011 .... = Opt Delta: 11
      .... 0001 = Opt Length: 1
      Block Number: 0
      .... 0... = More Flag: 0
      Block Size: 1024 (6 encoded)
   End of options marker: 255
 Payload: Payload Content-Format: application/link-format, Length: 207
      Payload Desc: application/link-format
      [Payload Length: 207]
```

Figura 2.24: Exemplo de resposta Acknowledgment (2.05 Content)

O conteúdo do *payload* do pacote exibido na figura 2.24 está destacado na figura 2.25. É possível observar que o servidor CoAP possui os seguintes recursos: /basic, /storage, /child, /separate, /etag, /big, /encoding, /advancedSeparate, /void, /advanced, /t, /long e /xml. Cabe ao atacante determinar, em seguida, qual recurso produzirá a maior resposta.

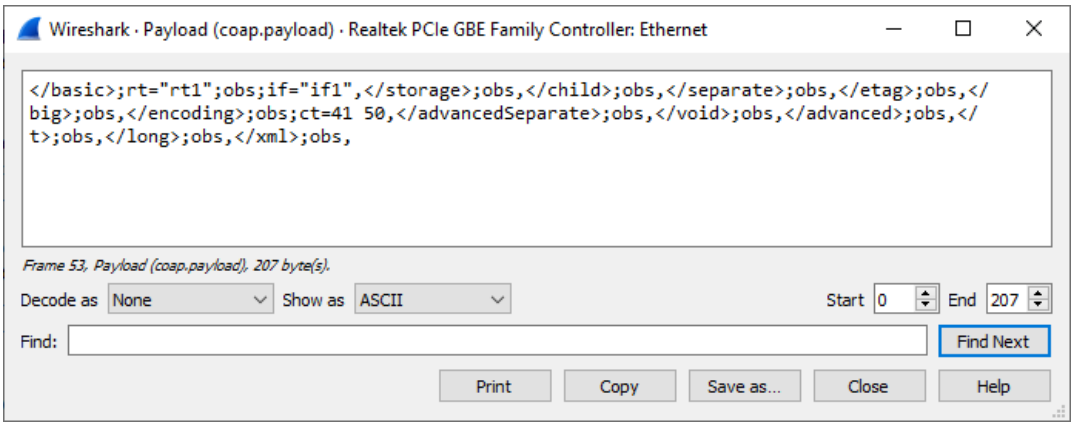

Figura 2.25: Conteúdo do *payload* da resposta da figura 2.24

Se a implementação do servidor CoAP for permissiva em relação ao campo **SZX** da requisição, aceitando o valor "7" para determinar que o tamanho do bloco pode ser de 2048 bytes, a resposta gerada poderá ter até 2104 bytes de tamanho (40 bytes do cabeçalho IPv6 + 8 bytes do

cabeçalho UDP + 8 bytes do cabeçalho CoAP + 2048 bytes de *payload*), que é um valor bem abaixo do que o SNMP pode alcançar.

A figura 2.26 exibe uma consulta ao recurso "/t" do servidor CoAP, que retornou uma resposta válida (**ACK** com código "2.05 Content") e com *payload* de 2048 bytes, apesar de o valor de **SZX** na resposta indicar que o tamanho do bloco é de apenas 1024 bytes.

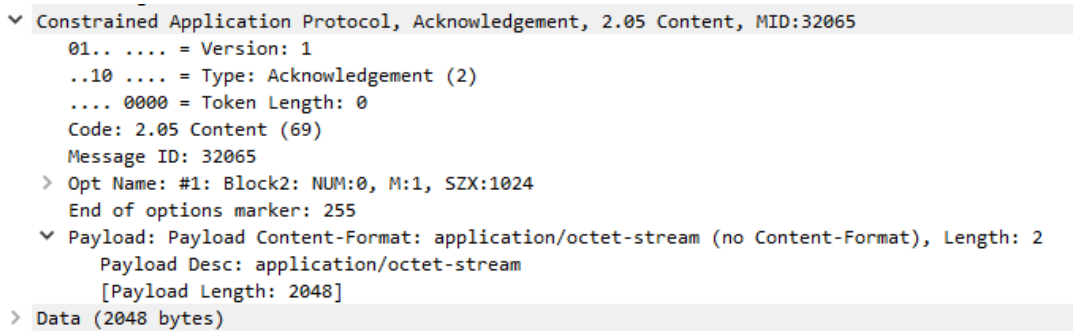

Figura 2.26: Exemplo de resposta Acknowledgment (2.05 Content) para bloco de 2048 bytes

2.3.3.1 Resumo do ataque via CoAP

Para se conseguir melhores taxas de amplificação em ataques AR-DDoS com o CoAP, segue um resumo dos passos a serem seguidos pelo atacante. A tabela 2.3 traz uma síntese dos parâmetros que podem ser modificados na mensagem Confirmable (CON).

1. Descobrir os recursos existentes no servidor CoAP enviando uma mensagem de requisição para a URI "/.well-known/core" e determinar o que possui maior tamanho;
2. Testar se o servidor aceita e gera respostas válidas para requisições com o campo **SZX** configurado para "7" (2048 bytes);
3. Gerar uma mensagem **CON** com método **GET**, com valor de **Message ID** aleatório para cada pacote. Em seguida, adicionar a opção do tipo 11 (**Uri-Path**) preenchida com o valor do recurso a ser consultado e a opção do tipo 23 (**Block2**) com valor de **SZX** ajustado para "6" (1024 bytes) ou "7" (2048 bytes), caso o servidor aceite;
4. Gerar um datagrama UDP com a mensagem **CON** criada no passo 3, com porta de origem aleatória e a porta de destino 5683 (ou a que o refletor esteja utilizando para o CoAP);
5. Adicionar o cabeçalho IPv4/IPv6 ao datagrama UDP, com o endereço de origem forjado para o IP da vítima e o endereço de destino ajustado para o IP do refletor. Em seguida, encaminhar o pacote ao refletor;
6. O refletor recebe a mensagem, a processa e responde desatentamente com uma mensagem do tipo **ACK** ao endereço da vítima, que estava forjado no pacote.

Tabela 2.3: Resumo dos parâmetros personalizáveis na mensagem Confirmable

Parâmetro	Valor
Message ID	Aleatório
Uri-Path	<caminho-do-recurso>
SZX	6 ou 7 (se suportado pelo servidor)

2.4 INTERNET DAS COISAS (IOT) EM ATAQUES DDOS

Com a popularização e a crescente expansão de dispositivos IoT no mundo inteiro, estes se tornaram alvos para a realização de ataques DDoS, o que contribuiu para um acréscimo na quantidade desses ataques [43], principalmente em função de não possuírem muitos mecanismos de segurança embutidos [44].

Desde a chegada do Mirai [10], que mostrou o potencial do emprego de dispositivos IoT em ataques DDoS, novas formas de ataques utilizando estes dispositivos surgiram. O artigo [45] cita alguns exemplos, entre eles um ataque a um cassino através de um termômetro de um aquário conectado à Internet e a descoberta de vulnerabilidades em dispositivos cardíacos, como marca-passos e desfibriladores.

No que diz respeito aos ataques DDoS, existem basicamente duas abordagens em relação à utilização de dispositivos IoT. A primeira, e provavelmente mais difundida, é através da criação de grandes *botnets*, como a da figura 2.27, contendo centenas de milhares desses equipamentos, que disparam ataques a alvos específicos. Para isso, no entanto, é necessário ter acesso aos dispositivos para controlá-los de maneira centralizada pelo atacante, geralmente infectando-os com algum *malware*, como o Mirai. Dessa forma, o atacante poderá lançar um ataque coordenado e sincronizado para atingir o seu objetivo.

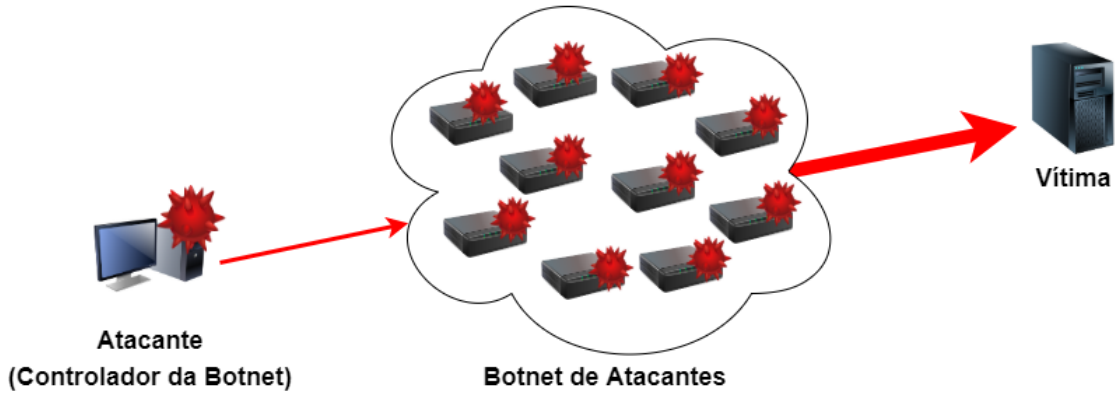

Figura 2.27: Botnet de dispositivos IoT

A segunda abordagem, na qual esse trabalho se baseia, utiliza dispositivos IoT como refletores em ataques de reflexão amplificada, como na figura 2.28. Nesse caso, o esforço de preparação do ataque é menor, pois não há a necessidade de se infectar ou controlar os dispositivos que atuam

como refletores, bastando descobrir equipamentos que rodem serviços baseados em UDP com potencial de amplificação para executar o ataque [22]. Ferramentas de busca como o Shodan[1], contudo, facilitam essa procura por eventuais refletores.

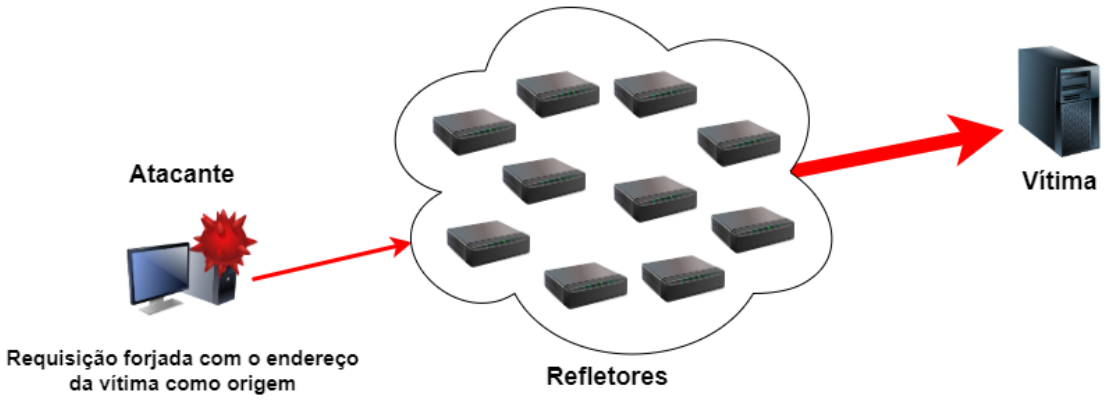

Figura 2.28: Dispositivos IoT atuando como refletores em ataques AR-DDoS

É possível ainda unir as duas abordagens em um só ataque, ou seja, utilizando *botnets* de dispositivos IoT para lançar ataques de reflexão amplificada em direção a outros dispositivos IoT que atuam como refletores, conforme a figura 2.29. Nesse caso, além do atacante precisar criar e controlar uma botnet de dispositivos IoT, será necessário localizar possíveis refletores e programar o ataque nos dispositivos da botnet. Por outro lado, essa técnica aumenta consideravelmente o poder do ataque e dificulta a sua mitigação, devido os dispositivos estarem muito distribuídos geograficamente.

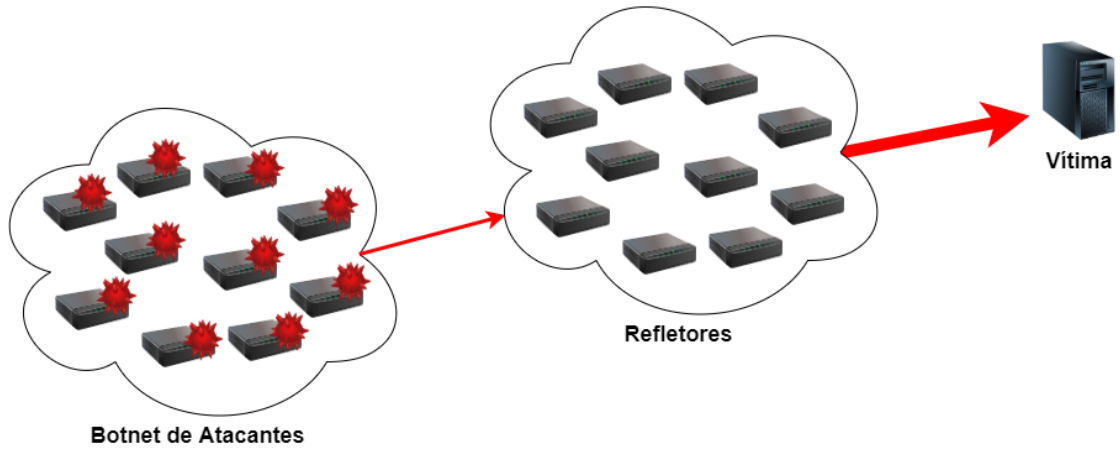

Figura 2.29: Dispositivos IoT em botnets e atuando como refletores em ataques AR-DDoS

[1]https://www.shodan.io/

2.5 TRABALHOS CORRELATOS

Há significativa produção referente à segurança em IoT, ataques DDoS e AR-DDoS, inclusive sobre IoT, mas estes últimos em menor escala. Apesar disso, poucos trabalhos na literatura abordam a utilização de dispositivos IoT como refletores em ataques AR-DDoS ou fazem uma análise da sua saturação durante a realização desses ataques, o que demonstra a relevância dessa pesquisa para compreender melhor o comportamento de tais dispositivos no decorrer de ataques reais.

A seguir, um breve resumo das principais publicações envolvendo os temas abordados nessa dissertação será apresentado.

2.5.1 Segurança em IoT

A segurança de IoT em geral e de dispositivos em particular tem se focado nos aspectos de prospecção de vulnerabilidades, aspectos de privacidade e da implementação de mecanismos de segurança em dispositivos IoT.

[46] descreveu os ataques mais comuns encarados por dispositivos IoT de consumidores e sugeriu algumas estratégias de mitigação para as ameaças encontradas, analisando os desafios da sua adoção.

[47] revisou as principais ameaças referentes aos dispositivos IoT, em suas diversas camadas: aplicação, percepção, rede e física. Em cada uma delas, elencou as questões de segurança relacionadas e as formas de proteção existentes.

[48] abordou as preocupações mais importantes que impedem a adoção em larga escala de dispositivos IoT. Os pontos estão relacionados à interoperabilidade, gerenciamento, segurança e privacidade em IoT.

[49] analisou as principais pesquisas relacionadas à segurança em IoT entre 2016 e 2018, assim como as tendências e questões pendentes acerca desse assunto, provendo um resumo do estado da arte de segurança em IoT e das principais ferramentas e simuladores utilizados.

[50] reviu as preocupações em torno da privacidade e segurança em dispositivos IoT, de forma a assegurar princípios como a confidencialidade, integridade, autenticação e controle de acesso exatos e assertivos entre estes dispositivos.

[51] fez uma revisão exaustiva das vulnerabilidades existentes em ambientes IoT, analisando pesquisas entre 2010 e 2018, classificando-as em diversas dimensões dentro do paradigma IoT e provendo uma taxonomia única, ressaltando a severidade dessas ameaças no contexto de IoT.

[52] utilizou técnicas de inteligência artificial e aprendizagem de máquinas para propor esquemas de autenticação, controle de acesso e detecção de *malwares* para privacidade de dados em dispositivos IoT. O trabalho também discutiu os desafios que precisam ser tratados para implementar esses esquemas em sistemas IoT práticos.

[53] mostrou como alguns ambientes de IoT podem se beneficiar de mecanismos de *Blockchain* para garantir a segurança de seus dispositivos.

2.5.2 Ataques DDoS

Os ataques DDoS são uma realidade na Internet desde o final da década de 90 e existe uma infinidade de pesquisas acerca do tema, principalmente envolvendo a sua detecção e mitigação.

[1] descreveu os modelos de ataques DDoS e apresentou uma taxonomia para caracterizar o escopo desses ataques, assim como os atributos das ferramentas de ataque utilizadas e das contramedidas disponíveis. A taxonomia mostrou similaridades e padrões em diferentes ataques DDoS e ferramentas e auxilia no desenvolvimento de soluções para conter esses ataques.

[2] realizou uma análise sistemática dos ataques DDoS, incluindo as suas motivações e evoluções, examinando as diferentes técnicas de ataque, prevenção e mitigação encontradas, assim como as possíveis limitações e desafios das pesquisas existentes.

[31] apresentou uma abordagem estruturada dos principais tipos de ataque DDoS e mecanismos de defesas existentes. As características de cada ataque e do seu respectivo método de defesa são descritas e as vantagens e desvantagens acerca delas são discutidas.

[54] analisou os três principais mecanismos de defesa existentes para ataques DDoS: detecção, mitigação e rastreamento de IP de origem. Em seguida, propôs um mecanismo para defesa de ataques DDoS localizados nas camadas de rede e aplicação, através do uso de listas de controle de acesso e *firewall*.

[55] resumiu as soluções de detecção e mitigação de ataques DDoS baseadas em Redes Definidas por *Software* (SDN, em inglês) e apresentou uma arquitetura para detecção e mitigação de ataques DDoS em uma rede de larga escala que engloba uma cidade inteligente construída em uma infraestrutura SDN.

[56] propôs um mecanismo que, além de detectar a presença de um ataque DDoS, também identifica a rota do ataque e inicia um processo de mitigação ainda no estágio inicial de identificação do ataque. A proposta utiliza um algoritmo de classificação integrado com um sistema de prevenção de intrusão para impedir a ação do ataque.

[57] apresentou uma nova técnica para autenticação de origem de pacotes em uma rede baseada em esquemas de verificação de assinatura designadas, que pode ser utilizada para prevenir ataques de negação de serviço. Como ela não é baseada em infraestruturas de chaves públicas, mas sim maquinário criptográfico leve, pode ser utilizado inclusive em ambientes IoT.

2.5.3 Ataques AR-DDoS

O estudo dos ataques AR-DDoS em geral motivou este trabalho no contexto de IoT e tem por foco a caracterização comportamental, principalmente do refletor.

[12] analisou quatorze protocolos de aplicação baseados no UDP que podem ser utilizados para gerar ataques amplificados, incluindo o SSDP e o SNMP. Determinou-se, em seguida, os fatores de amplificação de cada um deles, tanto em *bytes* quanto em pacotes. Do mesmo autor, [32] abordou os ataques DDoS amplificados utilizando protocolos de aplicação baseados no TCP, que são menos comuns para esse tipo de ataque.

[58] analisou o resultado de 1000 dias de coleta de tráfego de ataques DDoS amplificados através de uma rede de *honeypots* de refletores UDP, entre julho de 2014 e julho de 2017. Foi explorado o ciclo de vida dos ataques que utilizaram os refletores, desde a fase de escaneamento usada para detectar as máquinas até o seu uso nos ataques.

[59] desenvolveu uma metodologia para medir as propriedades de dispositivos individuais que participam em ataques DDoS, como limite de tráfego, fator de amplificação e velocidade, que permitem avaliar o grau de contribução de cada dispositivo para o ataque.

[60] propôs uma arquitetura de defesa distribuída para lidar com ataques AR-DDoS na rede. O método permite detectar se pacotes não solicitados de ataque estão sendo enviados para uma vítima dentro da rede de um provedor de serviços. Uma vez detectado, os roteadores na borda da rede automaticamente bloqueia as origens indevidas.

[20] e [33] realizaram ataques AR-DDoS em ambientes controlados, sendo que, no primeiro, os equipamentos rodavam os protocolos SSDP e SNMP e, no segundo, rodavam SNMP, SSDP, NTP e DNS. Em ambos os casos, os dispositivos do atacante, refletor e vítima foram interconectados através de um *switch* e as saturações de todos eles foram analisadas, à medida em que a quantidade de pacotes por segundo aumentava ao longo dos ataques. Os resultados indicaram que os refletores saturaram com uma taxa relativamente baixa de pacotes por segundo. No entanto, nenhum dos trabalhos utilizou dispositivos IoT como refletores nos testes, mas sim computadores convencionais.

2.5.4 Ataques DDoS/AR-DDoS em Ambientes IoT

O *malware* Mirai marca o emprego massivo de dispositivos IoT em *botnets* para a execução de ataques DDoS, sendo amplamente estudado. Em outra linha, encontramos a caracterização de AR-DDoS abusando de dispositivos IoT.

[11] e [61] descreveram como a utilização de dispositivos IoT podem potencializar o poder destrutivo de ataques DDoS, através da utilização de *malwares*, como o Mirai, para a formação de *botnets* com milhares desses dispositivos.

[43] investigou diversos protocolos utilizados em ataques DDoS envolvendo dispositivos IoT, entre o primeiro trimestre de 2013 e o primeiro trimestre de 2015, e chegou à conclusão que o SSDP, com 20,7%, foi o protocolo mais utilizados nesses ataques.

[62] analisou a eficiência de um ataque DDoS por reflexão amplificada em um ambiente IoT,

através de simulações realizadas no simulador de redes NS-3[2], utilizando o protocolo CoAP, mas sobre uma rede baseada no padrão *IPv6 over Low power Wireless Personal Area Networks* (6LoWPAN). Os resultados indicaram que um ataque nesse ambiente seria viável, mas requereria um alto número de redes IoT para ser efetivo.

[63] estendeu o trabalho realizado por [62], analisando o impacto do ataque DDoS de acordo com alguns parâmetros, como tamanho do pacote, número de pacotes e taxa de injeção de tráfego. Chegou-se à conclusão de que, para se obter um ataque efetivo, é necessário um bom entendimento da capacidade dos dispositivos IoT, assim como dos serviços disponíveis, os protocolos implementados e outras propriedades. Também observou-se que, do ponto de vista do atacante, um aumento na taxa de injeção de pacotes não necessariamente reflete um aumento do fator de amplificação do ataque.

Esses dois trabalhos, contudo, não utilizaram equipamentos reais nos testes, mas sim um simulador de redes, o que pode tornar os resultados menos suscetíveis a variações em virtudes de más implementações de *software* do dispositivo, por exemplo. Além disso, se limitaram a testar apenas o protocolo CoAP e utilizando um padrão encontrado apenas em equipamentos com menor capacidade computacional, como sensores.

[64] comparou o comportamento de quatro dispositivos IoT (Raspberry Pi, Google Home Assistant, câmera IP e impressora de rede) atuando como refletores durante ataques DDoS reais por reflexão, em redes IPv4 e IPv6, para analisar a potência destes ataques. Foi utilizado o *software* Scapy[3] para a geração de pacotes UDP, com endereços IP de origem forjados e portas UDP de destino inexistentes. Como os dispositivos IoT utilizados não estavam rodando serviços nas portas testadas, o tráfego refletido analisado pelo trabalho foi apenas o de mensagens ICMP/ICMPv6 do tipo *Port unreachable*. O estudo avaliou que há potencial de ataques dessa natureza envolvendo dispositivos IoT, tanto para IPv4 quanto para IPv6, e que, em redes IPv6, os ataques são mais eficientes e mais esmagadores do que em IPv4, em virtude do cabeçalho IPv6 possuir o dobro de tamanho do IPv4.

Entretanto, apesar deste trabalho ter utilizado equipamentos IoT como refletores, os testes não se preocuparam com a saturação dos dispositivos nem com a utilização de protocolos de aplicação que rodam sobre UDP, como o SSDP, por exemplo.

[21] avaliou o potencial de reflexão de oito dispositivos IoT domésticos, como câmeras IP, impressora, tomada, lâmpada, entre outros, utilizando três protocolos distintos (SSDP, SNMP e TCP). Foram medidos os fatores de amplificação, a capacidade de tráfego e a duração dos ataques em cada um dos equipamentos até o momento em que eles saturaram. Além disso, o trabalho utilizou um *malware* para enviar um comando UPnP ao *gateway* da rede, para permitir um ataque DDoS por reflexão amplificada a partir da Internet utilizando um dispositivo IoT interno como refletor. O comando habilitou um redirecionamento de porta para o dispositivo, de forma que requisições fossem direcionadas a uma porta específica no equipamento. Por fim, os oito dispo-

[2]https://www.nsnam.org/

[3]https://scapy.net/

sitivos foram distribuídos entre três residências para realizar um ataque combinado a uma vítima durante 24 horas para demonstrar a capacidade de amplificação do ataque.

O trabalho, no entanto, utilizou equipamentos de baixa capacidade computacional e com pouco suporte aos protocolos testados. Apenas dois deles rodavam o protocolo SNMP e metade suportava o SSDP. O envio de mensagens TCP SYN foi o único possível de ser realizado em todos os dispositivos, mas com baixas taxas de amplificação.

[65] analisou os doze maiores ataques DDoS realizados até a presente data para verificar quais deles poderiam utilizar dispositivos IoT como refletores. Para isso, pacotes similares aos utilizados nos ataques foram manualmente criados e enviados para um dispositivo de teste para examinar o seu comportamento. O dispositivo foi capaz de responder aos pacotes referentes aos protocolos ICMP, TCP e SSDP, sendo que o último obteve a maior taxa de amplificação. Por fim, foi feita uma projeção de um ataque utilizando 240 milhões de dispositivos IoT, onde se chegou a uma taxa de 36,28 Tbps, o que corresponde a um volume de tráfego 16 vezes superior ao do maior ataque DDoS registrado até hoje. Todavia, o estudo utilizou apenas um dispositivo e não se preocupou em avaliar a sua saturação durante os ataques.

[66] utilizou a ferramenta de busca Shodan para localizar 106 câmeras IoT (54 câmeras IP e 52 *webcams*) e 95 Gravadores de Vídeo Digital (DVR, na sigla em inglês) ao redor do mundo. Em seguida, foram realizados testes para determinar se estes estavam protegidos por firewall ou não, indicando que aproximadamente 84% dos dispositivos não estavam. Por fim, os dispositivos encontrados foram usados como refletores em ataques AR-DDoS reais. Entretanto, o único tipo de ataque testado foi o de TCP SYN e com uma taxa fixa de 10 pacotes para cada aparelho, não se preocupando em avaliar a saturação destes equipamentos.

[16] avaliou os possíveis impactos de ataques AR-DDoS em dispositivos IoT utilizando uma metodologia própria para prospeção ativa de vulnerabilidades desenvolvida por eles, chamada *Decision-Oriented Tool-Agnostic* (DOTA). Para comprovar essa metodologia, foi realizado um ataque AR-DDoS, em ambiente controlado, abusando do protocolo SNMP. Apesar de ser voltado para o ambiente IoT, o dispositivo utilizado no teste, contudo, foi um computador convencional.

[22] e [23], derivados da presente pesquisa, por sua vez, utilizaram a mesma metodologia de [20] e [33] para avaliar o comportamento de refletores com dispositivos IoT durante ataques AR-DDoS. O primeiro utilizou três dispositivos IoT diferentes como refletores (Raspberry Pi, *gateway* ADSL e câmera IP), todos rodando os protocolos SSDP e SNMP em IPv4, enquanto que o segundo utilizou um Raspberry Pi, rodando o protocolo CoAP tanto em IPv4 quanto em IPv6.

2.5.5 Resumo dos Trabalhos Correlatos

De forma a correlacionar essa pesquisa com os principais trabalhos encontrados na literatura, a tabela 2.4 resume as características e os resultados obtidos nos respectivos trabalhos correlatos e a tabela 2.5 compara os protocolos e informações mais relavantes encontradas nessa pesquisa com os demais trabalhos que possuem foco em AR-DDoS.

Tabela 2.4: Resumo dos Principais Trabalhos Correlatos

Trabalho	Foco	Escopo	Metodologia	Resultados
[58]	Análise de tráfego AR-DDoS em *honeypots*	65 nós rodando protocolos em UDP entre 2014 e 2017	Análise do ciclo de vida dos ataques, desde o escaneamento até o ataque	Média de 1450 escaneamentos por dia. DNS e NTP foram os protocolos mais utilizados
[59]	Criação de metodologia para medir a participação individual em ataques	Prova de conceito com NTP (mas pode ser aplicada a outros protocolos)	Cálculo de envio de 5Kbps a 31389 servidores NTP identificados na Internet	Até 20,55x de amplificação e 134 Gbps de tráfego total com a metodologia proposta
[20]	Análise de saturação de refletores em ataques	1 PC como refletor (SSDP e SNMP)	Rajada de pacotes crescentes a cada 10s	Saturação com baixa taxa de requisições/s
[33]	Análise de saturação de refletores em ataques	1 PC como refletor (SSDP, SNMP, NTP e DNS)	Rajada de pacotes crescentes a cada 10s	Saturação com baixa taxa de requisições/s
[43]	Análise de ataques DDoS prévios com IoT	Ataques realizados entre 2013 e 2015	Análise estatística de dados	SSDP foi o protocolo mais utilizado nos ataques (20,7%)
[62]	Análise da eficiência de um ataque AR-DDoS com IoT	Simulação com NS-3 (CoAP sobre 6LoWPAN)	Simulação com 5, 25, 50, 75 e 100 nós, com até 100 pacotes/s	Fator de amplificação de 20x, mas tráfego máximo de 16 Kbps
[63]	Extensão do trabalho [62], avaliando outros aspectos	Simulação com NS-3 (CoAP sobre 6LoWPAN)	Simulação com 1, 3, 5, 50 e 100 nós, com até 200 pacotes/s	Até 64 Kbps de tráfego gerado
[64]	Avaliação do potencial de ataques com dispositivos IoT	4 dispositivos IoT como refletores isoladamente (UDP, IPv4/IPv6)	Geração de tráfego UDP com portas aleatórias gerando reflexão com erros ICMP	Há potencial para o ataque e IPv6 gera mais tráfego refletido que IPv4

Trabalho	Foco	Escopo	Metodologia	Resultados
[21]	Avaliação do potencial de reflexão de dispositivos IoT domésticos	8 dispositivos IoT como refletores (SSDP, SNMP e TCP SYN)	Simulação de ataques interno e pela Internet	Fator de amplificação de 20x durante 24h com taxa de 1,2 Mbps para a vítima
[65]	Determinar o poder de um ataque DDoS usando dispositivos IoT	Análise dos 12 maiores ataques no último trimestre de 2015	Avaliou o fator de amplificação de 1 ataque com SSDP e multiplicou pela estimativa de dispositivos IoT existentes	Estimou que o ataque poderia chegar a 36,28 Tbps com 24 bilhões de dispositivos
[66]	Mostrar o potencial e a vulnerabilidade de dispositivos IoT em ataques AR-DDoS	106 câmeras IoT e 95 DVRs encontrados na Internet como refletores (TCP SYN)	Envio de 10 pacotes para cada dispositivo	75% dos dispositivos refletiu tráfego para a vítima
[16]	Análise de saturação de refletores em ataques	1 PC como refletor (SNMP)	Rajada de pacotes crescentes a cada 10s	Saturação com baixa taxa de requisições/s
[22]	Análise de saturação de refletores em ataques	3 dispositivos IoT como refletores isoladamente (SSDP e SNMP)	Rajada de pacotes crescentes a cada 10s	Saturação com baixa taxa de requisições/s
[23]	Análise de saturação de refletores em ataques	1 dispositivo IoT como refletor (CoAP)	Rajada de pacotes crescentes a cada 10s	Saturação com baixa taxa de requisições/s

É possível notar, analisando a tabela 2.5, que nenhum dos trabalhos encontrados na literatura, até onde se tem conhecimento, avalia os mesmos aspectos que este e de forma ampla, envolvendo diversos protocolos, equipamentos e características do ataque, tornando esta pesquisa relevante para a comunidade científica.

Tabela 2.5: Comparação entre os Principais Trabalhos Correlatos com Foco em AR-DDoS

Trabalho	SSDP	SNMP	CoAP	IPv4	IPv6	IoT
[58]	X	X		X		
[59]	X	X		X		
[20]	X	X		X		
[33]	X	X		X		
[43]	X	X		X		X
[62]			X		X*	X
[63]			X		X*	X
[64]				X	X	X
[21]	X	X		X		X
[65]	X			X		X
[66]				X		X
[16]		X		X		
[22]	X	X		X		X
[23]			X	X	X	X

* 6LoWPAN

2.6 RESUMO DO CAPÍTULO

Este capítulo discorreu sobre os principais conceitos utilizados nessa dissertação. Primeiramente, foi abordada como a técnica de falsificação de endereços IPs é empregada em ataques DoS/DDoS. Em seguida, foram explicados como esses ataques funcionam, assim como a sua taxonomia, destacando o tipo de ataque DDoS envolvido nesta pesquisa, o por reflexão amplificada (AR-DDoS), e as fórmulas para calcular os fatores de amplificação de *bytes* e pacotes, que são utilizadas para medir a potência de um ataque AR-DDoS.

O funcionamento dos três protocolos utilizados nessa dissertação (SSDP, SNMP e CoAP) foram apresentados, bem como a forma como eles são utilizados em ataques AR-DDoS, explicando como a modificação de alguns parâmetros dos pacotes de requisição afetam o tamanho da resposta que chega na vítima.

Foi visto ainda como os dispositivos IoT vêm sendo utilizados em ataques DDoS, tanto como atacantes, geralmente no formato de *botnets* com centenas de milhares de dispositivos, quanto como refletores, que é o caso em estudo neste trabalho.

Por fim, foi feita uma revisão dos principais trabalhos análogos a esta pesquisa, correlacionando alguns deles com esta.

Todos esses conceitos são importantes para compreender melhor o próximo capítulo, que explica em detalhes como os testes foram conduzidos, assim como o funcionamento da ferramenta utilizada nos ataques.

3 MATERIAIS E MÉTODOS

Os testes consistiram em executar o ataque usando os protocolos SSDP, SNMP e CoAP, em condições controladas, com o objetivo de caracterizar o comportamento de saturação dos refletores. Para este fim, utilizou-se uma ferramenta desenvolvida como implementação de referência dos ataques de interesse, bem como suporte metodológico para a realização dos testes.

A seguir são descritas a ferramenta, a metodologia e os procedimentos empregados.

3.1 FERRAMENTA LINDERHOF

Para realizar os testes desse trabalho, foi utilizada a ferramenta **Linderhof** [1], desenvolvida em linguagem C por estudantes de Engenharia da Computação da Universidade de Brasília (UnB), com o objetivo de servir como apoio em pesquisas relacionadas aos ataques AR-DDoS produzidas na referida Universidade.

A ferramenta começou a ser desenvolvida por [67], com a construção de toda a arquitetura e implementação do primeiro "espelho", do Memcached. Houve uma preocupação de deixá-la o mais modular possível, para que sua expansão acontecesse de forma simplificada. Isso permitiu que, posteriormente, os protocolos NTP [68], DNS [69] e CoAP [70] fossem integrados à ela.

Contudo, esse crescimento aconteceu na forma de projetos independentes compartilhando um núcleo de código comum e, a partir desse trabalho, foi iniciado um esforço em conjunto com dois alunos de graduação em Engenharia da Computação da UnB[2] para que toda a ferramenta passasse por uma reorganização arquitetural, padronização, documentção e novas funcionalidades fossem acrescentadas a ela, permitindo um maior controle e personalização dos ataques. Além disso, os protocolos SNMP e SSDP também foram incorporados ao Linderhof, que chegou à sua versão 1.0.0. Todas as melhorias desenvolvidas nessa versão estão detalhadas na seção 3.1.2.

3.1.1 Arquitetura

A arquitetura do Linderhof, ilustrada na figura 3.1, é composta por 4 módulos principais, descritos a seguir:

- ***Interface*:** é o responsável pela interação com o usuário, recebendo como entrada os parâ-

[1] Linderhof é o nome de um palácio real alemão, construído no século XIX, no sudoeste da região da Baviera. A ferramenta possui esse nome em alusão à sua Galeria dos Espelhos (*Hall of Mirrors*), cujos espelhos refletiam e amplificavam a luz das velas dos candelabros que ali existiam, de forma similar ao que ocorre com os pacotes nos ataques AR-DDoS. Nesse sentido, cada protocolo implementado na ferramenta representa um espelho diferente da sua galeria.

[2] Amanda Lopes Dantas e Matheus de Oliveira Vieira atuaram como programadores, e Alan como líder de projeto e testador das funcionalidades da ferramenta.

metros do ataque e devolvendo na saída um resumo dos pacotes gerados e enviados;

- ***Commander*:** é o módulo onde ocorre o planejamento do ataque, de acordo com o protocolo (espelho) selecionado e os parâmetros escolhidos pelo usuário;
- ***Hall of Mirrors*:** é o encarregado pela geração dos pacotes e pela iniciação dos ataques. Também é onde os espelhos são adicionados à ferramenta;
- ***Injector*:** é onde o ataque é, de fato, realizado, ocorrendo o envio dos pacotes para o(s) refletor(es) e a geração de um resumo dos pacotes gerados e enviados.

Cada um desses módulos foi dividido em outros submódulos, para que a divisão de tarefas ficasse mais otimizada. Nas próximas subseções, as funções de cada um deles serão detalhadas.

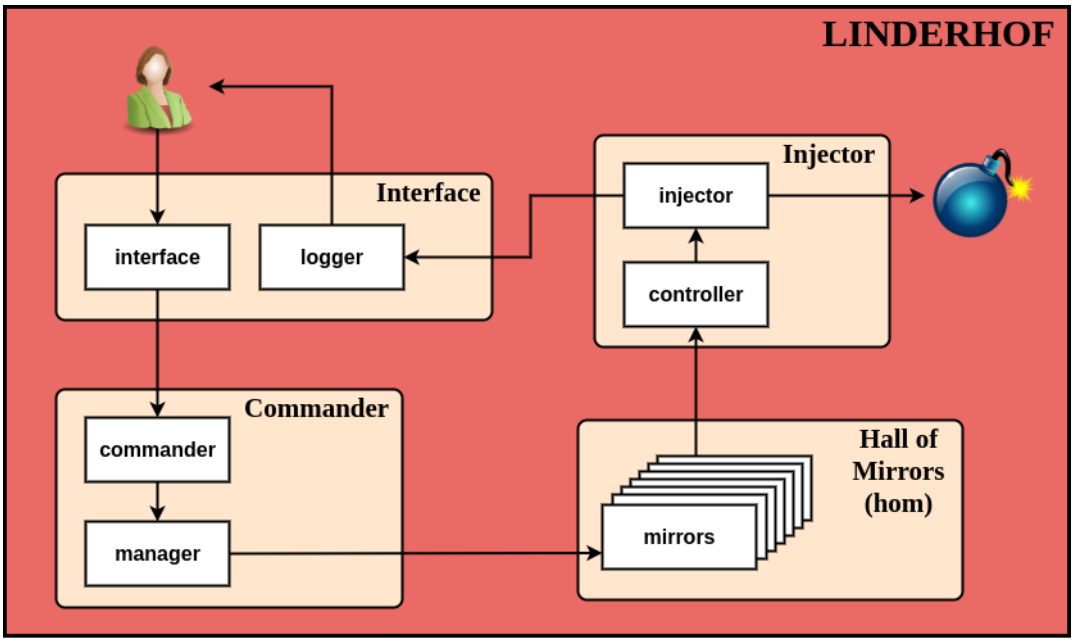

Figura 3.1: Arquitetura do Linderhof

3.1.1.1 Interface

Após a inicialização do programa por parte do usuário, o submódulo **interface** é chamado para ler os parâmetros e argumentos passados via linha de comando ou via arquivo de configuração, ou então assumir os valores padrões dos parâmetros necessários para o ataque em questão. Com a ajuda de dois analisadores sintáticos, um para cada formato de entrada, é criada a estrutura do ataque, que posteriormente é enviada ao submódulo **commander**.

A tabela 3.1 contem uma lista com todos os parâmetros globais do Linderhof, ou seja, os que podem ser utilizados com qualquer espelho, enquanto que a tabela 3.2 possui a lista de parâmetros específicos para cada espelho implementado na ferramenta, ambos para utilização via linha de comando. A tabela 3.3, por sua vez, mostra um exemplo de arquivo de configuração com todos os parâmetros que podem ser utilizados.

Tabela 3.1: Parâmetros globais do Linderhof

Parâmetro Curto	Parâmetro Longo	Parâmetro Obrigatório	Argumento Obrigatório	Argumento Padrão	Descrição
-m	--mirror	Sim	Sim	<Nenhum>	Espelho a ser utilizado
-t	--target	Sim	Sim	<Nenhum>	Endereço IP da vítima
-r	--reflector	Sim	Sim	<Nenhum>	Endereço IP do refletor ou arquivo com refletores
-g	--targport	Não	Sim	Aleatório (40000-60000)	Porta de origem da requisição
-p	--reflecport	Não	Sim	Padrão do Espelho	Porta de destino da requisição
-l	--level	Não	Sim	1	Nível do ataque
-d	--timer	Não	Sim	Ilimitado	Duração do ataque (em segundos)
-i	--inc	Não	Sim	Ilimitado	Duração de cada nível (em segundos)
-c	--config	Não	Sim	linderhof.conf	Arquivo de configuração
-a	--aggressive	Não	Não	-----	Modo agressivo
-f	--flood	Não	Não	-----	Modo *flooding*
-h	--help	Não	Não	-----	Ajuda

Tabela 3.2: Parâmetros dos espelhos do Linderhof

Parâmetro Curto	Parâmetro Longo	Parâmetro Obrigatório	Argumento Obrigatório	Argumento Padrão	Descrição
-D	--domain-name	Não	Sim	ddos.dns.com	DNS - Nome de domínio
-V	--upnp-version	Não	Sim	1.0	SSDP - Versão do UPnP
-U	--unicast	Não	Não	-----	SSDP - M-SEARCH no formato unicast
-C	--community-string	Não	Sim	public	SNMP - Comunidade
-R	--max-repetitions	Não	Sim	2000	SNMP - Campo max-repetitions
-Z	--szx	Não	Sim	6	CoAP - Campo SZX
-P	--uri-path	Não	Sim	/.well-known/core	CoAP - Caminho da URI

Tabela 3.3: Exemplo de arquivo de configuração do Linderhof

```
# Linderhof configuration
# These settings will be replaced with subsequent command line arguments.

[General]
# Mirror type (string)
MIRROR=
# Target IP (string)
TARGET=
# Target port (integer)
TARGET_PORT=
# Reflector IP (string)
REFLECTOR=
# Reflector port (integer)
REFLECTOR_PORT=
# Attack level (integer)
LEVEL=1
# Attack duration (integer)
DURATION=
# Increment attack delay (integer)
INCREMENT=
# Set aggressive mode on (boolean)
AGGRESSIVE=false
[DNS]
# Domain (string)
DOMAIN_NAME=ddos.dns.com

[SSDP]
# UPnP Version (1.0, 1.1 or 2.0)
UPNP_VERSION=1.0
# Set host field to unicast address (boolean)
UNICAST=false

[SNMP]
# Community String field (string)
COMMUNITY_STRING=public
# Max-Repetitions field (integer)
MAX_REPETITIONS=2000

[CoAP]
# SZX field (integer) (0-7)
SZX=6
# Uri-Path field (string)
URI_PATH=/.well-known/core
```

Durante a realização do ataque, um resumo das opções escolhidas e a quantidade de pacotes gerados e enviados em cada nível e para cada um dos refletores vai sendo exibida em tela, conforme a figura 3.2. O submódulo responsável por isso é o **logger**, que recebe essas informações do **injector**.

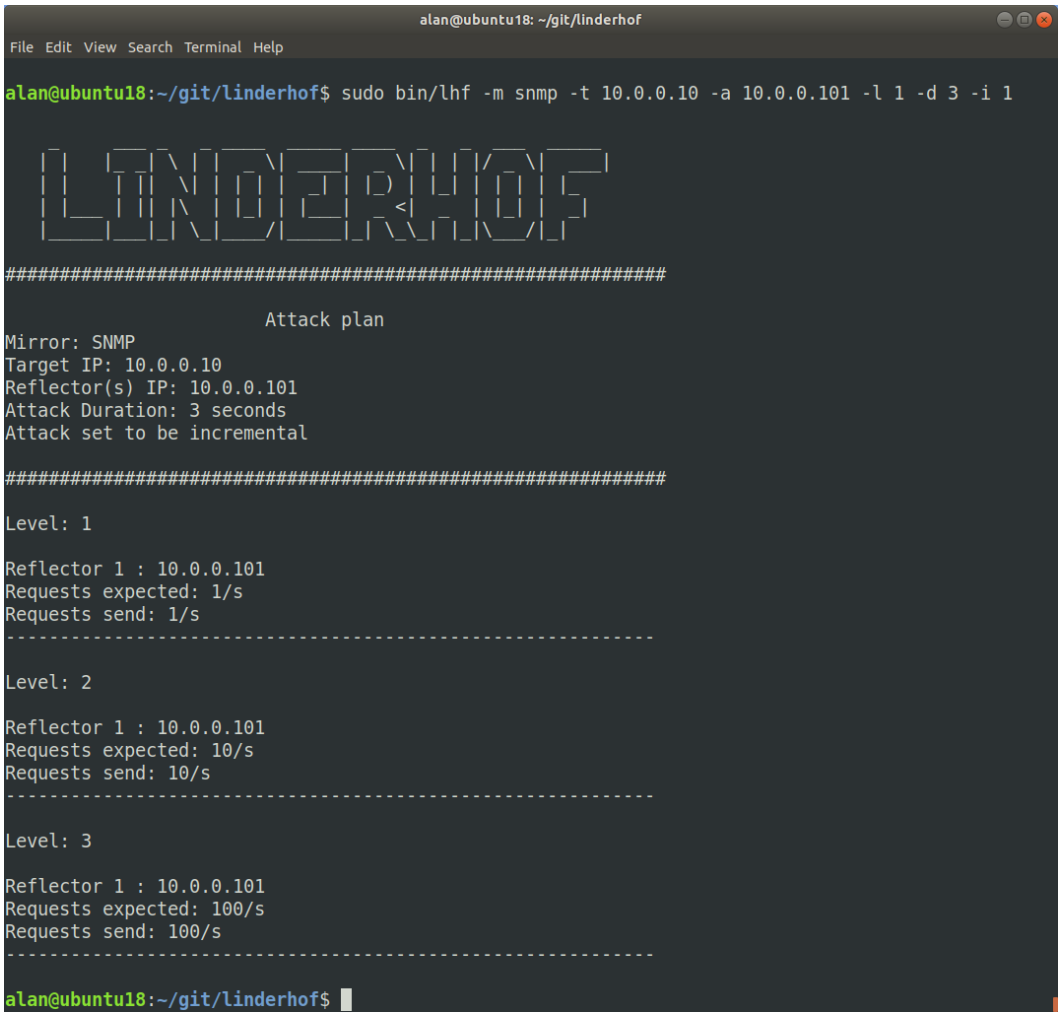

Figura 3.2: Saída em tela do Linderhof

3.1.1.2 Commander

Neste módulo são executadas as principais funções de planejamento do ataque. Após o submódulo **commander** receber a estrutura do ataque, ele faz uma validação desses dados e depois aciona o submódulo **manager**, que vai montar um plano de ataque, de acordo com o espelho escolhido e a estrutura recebida. Posteriormente, o plano de ataque será enviado ao espelho correspondente no módulo ***hall of mirrors*** **(hom)**.

3.1.1.3 Hall of Mirrors (hom)

Após receber o plano de ataque, o espelho irá gerar o pacote, de acordo com o formato de mensagem definido nele. Cada protocolo existente na Galeria dos Espelhos (Memcached, NTP, DNS, CoAP, SNMP e SSDP) é adicionado em seu respectivo diretório e contem as suas próprias funções e formatos de mensagens.

O espelho montará primeiro a mensagem de requisição do seu protocolo, que corresponderá à parte de Dados do pacote, e, em seguida, serão acrescentados os cabeçalhos UDP e IP (IPv4 ou IPv6). Cabe lembrar que o endereço IP de origem do pacote será forjado com o IP da vítima, de forma a possibilitar a reflexão do ataque, utilizando a técnica de IP *spoofing*, conforme visto na seção 2.2.2. O cabeçalho da camada de enlace varia de acordo com a tecnologia de interconexão existente na rede e, portanto, ele é adicionado pela interface de rede do equipamento do atacante e não pela ferramenta. A figura 3.3 ilustra a montagem de um pacote SNMP feita pelo Linderhof, com os respectivos tamanhos de cada parte do pacote.

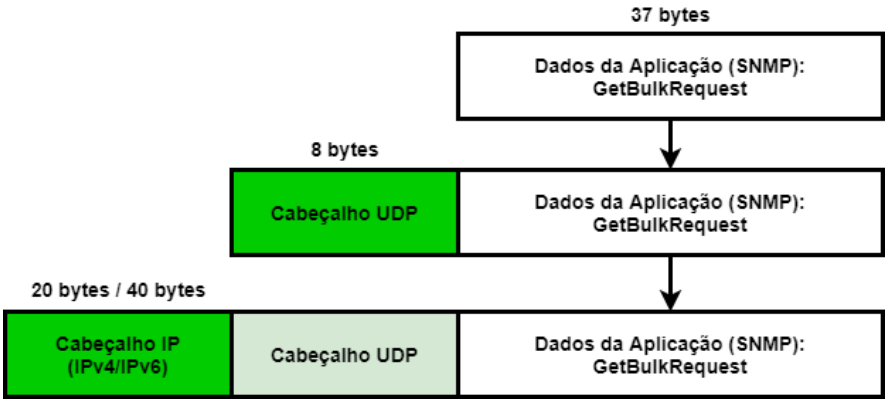

Figura 3.3: Pacote SNMP gerado pelo Linderhof

Por fim, o espelho inicia a execução do ataque, encaminhando ao submódulo **controller** o pacote gerado para envio ao(s) refletor(es).

3.1.1.4 Injector

Uma vez que o ataque é iniciado pelo espelho, o submódulo **controller** será responsável pelo gerenciamento das *threads* dos injetores, ou seja, ele informa ao submódulo **injector** quando elas devem ser criadas ou destruídas, sendo que será criada uma *thread* para cada injetor e cada refletor utilizado no ataque usará um injetor (*thread*) diferente.

Além disso, o **controller** também regula a quantidade e a taxa de injeção de pacotes de cada injetor, através da utilização de um balde (*bucket*), de acordo com as especificações de duração do ataque, o seu nível de intensidade e se o mesmo será incremental ou não. Para isso, o **controller** informa ao **injector** quantos pacotes devem ser enviados a cada intervalo de tempo determinado.

O submódulo **injector** é quem envia efetivamente os pacotes a(os) refletor(es). A quantidade

de pacotes enviadas por cada injetor varia em função do nível de intensidade do ataque, que é baseado em potência de 10, de acordo com a fórmula (3.1), onde Pps significa "Pacotes por segundo". O Linderhof possui 10 níveis disponíveis (1 a 10) e cada um possui uma intensidade diferente, que vai crescendo de forma exponencial, conforme a tabela 3.4.

$$Pps = 10^{\text{nível-1}} \tag{3.1}$$

Tabela 3.4: Pacotes por segundo gerados em cada nível de ataque do Linderhof

Nível	**Pacotes por Segundo**
1	1
2	10
3	100
4	1.000
5	10.000
6	100.000
7	1.000.000
8	10.000.000
9	100.000.000
10	1.000.000.000

Além desses 10 níveis, foi adicionado um modo inundação (*flooding*), através do parâmetro "-f", que não possui um balde limitando a quantidade de pacotes a serem enviados, ou seja, a cada segundo a ferramenta injeta "infinitos" pacotes na rede.

É imprescindível destacar, no entanto, que a quantidade de pacotes por segundo enviada em um ataque representa apenas a taxa de injeção desejada, ou seja, não há garantias de que o equipamento do atacante conseguirá gerar e enviar todos esses pacotes e nem que eles chegarão ao refletor ou conseguirão ser refletidos por ele.

Em um ataque incremental, a ferramenta passará para o nível seguinte quando o tempo definido no parâmetro "-i <tempo_em_s>" for atingido. Quando a ferramenta chegar ao nível 10 e ainda houver incrementos a serem realizados, ela permanecerá nele até o término do ataque.

O submódulo **injector** também é responsável por enviar informações sobre a geração e o envio de pacotes para o submódulo **logger**, que as exibirá ao usuário.

3.1.2 Melhorias Desenvolvidas

Na versão 1.0.0 do Linderhof, desenvolvida para este trabalho, foi feita uma revisão completa da arquitetura e do código, assim como foram realizadas várias melhorias no funcionamento da ferramenta e acrescentadas novas funcionalidades, destacadas a seguir:

- **Remoção de código morto:** o código foi inteiramente revisto e as partes que não eram

utilizadas foram removidas;

- **Padronização e documentação de todo o código-fonte:** todas as funções foram revistas e algumas tiveram seus nomes ajustados para refletir os seus objetivos ou movidas para outro submódulo, por terem mais correlação. As demais partes do código também foram revistas e documentadas com comentários que explicam o seu funcionamento;

- **Padronização e ajustes dos nomes dos arquivos e diretórios:** alguns nomes de módulos e submódulos não seguiam um padrão lógico e outros possuiam nomes de deuses da mitologia romana. Foi feito um ajuste completo em todos os arquivos e diretórios da ferramenta, para que refletissem a sua função principal e o Linderhof ficasse com uma apresentação mais profissional;

- **Adição do espelho SSDP:** foi implementado o suporte ao protocolo SSDP, que permite a geração de requisições do tipo M-SEARCH. É possível personalizar o ataque utilizando o parâmetro "-V" para definir a versão do UPnP do refletor (1.0, 1.1 ou 2.0) e o parâmetro "-U" para indicar que a mensagem M-SEARCH será do tipo *unicast* (o padrão é *multicast*);

- **Adição do espelho SNMP:** com a inclusão do protocolo SNMP à ferramenta, mensagens do tipo GetBulkRequest podem ser geradas e enviadas aos refletores. Através da utilização dos parâmetros "-C <comunidade_snmp>" e "-R <max_repetitions>" pode-se personalizar o ataque, de acordo com o cenário encontrado;

- **Ajustes no suporte a múltiplos refletores:** anteriormente havia sido implementado um suporte parcial a múltiplos refletores, onde era necessário definí-los diretamente no código e compilá-lo novamente a cada alteração. Com a nova versão, os refletores são adicionados em um arquivo de texto e o mesmo é referenciado através do parâmetro "-r <arquivo>". O modo de balanceamento padrão é o colaborativo, onde a quantidade máxima de pacotes a serem enviados em cada nível é dividida igualmente entre todos os refletores. Caso a divisão não seja igual, os primeiros enviarão alguns pacotes a mais, até que se atinja o total do nível;

- **Adição do modo agressivo:** quando múltiplos refletores forem referenciados e esse modo estiver ativado, através do parâmetro "-a", a ferramenta irá gerar, para cada um dos refletores, a quantidade máxima de pacotes prevista para cada nível, ao invés de atuar de modo colaborativo, que é o padrão;

- **Adição de suporte ao protocolo IPv6:** além de realizar ataques informando os endereços IPv4 de refletor(es) e vítima, também é possível direcioná-los aos seus endereços IPv6 nessa versão;

- **Adição de suporte a arquivo de configuração:** ao invés de inserir vários parâmetros via linha de comando para personalizar o ataque, agora é possível fazer a personalização de todas as opções do ataque em um único arquivo de configuração e referenciá-lo através do parâmetro "-c <arquivo>". Se o parâmetro for utilizado, mas nenhum argumento for inserido, será buscado o arquivo **linderhof.conf**;

- **Adição de novos parâmetros para personalização dos ataques:** além dos parâmetros citados anteriormente para o SSDP e o SNMP, também foram adicionados outros nos espelhos DNS e CoAP para alterar itens específicos das mensagens de requisição dos respectivos protocolos, a fim de se flexibilizar o ataque e obter taxas maiores de amplificação, de acordo com os refletores utilizados;

- **Ajustes no método de atribuição da porta de origem dos pacotes:** ao invés de se utilizar a mesma porta fixa de origem para todos os pacotes enviados em todos os níveis do ataque, como acontecia anteriormente, optou-se pela geração de portas de origem aleatórias em cada um dos níveis do ataque, para dificultar a sua mitigação, assim como para facilitar a separação dos fluxos de cada nível a serem analisados em trabalhos como esse.

3.2 METODOLOGIA DOS TESTES

Para simular os ataques AR-DDoS com os três protocolos descritos na seção 2.3 através do Linderhof, os testes foram realizados em um ambiente controlado, seguindo a mesma metodologia adotada nos artigos [16], [20], [33], [22] e [23], detalhada a seguir.

Foram montados alguns cenários, com pelo menos três dispositivos em cada (atacante, refletor e vítima), interligados por um *switch*. Isso significa que todos os testes foram realizados com cabos UTP categoria 5e, com capacidade máxima de transmissão de 1Gbps.

Todos os refletores possuíam capacidade computacional limitada, sendo classificados como dispositivos IoT. Quando suportado pelo refletor, os ataques foram realizados tanto em IPv4 quanto em IPv6, para efeito de comparação, sendo que o IP de origem dos pacotes no atacante foram forjados com o IP da vítima.

Os ataques foram executados com diferentes intensidades (níveis 1 a 7 do Linderhof). Cada nível de intensidade tinha a duração de 10 segundos e, assim que esse período encerrava, a ferramenta automaticamente passava para o nível seguinte, aumentando exponencialmente a quantidade de pacotes enviados por segundo, de acordo com a equação (3.1).

Em alguns casos, em virtude de divergência de resultados ou comportamentos anômalos, uma metodologia alternativa foi utilizada, onde, ao invés de a ferramenta passar automaticamente para o próximo nível após os 10 segundos com apenas 1 comando, foram rodados 7 comandos individuais, um para cada nível. A diferença é que antes de rodar o comando do nível seguinte, foram esperados que todos os pacotes refletidos naquele nível chegassem à vítima.

Foram realizadas capturas de tráfego em todos os equipamentos envolvidos, diretamente nas interfaces de rede dos dispositivos, quando possível, via tcpdump ou Wireshark, sendo desprezados o custo computacional e a interferência na camada de enlace eventualmente causados pelo analisador de protocolos. Em seguida, elas foram analisadas para se levantar a quantidade de bytes e pacotes enviados ou recebidos pelos dispositivos, a fim de se descobrir os valores de FAB

(equação (2.1)) e FAP (equação (2.2)), assim como o momento em que eles saturavam, ou seja, que não eram mais capazes de atingir a taxa máxima de envio de pacotes/bytes de cada nível.

As próximas subseções detalharão os dispositivos e softwares utilizados nos testes, assim como os cenários simulados e os procedimentos de testes adotados em cada cenário.

3.2.1 Equipamentos Utilizados

Ao todo, foram utilizados seis dispositivos durante os testes, com poderes computacionais variados, sendo um atacante, responsável por gerar os ataques utilizando a ferramenta Linderhof, enviando as requisições para os refletores; três refletores, equipamentos que rodam os serviços explorados pelo atacante para atingir a vítima; uma vítima, dispositivo alvo do atacante, recebendo o tráfego amplificado dos refletores; e um *switch*, equipamento de rede utilizado para interligar todos os demais, quando necessário.

- **Atacante (*Desktop*):** Foi o equipamento mais robusto utilizado, sendo um computador do tipo *desktop* rodando Ubuntu Linux 18.04 LTS, com a seguinte configuração:
  - Intel Core i9-9900KS @ 4.00GHz
  - 32GB DDR4 @ 2666MHz RAM
  - Adaptador GigabitEthernet Intel I219-V
  - Ubuntu 18.04 LTS x64
  - Linderhof 1.0.0
  - tcpdump
- **Refletor 01 (*Gateway* ADSL):** o primeiro refletor utilizado foi um *gateway* ADSL doméstico mais antigo (lançado em 2006), cujo modelo é o Linksys WAG200G. O dispositivo rodava os serviços SSDP e SNMP e possuía a seguinte configuração:
  - MIPS 4KEc V4.8 @ 211MHz
  - 16MB RAM
  - 4 portas FastEthernet
  - 1 porta ADSL
- **Refletor 02 (Câmera IP):** o segundo refletor utilizado foi uma câmera IP corporativa, cujo modelo é o Axis M1125. O dispositivo rodava os serviços SSDP e SNMP e possuía a seguinte configuração:
  - 512MB RAM
  - 256MB flash
  - 1 porta FastEthernet PoE

- **Refletor 03 (Raspberry Pi):** o terceiro refletor utilizado foi um Raspberry Pi 3 B+, com os serviços SSDP, SNMP e CoAP rodando em uma versão adaptada do Debian para o equipamento, chamado de Raspbian. O dispositivo possuía a seguinte configuração:
  - Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
  - 1GB LPDDR2 SDRAM
  - Adaptador GigabitEthernet sobre USB 2.0 (máximo de 300 Mbps)
  - Raspbian Stretch Lite
  - MiniUPnPd 1.8
  - snmpd 5.7.3
  - CoAPthon 4.0.2
  - tcpdump
- **Vítima:** Foi utilizado um notebook HP EliteBook 840 G1 rodando Windows 10 apenas com o Wireshark ativado, coletando o tráfego recebido. O equipamento possuía a seguinte configuração:
  - Intel Core i5-4300U @ 1.90GHz
  - 8GB DDR3 @ 1600MHz RAM
  - Adaptador GigabitEthernet Intel I218-LM
  - Windows 10 Pro x64 versão 1809
  - Wireshark 3.2.3
- **Switch (Enterasys C3G124-48P):** Foi utilizado o modelo Enterasys C3G124-48P, que possuía 48 portas 10/100/1000Mbps com suporte a *Power over Ethernet* (PoE), que permitiu utilizar o refletor 02 sem fonte de alimentação.

3.2.2 Cenários

Foram montados três cenários distintos para simular os ataques, cada um contendo um refletor diferente (*Gateway* ADSL, câmera IP e Raspberry Pi).

No cenário 1 (figura 3.4) foi utilizado como refletor o *gateway* ADSL, que suportava os protocolos SSDP e SNMP, mas apenas em IPv4. No cenário 2 (figura 3.5) foi utilizado como refletor a câmera IP, que suportava os mesmos protocolos que o *gateway* ADSL, mas o servidor SSDP só rodava em IPv4, enquanto que o SNMP rodava em IPv4 e IPv6. No cenário 3 (figura 3.6) foi utilizado como refletor o Raspberry Pi. Dos refletores testados, esse era o mais robusto e o único capaz de suportar todos os protocolos testados (SSDP, SNMP e CoAP), tanto em IPv4 quanto em IPv6.

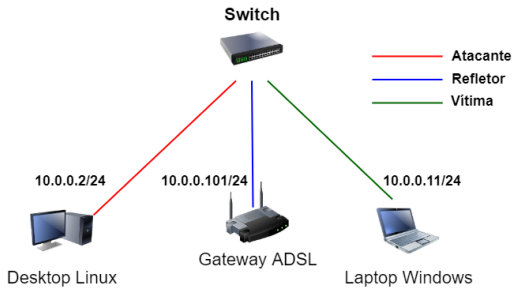

Figura 3.4: Cenário 1

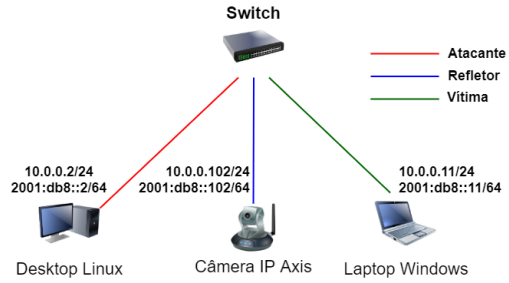

Figura 3.5: Cenário 2

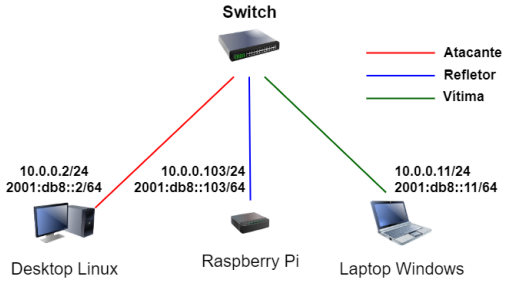

Figura 3.6: Cenário 3

Os protocolos suportados por cada um deles estão resumidos na tabela 3.5.

Tabela 3.5: Resumo dos equipamentos utilizados e protocolos suportados

Equipamentos	**SSDP (IPv4)**	**SSDP (IPv6)**	**SNMP (IPv4)**	**SNMP (IPv6)**	**CoAP (IPv4)**	**CoAP (IPv6)**
Gateway ADSL	X		X			
Câmera IP	X		X	X		
Raspberry Pi	X	X	X	X	X	X

3.2.3 Procedimentos de Testes

Os experimentos foram conduzidos de forma similar nos três cenários, mas devido os refletores serem implementados de formas distintas e possuírem capacidades computacionais variadas, foi necessário verificar quais parâmetros dos protocolos SSDP, SNMP e CoAP deveriam ser ajustados para se obter o máximo fator de amplificação possível durante os ataques. A seguir serão detalhados os procedimentos utilizados nos testes de cada um dos cenários.

3.2.3.1 Cenário 1

O *gateway* ADSL rodava tanto servidores SSDP quanto SNMP, mas com algumas particularidades. Por ser muito antigo, ele não suportava o protocolo IPv6, então todos os testes foram realizados apenas em IPv4. Além disso, o servidor SSDP só suportava a versão 1.0 do UPnP e,

portanto, as mensagens M-SEARCH tiveram que ser do tipo *multicast*, com campo HOST preenchido com o endereço "239.255.255.250" e o campo MX, que agora passou a ser obrigatório, preenchido com o valor "1". O campo ST permaneceu preenchido com "ssdp:all". Cabe ressaltar que, apesar de a mensagem ser no formato *multicast*, ela foi enviada via *unicast* para o endereço IPv4 do *gateway*. A porta UDP de destino identificada no refletor foi a padrão (1900).

O servidor SNMP encontrado no equipamento era compatível com a versão 2c do protocolo e rodava na sua porta UDP padrão (161). A comunidade de leitura existente também era a "public" e foram testados vários valores para o campo max-repetitions, chegando-se ao número "7758".

Os comandos do Linderhof para gerar os ataques do cenário 1 foram os seguintes:

bin/lhf -m ssdp -t 10.0.0.11 -r 10.0.0.101 -l 1 -d 70 -i 10 -V 1.0

bin/lhf -m snmp -t 10.0.0.11 -r 10.0.0.101 -l 1 -d 70 -i 10 -C public -R 7758

Enquanto que os comandos do Linderhof para gerar os ataques do cenário 1 com a metodologia alternativa foram os seguintes, trocando **X** por valores de 1 a 7, de acordo com o nível testado:

bin/lhf -m ssdp -t 10.0.0.11 -r 10.0.0.101 -l **X** -d 10 -V 1.0

bin/lhf -m snmp -t 10.0.0.11 -r 10.0.0.101 -l **X** -d 10 -C public -R 7758

3.2.3.2 Cenário 2

Assim como o *gateway* ADSL, a câmera IP igualmente suportava os protocolos SSDP e SNMP com algumas particularidades. Apesar de ser compatível com o protocolo IPv6, somente era possível atribuir um endereço através de um servidor DHCPv6. Para isso, foi configurado um servidor DHCPv6 na máquina do atacante, de forma similar ao tutorial encontrado em [71].

Nesse dispositivo, o servidor SSDP também só suportava a versão 1.0 do UPnP e os campos HOST, MX e ST das mensagens M-SEARCH foram preenchidos de forma idêntica aos das mensagens enviadas ao *gateway* ADSL. As mensagens foram igualmente enviadas via *unicast* para o endereço IPv4 da câmera e para a mesma porta UDP de destino (1900).

O servidor SNMP da câmera era compatível com a versão 2c do protocolo e rodava na sua porta UDP padrão (161), com a comunidade de leitura padrão "public". O servidor era muito mais limitado e, a partir do valor "40" do campo max-repetitions, o tamanho da resposta não variava mais de tamanho.

Os comandos do Linderhof para gerar os ataques do cenário 2 foram os seguintes:

bin/lhf -m ssdp -t 10.0.0.11 -r 10.0.0.102 -l 1 -d 70 -i 10 -V 1.0

bin/lhf -m snmp -t 10.0.0.11 -r 10.0.0.102 -l 1 -d 70 -i 10 -C public -R 40

bin/lhf -m snmp -t 2001:db8::11 -r 2001:db8::102 -l 1 -d 70 -i 10 -C public -R 40

3.2.3.3 Cenário 3

O servidor SSDP utilizado no Raspberry Pi foi o **MiniUPnPd 1.8**, sendo compatível com a versão 1.1 do UPnP. Com isso, foi possível enviar mensagens do tipo M-SEARCH via *unicast*, economizando-se alguns bytes na requisição, por possuírem menos campos obrigatórios. O campo ST foi preenchido com o valor "ssdp:all" e o campo HOST com o IPv4 (10.0.0.101) ou o IPv6 (2001:db8::101) do Raspberry, de acordo com o protocolo IP utilizado no ataque. A porta UDP de destino identificada no refletor foi a padrão (1900).

Para rodar o protocolo SNMP, foi utilizado o servidor **snmpd 5.7.3**, compatível com a versão 2c do protocolo SNMP. A comunidade de leitura existente era a padrão "public". Foram testados vários valores para o campo max-repetitions da mensagem GetBulkRequest, a fim de se obter o maior pacote de resposta, e o valor que gerou melhores resultados foi o "2650". A porta UDP de destino identificada no refletor foi a padrão (161).

O servidor CoAP foi disponibilizado no refletor através do software **CoAPthon 4.0.2**, que roda em cima de Python. Após testes iniciais, foi constatado que o servidor aceitava requisições com valor de SZX igual a 7. Desse modo, foi adicionado a ele um recurso que possuía mais de 2048 bytes de tamanho, para simular que a resposta fosse a maior possível, e o caminho do recurso (URI) foi configurado para ter apenas 1 caractere, chamado de “/r”, para que o tamanho da requisição fosse a menor possível. A porta UDP de destino identificada no refletor foi a padrão (5683).

Os comandos do Linderhof para gerar os ataques do cenário 3 foram os seguintes:

bin/lhf -m ssdp -t 10.0.0.11 -r 10.0.0.103 -l 1 -d 70 -i 10 -U -V 1.1

bin/lhf -m ssdp -t 2001:db8::11 -r 2001:db8::103 -l 1 -d 70 -i 10 -U -V 1.1

bin/lhf -m snmp -t 10.0.0.11 -r 10.0.0.103 -l 1 -d 70 -i 10 -C public -R 2650

bin/lhf -m snmp -t 2001:db8::11 -r 2001:db8::103 -l 1 -d 70 -i 10 -C public -R 2650

bin/lhf -m coap -t 10.0.0.11 -r 10.0.0.103 -l 1 -d 70 -i 10 -Z 7 -P /r

bin/lhf -m coap -t 2001:db8::11 -r 2001:db8::103 -l 1 -d 70 -i 10 -Z 7 -P /r

Enquanto que os comandos do Linderhof para gerar os ataques do cenário 3 com a metodologia alternativa foram os seguintes, trocando **X** por valores de 1 a 7, de acordo com o nível testado:

bin/lhf -m snmp -t 10.0.0.11 -r 10.0.0.103 -l **X** -d 10 -C public -R 2650

bin/lhf -m snmp -t 2001:db8::11 -r 2001:db8::103 -l **X** -d 10 -C public -R 2650

3.3 RESUMO DO CAPÍTULO

Este capítulo apresentou a ferramenta utilizada nos testes dessa dissertação, detalhando a sua arquitetura e como os seus módulos se relacionam uns com os outros, explicando ainda como ela pode ser utilizada para gerar ataques AR-DDoS e todos os parâmetros disponíveis para personalizá-los. Ao final, todas as melhorias e correções desenvolvidas na ferramenta para esta pesquisa foram resumidas.

Em seguida, a metodologia utilizada nos testes foi esmiúçada. Foram apresentados os equipamentos utilizados, com as suas respectivas especificações técnicas. Os três cenários preparados para os experimentos foram descritos, assim como os protocolos suportados em cada um deles, tanto para IPv4 quanto para IPv6. Ao final, foram explicados os procedimentos dos testes, sendo mostrados os parâmetros utilizados em cada pacote de requisição, assim como os comandos executados com a ferramenta.

O próximo capítulo apresenta os resultados obtidos nos testes de cada cenário, incluindo os cálculos dos fatores de amplificação, explicando e discutindo o que significa cada um, comparando-os com trabalhos passados.

4 RESULTADOS DOS TESTES

As capturas de tráfego realizadas em cada equipamento durante os ataques foram analisadas e, com os resultados obtidos, foram montados os gráficos e as tabelas a seguir, que serão detalhados ao longo de cada cenário nas subseções seguintes, para que se compreenda melhor o comportamento de cada refletor durante os ataques AR-DDoS efetuados.

4.1 RESULTADOS

Nos cenários 1 e 2 (*gateway* ADSL e câmera IP), devido aos dispositivos possuírem sistema operacional embarcado, não foi possível realizar a captura de tráfego diretamente neles e, portanto, a saturação foi medida a partir do que chegava na vítima em relação ao que foi enviado pelo atacante. No cenário 3 (Raspberry Pi), no entanto, foi possível realizar as capturas diretamente no refletor e analisar melhor o comportamento dele durante os ataques.

4.1.1 Cenário 1 (*Gateway* ADSL)

Este cenário possuía o dispositivo mais antigo e com poder computacional mais limitado. O equipamento não era compatível com IPv6 e todos os testes foram realizados apenas com IPv4.

4.1.1.1 SSDP

No ataque com o protocolo SSDP (tabela 4.1), houve saturação do atacante apenas a partir do nível 6 da ferramenta Linderhof, onde era esperado o envio de 100.000 pacotes por segundo, sendo gerados pelo atacante apenas 70.540,8 pacotes/s.

O comportamento do refletor, no entanto, foi o que mais supreendeu, pois a partir do nível 3, ele sequer foi capaz de gerar algum pacote de resposta para a vítima, sofrendo um ataque de negação de serviço, que não era o objetivo do atacante, já que o alvo era a vítima. Além disso, o ataque, que deveria ter durado em torno de 10 segundos por nível na vítima, levou mais do que o triplo do tempo para ser concluído em cada nível, conforme se observa na tabela 4.1.

A saturação do refletor ocorre ainda no nivel 2, pois ele não conseguiu gerar uma quantidade de *bytes* e consequentes taxas de Kbps e pacotes/s compatíveis com esse nível, conforme se observa nas figuras 4.1 e 4.2.

Como os pacotes gerados pelo atacante possuíam portas UDP de origem aleatórias e altas, quando o trafégo chegava na vítima (dessa vez como portas de destino), o sistema operacional não as reconhecia como estando abertas e gerava mensagens ICMP "*port unreachable*" como resposta

para o refletor, ocasionando em ainda mais informações a serem processadas por ele. A tabela 4.2 traz a quantidade de pacotes ICMP gerados pela vítima, assim como o total de *bytes* dessas mensagens, em cada nível. É possível notar que, o tráfego gerado pela vítima foi exatamente o mesmo recebido por ela. Isso se deve porque, de acordo com a RFC 1812 [72], as mensagens ICMP devem conter o máximo de informação possível da mensagem original, sem extrapolar o limite de 576 *bytes*, que corresponde ao MTU mínimo do IPv4. No caso do IPv6, o limite é de 1280 *bytes*. Uma vez que as mensagens SSDP geradas por esse refletor estavam abaixo de 400 *bytes*, era possível inserir toda a mensagem original no pacote ICMP.

Durante as capturas, também se observou um comportamento atípico deste refletor com o protocolo SSDP. Apesar de o serviço rodar na porta UDP/1900, as respostas tinham como porta de origem sempre a UDP/2051. Todavia, o conteúdo dos pacotes era de tráfego SSDP legítimo, não inviabilizando o ataque.

Tabela 4.1: Cenário 1 (SSDP) - IPv4

	Saída do Atacante			**Entrada da Vítima**			
Nível	***Bytes***	***Kbps***	***Pacotes/s***	***Bytes***	***Kbps***	***Pacotes/s***	***Duração (s)***
1	1220	1,0	1,0	107840	28,1	9,1	30,75
2	12200	9,8	10,0	118624	26,8	8,7	35,41
3	122000	97,6	100,0	0	0,0	0,0	0,0
4	1220000	976,0	1000,0	0	0,0	0,0	0,0
5	1220000	9760,0	10000,0	0	0,0	0,0	0,0
6	86059776	68847,8	70540,8	0	0,0	0,0	0,0
7	808488632	646790,9	662695,6	0	0,0	0,0	0,0

Tabela 4.2: Cenário 1 (SSDP) - IPv4 - ICMP

	Saída da Vítima	
Nível	***Pacotes ICMP***	***Bytes ICMP***
1	280	107840
2	308	118624
3	0	0
4	0	0
5	0	0
6	0	0
7	0	0

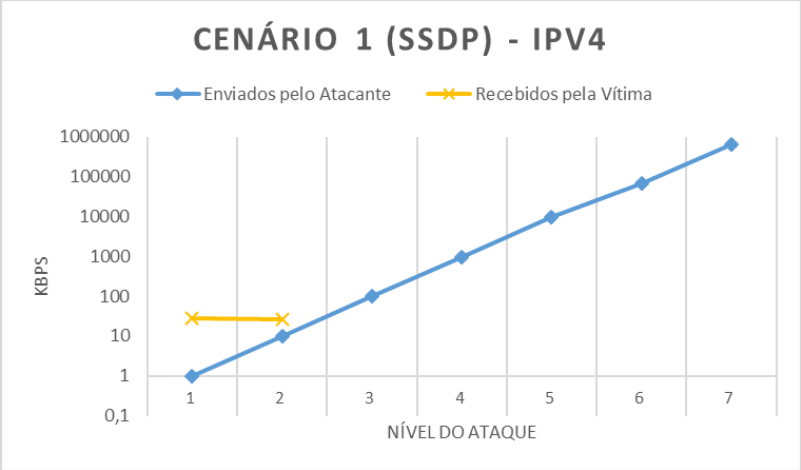

Figura 4.1: Cenário 1 (SSDP) - IPv4 - Kbps

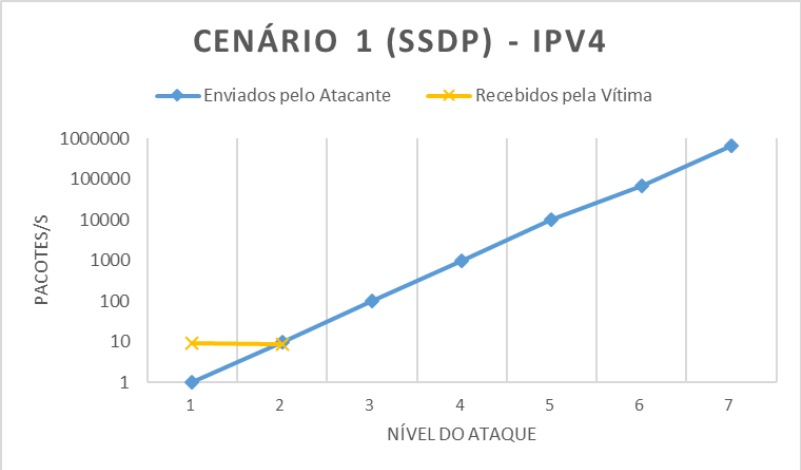

Figura 4.2: Cenário 1 (SSDP) - IPv4 - Pacotes/s

4.1.1.2 SSDP (Metodologia Alternativa)

Como os resultados obtidos com este refletor nesta pesquisa foram divergentes dos alcançados no artigo [22] e as metodologias utilizadas foram um pouco diferentes, optou-se por se refazer os testes usando a metodologia alternativa descrita na seção 3.2, onde cada nível foi testado isoladamente, aguardando-se até que todos os pacotes refletidos pelo *gateway* ADSL chegassem até à vítima, para que o nível seguinte fosse iniciado no atacante.

Analisando a tabela 4.3, é possível perceber que, à medida que a quantidade de pacotes/s enviada pelo atacante aumentava, o tempo que o ataque durava na vítima aumentava consideravalmente, chegando a durar até 5h, com apenas 10s de fluxo gerado pelo atacante. Isso talvez explique o travamento do equipamento no nível 3 com a metodologia convencional, pois enquanto ele ainda estava lidando com o processamento e envio dos pacotes dos níveis iniciais, chegavam mais milhares de pacotes dos níveis finais para serem processados.

A saturação, com essa metodologia, também ocorreu depois, a partir do nível 3, conforme se

observa na tabela 4.3. A partir do nível 5, a quantidade total de dados recebida pela vítima já era menor do que a enviada pelo atacante, tornando o ataque menos atrativo desse ponto em diante. Apesar de as figuras 4.3 e 4.4 mostrarem que houve uma possível saturação a partir do nível 2, o que na verdade ocorreu foi que o refletor enviou as respostas a taxas constantes (aproximadamente 28 Kbps e 9,1 pacotes/s) durante toda a duração do ataque. Esse comportamento somente foi observado com esse refletor e ficou mais evidente quando a metodologia alternativa foi aplicada.

A tabela 4.4 traz a quantidade e o tamanho dos pacotes ICMP gerados pela vítima para o refletor durante todo o ataque. A quantidade de *bytes* gerada pela vítima e, consequentemente, processada pelo refletor é relativamente alta e não deve ser desprezada.

Tabela 4.3: Cenário 1 (SSDP) - IPv4 - Metodologia Alternativa

	Saída do Atacante			**Entrada da Vítima**			
Nível	***Bytes***	***Kbps***	***Pacotes/s***	***Bytes***	***Kbps***	***Pacotes/s***	***Duração (s)***
1	1220	1,0	1,0	107840	28,1	9,1	30,69
2	12200	9,8	10,0	1078400	28,0	9,1	308,05
3	122000	97,6	100,0	5176320	28,0	9,1	1479,00
4	1220000	976,0	1000,0	7236064	28,0	9,1	2067,64
5	12200000	9760,0	10000,0	9597760	28,0	9,1	2745,74
6	81937030	65549,6	67161,5	16413248	28,0	9,1	4697,07
7	747180460	597744,4	612443,0	35166624	15,3	5,0	18393,24

Tabela 4.4: Cenário 1 (SSDP) - IPv4 - ICMP - Metodologia Alternativa

	Saída da Vítima	
Nível	***Pacotes ICMP***	***Bytes ICMP***
1	280	115680
2	1643	678728
3	9126	3770170
4	13060	5395843
5	17272	7135728
6	29973	12383201
7	62649	25883194

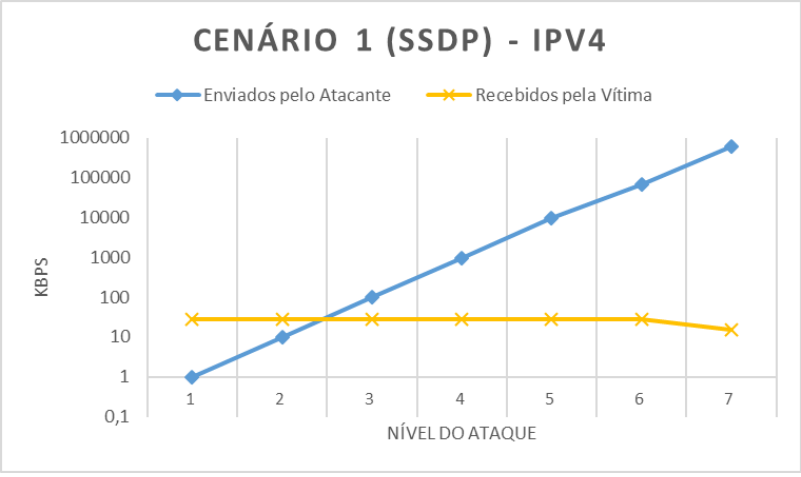

Figura 4.3: Cenário 1 (SSDP) - IPv4 - Kbps - Metodologia Alternativa

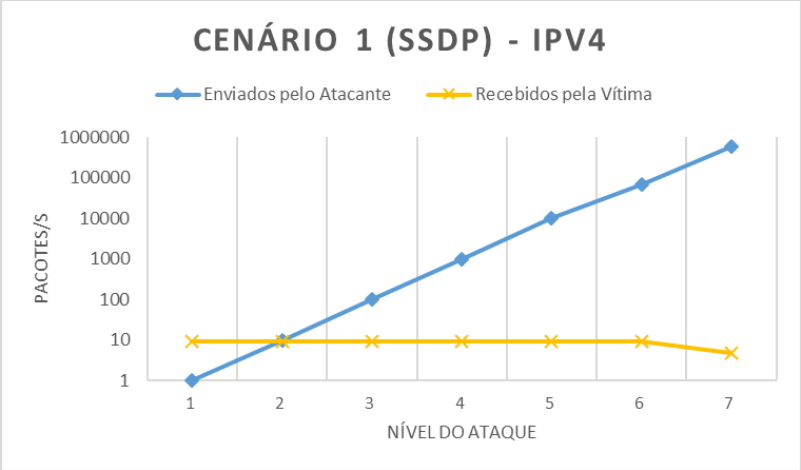

Figura 4.4: Cenário 1 (SSDP) - IPv4 - Pacotes/s - Metodologia Alternativa

4.1.1.3 SNMP

No ataque com o protocolo SNMP não houve saturação do atacante e ele foi capaz de enviar todos os pacotes esperados para cada nível do Linderhof, conforme se observa na tabela 4.5. Dessa vez, a duração do ataque praticamente não extrapolou os 10 segundos (ou pelo menos se manteve próximo a isso), com exceção do nível 2, que durou 31,13 segundos para ser concluído na vítima, mas ainda assim com uma taxa de transmissão razoável (892,3 Kbps e 76,6 pacotes/s), conforme se observa nas figuras 4.5 e 4.6.

A partir do nível 2 já se observou saturação do refletor, em relação à quantidade de *bytes* gerados por ele. Esperava-se que a vítima recebesse 6545800 *bytes* e ela recebeu apenas 3471870 *bytes*. A partir do nível 3, a figura 4.6 evidencia que a quantidade de pacotes/s recebidos pela vítima já foi menor do que a enviada pelo atacante, indicando que o refletor foi incapaz de processar tantos pacotes de requisição, mesmo a quantidade não sendo tão alta.

A tabela 4.6 traz a quantidade e o tamanho dos pacotes ICMP gerados pela vítima para o refletor durante todo o ataque. Nesse caso, é possível notar (pelo nível 1) que cada mensagem ICMP gerada possuía 576 *bytes* de tamanho (Total_Bytes_ICMP/Total_Pacotes_ICMP = 5760/10). Isso é porque as mensagens SNMP recebidas pela vítima eram muito superior ao limite de 576 *bytes* do ICMP e, portanto, tiveram que ser suprimidas. O mesmo se repete nos demais níveis.

Tabela 4.5: Cenário 1 (SNMP) - IPv4

	Saída do Atacante			**Entrada da Vítima**			
Nível	***Bytes***	***Kbps***	***Pacotes/s***	***Bytes***	***Kbps***	***Pacotes/s***	***Duração (s)***
1	650	0,5	1,0	654580	591,9	50,9	8,85
2	6500	5,2	10,0	3471870	892,3	76,6	31,13
3	65000	52,0	100,0	1048804	925,1	79,4	9,07
4	650000	520,0	1000,0	1114298	790,9	67,9	11,27
5	6500000	5200,0	10000,0	1114400	747,5	64,1	11,93
6	65000000	52000,0	100000,0	1047140	934,7	80,3	8,96
7	650000000	520000,0	1000000,0	720256	962,2	82,7	5,99

Tabela 4.6: Cenário 1 (SNMP) - IPv4 - ICMP

Nível	Saída da Vítima	
	Pacotes ICMP	***Bytes ICMP***
1	10	5760
2	53	30528
3	16	9216
4	17	9792
5	17	9792
6	16	9216
7	11	6336

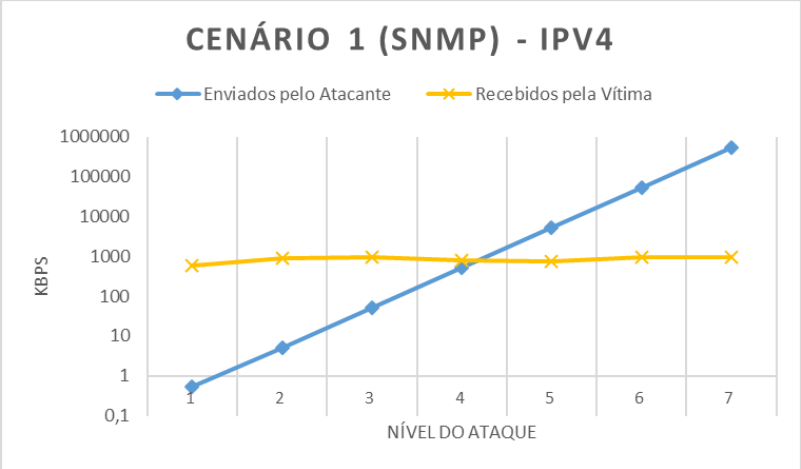

Figura 4.5: Cenário 1 (SNMP) - IPv4 - Kbps

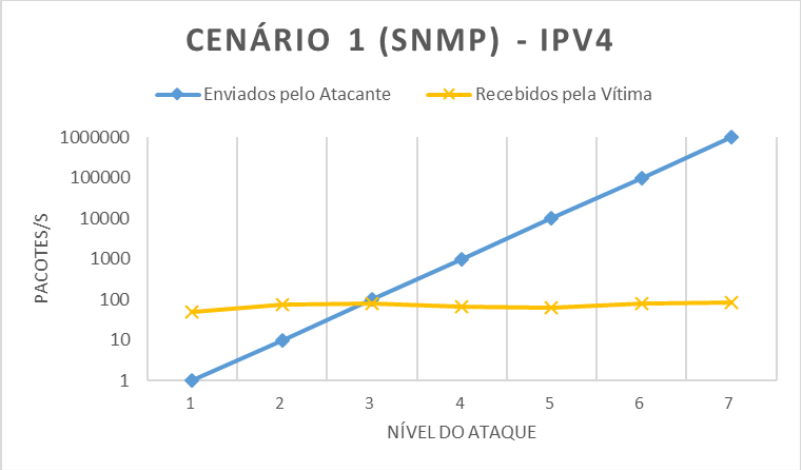

Figura 4.6: Cenário 1 (SNMP) - IPv4 - Pacotes/s

4.1.1.4 SNMP (Metodologia Alternativa)

Do mesmo modo que com SSDP, os testes com SNMP apresentaram divergências em relação ao artigo [22] e foram refeitos com a metodologia alternativa. O comportamento observado também foi diferente daquele obtido com a metodologia convencional, conforme mostram a tabela

4.7 e as figuras 4.7 e 4.8.

Dessa vez, houve saturação do atacante no nível 7 e, a duração do ataque, com exceção do nível 1, foi praticamente a mesma em todos os demais níveis (ao redor de 30 segundos), gerando taxas de Kbps e de pacotes/s constantes, assim como o SSDP.

Em relação ao tráfego ICMP (tabela 4.8), apesar de a quantidade de pacotes e *bytes* enviados pela vítima ter sido, em média, maior do que a metodologia convencional, por algum motivo não foram gerados pacotes ICMP nos níveis 2 e 3.

Tabela 4.7: Cenário 1 (SNMP) - IPv4 - Metodologia Alternativa

	Saída do Atacante			**Entrada da Vítima**			
Nível	***Bytes***	***Kbps***	***Pacotes/s***	***Bytes***	***Kbps***	***Pacotes/s***	***Duração (s)***
1	650	0,5	1,0	654580	523,7	50,7	8,86
2	6500	5,2	10,0	2882460	524,1	64,1	30,90
3	65000	52,0	100,0	3406588	524,1	77,0	30,40
4	650000	520,0	1000,0	3472082	524,1	76,7	31,10
5	6500000	5200,0	10000,0	3406900	524,1	76,6	30,53
6	65000000	52000,0	100000,0	3401180	523,3	75,6	30,96
7	578513650	462810,9	890021,0	3139908	523,3	68,5	31,54

Tabela 4.8: Cenário 1 (SNMP) - IPv4 - ICMP - Metodologia Alternativa

	Saída da Vítima	
Nível	***Pacotes ICMP***	***Bytes ICMP***
1	9	5184
2	0	0
3	0	0
4	35	20160
5	52	29952
6	52	29952
7	48	27648

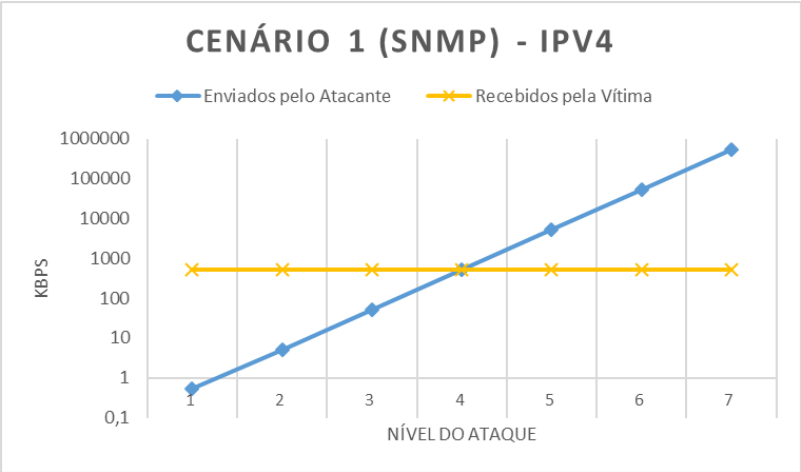

Figura 4.7: Cenário 1 (SNMP) - IPv4 - Kbps - Metodologia Alternativa

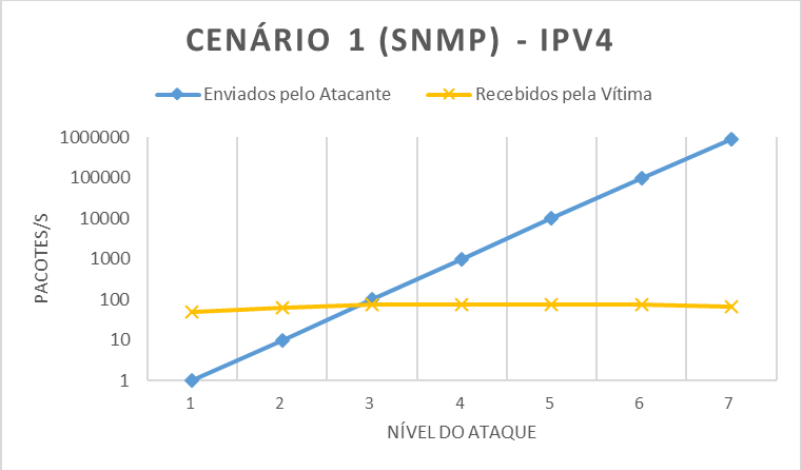

Figura 4.8: Cenário 1 (SNMP) - IPv4 - Pacotes/s - Metodologia Alternativa

4.1.2 Cenário 2 (Câmera IP)

Neste cenário, apesar de o dispositivo suportar IPv6, ele funcionava apenas para o protocolo SNMP. Com IPv4, além do próprio SNMP, também foi testado o protocolo SSDP.

4.1.2.1 SSDP

No ataque via SSDP, a tabela 4.9 indica que houve saturação do atacante a partir do nível 6, assim como no cenário 1, com ambas as metodologias. A tabela também aponta que houve saturação do refletor a partir do nível 3 da ferramenta, ficando esse comportamento mais evidenciado a partir do nível 4, como é possível observar nas figuras 4.9 e 4.10. No nível 4, inclusive, a quantidade de pacotes/s recebida pela vítima já era menor do que a enviada pelo atacante, o que continuou até o final do ataque.

O tempo de duração do ataque na vítima, foi compatível com o que se esperava, permanencendo constante e próximo aos 10 segundos. Nesse ataque, curiosamente, não foram geradas mensagens ICMP nos níveis 1 ao 5, começando a serem geradas apenas no nível 6, de acordo com a tabela 4.10.

Durante as capturas, de modo similar ao que ocorreu com o *gateway* ADSL, também se observou um comportamento atípico da câmera IP utilizando o protocolo SSDP, porém de outra natureza. Apesar de o serviço rodar na porta UDP/1900, as respostas tinham como porta de origem sempre portas dinâmicas, geralmente acima de UDP/30000, mas o conteúdo dos pacotes também era de tráfego SSDP legítimo, não prejudicando o ataque.

Nesse cenário não foram encontradas divergências ou comportamentos anômalos que justificassem novos testes com a metodologia alternativa.

Tabela 4.9: Cenário 2 (SSDP) - IPv4

Nível	Saída do Atacante			Entrada da Vítima			
	Bytes	***Kbps***	***Pacotes/s***	***Bytes***	***Kbps***	***Pacotes/s***	***Duração (s)***
1	1220	1,0	1,0	18740	16,7	4,40	9,00
2	12200	9,8	10,0	187400	166,4	44,4	9,01
3	122000	97,6	100,0	1710962	1501,7	400,7	9,12
4	1220000	976,0	1000,0	1992062	1745,9	465,8	9,13
5	12200000	9760,0	10000,0	2413712	2118,2	565,2	9,12
6	84256128	67404,9	69062,4	3566222	2903,8	774,8	9,83
7	800557412	640445,9	656194,6	12021710	9546,7	2547,2	10,07

Tabela 4.10: Cenário 2 (SSDP) - IPv4 - ICMP

Nível	Saída da Vítima	
	Pacotes ICMP	***Bytes ICMP***
1	0	0
2	0	0
3	0	0
4	0	0
5	0	0
6	808	400830
7	2196	1089892

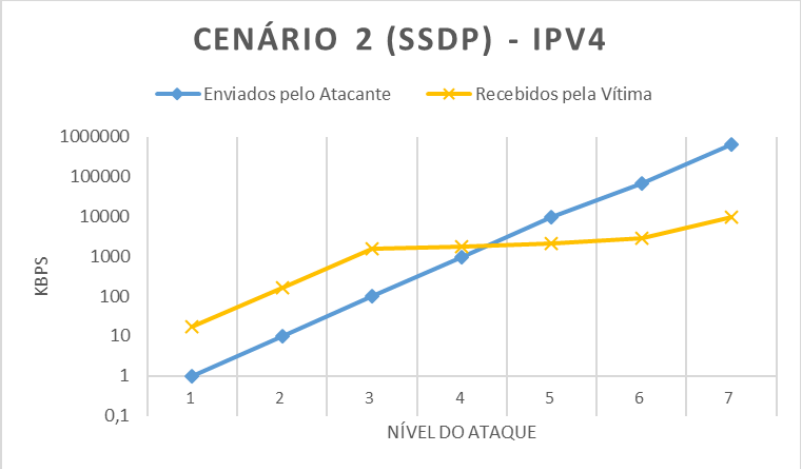

Figura 4.9: Cenário 2 (SSDP) - IPv4 - Kbps

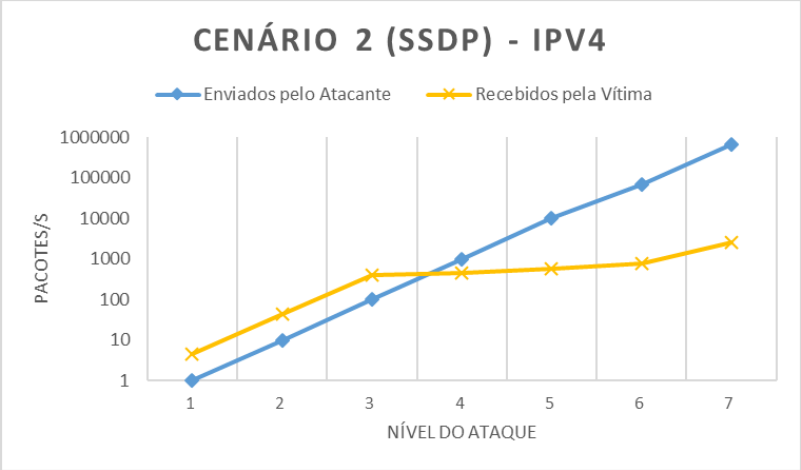

Figura 4.10: Cenário 2 (SSDP) - IPv4 - Pacotes/s

4.1.2.2 SNMP

No ataque via SNMP, foi possível analisar variações de comportamento do ataque quando a versão do protocolo IP utilizada nele era trocada. Observando as tabelas 4.11 e 4.13, é possível notar que o atacante saturou apenas com IPv6, ocorrendo a partir do nível 5. Entretanto, o comportamento do refletor durante o ataque se manteve praticamente o mesmo nas duas versões do protocolo IP.

As figuras 4.11 e 4.12 (IPv4) e 4.13 e 4.14 (IPv6) demonstram comportamentos muito parecidos entre as duas versões, sendo que o dispositivo começou a saturar ainda no nível 2 e, a partir do nível 3, a taxa de Kbps recebida pela vítima já foi menor do que a enviada pelo atacante, conforme mostra as figuras 4.11 e 4.13.

O tempo de duração do ataque na vítima também foi compatível com o que se esperava, com tempos parecidos aos observados no protocolo SSDP.

O fato de a vítima ter recebido apenas 1,1 pacote por segundo no nível 1 (mas com duração total de 9,00 segundos) também indica que não houve fragmentação nem amplificação de pacotes nesse cenário, com as respostas geradas pelo refletor sendo de 1321 *bytes*, abaixo do limite de 1500 *bytes* do MTU do enlace. Entretanto, como excederam o limite de 576 *bytes* do ICMP e 1280 *bytes* do ICMPv6, as mensagens ICMP e ICMPv6 geradas pelo refletor via SNMP tiveram que suprimir uma parte da mensagem original, de acordo com as tabelas 4.12 e 4.14.

Nesse cenário também não foram encontradas divergências ou comportamentos anômalos que justificassem novos testes com a metodologia alternativa.

Tabela 4.11: Cenário 2 (SNMP) - IPv4

Nível	Saída do Atacante			Entrada da Vítima			
	Bytes	***Kbps***	***Pacotes/s***	***Bytes***	***Kbps***	***Pacotes/s***	***Duração (s)***
1	650	0,5	1,0	13210	11,7	1,1	9,00
2	6500	5,2	10,0	13212	11,7	1,1	9,00
3	65000	52,0	100,0	38304	34,0	3,2	9,02
4	650000	520,0	1000,0	39636	35,2	3,3	9,02
5	6500000	5200,0	10000,0	44890	39,8	3,8	9,03
6	65000000	52000,0	100000,0	121548	106,7	10,1	9,11
7	650000000	520000,0	1000000,0	962862	774,4	73,3	9,95

Tabela 4.12: Cenário 2 (SNMP) - IPv4 - ICMP

Nível	Saída da Vítima	
	Pacotes ICMP	***Bytes ICMP***
1	10	5760
2	10	5760
3	29	16704
4	30	17280
5	34	19584
6	92	52992
7	729	419904

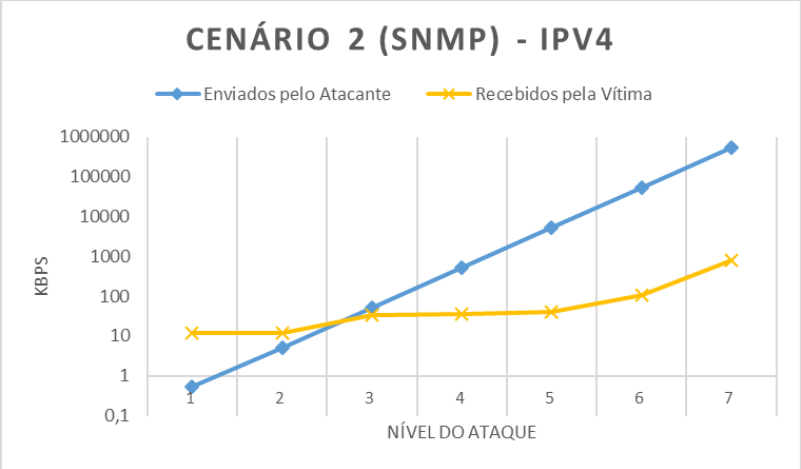

Figura 4.11: Cenário 2 (SNMP) - IPv4 - Kbps

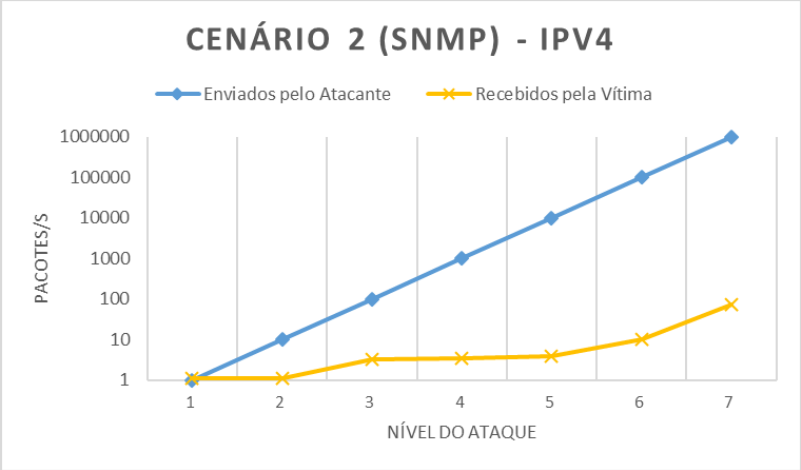

Figura 4.12: Cenário 2 (SNMP) - IPv4 - Pacotes/s

Tabela 4.13: Cenário 2 (SNMP) - IPv6

	Saída do Atacante			Entrada da Vítima			
Nível	***Bytes***	***Kbps***	***Pacotes/s***	***Bytes***	***Kbps***	***Pacotes/s***	***Duração (s)***
1	850	0,7	1,0	13406	11,9	1,1	9,00
2	8500	6,8	10,0	13410	11,9	1,1	9,00
3	85000	68,0	100,0	40236	35,7	3,3	9,02
4	850000	680,0	1000,0	40230	35,7	3,3	9,02
5	3429750	2743,8	4035,0	42914	38,1	3,5	9,02
6	11143500	8914,8	13110,0	140804	122,4	11,4	9,21
7	51852975	41482,4	61003,5	607217	485,3	45,3	10,01

Tabela 4.14: Cenário 2 (SNMP) - IPv6 - ICMP

	Saída da Vítima	
Nível	***Pacotes ICMP***	***Bytes ICMP***
1	10	12800
2	10	12800
3	30	38400
4	30	38400
5	32	40960
6	105	134400
7	453	579840

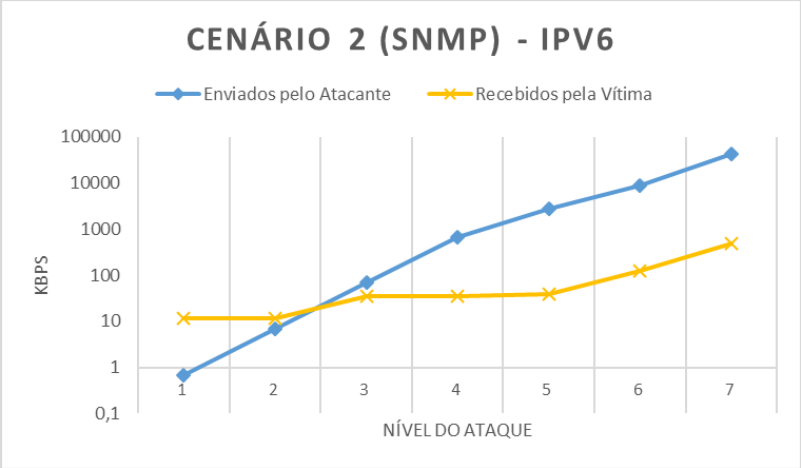

Figura 4.13: Cenário 2 (SNMP) - IPv6 - Kbps

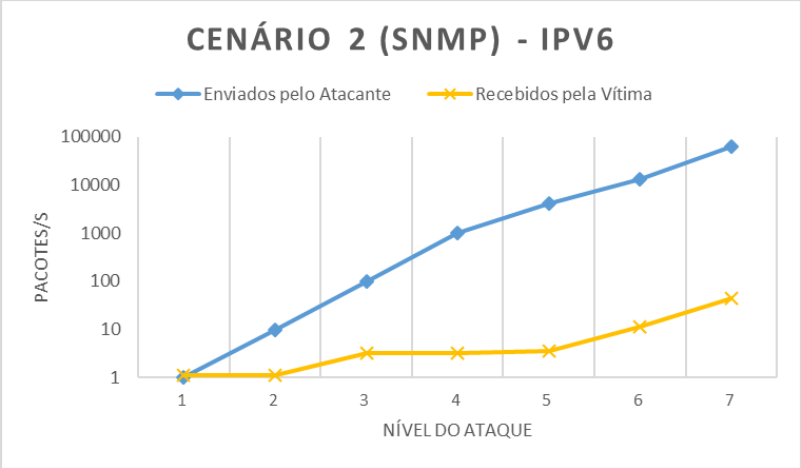

Figura 4.14: Cenário 2 (SNMP) - IPv6 - Pacotes/s

4.1.3 Cenário 3 (Raspberry Pi)

No cenário 3, além do refletor possuir maior capacidade computacional, foi possível realizar análises mais detalhadas e assertivas a partir das capturas realizadas nele. Para facilitar a leitura dos resultados, as tabelas foram separadas em Kbps e Pacotes/s. Também foi exequível testar neste cenário os três protocolos estudados neste trabalho (SSDP, SNMP e CoAP), tanto em IPv4 quanto em IPv6.

4.1.3.1 SSDP

Analisando as tabelas 4.15 e 4.16 (IPv4) e 4.18 e 4.19 (IPv6), é possível perceber que o atacante satura a partir do nível 6 para o IPv4 e do nível 5 para o IPv6, nos testes com o protocolo SSDP. O refletor, por sua vez, não consegue receber a mesma quantidade de pacotes do atacante a partir do nível 4, para ambas as versões, ocorrendo a sua saturação. A partir desse nível o refletor

também não consegue mais gerar pacotes na mesma taxa em que os recebe. A vítima, finalmente, recebe praticamente a mesma quantidade de pacotes do refletor, com pequenas variações.

As figuras 4.15 e 4.16 (IPv4) e 4.17 e 4.18 (IPv6) apresentam comportamentos similares. As principais diferenças ocorrem a partir do nível 5, onde observa-se que, via IPv6, a vítima recebe taxas maiores de Kbps em relação ao que o atacante envia do que com IPv4.

O tempo de duração do ataque na vítima, tanto via IPv4 quanto via IPv6, foi compatível com o que se esperava, ficando abaixo de 10 segundos por nível.

As tabelas 4.17 e 4.20 trazem a quantidade de pacotes ICMP e seus respectivos *bytes* gerados pela vítima e recebidos pelo refletor. O que vale destacar aqui é que, além da vítima não conseguir gerar mensagens ICMP na mesma quantidade de pacotes SSDP que recebe, o refletor, a partir do nível 6, não consegue processar todas as mensagens ICMP geradas pela vítima, evidenciando o quanto ele já estava saturado.

Tabela 4.15: Cenário 3 (SSDP) - IPv4 - Kbps

	Saída do Atacante		**Entrada do Refletor**		**Saída do Refletor**		**Entrada da Vítima**		
Nível	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Duração (s)***
1	1100	0,9	1100	0,9	55390	49,2	55390	49,2	9,00
2	11000	8,8	11000	8,8	553900	491,9	553900	491,9	9,01
3	110000	88,8	110000	88,8	5539000	4884,7	5122221	4517,1	9,07
4	1100000	880,8	949960	760,0	11687290	10204,1	11687290	10204,1	9,16
5	11000000	8800,8	2115300	1692,2	12075020	10606,3	12075020	10606,3	9,11
6	81030400	64824,3	5610000	4488,0	12839402	11260,4	12389745	10866,1	9,12
7	763420350	610736,3	69747920	55798,3	42733212	34550,0	52798286	42687,7	9,89

Tabela 4.16: Cenário 3 (SSDP) - IPv4 - Pacotes/s

	Saída do Atacante	**Entrada do Refletor**	**Saída do Refletor**	**Entrada da Vítima**
Nível	***Pacotes/s***	***Pacotes/s***	***Pacotes/s***	***Pacotes/s***
1	1,0	1,0	14,4	14,4
2	10,0	10,0	144,3	144,3
3	100,0	100,0	1433,0	1325,2
4	1000,0	863,6	2993,6	2993,6
5	10000,0	1923,0	3111,6	3111,6
6	73664,0	5100,0	3303,5	3187,7
7	694018,5	63407,2	10135,4	12523,3

Tabela 4.17: Cenário 3 (SSDP) - IPv4 - ICMP

	Saída da Vítima		**Entrada do Refletor**	
Nível	***Pacotes ICMP***	***Bytes ICMP***	***Pacotes ICMP***	***Bytes ICMP***
1	130	59030	130	59030
2	1036	470317	1036	470317
3	1000	454536	1000	454536
4	1000	454527	1000	454527
5	1000	454279	1000	454279
6	1000	454501	93	42507
7	2100	954758	225	103090

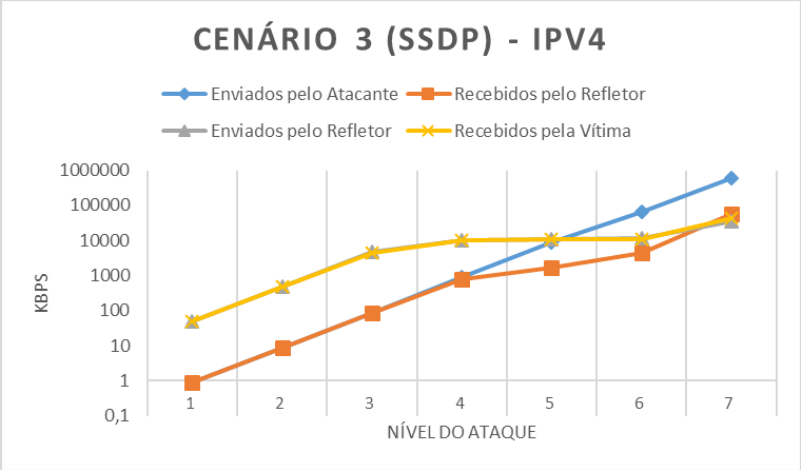

Figura 4.15: Cenário 3 (SSDP) - IPv4 - Kbps

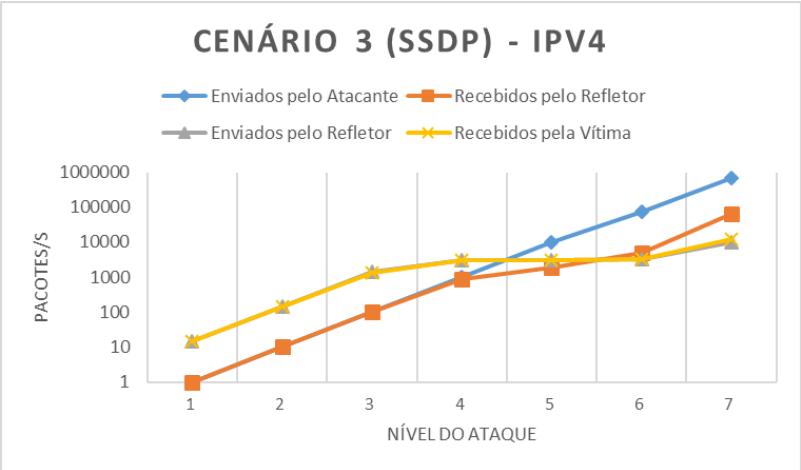

Figura 4.16: Cenário 3 (SSDP) - IPv4 - Pacotes/s

Tabela 4.18: Cenário 3 (SSDP) - IPv6 - Kbps

	Saída do Atacante		**Entrada do Refletor**		**Saída do Refletor**		**Entrada da Vítima**		
Nível	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Duração (s)***
1	1330	1,1	1330	1,1	59940	53,3	59940	53,3	9,00
2	13300	10,6	13300	10,6	599400	532,3	599400	532,3	9,01
3	133000	106,4	133000	106,4	5994000	5286,3	5994000	5286,3	9,07
4	1330000	1064,0	1093393	874,7	11166822	9746,5	11036752	9633,0	9,17
5	5374530	4299,6	2122015	1697,6	10639350	9294,4	10639350	9294,4	9,16
6	17466890	13973,5	8111005	6488,8	19024956	16628,6	18854821	16479,9	9,15
7	54820206	43856,2	15911056	12728,8	51727840	41565,2	81798642	65728,1	9,96

Tabela 4.19: Cenário 3 (SSDP) - IPv6 - Pacotes/s

Nível	Saída do Atacante	Entrada do Refletor	Saída do Refletor	Entrada da Vítima
	Pacotes/s	*Pacotes/s*	*Pacotes/s*	*Pacotes/s*
1	1,0	1,0	14,4	14,4
2	10,0	10,0	144,3	144,3
3	100,0	100,0	1433,1	1433,1
4	1000,0	822,1	2642,3	2611,6
5	4041,0	1595,5	2519,8	2519,8
6	13133,0	6098,5	4508,1	4467,8
7	41218,2	11963,2	11268,5	17819,3

Tabela 4.20: Cenário 3 (SSDP) - IPv6 - ICMP

Nível	Saída da Vítima		Entrada do Refletor	
	Pacotes ICMP	*Bytes ICMP*	*Pacotes ICMP*	*Bytes ICMP*
1	118	60085	118	60085
2	1070	544868	1070	544868
3	1000	509252	1000	509252
4	1000	509294	1000	509294
5	1000	509563	1000	509563
6	1700	865877	1661	845992
7	2100	1069268	1200	611000

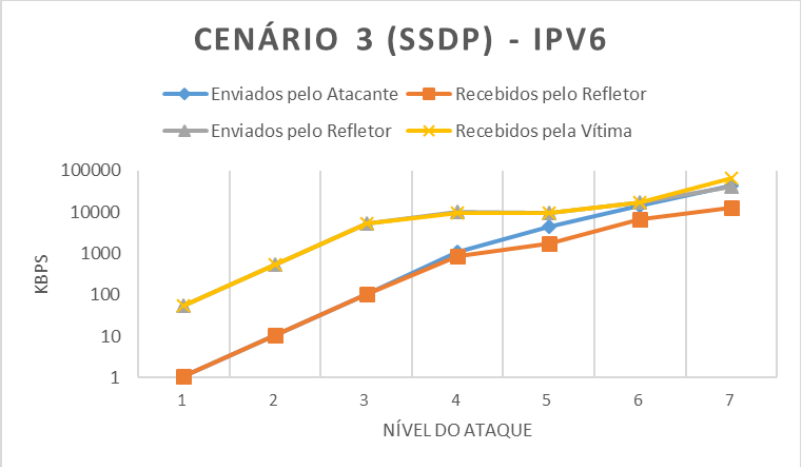

Figura 4.17: Cenário 3 (SSDP) - IPv6 - Kbps

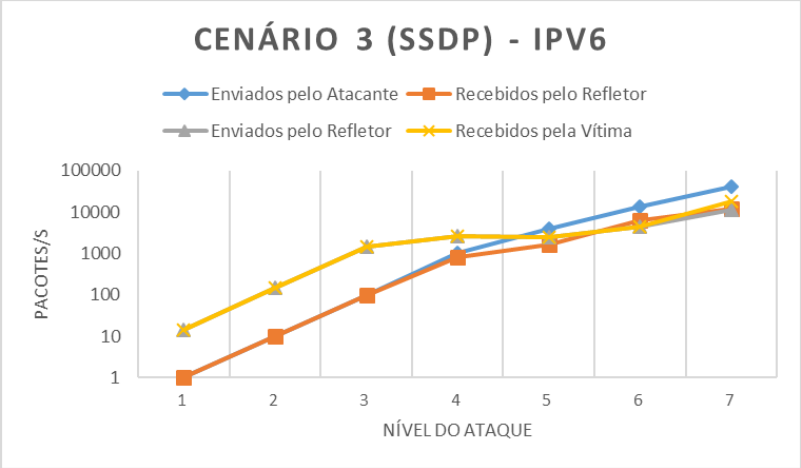

Figura 4.18: Cenário 3 (SSDP) - IPv6 - Pacotes/s

4.1.3.2 SNMP

Nos ataques com SNMP, após verificar as tabelas 4.21 e 4.22 (IPv4) e 4.24 e 4.25 (IPv6), é possível perceber que não ocorre satuação do atacante via IPv4, mas ela ocorre a partir do nível 5 com IPv6. O refletor, por sua vez, não consegue receber a mesma quantidade de pacotes do atacante a partir do nível 4, para ambas as versões, ocorrendo a sua saturação. A partir do nível 3, o refletor também não consegue mais gerar pacotes na mesma taxa em que os recebe. A vítima, finalmente, recebe praticamente a mesma quantidade de pacotes do refletor, com pequenas variações via IPv4.

As figuras 4.19 e 4.20 (IPv4) e 4.21 e 4.22 (IPv6) apresentam comportamentos muito parecidos entre as duas versões do protocolo IP, corroborando o que foi destacado acima.

Em relação aos pacotes ICMP, é possível notar que, via IPv4, o refletor não consegue processar todas as mensagens recebidas nos níveis 3, 6 e 7 (tabela 4.23), enquanto que, via IPv6, todos os pacotes foram processados normalmente pelo refletor (tabela 4.26).

Nesse cenário se observou um comportamento anômalo, não observado em nenhum outro ataque com o protocolo SNMP. No nível 5, tanto via IPv4 quanto via IPv6, o refletor foi incapaz de gerar pacotes de resposta para a vítima, mesmo repetindo-se o teste outras duas vezes com a metodologia convencional. O tempo de duração do ataque também variou bastante na vítima, tanto via IPv4 quanto via IPv6, chegando a 28,15 segundos no nível 3. Em virtude disso, optou-se por realizar os testes com a metodologia alternativa, cujos resultados estão exibidos na seção 4.1.3.3.

Tabela 4.21: Cenário 3 (SNMP) - IPv4 - Kbps

	Saída do Atacante		Entrada do Refletor		Saída do Refletor		Entrada da Vítima		
Nível	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Duração (s)***
1	650	0,5	650	0,5	632370	569,5	632370	569,5	8,88
2	6500	5,2	6500	5,2	6316260	3605,8	6316260	3605,8	14,01
3	65000	52,0	65000	52,0	12838768	3648,8	12838768	3648,8	28,15
4	650000	520,0	631410	505,1	191522	5547,3	191522	5547,3	0,28
5	6500000	5200,0	1657760	1326,2	0	0,0	0	0,0	0,0
6	65000000	52000,0	4950855	3960,7	4131870	3365,7	4449640	3624,6	9,82
7	650000000	520000,0	37565515	30052,4	3602519	3436,0	3792080	3616,8	8,39

Tabela 4.22: Cenário 3 (SNMP) - IPv4 - Pacotes/s

	Saída do Atacante	Entrada do Refletor	Saída do Refletor	Entrada da Vítima
Nível	***Pacotes/s***	***Pacotes/s***	***Pacotes/s***	***Pacotes/s***
1	1,0	1,0	48,4	48,4
2	10,0	10,0	306,8	306,8
3	100,0	100,0	308,6	308,6
4	1000,0	971,4	467,1	467,1
5	10000,0	2550,4	0,0	0,0
6	100000,0	7616,7	284,6	306,5
7	1000000,0	57793,1	292,2	307,6

Tabela 4.23: Cenário 3 (SNMP) - IPv4 - ICMP

	Saída da Vítima		Entrada do Refletor	
Nível	***Pacotes ICMP***	***Bytes ICMP***	***Pacotes ICMP***	***Bytes ICMP***
1	10	5760	10	5760
2	100	57600	100	57600
3	202	116352	201	115776
4	3	1728	3	1728
5	0	0	0	0
6	70	40320	48	27648
7	60	34560	14	8064

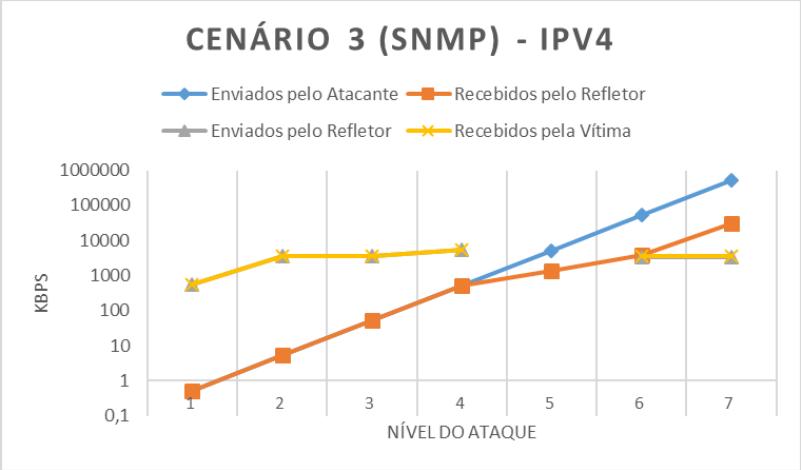

Figura 4.19: Cenário 3 (SNMP) - IPv4 - Kbps

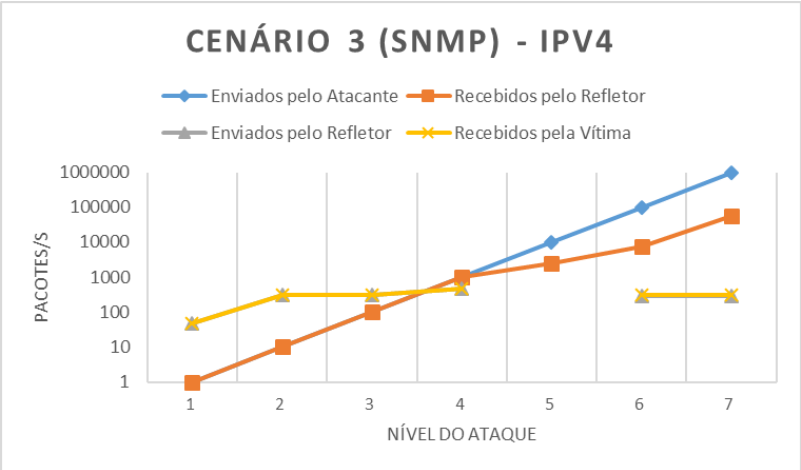

Figura 4.20: Cenário 3 (SNMP) - IPv4 - Pacotes/s

Tabela 4.24: Cenário 3 (SNMP) - IPv6 - Kbps

	Saída do Atacante		**Entrada do Refletor**		**Saída do Refletor**		**Entrada da Vítima**		
Nível	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Duração (s)***
1	850	0,7	850	0,7	627210	565,2	627210	565,2	8,88
2	8500	6,8	8500	6,8	6261680	3208,7	6261680	3208,7	15,61
3	85000	68,0	85000	68,0	11334919	3221,3	11334919	3221,3	28,15
4	850000	680,0	783955	627,2	188737	4856,5	188737	4856,5	0,31
5	3426350	2741,1	1701190	1361,0	0	0,0	0	0,0	0,00
6	9831270	7865,0	5298220	4238,6	5199170	3229,7	5199170	3229,7	12,88
7	31179785	24943,8	15684795	12547,8	1503808	3369,9	1503808	3369,9	3,57

Tabela 4.25: Cenário 3 (SNMP) - IPv6 - Pacotes/s

Nível	Saída do Atacante	Entrada do Refletor	Saída do Refletor	Entrada da Vítima
	Pacotes/s	*Pacotes/s*	*Pacotes/s*	*Pacotes/s*
1	1,0	1,0	49,6	49,6
2	10,0	10,0	281,8	281,8
3	100,0	100,0	282,9	282,9
4	1000,0	922,3	424,6	424,6
5	4031,0	2001,4	0,0	0,0
6	11566,2	6233,2	283,6	283,6
7	36682,1	18452,7	295,8	295,8

Tabela 4.26: Cenário 3 (SNMP) - IPv6 - ICMP

Nível	Saída da Vítima		Entrada do Refletor	
	Pacotes ICMP	*Bytes ICMP*	*Pacotes ICMP*	*Bytes ICMP*
1	10	12800	10	12800
2	100	128000	100	128000
3	181	231680	181	231680
4	3	3840	3	3840
5	0	0	0	0
6	83	106240	83	106240
7	24	30720	24	30720

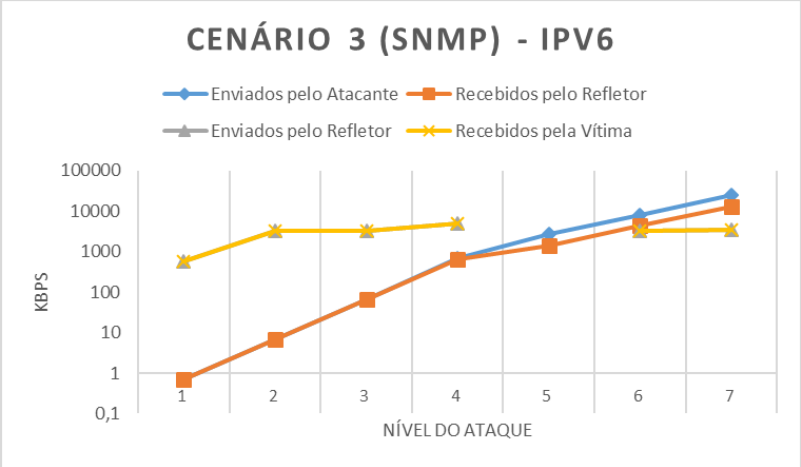

Figura 4.21: Cenário 3 (SNMP) - IPv6 - Kbps

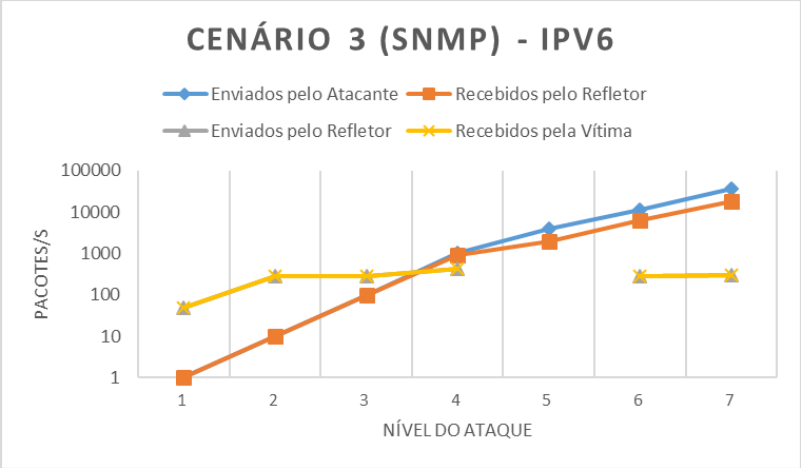

Figura 4.22: Cenário 3 (SNMP) - IPv6 - Pacotes/s

4.1.3.3 SNMP (Metodologia Alternativa)

Conforme explicado na subseção anterior, em virtude dos comportamentos anômalos encontrados nesse cenário utilizando a metodologia convencvional, os testes foram todos refeitos com a metodologia alternativa, tanto para IPv4 quanto para IPv6.

Dessa vez, como se observa nas tabelas 4.27 e 4.28 (IPv4) e 4.30 e 4.31 (IPv6), é possível perceber que ocorre satuação do atacante via IPv4, mas apenas no nível 7, enquanto que, com IPv6, ela continua ocorrendo a partir do nível 5.

O refletor continua sem conseguir processar a mesma quantidade de pacotes do atacante a partir do nível 4, para ambas as versões, ocorrendo a sua saturação. A partir do nível 3, o refletor também não consegue mais gerar pacotes na mesma taxa em que os recebe. Contudo, nesse cenário, a vítima recebe a mesma quantidade de pacotes do refletor, tanto via IPv4 quanto via IPv6.

As figuras 4.19 e 4.20 (IPv4) e 4.21 e 4.22 (IPv6) apresentam comportamentos muito parecidos entre as duas versões do protocolo IP, ficando mais próximo do esperado, ao contrário da metodologia convencional.

Em relação aos pacotes ICMP, não foram geradas respostas entre os níveis 1 e 3, via IPv4 (tabela 4.29), ao contrário do que se observa com IPv6 (tabela 4.32), onde foram geradas respostas ICMPv6 em todos os níveis. Via IPv4, apenas no nível 4 o refletor conseguiu receber e processar todos os pacotes enviados pela vítima, enquanto que, via IPv6, todos os pacotes foram processados normalmente em todos os níveis.

Cabe ressaltar ainda que o tempo de duração do ataque, utilizando a metodologia alternativa, durou mais de 30 segundos, a partir do nível 4 (IPv4) e do nível 3 (IPv6).

Tabela 4.27: Cenário 3 (SNMP) - IPv4 - Kbps - Metodologia Alternativa

	Saída do Atacante		Entrada do Refletor		Saída do Refletor		Entrada da Vítima		
Nível	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Duração (s)***
1	650	0,5	650	0,5	632370	550,2	632370	550,2	9,19
2	6500	5,2	6500	5,2	6316260	3627,8	6316260	3627,8	13,93
3	65000	52,0	65000	52,0	13029430	3687,6	13029430	3687,6	28,27
4	650000	520,0	597805	478,2	15889360	3664,3	15889360	3664,3	34,69
5	6500000	5200,0	1965795	1572,6	15889360	3664,3	15889360	3664,3	34,69
6	65000000	52000,0	4683315	3746,7	15889360	3643,5	15889360	3643,5	34,89
7	649751180	519800,9	43872465	35098,0	14912992	3387,7	14912992	3387,7	35,22

Tabela 4.28: Cenário 3 (SNMP) - IPv4 - Pacotes/s - Metodologia Alternativa

	Saída do Atacante	Entrada do Refletor	Saída do Refletor	Entrada da Vítima
Nível	***Pacotes/s***	***Pacotes/s***	***Pacotes/s***	***Pacotes/s***
1	1,0	1,0	46,8	46,8
2	10,0	10,0	308,7	308,7
3	100,0	100,0	311,9	311,9
4	1000,0	919,7	309,9	309,9
5	10000,0	3024,3	309,9	309,9
6	100000,0	7205,1	308,1	308,1
7	999617,2	67496,1	288,2	288,2

Tabela 4.29: Cenário 3 (SNMP) - IPv4 - ICMP - Metodologia Alternativa

	Saída da Vítima		Entrada do Refletor	
Nível	***Pacotes ICMP***	***Bytes ICMP***	***Pacotes ICMP***	***Bytes ICMP***
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	240	138240	240	138240
5	250	144000	249	143424
6	250	144000	245	141120
7	236	135936	181	104256

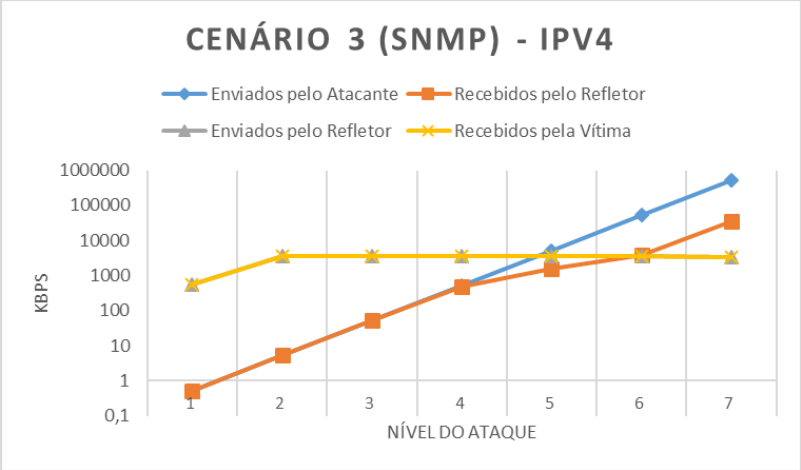

Figura 4.23: Cenário 3 (SNMP) - IPv4 - Kbps - Metodologia Alternativa

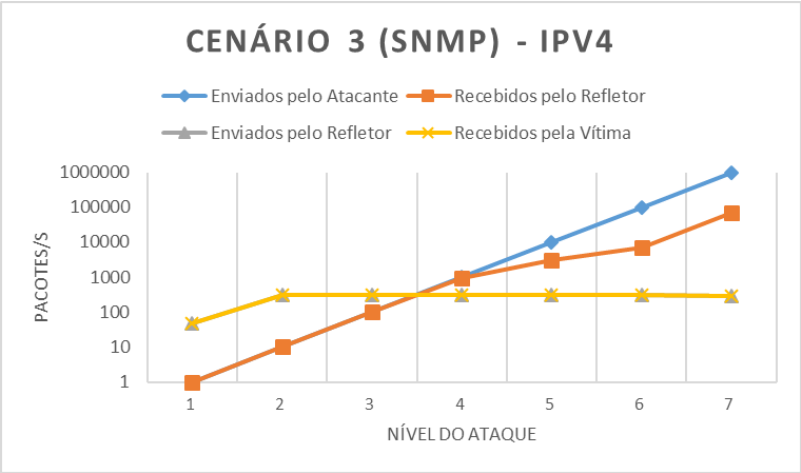

Figura 4.24: Cenário 3 (SNMP) - IPv4 - Pacotes/s - Metodologia Alternativa

Tabela 4.30: Cenário 3 (SNMP) - IPv6 - Kbps - Metodologia Alternativa

	Saída do Atacante		**Entrada do Refletor**		**Saída do Refletor**		**Entrada da Vítima**		
Nível	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Duração (s)***
1	850	0,7	850	0,7	627210	545,2	627210	545,2	9,20
2	8500	6,8	8500	6,8	6261680	3160,6	6261680	3160,6	15,85
3	85000	68,0	85000	68,0	13088251	3137,3	13088251	3137,3	33,37
4	850000	680,0	749615	599,7	11147062	3142,1	11147062	3142,1	28,38
5	3441650	2753,3	1729410	1383,5	14591107	3193,2	14591107	3193,2	36,56
6	10451855	8361,5	5623855	4499,1	11149020	3160,6	11149020	3160,6	28,22
7	31327770	25062,2	16292630	13034,1	14591806	3176,1	14591806	3176,1	36,75

Tabela 4.31: Cenário 3 (SNMP) - IPv6 - Pacotes/s - Metodologia Alternativa

Nível	Saída do Atacante	Entrada do Refletor	Saída do Refletor	Entrada da Vítima
	Pacotes/s	*Pacotes/s*	*Pacotes/s*	*Pacotes/s*
1	1,0	1,0	47,8	47,8
2	10,0	10,0	277,6	277,6
3	100,0	100,0	275,5	275,5
4	1000,0	881,9	276,0	276,0
5	4049,0	2034,6	280,4	280,4
6	12296,3	6616,3	277,5	277,5
7	36856,2	19167,8	278,9	278,9

Tabela 4.32: Cenário 3 (SNMP) - IPv6 - ICMP - Metodologia Alternativa

Nível	Saída da Vítima		Entrada do Refletor	
	Pacotes ICMP	*Bytes ICMP*	*Pacotes ICMP*	*Bytes ICMP*
1	10	12800	10	12800
2	100	128000	100	128000
3	209	267520	209	267520
4	178	227840	178	227840
5	233	298240	233	298240
6	178	227840	178	227840
7	233	298240	233	298240

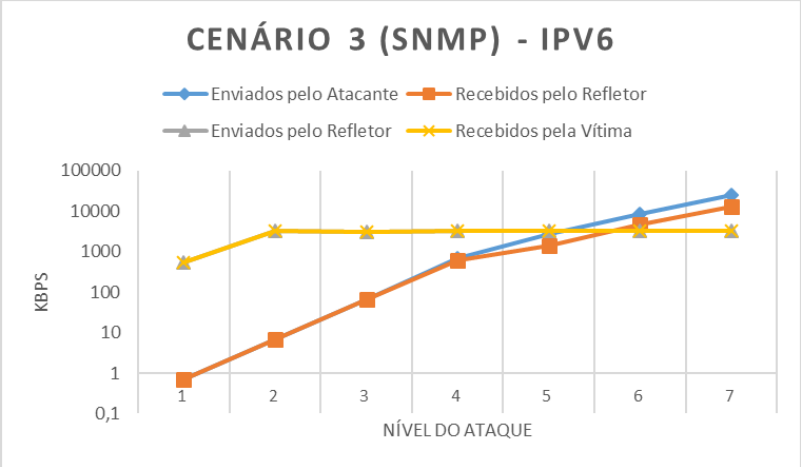

Figura 4.25: Cenário 3 (SNMP) - IPv6 - Kbps - Metodologia Alternativa

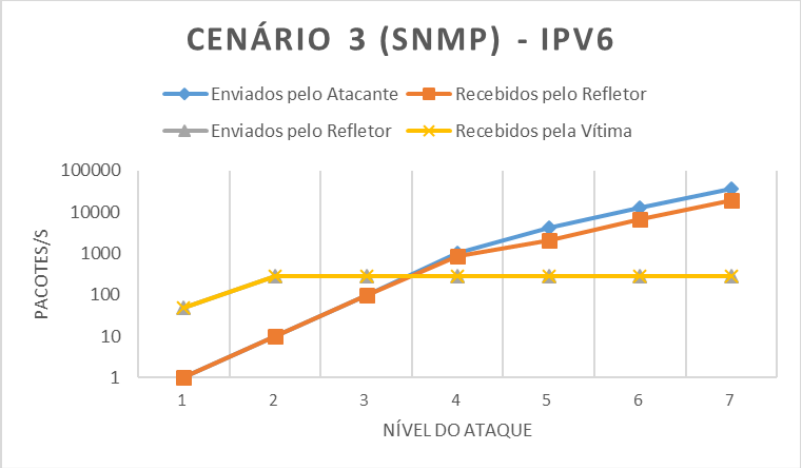

Figura 4.26: Cenário 3 (SNMP) - IPv6 - Pacotes/s - Metodologia Alternativa

4.1.3.4 CoAP

No ataque utilizando CoAP, ocorreu saturação do atacante apenas no nível 7 (IPv4) e a partir do nível 5 (IPv6), conforme se observa nas tabelas 4.33 e 4.34 (IPv4) e 4.36 e 4.37 (IPv6). O refletor, contudo, não consegue receber a mesma quantidade de pacotes do atacante a partir do nível 5, para ambas as versões, ocorrendo a sua saturação. O refletor não consegue mais gerar pacotes na mesma taxa em que os recebe a partir do nível 3, tanto para IPv4 quanto para IPv6. A vítima, por sua vez, recebe exatamente a mesma quantidade de pacotes enviados pelo refletor.

As figuras 4.27 e 4.28 (IPv4) e 4.29 e 4.30 (IPv6) refletem essas diferenças descritas anteriormente. No restante, o comportamento é muito similar nas duas versões do protocolo IP.

O tempo de duração do ataque na vítima se manteve quase sempre dentro do esperado, com exceção dos níveis 6 (IPv4) e 3 (IPv6), que duraram 13,31 e 13,75 segundos, respectivamente, o que também não é nada elevado.

Em relação aos pacotes ICMP, pode-se observar nas tabelas 4.35 e 4.38, que os refletores são incapazes de processar todos os pacotes ICMP recebidos a partir do nível 4.

Tabela 4.33: Cenário 3 (CoAP) - IPv4 - Kbps

	Saída do Atacante		**Entrada do Refletor**		**Saída do Refletor**		**Entrada da Vítima**		
Nível	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Duração (s)***
1	360	0,3	360	0,3	21040	18,7	21040	18,7	9,00
2	3600	2,9	3600	2,9	210400	183,2	210400	183,2	9,19
3	36000	28,8	36000	28,8	1041480	744,7	1041480	744,7	11,19
4	360000	288,0	360000	288,0	940488	694,0	940488	694,0	10,84
5	3600000	2880,0	988344	790,7	591224	667,9	591224	667,9	7,08
6	36000000	28800,0	2217060	1773,6	1144576	688,2	1144576	688,2	13,31
7	318346380	254677,1	14507316	11605,9	749024	656,0	749024	656,0	9,14

Tabela 4.34: Cenário 3 (CoAP) - IPv4 - Pacotes/s

Nível	Saída do Atacante	Entrada do Refletor	Saída do Refletor	Entrada da Vítima
	Pacotes/s	*Pacotes/s*	*Pacotes/s*	*Pacotes/s*
1	1,0	1,0	2,2	2,2
2	10,0	10,0	21,8	21,8
3	100,0	100,0	88,5	88,5
4	1000,0	1000,0	82,5	82,5
5	10000,0	2745,4	79,4	79,4
6	100000,0	6158,5	81,8	81,8
7	884295,5	40298,1	77,9	77,9

Tabela 4.35: Cenário 3 (CoAP) - IPv4 - ICMP

Nível	Saída da Vítima		Entrada do Refletor	
	Pacotes ICMP	*Bytes ICMP*	*Pacotes ICMP*	*Bytes ICMP*
1	10	5760	10	5760
2	100	57600	100	57600
3	495	285120	495	285120
4	447	257472	444	255744
5	281	161856	276	158976
6	544	313344	389	224064
7	356	205056	159	91584

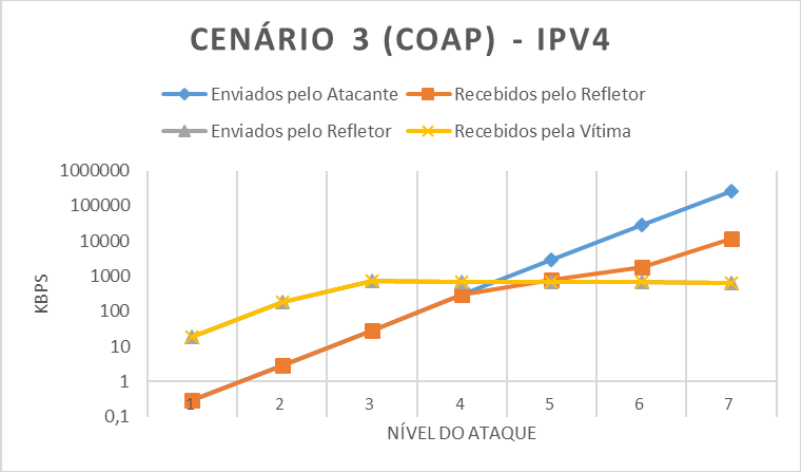

Figura 4.27: Cenário 3 (CoAP) - IPv4 - Kbps

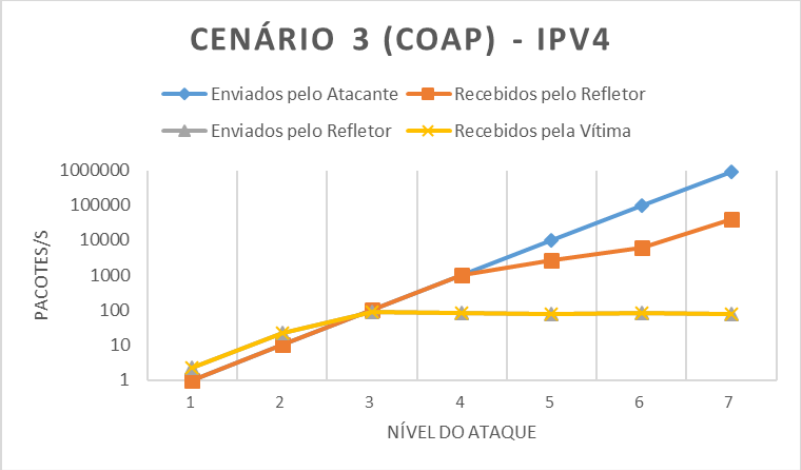

Figura 4.28: Cenário 3 (CoAP) - IPv4 - Pacotes/s

Tabela 4.36: Cenário 3 (CoAP) - IPv6 - Kbps

	Saída do Atacante		Entrada do Refletor		Saída do Refletor		Entrada da Vítima		
Nível	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Bytes***	***Kbps***	***Duração (s)***
1	560	0,4	560	0,4	21440	19,1	21440	19,1	8,99
2	5600	4,5	5600	4,5	214400	185,7	214400	185,7	9,24
3	56000	44,8	56000	44,8	1309984	762,0	1309984	762,0	13,75
4	560000	448,0	560000	448,0	694656	688,7	694656	688,7	8,07
5	2257360	1805,9	1193024	954,4	849024	718,8	849024	718,8	9,45
6	6675144	5340,1	3911544	3129,2	849024	719,1	849024	719,1	9,45
7	25340056	20272,0	16371544	13097,2	872608	751,7	872608	751,7	9,29

Tabela 4.37: Cenário 3 (CoAP) - IPv6 - Pacotes/s

	Saída do Atacante	Entrada do Refletor	Saída do Refletor	Entrada da Vítima
Nível	***Pacotes/s***	***Pacotes/s***	***Pacotes/s***	***Pacotes/s***
1	1,0	1,0	2,2	2,2
2	10,0	10,0	21,7	21,7
3	100,0	100,0	88,8	88,8
4	1000,0	1000,0	80,3	80,3
5	4031,0	2130,4	83,8	83,8
6	11919,9	6984,9	83,8	83,8
7	45250,1	29234,9	87,6	87,6

Tabela 4.38: Cenário 3 (CoAP) - IPv6 - ICMP

	Saída da Vítima		Entrada do Refletor	
Nível	***Pacotes ICMP***	***Bytes ICMP***	***Pacotes ICMP***	***Bytes ICMP***
1	10	12800	10	12800
2	100	128000	100	128000
3	611	782080	611	782080
4	324	414720	323	413440
5	396	506880	395	505600
6	396	506880	391	500480
7	407	520960	397	508160

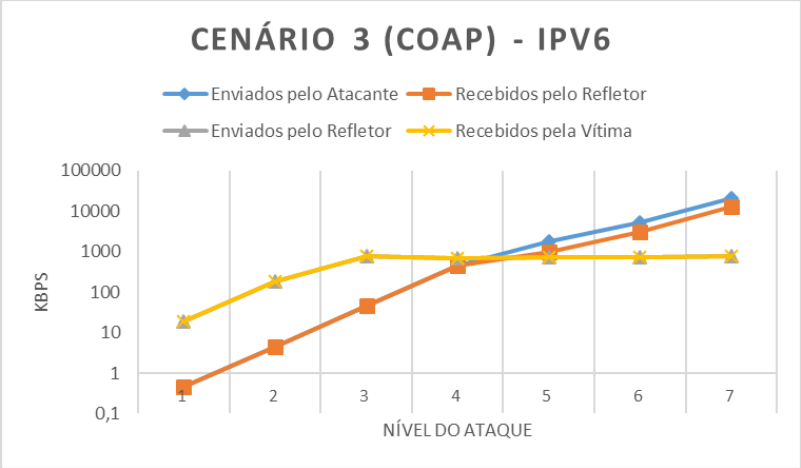

Figura 4.29: Cenário 3 (CoAP) - IPv6 - Kbps

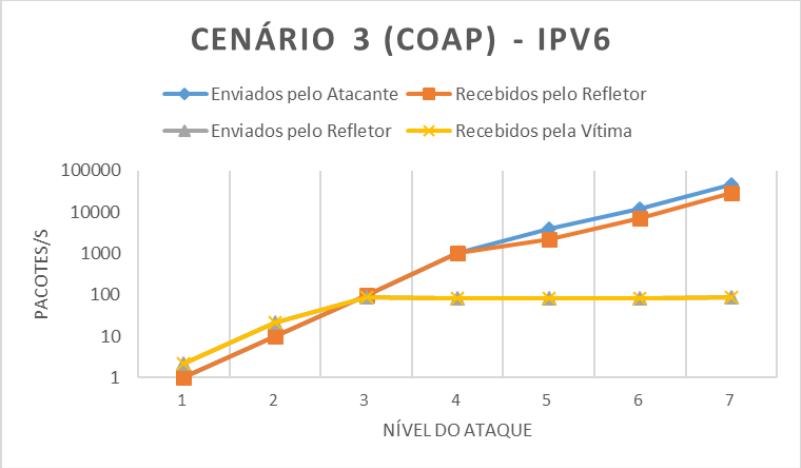

Figura 4.30: Cenário 3 (CoAP) - IPv6 - Pacotes/s

4.1.4 Fatores de Amplificação

Nesta subseção foram calculados os fatores de amplificação de *bytes* e de pacotes em todos os ataques realizados em cada cenário, de acordo com as fórmulas (2.1) e (2.2), descritas anteriormente na seção 2.2.2.

4.1.4.1 Cenário 1

A tabela 4.39 traz os fatores de amplificação obtidos no cenário 1, com a metodologia convencional, utilizando os protocolos SSDP (IPv4) e SNMP (IPv4), respectivamente. As taxas obtidas com esse refletor foram significativas, mas somente até o momento em que houve saturação no equipamento, caindo consideravelmente a partir do nível 2, para ambos os protocolos.

O fator de amplificação máximo de *bytes* de 1007x com SNMP foi alto para um equipamento tão limitado – o mais alto obtido em todos os testes realizados –, significando que para cada *byte*

enviado pelo atacante, 1007 *bytes* chegaram na vítima. Contudo, só foi possível obter esse fator com uma taxa de transmissão de 1 pacote/s, o que é muito baixo para um ataque volumétrico. Com SSDP, o fator de amplificação máximo de *bytes* foi de 88x, no nível 1. As figuras 4.31, 4.32, 4.33 e 4.34 ilustram bem o declínio dos fatores de amplificação a partir do nível 2 do ataque.

O fator de amplificação máximo de pacotes obtido com SSDP foi de 28x e com SNMP foi de 45x, ambos no nível 1, o que significa que para cada pacote de requisição enviado pelo atacante, foram refletidos 28 e 45 pacotes (ou fragmentos IP) de resposta para a vítima.

Com o protocolo SNMP, a partir do nível 5 (*bytes*) e do nível 3 (pacotes), se observou um comportamento chamado de atenuação, onde o refletor não consegue refletir tráfego a uma taxa maior que 1, ou seja, o refletor não é mais capaz de amplificar *bytes* ou pacotes, produzindo menos tráfego do que recebeu.

Tabela 4.39: Fatores de Amplificação - Cenário 1

	SSDP - IPv4		**SNMP - IPv4**	
Nível	***Bytes***	***Pacotes***	***Bytes***	***Pacotes***
1	88,393	28,000	1007,046	45,000
2	9,723	3,080	534,134	23,850
3	0,0	0,0	16,135	0,720
4	0,0	0,0	1,714	0,077
5	0,0	0,0	0,171	0,008
6	0,0	0,0	0,016	0,001
7	0,0	0,0	0,001	0,000

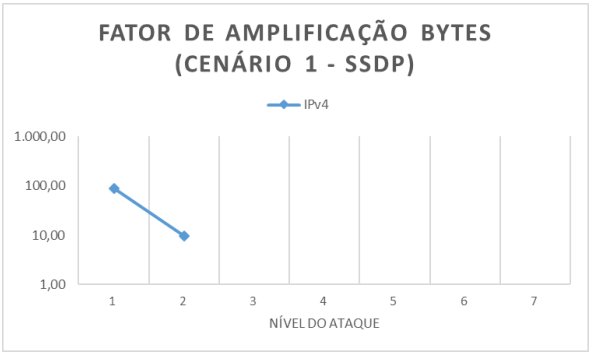

Figura 4.31: Fator de Amplificação *Bytes* (Cenário 1 - SSDP)

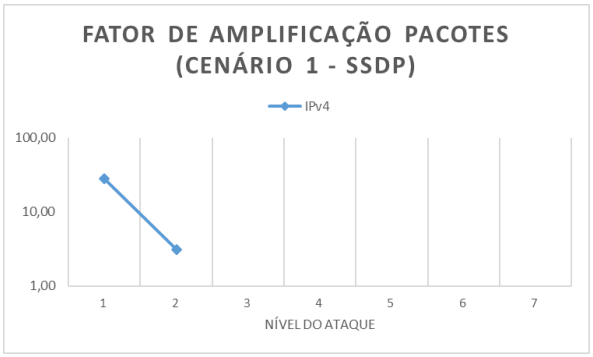

Figura 4.32: Fator de Amplificação Pacotes (Cenário 1 - SSDP)

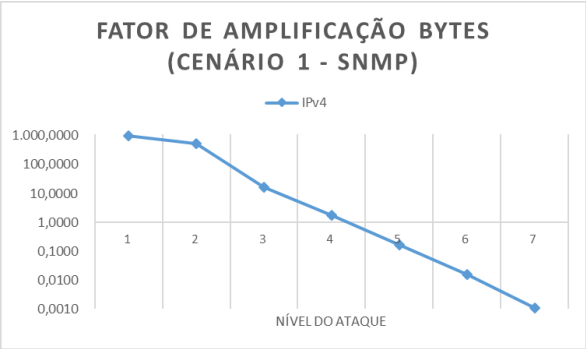

Figura 4.33: Fator de Amplificação *Bytes* (Cenário 1 - SNMP)

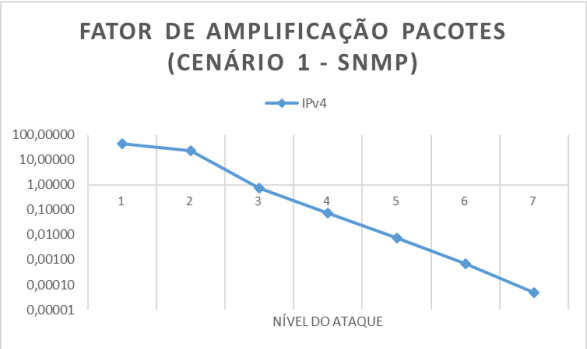

Figura 4.34: Fator de Amplificação Pacotes (Cenário 1 - SNMP)

4.1.4.2 Cenário 1 (Metodologia Alternativa)

Os fatores de amplificação obtidos usando a metodologia alternativa no cenário 1 foram compatíveis com os da convencional, porque em ambas foram consideradas as quantidades de *bytes* ou pacotes recebidos pela vítima. A diferença se dá no volume de dados recebidos nesse teste, que foi superior ao do primeiro.

Os fatores máximo de amplificação de *bytes* e pacotes obtidos são iguais aos da metodologia convencional, mas no caso do SSDP, eles também foram obtidos no nível 2. A tabela 4.40 e as figuras 4.35 e 4.36 (SSDP) e 4.37 e 4.38 (SNMP) ilustram melhor esses valores.

É possível notar uma atenuação via SSDP, a partir do nível 5, tanto para *bytes* quanto para pacotes. Com SNMP, a atenuação também ocorre a partir do nível 5 para *bytes*, mas inicia no nível 4 para pacotes.

Tabela 4.40: Fatores de Amplificação - Cenário 1 - Metodologia Alternativa

	SSDP - IPv4		**SNMP - IPv4**	
Nível	***Bytes***	***Pacotes***	***Bytes***	***Pacotes***
1	88,393	28,000	1007,046	45,000
2	88,393	28,000	443,455	19,800
3	42,429	13,440	52,409	2,340
4	5,931	1,879	5,342	0,239
5	0,787	0,249	0,524	0,023
6	0,200	0,063	0,052	0,002
7	0,047	0,015	0,005	0,000

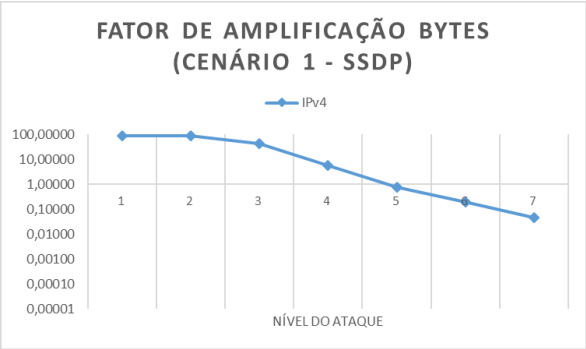

Figura 4.35: Fator de Amplificação *Bytes* (Cenário 1 - SSDP) - Metodologia Alternativa

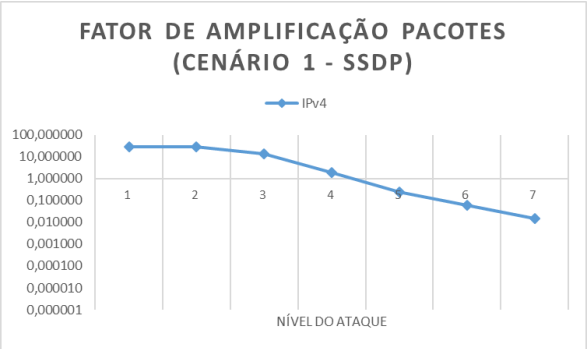

Figura 4.36: Fator de Amplificação Pacotes (Cenário 1 - SSDP) - Metodologia Alternativa

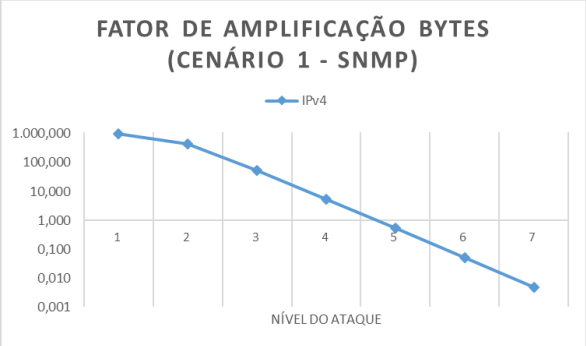

Figura 4.37: Fator de Amplificação *Bytes* (Cenário 1 - SNMP) - Metodologia Alternativa

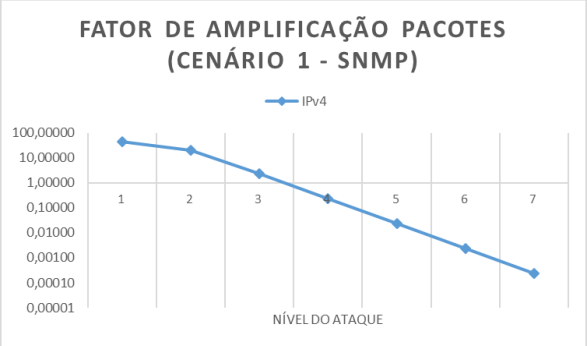

Figura 4.38: Fator de Amplificação Pacotes (Cenário 1 - SNMP) - Metodologia Alternativa

4.1.4.3 Cenário 2

A tabela 4.41 resume as taxas de amplificação obtidas no cenário 2, também utilizando os protocolos SSDP (IPv4) e SNMP (IPv4 e IPv6), respectivamente. Esse refletor foi o que apresentou fatores de amplificação menores, com respostas inferiores às dos demais em tamanho e com menos pacotes/fragmentos.

Com SSDP, a taxa se manteve praticamente constante até o nível 3, quando a câmera saturou, ocorrendo atenuação a partir do nível 5 (*bytes*) e do nível 4 (pacotes), conforme se observa nos gráficos 4.39 e 4.40. Com SNMP, a saturação aconteceu logo no nível 2, deixando os fatores praticamente idênticos a partir desse ponto.

O fator de amplificação máximo de *bytes* obtido com SSDP foi de 15x e com SNMP foi de 20x para IPv4 e 15x para IPv6, enquanto que o fator de amplificação máximo de pacotes obtido com SSDP foi de 4x e com SNMP foi de 1x, tanto para IPv4 quanto para IPv6, indicando que não houve amplificação de pacotes com SNMP. A diferença na amplificação de *bytes* entre IPv4 e IPv6 provavelmente está relacionado ao tamanho menor do cabeçalho IPv4, que produz requisições menores. Essa diferença pode ser melhor visualizada nos gráficos 4.41 e 4.42.

A atenuação, tanto com IPv4 quanto com IPv6, ocorreu no mesmo momento, a partir do nível

3 (*bytes*) e do nível 2 (pacotes).

Tabela 4.41: Fatores de Amplificação - Cenário 2

Nível	SSDP - IPv4		SNMP - IPv4		SNMP - IPv6	
	Bytes	***Pacotes***	***Bytes***	***Pacotes***	***Bytes***	***Pacotes***
1	15,361	4,000	20,323	1,000	15,772	1,000
2	15,361	4,000	2,033	0,100	1,578	0,100
3	14,024	3,652	0,589	0,029	0,473	0,030
4	1,633	0,425	0,061	0,003	0,047	0,003
5	0,198	0,052	0,007	0,000	0,013	0,001
6	0,042	0,011	0,002	0,000	0,013	0,001
7	0,015	0,004	0,001	0,000	0,012	0,001

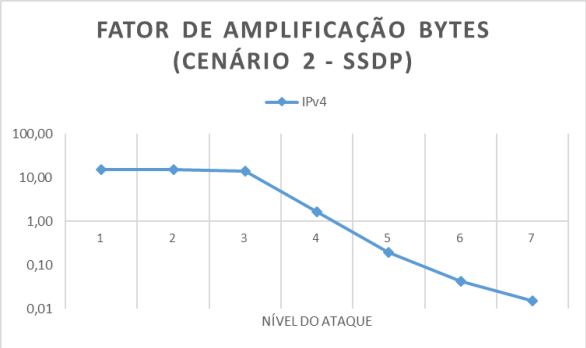

Figura 4.39: Fator de Amplificação *Bytes* (Cenário 2 - SSDP)

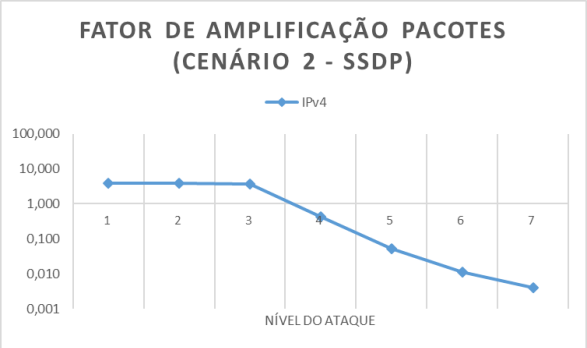

Figura 4.40: Fator de Amplificação Pacotes (Cenário 2 - SSDP)

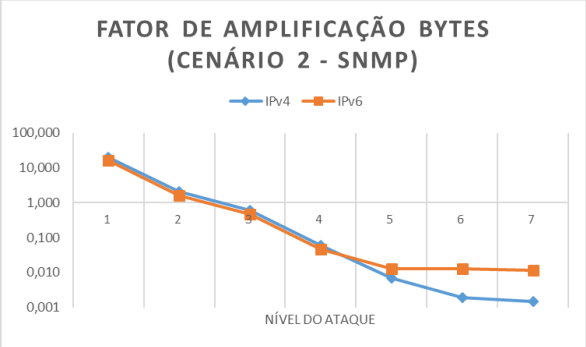

Figura 4.41: Fator de Amplificação *Bytes* (Cenário 2 - SNMP)

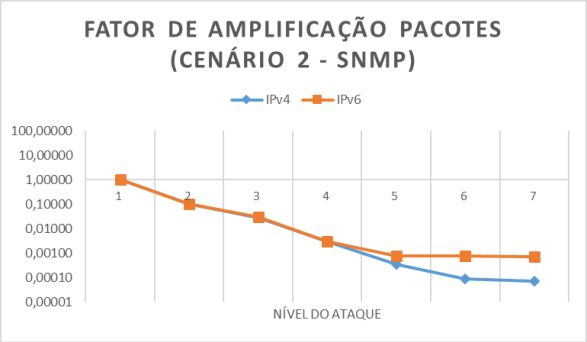

Figura 4.42: Fator de Amplificação Pacotes (Cenário 2 - SNMP)

4.1.4.4 Cenário 3

A tabela 4.42 apresenta as taxas de amplificação obtidas no cenário 3, utilizando os protocolos SSDP (IPv4 e IPv6), SNMP (IPv4 e IPv6) e CoAP (IPv4 e IPv6), respectivamente.

Com SSDP, os fatores de amplificação máximo de *bytes* obtidos foram 50x (IPv4) e 45x (IPv6) e de pacotes foi de 13x (IPv4/IPv6), se mantendo constante até praticamente o final do nível 3 (IPv4) e até o final do nível 3 (IPv6), quando houve saturação do equipamento, conforme se

observa nos gráficos 4.43 e 4.44.

Ocorreu atenuação na amplificação de pacotes tanto com IPv4 quanto com IPv6, ambos a partir do nível 5. Mas o curioso é que houve atenuação na amplificação de *bytes* apenas com IPv4 (a partir do nível 6). Com IPv6 (*bytes*), a partir do nível 5, foi observado uma amplificação residual sustentada, um comportamento que acontece quando o fator de amplificação se mantem maior que 1 após saturação e se sustenta até o final do ataque. Existe ainda a amplificação residual não sustentada, que dura por apenas um período, logo se transformando em atenuação. Entretanto, optamos por não caracterizá-la nesse trabalho, pois este foi o comportamento padrão encontrado em quase todos os cenários.

Com SNMP, o Raspberry atingiu fatores de amplificação máximo de *bytes* muito altos, próximos à do *gateway* ADSL, mas por apenas 2 níveis, sendo que as taxas foram maiores via IPv4 (972x) do que em relação ao IPv6 (737x). Os fatores de amplificação máximo de pacotes foram de 43x (IPv4) e 44x (IPv6), de acordo com os gráficos 4.45 e 4.46. Ocorreu atenuação, tanto de *bytes* quanto de pacotes, a partir do nível 4, em ambas as versões do protocolo IP.

Com CoAP, os fatores de amplificação máximo de *bytes* também foram maiores via IPv4 (58x) do que via IPv6 (38x), mas os de pacotes se mantiveram constantes tanto via IPv4 quanto via IPv6, a uma taxa de 2x. As taxas permaneceram praticamente constantes até o nível 3, quando houve saturação do refletor, acarretando em uma posterior atenuação, a partir do nível 5 (*bytes* - IPv4/IPv6) e dos níveis 3 (pacotes - IPv4) e 4 (pacotes - IPv6), como pode ser visto nos gráficos 4.47 e 4.48.

Tabela 4.42: Fatores de Amplificação - Cenário 3

	SSDP - IPv4		**SSDP - IPv6**		**SNMP - IPv4**		**SNMP - IPv6**		**CoAP - IPv4**		**CoAP - IPv6**	
Nível	***Bytes***	***Pacotes***	***Bytes***	***Pacotes***	***Bytes***	***Pacotes***	***Bytes***	***Pacotes***	***Bytes***	***Pacotes***	***Bytes***	***Pacotes***
1	50,355	13,000	45,068	13,000	972,877	43,000	737,894	44,000	58,444	2,000	38,286	2,000
2	50,355	13,000	45,068	13,000	971,732	43,000	736,668	44,000	58,444	2,000	38,286	2,000
3	46,566	12,022	45,068	13,000	197,520	8,686	133,352	7,964	28,930	0,990	23,393	1,222
4	10,625	2,743	8,298	2,394	0,295	0,013	0,222	0,013	2,612	0,089	1,240	0,065
5	1,098	0,283	1,980	0,571	0,000	0,000	0,000	0,000	0,164	0,006	0,376	0,020
6	0,153	0,039	1,079	0,311	0,068	0,003	0,529	0,032	0,032	0,001	0,127	0,007
7	0,069	0,018	1,492	0,430	0,006	0,000	0,048	0,003	0,002	0,000	0,034	0,002

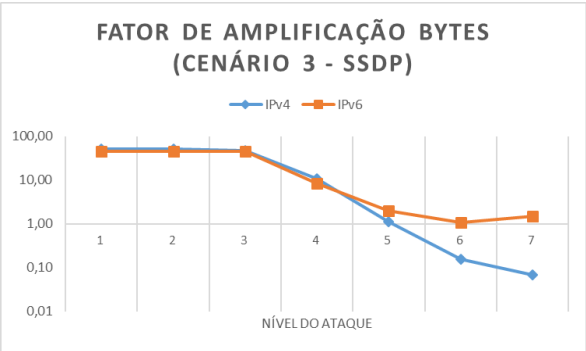

Figura 4.43: Fator de Amplificação *Bytes* (Cenário 3 - SSDP)

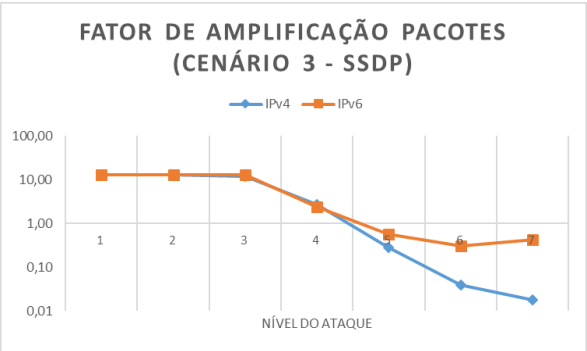

Figura 4.44: Fator de Amplificação Pacotes (Cenário 3 - SSDP)

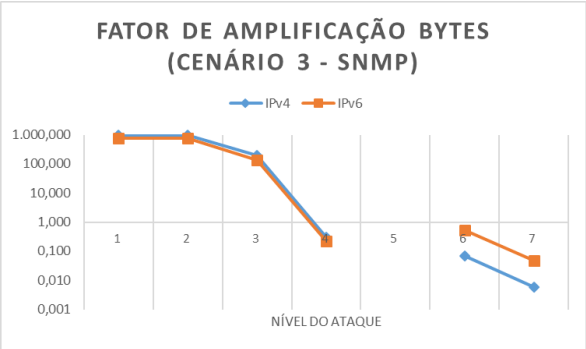

Figura 4.45: Fator de Amplificação *Bytes* (Cenário 3 - SNMP)

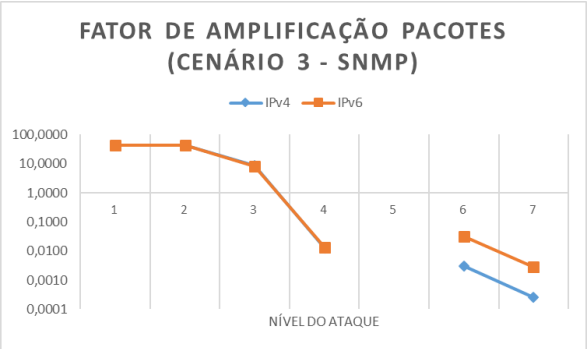

Figura 4.46: Fator de Amplificação Pacotes (Cenário 3 - SNMP)

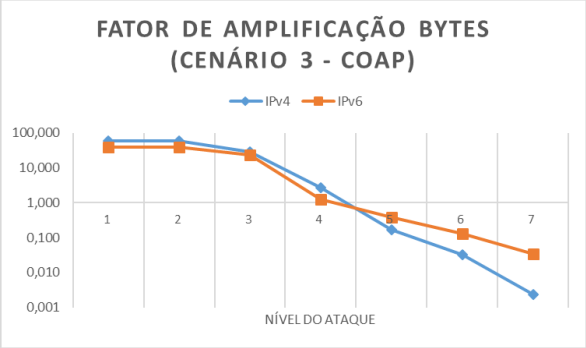

Figura 4.47: Fator de Amplificação *Bytes* (Cenário 3 - CoAP)

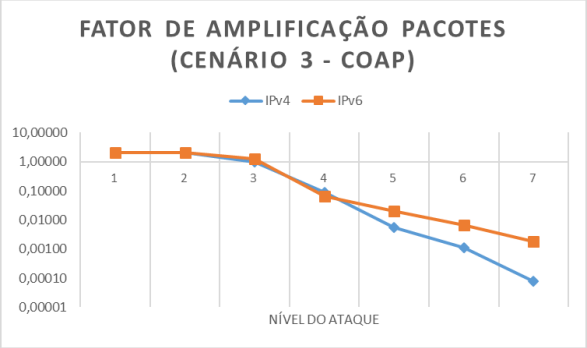

Figura 4.48: Fator de Amplificação Pacotes (Cenário 3 - CoAP)

4.1.4.5 Cenário 3 (Metodologia Alternativa)

Os fatores de amplificação obtidos usando a metodologia alternativa no cenário 3 foram similares com os da convencional, sendo que os fatores de amplificação máximo, tanto em *bytes* quanto em pacotes, foram idênticos. As divergências ocorrem principalmente a partir do nível 3. A tabela 4.43 e as figuras 4.49 e 4.50 ilustram melhor essa diferença.

A atenuação em *bytes*, dessa vez, ocorreu apenas no nível 6 (IPv4) e no nível 7 (IPv6), enquanto que a atenuação de pacotes ocorreu a partir do nível 5 (IPv4) e do nível 4 (IPv6).

Tabela 4.43: Fatores de Amplificação - Cenário 3 - Metodologia Alternativa

	SNMP - IPv4		**SNMP - IPv6**	
Nível	***Bytes***	***Pacotes***	***Bytes***	***Pacotes***
1	972,877	43,000	737,894	44,000
2	971,732	43,000	736,668	44,000
3	200,453	8,815	153,979	9,196
4	24,445	1,075	13,114	0,783
5	2,445	0,108	4,240	0,253
6	0,244	0,011	1,067	0,064
7	0,023	0,001	0,466	0,028

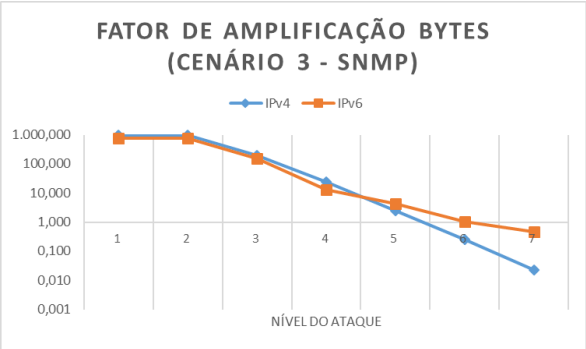

Figura 4.49: Fator de Amplificação *Bytes* (Cenário 3 - SNMP) - Metodologia Alternativa

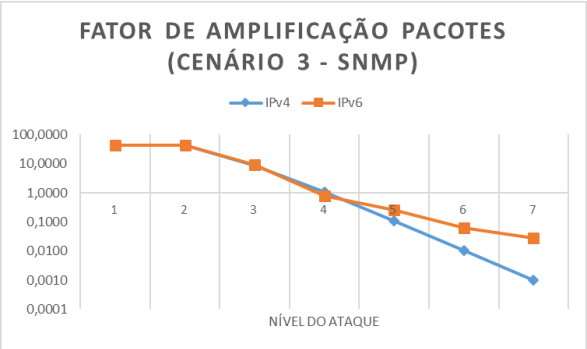

Figura 4.50: Fator de Amplificação Pacotes (Cenário 3 - SNMP) - Metodologia Alternativa

4.2 RESUMO DOS RESULTADOS

Esta seção apresenta um pequeno resumo de todos os resultados obtidos nessa dissertação. A tabela 4.44 exibe em que nível a saturação em *bytes* começou a ocorrer no refletor em cada um dos testes realizados, assim como o nível em que ela se consolidou, e a tabela 4.45 apresenta os mesmos resultados, mas para saturação de pacotes. As tabelas também exibem em qual nível começou a ocorrer atenuação e amplificação residual sustentada.

Tabela 4.44: Resumo da Saturação dos Refletores (Bytes)

Cenário 1								
	Nível Inicial		***Consolidação***		***Atenuação***		***Residual***	
Protocolo	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**
SSDP	2		2		-		-	
SNMP	2		2		5		-	

Cenário 1 (Metodologia Alternativa)								
	Nível Inicial		***Consolidação***		***Atenuação***		***Residual***	
Protocolo	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**
SSDP	3		4		5		-	
SNMP	2		3		5		-	

Cenário 2								
	Nível Inicial		***Consolidação***		***Atenuação***		***Residual***	
Protocolo	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**
SSDP	3		4		5		-	
SNMP	2	2	2	2	3	3	-	-

Cenário 3								
	Nível Inicial		***Consolidação***		***Atenuação***		***Residual***	
Protocolo	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**
SSDP	4	4	4	4	6	-	-	5
SNMP	2	2	3	3	4	4	-	-
CoAP	3	3	4	4	5	5	-	-

Cenário 3 (Metodologia Alternativa)								
	Nível Inicial		***Consolidação***		***Atenuação***		***Residual***	
Protocolo	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**
SNMP	2	2	3	3	6	7	-	-

Em todos os cenários, a saturação iniciou e se consolidou entre os níveis 2 a 4, tanto para *bytes* quanto para pacotes. Houve atenuação, quando não há mais amplificação, em praticamente todos os casos, com exceção do cenário 1 (SSDP), tanto para *bytes* quanto para pacotes, em virtude do travamento do equipamento e no cenário 3 (SSDP, via IPv6), somente em *bytes*, por este ter apresentado uma amplificação residual sustentada, o único caso em que isso ocorreu.

Tabela 4.45: Resumo da Saturação dos Refletores (Pacotes)

Cenário 1								
	Nível Inicial		***Consolidação***		***Atenuação***		***Residual***	
Protocolo	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**
SSDP	2		2		-		-	
SNMP	2		2		3		-	

Cenário 1 (Metodologia Alternativa)								
	Nível Inicial		***Consolidação***		***Atenuação***		***Residual***	
Protocolo	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**
SSDP	3		4		5		-	
SNMP	2		3		4		-	

Cenário 2								
	Nível Inicial		***Consolidação***		***Atenuação***		***Residual***	
Protocolo	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**
SSDP	3		4		4		-	
SNMP	2	2	2	2	2	2	-	-

Cenário 3								
	Nível Inicial		***Consolidação***		***Atenuação***		***Residual***	
Protocolo	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**
SSDP	4	4	4	4	5	5	-	-
SNMP	2	2	3	3	4	4	-	-
CoAP	2	2	3	3	3	4	-	-

Cenário 3 (Metodologia Alternativa)								
	Nível Inicial		***Consolidação***		***Atenuação***		***Residual***	
Protocolo	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**	**IPv4**	**IPv6**
SNMP	2	2	3	3	5	4	-	-

As figuras 4.51 e 4.52 agrupam os gráficos dos fatores de amplificação em *bytes* e pacotes, respectivamente, de todos os cenários com SSDP. Ignorando o comportamento atípico do cenário 1, com a metodologia convencional, e a amplificação residual sustentada no cenário 3, o comportamento de saturação observado é muito similar, com variações apenas no valor da amplificação.

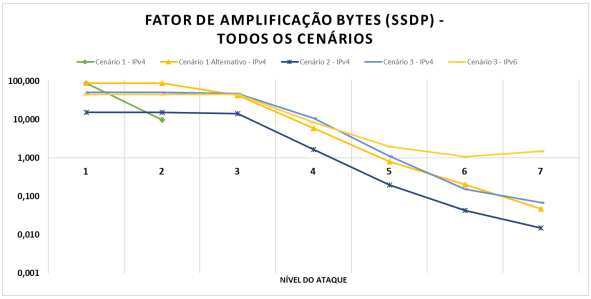

Figura 4.51: Fator de Amplificação *Bytes* - SSDP (Todos os Cenários)

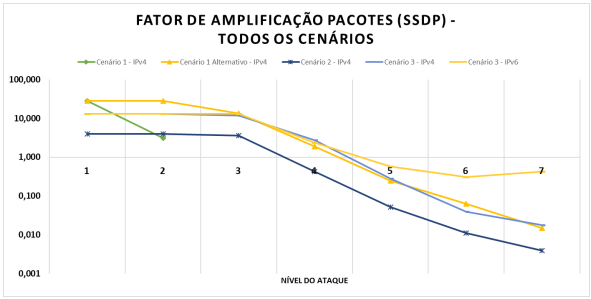

Figura 4.52: Fator de Amplificação Pacotes - SSDP (Todos os Cenários)

As figuras 4.53 e 4.54 agrupam os gráficos dos fatores de amplificação em *bytes* e pacotes, respectivamente, de todos os cenários com SNMP. Com exceção do comportamento atípico do cenário 3, com a metodologia convencional, onde houve uma queda abrupta no nível 4 e supressão da reflexão no nível 5, o comportamento de saturação observado também foi muito similar entre todos os dispositivos, inclusive com a metodologia alternativa aplicada no cenário 3. As maiores diferenças continuam sendo apenas no valor de amplificação.

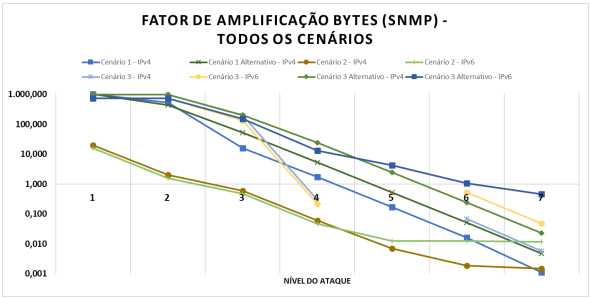

Figura 4.53: Fator de Amplificação *Bytes* - SNMP (Todos os Cenários)

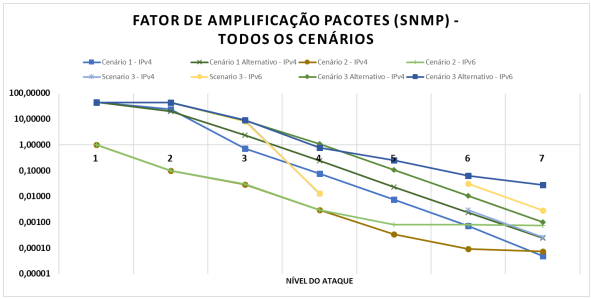

Figura 4.54: Fator de Amplificação Pacotes - SNMP (Todos os Cenários)

As figuras 4.55 e 4.56 agrupam os gráficos dos fatores de amplificação em *bytes* e pacotes, respectivamente, do protocolo CoAP, que ocorreram apenas no cenário 3. O comportamento de saturação observado continua parecido com os dos demais cenários.

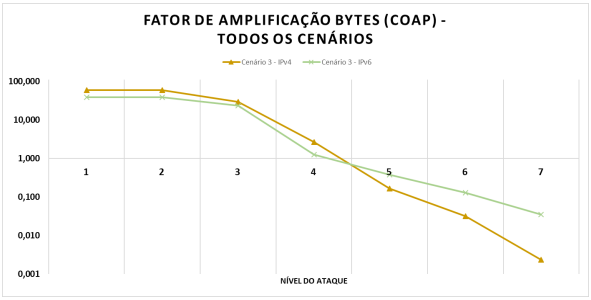

Figura 4.55: Fator de Amplificação *Bytes* - CoAP (Todos os Cenários)

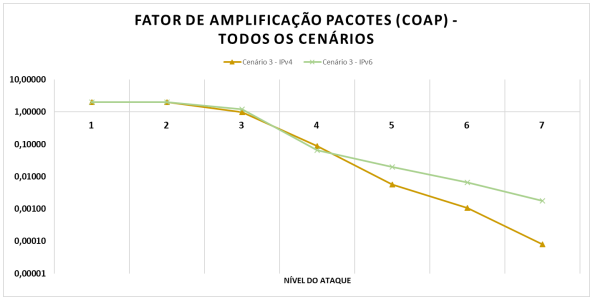

Figura 4.56: Fator de Amplificação Pacotes - CoAP (Todos os Cenários)

As figuras 4.57 e 4.58 reunem os gráficos dos fatores de amplificação em *bytes* e pacotes, respectivamente, de todos os protocolos e todos os cenários testados. Neles, fica ainda mais evidente o comportamento similar de saturação encontrado nessa dissertação, com a utilização de dispositivos IoT. Em geral, os dispositivos respondem bem até o nível 2 ou 3, quando começam a saturar e chegam ao ponto de não conseguirem mais amplificar tráfego.

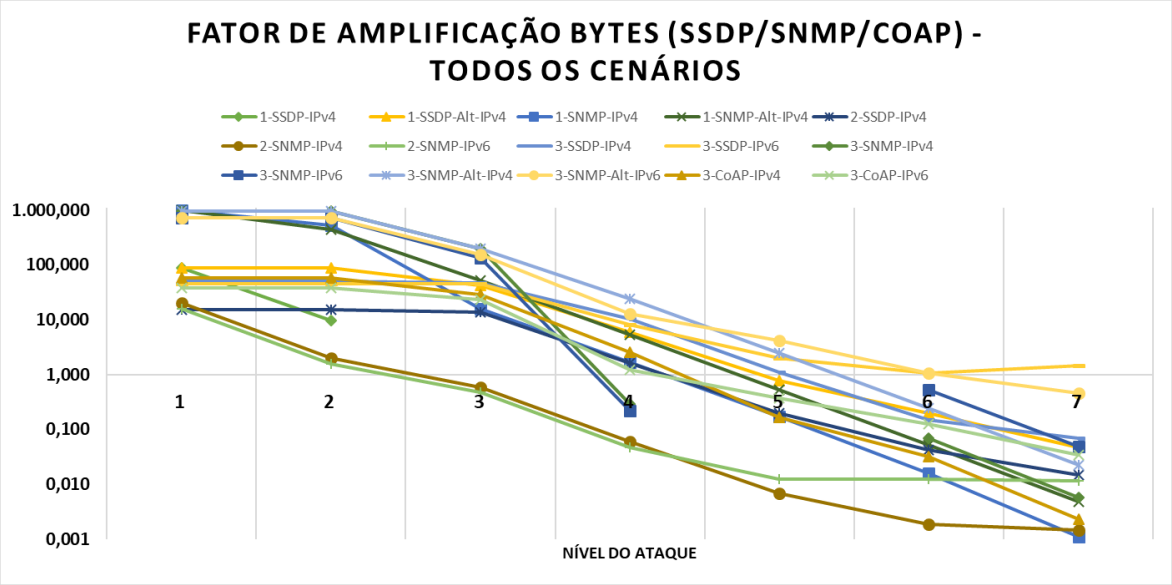

Figura 4.57: Fator de Amplificação *Bytes* - SSDP/SNMP/CoAP (Todos os Cenários)

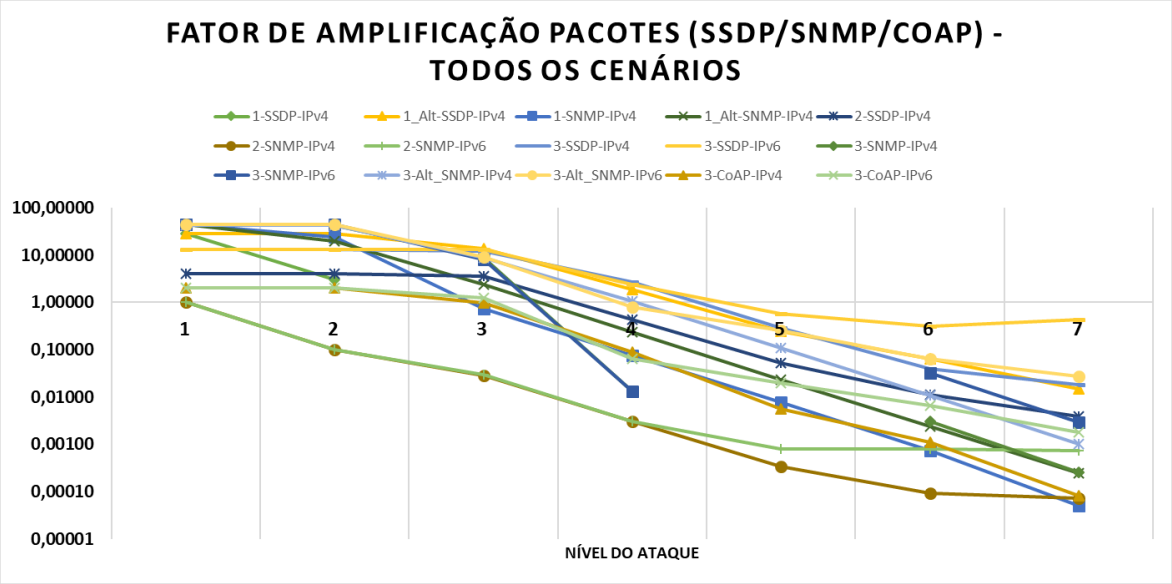

Figura 4.58: Fator de Amplificação Pacotes - SSDP/SNMP/CoAP (Todos os Cenários)

4.3 DISCUSSÃO

Os resultados dos testes indicaram um padrão similar de saturação dos refletores, ocorrida geralmente entre os níveis 2 a 4 da ferramenta utilizada e os fatores de amplificação, em geral, apresentaram comportamento compatível com a saturação, ou seja, caindo consideravelmente depois do dispositivo não conseguir mais refletir todo o tráfego recebido. Em quase todos as situações, se observou uma atenuação do tráfego posterior à saturação do refletor. Em apenas um caso observou-se amplificação residual após saturação.

Esse comportamento é consistente com o relatado em trabalhos anteriores ([16], [20], [22] e [33]). Deve-se notar que alguns desses trabalhos envolveram outros protocolos e diferentes especificações de refletores, sendo que alguns deles nem se tratavam de dispositivos IoT. Em todos os casos, a saturação do refletor se deu, tanto em IPv4 quanto em IPv6, geralmente no mesmo intervalo. Ou seja, a ocorrência do comportamento comum de saturação para níveis relativamente baixos de injeção de pacotes foi corroborada.

De um modo geral, as poucas diferenças se dão em fatores geralmente menores de amplificação e na ocorrência de saturação em níveis mais baixos, para alguns casos com dispositivos IoT, o que em um ataque real demandaria um recrutamento de mais dispositivos a serem utilizados no ataque, exigindo um planejamento e uma orquestração maior por parte do atacante.

Como esperado, as taxas de amplificação via SNMP foram bem superiores às do SSDP e CoAP, com exceção do cenário 2, onde elas ficaram muito próximas às do SSDP. Isso se deve ao fato da resposta GetBulkRequest da câmera IP não retornar muitos dados para a vítima, ao contrário dos outros dois cenários, supostamente por sua MIB não possuir muitas tabelas.

O *gateway* ADSL, provavelmente por ter o hardware mais limitado, teve uma saturação extrema com o protocolo SSDP utilizando a metodologia convencional, onde nenhuma resposta foi gerada por ele do nível 3 em diante. Se mais dispositivos IoT com poder limitado de processamento tivessem sido testados, como lâmpadas, impressoras ou sensores, presume-se que esse comportamento teria se repetido outras vezes.

Os testes com o cenário 1 (SSDP e SNMP) e com o cenário 3 (SNMP), por terem apresentado divergência com resultados anteriores, ou devido a ocorrência de comportamentos anômalos, foram refeitos utilizando-se uma metodologia alternativa. Os novos resultados alcançados revelaram situações bem peculiares, explicadas a seguir.

Com o protocolo SSDP (cenário 1), os resultados revelaram que a amplificação não ocorre na taxa de *bytes* enviados pelo refletor, que se mantém constante ao longo do tempo, mas sim no volume total de dados gerados, que vai se estendendo no tempo, como se ocorresse uma dilatação da duração do ataque. Ou seja, ao invés de todos os pacotes serem refletidos dentro do período de duração do ataque em cada nível (10 segundos), o refletor o fez com tempos variados, chegando a durar até 5 horas no nível 7.

Com o protocolo SNMP, tanto nos cenários 1 como 3, o tempo de duração em cada nível foi de aproximadamente 30 segundos, com algumas poucas exceções, indicando que, para o protocolo SNMP, o esforço computacional exigido do refletor é ainda maior que o dos demais protocolos. No cenário 1, comparando os resultados obtidos no trabalho [22], ocorreram algumas divergências, mas principalmente devido ao valor de max-repetitions utilizado na mensagem **GetBulkRequest** (2000 no trabalho anterior e 7758 nesse), gerando respostas de tamanhos diferentes.

Essas diferenças de comportamento reforçam que, do ponto de vista do atacante, não existe uma única forma de executar o ataque e que diversas variáveis devem ser consideradas, principalmente quando dispositivos IoT estão envolvidos.

O comportamento anômalo observado com o protocolo SSDP no cenário 2 é chamado de SSDP *Diffraction* e está descrito em [73]. Isso se deve à uma falha na biblioteca UPnPlibrary, possivelmente utilizada pela câmera IP, que leva o equipamento a responder com portas UDP efêmeras altas alocadas de forma pseudo-aleatória, ao invés da padrão 1900. Entretanto, no cenário 1, a porta de origem, apesar de não ser a padrão, é fixa, indicando outra possível falha e um desrespeito às especificações do protocolo. Todavia, em ambos os casos, os ataques puderam ser realizados normalmente, não interferindo nos resultados.

Apesar de as taxas de amplificação para CoAP terem sido satisfatórias, a quantidade de *bytes* que chegou à vítima foram geralmente menores do que em relação aos outros dois protocolos testados nesse trabalho, principalmente devido ao limite do tamanho da resposta imposto pelas especificações do protocolo. Deve-se ressaltar, contudo, que essas taxas foram obtidas graças ao dispositivo aceitar requisições com valor SZX igual a 7, que engana o servidor CoAP para indicar que as respostas devam ter 2048 *bytes* de tamanho, o que não é aceito por todas as implementações do CoAP. Além disso, geralmente os tamanhos dos recursos encontrados em dispositivos com CoAP costumam ser menores que 1024 *bytes*. Assim, em um ataque real espera-se observar uma taxa bem menor para este protocolo.

Na maior parte dos testes, a partir do nível 5, observou-se ainda que os atacantes não foram mais capazes de gerar pacotes na taxa máxima da ferramenta, indicando que eles também foram saturados. Em um ataque real, esses níveis não seriam recomendados, pois o ataque não seria eficiente, tanto do ponto de vista do atacante quanto do refletor. Foram poucos os casos em que não houve nenhuma saturação do atacante, e somente com o protocolo SNMP em IPv4. O interessante é que, mesmo o CoAP produzindo pacotes de requisição bem menores que o SSDP e SNMP, houve saturação do Raspberry com este protocolo, tanto via IPv4 (apenas no nível 7) quando via IPv6 (a partir do nível 5).

Também foi notada uma diferença no comportamento do tráfego refletido após a saturação, principalmente nos últimos níveis (5 a 7), onde o tráfego que chegava à vítima costumava ser menor do que o enviado pelo atacante, tornando esses níveis ainda menos interessantes para a realização de um ataque.

O tráfego ICMP e ICMPv6 gerado pelas vítimas em direção aos refletores revelou que essa sobrecarga de gerar e processar mais pacotes é prejudicial não somente para a vítima, que está sofrendo um ataque DDoS, mas também para os refletores, que precisam receber tais pacotes e processá-los em conjunto com o todo o tráfego do ataque que está sendo refletido, sobrecarregando-os ainda mais e possivelmente contribuindo para a sua prévia saturação.

Por fim, as taxas de amplificação geralmente menores para os ataques sobre IPv6 são consistentes com o esperado devido ao seu cabeçalho possuir o dobro do tamanho do IPv4, 40 *bytes* contra 20 *bytes*, provavelmente gerando uma sobrecarga um pouco maior de processamento nos dispositivos para tratar uma quantidade tão alta de pacotes.

4.4 RESUMO DO CAPÍTULO

Este capítulo apresentou os resultados dos onze ataques AR-DDoS realizados nessa dissertação, em ambiente controlado, utilizando três dispositivos IoT distintos como refletores (*gateway* ADSL, câmera IP e Raspberry Pi). Foram analisadas as capturas de tráfego e montadas tabelas e gráficos a partir delas, onde foram detalhados os comportamentos observados em cada um dos onze testes, tanto em relação ao tráfego gerado (em Kbps) quanto à quantidade de pacotes/s enviados e recebidos pelos equipamentos envolvidos, sendo destacados os momentos em que houve saturação dos equipamentos.

Em seguida, foram calculados os fatores de amplificação dos ataques e apresentada uma breve discussão sobre os resultados.

Devido à diferença nos resultados obtidos com o equipamento do cenário 1 entre esse trabalho e outro prévio, os testes do cenário 1 foram refeitos utilizando a metodologia utilizado no outro, chegando-se dessa vez a resultados bem similares. Do mesmo modo, os testes com SNMP no cenário 3 foram refeitos, devido a uma inconsistência nos resultados com a metodologia convencional.

5 CONCLUSÃO

Através dos testes realizados neste trabalho foi possível compreender, no contexto de IoT, como os ataques DDoS por reflexão amplificada funcionam com os protocolos SSDP, SNMP e CoAP, de forma que ficasse claro como cada parâmetro modificado do pacote de requisição influenciava na obtenção de maiores taxas de amplificação dos ataques.

Esse estudo também permitiu assimilar como dispositivos IoT utilizados como refletores se comportam em ataques dessa natureza, com as mais variadas taxas de requisições por segundo, de forma a entender até que ponto eles podem ser utilizados em ataques reais.

A versão 1.0.0 da ferramenta Linderhof, que foi revisada e aprimorada para ser utilizada nesse trabalho, se mostrou um excelente meio de se avaliar todos os aspectos dos ataques AR-DDoS analisados nesse trabalho.

Os resultados obtidos mostram que o ataque AR-DDoS sobre dispositivos IoT explorando os três protocolos estudados neste trabalho configuram uma ameaça que não pode ser desprezada, corroborando com trabalhos anteriores que analisaram o comportamento no contexto de equipamentos convencionais.

Os ataques revelaram, entretanto, que para dispositivos IoT, a taxa de transmissão de pacotes/s no momento da sua saturação eram relativamente baixas e, quanto mais limitado computacionalmente o equipamento, mais cedo eles saturavam. Como esperado, o Raspberry Pi foi o dispositivo que apresentou resultados mais consistentes com os três protocolos explorados, apesar de não ter obtido a melhor taxa de amplificação em alguns casos.

Ainda que o fator de amplificação obtido seja entre médio e baixo para taxas razoáveis de requisições/s e que estes equipamentos tenham a tendência de saturar mais rapidamente, o grande número de dispositivos IoT existentes no mundo, em sua maioria com escassos recursos de segurança ativados, oferece uma superfície de ataque significativa.

Contudo, para que um ataque real seja efetivo, seriam necessários centenas de milhares de dispositivos atuando como refletores, a uma taxa baixa de requisições por segundo, para que eles não saturem, podendo, dessa forma, sustentar o ataque por períodos mais longos. Todavia, a coordenação do ataque por parte do atacante seria mais complexa.

Este trabalho também possibilitou analisar as diferenças no comportamento de ataques AR-DDoS com as duas versões do protocolo IP (IPv4 e IPv6). Talvez a distinção mais significativa tenha sido nos fatores de amplificação obtidos, onde os testes com IPv6 geralmente resultaram em taxas menores, devido ao cabeçalho desta versão possuir o dobro do tamanho da versão mais antiga, fazendo com que o tamanho das requisições fossem maiores, influenciando diretamente no cálculo dos fatores de amplificação. Outro comportamento divergente foi em relação ao tráfego ICMPv6 gerado pela vítima em direção ao refletor, que foi maior do que o ICMP do IPv4, pois

o limite de tamanho da mensagem ICMPv6 é mais do que o dobro maior do que a do ICMP. No restante, não foram observadas diferenças entre as duas versões do protocolo IP.

Ainda em relação a esse tráfego ICMP/ICMPv6 gerado pela vítima, um atacante mais habilidoso, com poder de controlar não apenas *botnets* de atacantes, mas também de refletores, poderia inserir um código malicioso nestes, informando-os que, além de refletir o tráfego, eles deveriam forjar o endereço IP de origem dos pacotes refletidos, para que a vítima gerasse respostas ICMP/ICMPv6 para uma 2ª vítima ao invés de gerar para eles, causando um duplo efeito de segunda ordem no ataque.

Por fim, a diferença dos resultados quando diferentes metodologias foram adotadas nos cenários 1 e 3 mostrou como um atacante pode variar seu repertório para a condução de um ataque e como os sistemas de mitigação devem estar preparados para essa diversidade.

Estima-se que para emprego de dispositivos IoT em um ataque real eficaz, o atacante deveria conduzir as ações de forma bem mais sofisticada que a observada em ataques AR-DDoS até então, não só para obter a máxima eficiência na amplificação mas até para evitar a incapacitação do refletor. Deve-se também registrar que o comportamento observado com a metodologia alternativa no cenário 1, em que o ataque amplifica o volume global de tráfego mantendo uma taxa constante que se alonga no tempo, pode ser explorado por um atacante levando a ataques furtivos quanto à origem dos *probes*. Este seria um nível a mais de sofisticação na condução de ataques AR-DDoS.

5.1 TRABALHOS FUTUROS

Como sugestões de trabalhos futuros, podem ser testados e analisados outros dispositivos IoT, como impressoras, lâmpadas, sensores, entre outros, para que o leque de comparação entre eles seja maior. Poderia haver, ainda, um acréscimo no número de refletores nos ataques, a fim de se determinar o esforço necessário para que ele seja bem sucedido.

Como desdobramento deste trabalho, sugere-se ainda a realização dos testes em outros meios de interconexão dos dispositivos IoT, como o Wi-Fi, que é uma das formas mais comuns deles serem conectados a uma rede doméstica, por exemplo.

REFERÊNCIAS BIBLIOGRÁFICAS

1 SPECHT, S.; LEE, R. Distributed denial of service: Taxonomies of attacks, tools, and countermeasures. p. 543–550, 01 2004.

2 MAHJABIN, T.; XIAO, Y.; SUN, G.; JIANG, W. A survey of distributed denial-of-service attack, prevention, and mitigation techniques. *International Journal of Distributed Sensor Networks*, v. 13, n. 12, p. 1550147717741463, 2017.

3 HILTON, S. *Dyn Analysis Summary Of Friday October 21 Attack*. 2016. Disponível em: https://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack/. Acessado em: 16/10/2019.

4 VARGHESE, S. *DDoS attack on Dyn costly for company: claim*. 2017. Disponível em: https://www.itwire.com/security/76717-ddos-attack-on-dyn-costly-for-company-claim.html. Acessado em: 16/10/2019.

5 KOTTLER, S. *February 28th DDoS Incident Report*. 2018. Disponível em: https://github.blog/2018-03-01-ddos-incident-report/. Acessado em: 16/10/2019.

6 MORALES, C. *NETSCOUT Arbor Confirms 1.7 Tbps DDoS Attack; The Terabit Attack Era Is Upon Us*. 2018. Disponível em: https://www.netscout.com/blog/asert/netscout-arbor-confirms-17-tbps-ddos-attack-terabit-attack-era. Acessado em: 16/10/2019.

7 SHIELD, A. *Threat Landscape Report – Q1 2020*. 2020. Disponível em: https://aws-shield-tlr.s3.amazonaws.com/2020-Q1$_{A}WS_{S}hield_{T}LR.pdf.Acessadoem$: $16/07/2020$.

8 CLOUDFARE. *AWS hit by Largest Reported DDoS Attack of 2.3 Tbps*. 2020. Disponível em: https://www.a10networks.com/blog/aws-hit-by-largest-reported-ddos-attack-of-2-3-tbps/. Acessado em: 27/07/2020.

9 HASSAN, Z.; ALI, H.; BADAWY, M. Internet of things (iot): Definitions, challenges, and recent research directions. *International Journal of Computer Applications*, v. 128, p. 975–8887, 10 2015.

10 GAMBLIN, J. *Mirai Source-Code*. 2019. Disponível em: https://github.com/jgamblin/Mirai-Source-Code. Acessado em: 16/10/2019.

11 KOLIAS, C.; KAMBOURAKIS, G.; STAVROU, A.; VOAS, J. Ddos in the iot: Mirai and other botnets. *Computer*, v. 50, n. 7, p. 80–84, 2017. ISSN 0018-9162.

12 ROSSOW, C. Amplification Hell: Revisiting Network Protocols for DDoS Abuse. February 2014.

13 MEMCACHED. 2019. Disponível em: https://memcached.org/. Acessado em: 16/10/2019.

14 MUSTAPHA, H.; ALGHAMDI, A. M. Ddos attacks on the internet of things and their prevention methods. ACM, New York, NY, USA, p. 4:1–4:5, 2018. Disponível em: <http://doi.acm.org/10.1145/3231053.3231057>.

15 ZHANG, C.; GREEN, R. Communication security in internet of thing: Preventive measure and avoid ddos attack over iot network. Society for Computer Simulation International, San Diego, CA, USA, p. 8–15, 2015. Disponível em: <http://dl.acm.org/citation.cfm?id=2872550.2872552>.

16 GONDIM, J. J. C.; ALBUQUERQUE, R. D. O.; NASCIMENTO, A. C. A.; VILLALBA, L. J. G.; KIM, T.-H. A methodological approach for assessing amplified reflection distributed denial of service on the internet of things. *Sensors*, v. 16, n. 11, 2016. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/16/11/1855>.

17 NETWORKS, A. *The State of DDoS Weapons: Special Report by A10 Security Research*. 2019. Disponível em: https://www.a10networks.com/sites/default/files/A10-EB-14115-EN.pdf. Acessado em: 16/10/2019.

18 REVIEW, M. T. *The first DDoS attack was 20 years ago. This is what we've learned since*. 2019. Disponível em: https://www.technologyreview.com/2019/04/18/103186/the-first-ddos-attack-was-20-years-ago-this-is-what-weve-learned-since/. Acessado em: 27/07/2020.

19 BURHAN, M.; REHMAN, R. A.; KIM, B.-S.; KHAN, B. Iot elements, layered architectures and security issues: A comprehensive survey. *Sensors*, v. 18, 08 2018.

20 GONDIM, J. J. C.; ALBUQUERQUE, R. d. O. Mirror saturation in amplified reflection ddos. *V Jornadas Nacionales de Investigación en Ciberseguridad, JNIC 2019*, p. 185–190, June 2019.

21 LYU, M.; SHERRATT, D.; SIVANATHAN, A.; GHARAKHEILI, H. H.; RADFORD, A.; SIVARAMAN, V. Quantifying the reflective ddos attack capability of household iot devices. ACM, New York, NY, USA, p. 46–51, 2017. Disponível em: <http://doi.acm.org/10.1145/3098243.3098264>.

22 VASQUES, A.; GONDIM, J. Amplified reflection ddos attacks over iot mirrors: A saturation analysis. Oct 2019.

23 VASQUES, A.; GONDIM, J. Ataques ddos por reflexão amplificada sobre refletor iot rodando coap. Jun 2020.

24 DANTAS, A.; VIEIRA, M.; VASQUES, A.; GONDIM, J. Linderhof: uma ferramenta para avaliação de sistemas de mitigação de ataques reflexivos volumétricos (ddos). Dez 2020.

25 (ISOC), I. S. *Addressing the Challenge of IP Spoofing*. 2015. Disponível em: https://www.internetsociety.org/resources/doc/2015/addressing-the-challenge-of-ip-spoofing/. Acessado em: 18/11/2019.

26 NETWORK Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. RFC Editor, 2000. BCP 38. (Best Current Practice, 38). Disponível em: <https://tools.ietf.org/html/bcp38>.

27 INGRESS Filtering for Multihomed Networks. RFC Editor, 2004. BCP 84. (Best Current Practice, 84). Disponível em: <https://tools.ietf.org/html/bcp84>.

28 (CAIDA), C. for A. I. D. A. *State of IP Spoofing*. 2020. Disponível em: https://spoofer.caida.org/summary.php. Acessado em: 12/05/2020.

29 CYBERSECURITY, T. N.; (NCCIC), C. I. C. *Understanding Denial-of-Service Attacks*. 2009. Disponível em: https://www.us-cert.gov/ncas/tips/ST04-015. Acessado em: 18/11/2019.

30 RADWARE. *History of DDoS Attacks*. 2017. Disponível em: https://security.radware.com/ddos-knowledge-center/ddos-chronicles/ddos-attacks-history/. Acessado em: 18/11/2019.

31 DOULIGERIS, C.; MITROKOTSA, A. Ddos attacks and defense mechanisms: Classification and state-of-the-art. *Computer Networks*, v. 44, p. 643–666, 04 2004.

32 KüHRER, M.; HUPPERICH, T.; ROSSOW, C.; HOLZ, T. Hell of a handshake: Abusing tcp for reflective amplification ddos attacks. USENIX Association, Berkeley, CA, USA, p. 4–4, 2014. Disponível em: <http://dl.acm.org/citation.cfm?id=2671293.2671297>.

33 GONDIM, J.; ALBUQUERQUE, R.; OROZCO, A. L. Mirror saturation in amplified reflection distributed denial of service: A case of study using snmp, ssdp, ntp and dns protocols. *Future Generation Computer Systems*, v. 108, 2020. ISSN 0167-739X. Disponível em: <https://www.sciencedirect.com/science/article/abs/pii/S0167739X19322745>.

34 FORUM, U. *UPnP Device Architecture 1.0*. 2008. Disponível em: http://upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.0.pdf. Acessado em: 19/11/2018.

35 FORUM, U. *UPnP Device Architecture 1.1*. 2008. Disponível em: http://upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf. Acessado em: 19/11/2018.

36 FORUM, U. *UPnP Device Architecture 2.0*. 2015. Disponível em: http://upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v2.0.pdf. Acessado em: 19/11/2018.

37 MAJKOWSKI, M. *Stupidly Simple DDoS Protocol (SSDP) generates 100 Gbps DDoS*. 2017. Disponível em: https://blog.cloudflare.com/ssdp-100gbps/. Acessado em: 16/04/2020.

38 A Simple Network Management Protocol (SNMP). RFC Editor, 1990. RFC 1157. (Request for Comments, 1157). Disponível em: <https://rfc-editor.org/rfc/rfc1157.txt>.

39 INTRODUCTION to Community-based SNMPv2. RFC Editor, 1996. RFC 1901. (Request for Comments, 1901). Disponível em: <https://rfc-editor.org/rfc/rfc1901.txt>.

40 INTRODUCTION and Applicability Statements for Internet Standard Management Framework. RFC Editor, 2002. RFC 3410. (Request for Comments, 3410). Disponível em: <https://rfc-editor.org/rfc/rfc3410.txt>.

41 THE Constrained Application Protocol (CoAP). RFC Editor, 2014. RFC 7252. (Request for Comments, 7252). Disponível em: <https://rfc-editor.org/rfc/rfc7252.txt>.

42 BLOCK-WISE Transfers in the Constrained Application Protocol (CoAP). RFC Editor, 2016. RFC 7959. (Request for Comments, 7959). Disponível em: <https://rfc-editor.org/rfc/rfc7959.txt>.

43 PERAKOVIC, D.; PERIšA, M.; CVITIć, I. Analysis of the iot impact on volume of ddos attacks. 12 2015.

44 SALIM, M.; RATHORE, S.; PARK, J. Distributed denial of service attacks and its defenses in iot: a survey. *The Journal of Supercomputing*, 07 2019.

45 OLSHANSKY, S.; WILTON, R. *Internet of Things Devices as a DDoS Vector*. 2019. Disponível em: https://www.internetsociety.org/blog/2019/04/internet-of-things-devices-as-a-ddos-vector/. Acessado em: 25/11/2019.

46 ALLADI, T.; CHAMOLA, V.; SIKDAR, B.; CHOO, K.-K. R. Consumer iot: Security vulnerability case studies and solutions. *IEEE Consumer Electronics Magazine*, v. 9, n. 2, p. 17–25, 2020.

47 Kumar, S. A.; Vealey, T.; Srivastava, H. Security in internet of things: Challenges, solutions and future directions. p. 5772–5781, 2016.

48 ELKHODR, M.; SHAHRESTANI, S. A.; CHEUNG, H. The internet of things: New interoperability, management and security challenges. *ArXiv*, abs/1604.04824, 2016.

49 NOOR, M.; HASSAN, W. Current research on internet of things (iot) security: A survey. *Computer Networks*, v. 148, 2019. ISSN 1389-1286. Disponível em: <https://www.sciencedirect.com/science/article/abs/pii/S1389128618307035>.

50 SIRISHA, U.; LAKSHMEESWARI, G. A survey on internet of things : Applications and layered wise security issue. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT)*, v. 5, p. 171–180, 2019. ISSN 2456-3307.

51 Neshenko, N.; Bou-Harb, E.; Crichigno, J.; Kaddoum, G.; Ghani, N. Demystifying iot security: An exhaustive survey on iot vulnerabilities and a first empirical look on internet-scale iot exploitations. *IEEE Communications Surveys Tutorials*, v. 21, n. 3, p. 2702–2733, 2019.

52 XIAO, L.; WAN, X.; LU, X.; ZHANG, Y.; WU, D. Iot security techniques based on machine learning: How do iot devices use ai to enhance security? *IEEE Signal Processing Magazine*, v. 35, n. 5, p. 41–49, 2018.

53 MINOLI, D.; OCCHIOGROSSO, B. Blockchain mechanisms for iot security. *Internet of Things*, v. 1-2, n. 5, p. 1–13, 2018. ISSN 2542-6605.

54 G., D.; T.V., R.; D., B. B.; DURGA, S. N. Ddos attacks—analysis and prevention. *Innovations in Computer Science and Engineering. Lecture Notes in Networks and Systems*, v. 32, p. 1–10, 2019.

55 BAWANY, N.; SHAMSI, J.; SALAH, K. Ddos attack detection and mitigation using sdn: Methods, practices, and solutions. *Arabian Journal for Science and Engineering*, v. 42, p. 425–441, 2017.

56 ABUBAKAR, R.; ALDEGHEISHEM, A.; MAJEED, M. F.; MEHMOOD, A.; MARYAM, H.; ALRAJEH, N. A.; MAPLE, C.; JAWAD, M. An effective mechanism to mitigate real-time ddos attack. *IEEE Access*, v. 8, p. 126215–126227, 2020.

57 ALMEIDA, M. P. de; JúNIOR, R. T. de S.; VILLALBA, L. J. G.; KIM, T.-H. New dos defense method based on strong designated verifier signatures. *Sensors*, v. 18, n. 9, 2018. ISSN 1424-8220. Disponível em: <http://www.mdpi.com/1424-8220/18/9/2813>.

58 Thomas, D. R.; Clayton, R.; Beresford, A. R. 1000 days of udp amplification ddos attacks. p. 79–84, 2017.

59 Lavrenovs, A. Towards measuring global ddos attack capacity. v. 900, p. 1–15, 2019.

60 KHOOI, X.; CSIKOR, L.; DIVAKARAN, D. M.; KANG, M. Dida: Distributed in-network defense architecture against amplified reflection ddos attacks. 06 2020.

61 Vlajic, N.; Zhou, D. Iot as a land of opportunity for ddos hackers. *Computer*, v. 51, n. 7, p. 26–34, 2018.

62 PACHECO, L.; GONDIM, J.; BARRETO, P.; ALCHIERI, E. Evaluation of distributed denial of service threat in the internet of things. p. 89–92, 2016.

63 OLIVEIRA, S.; LINHARES, C.; TRAVENçOLO, B.; MIANI, R. Investigation of amplification-based ddos attacks on iot devices. *INFOCOMP Journal of Computer Science*, v. 19, 2020. Disponível em: <http://infocomp.dcc.ufla.br/index.php/infocomp/article/view/765>.

64 ŠIMON, M.; HURAJ, L. A study of ddos reflection attack on internet of things in ipv4/ipv6 networks. *Computer Science On-line Conference (CSOC) 2019: Software Engineering Methods in Intelligent Algorithm*, p. 109–118, May 2019.

65 HENGST, K. Ddos through the internet of things. 2016.

66 VLAJIC, N.; ZHOU, D.; TUNG, J. Iot cameras and dvrs as ddos reflectors: Pros and cons from hacker's perspective. p. 181–187, 10 2018.

67 MIRANDA, I. F.; GONDIM, J. J. C. *Ataque de negação de serviço por reflexão amplificada explorando Memcached*. 2019. Trabalho de Conclusão de Curso (Bacharelado em Engenharia da Computação)—Universidade de Brasília, Brasília.

68 VIEIRA, A. A. d. S.; GONDIM, J. J. C. *Ataque de negação de serviço por reflexão amplificada explorando Memcached*. 2019. Trabalho de Conclusão de Curso (Bacharelado em Engenharia da Computação)—Universidade de Brasília.

69 SALDANHA, R. d. S.; GONDIM, J. J. C. *Ataque de negação de serviço por reflexão amplificada explorando Memcached*. 2019. Trabalho de Conclusão de Curso (Bacharelado em Engenharia da Computação)—Universidade de Brasília.

70 PEREIRA, P. H. M.; GONDIM, J. J. C. *Internet das coisas e seus riscos: uma análise da exploração de servidores CoAP como refletores de ataques de negação de serviço amplificados*. 2019. Trabalho de Conclusão de Curso (Bacharelado em Engenharia da Computação)—Universidade de Brasília.

71 BRITO, S. H. B. *Configuração de Servidor DHCPv6 no Linux*. 2015. Disponível em: http://labcisco.blogspot.com/2015/09/configuracao-de-servidor-dhcpv6-no-linux.html. Acessado em: 20/04/2020.

72 REQUIREMENTS for IP Version 4 Routers. RFC Editor, 1995. RFC 1812. (Request for Comments, 1812). Disponível em: <https://rfc-editor.org/rfc/rfc1812.txt>.

73 DOBBINS, R. *The Importance of Being Accurate: SSDP Diffraction Attacks, UDP Refraction Attacks, and UPnP NAT Bypass*. 2018. Disponível em: https://br.netscout.com/blog/asert/importance-being-accurate-ssdp-diffraction-attacks-udp. Acessado em: 20/07/2020.