DamBuster
Uma ferramenta de avaliação de soluções de mitigação de DoS volumétrico direto

Eduardo Sousa da Silva, Paulo Mauricio Costa Lopes, João José Costa Gondim

[1]Departamento de Ciência da Computação - Universidade de Brasília (UnB)
Campus Universitário Darcy Ribeiro, Brasília-DF | CEP 70910-900

edusousa1996@hotmail.com, reset.req@gmail.com, gondim@unb.br

Abstract. *DamBuster is a dual application tool for studying volumetric attacks and also evaluating and benchmarking mitigation systems for volumetric direct denial of service flooding attacks, implemented in a modular architecture. The tool implements attacks abusing various protocols, under customized conduction tactics and controlled intensity. Application performance is analyzed by comparing the results of available attacks, running on different hardware configurations and against a commonly used reference tool (T50). The results show that DamBuster is a scalable and efficient tool for packet injection and traffic generation. The average packet generation and forwarding rates obtained were higher and more consistent than those of the T50.*

Resumo. *DamBuster é uma ferramenta de aplicação dupla para estudo de ataques volumétricos e também avaliação e benchmarking de sistemas de mitigação para ataques volumétricos de negação direta de serviço e inundação, implementados em uma arquitetura modular. A ferramenta implementa ataques que abusam de diversos protocolos, sob táticas de condução customizadas e intensidade controlada. O desempenho do aplicativo é analisado comparando os resultados dos ataques disponíveis, executados em diferentes configurações de hardware e com uma ferramenta de referência comumente usada (T50). Os resultados mostram que o DamBuster é uma ferramenta escalável e eficiente para injeção de pacotes e geração de tráfego. As taxas médias de geração e encaminhamento de pacotes obtidas foram maiores e mais consistentes que as do T50.*

1. Introdução

DamBuster é uma ferramenta de uso dual com foco em estudar e aferir a execução de ataques de negação de serviço volumétricos (Volumetric Denial of Service - DoS) diretos, auxiliando a avaliação de sistemas de mitigação e estudos. A ferramenta suporta diferentes formas do ataque, tanto baseados em protocolos TCP quanto UDP, com total controle da condução dos ataques. DamBuster é um programa de computador registrado junto ao INPI sob o certificado BR512023001023-5

A ferramenta DamBuster foi desenvolvida na linguagem C, para sistemas Linux e é baseada no Linderhof Versão 2.0.0 [Vieira et al. 2021], [Dantas et al. 2020], que é voltada ao estudo de ataques de negação de serviço volumétricos, em específico os de

reflexão amplificada. utilizando a *engine* de geração de pacotes desta e incorporou diversas melhorias, onde se destacam: alterações na injeção de pacotes para ser utilizada por ataques volumétricos diretos; implementação de seis novos ataques, baseados em três protocolos; refatoração completa da lógica de criação de cabeçalhos e implementação de protocolos; refatoração da comunicação com o usuário por meio da criação do módulo *logger*, o qual gera arquivos de saída com relatórios de execução; e aperfeiçoamento das interfaces gráficas e de linha de comando.

Este artigo está organizado da seguinte forma: a Seção 2 apresenta a arquitetura e as funcionalidades da ferramenta; Seção 3, os resultados dos testes de desempenho são apresentados e comparados com uma implementação de referência; Seção 4 explica como será a demonstração da utilização da ferramenta no salão, e indica onde seu código e sua versão *dockerizada* estão disponíveis; por fim, a seção 5 fecha com as conclusões.

2. Arquitetura e Funcionalidades

O DamBuster se baseia na ferramenta Linderhof 2.0.0 [Vieira et al. 2021], que também foi desenvolvida na linguagem C, o qual possui porte aos protocolos CoAP, DNS, Memcached, NTP, SNMP, SSDP e CLDAP ([Zeilenga 2003], [Fedor et al. 1990], [Case et al. 1996] e [Fuller et al. 1993]). No Linderhof 2.0.0 os pacotes eram gerados e enviados para refletores (mirrors) e os refletores enviavam uma resposta ainda maior para a(s) vítima(s), no DamBuster o comportamento é diferente pois nele existem implementações para ataques volumétricos diretos. Específicamente, o DamBuster utiliza porções do código da *engine* de geração de pacotes. Como foram alteradas funcionalidades e arquitetura a partir do Linderhof 2.0.0 para implementar ataques DDoS diretos de *flooding*, a ferramenta resultante recebeu o nome de DamBuster, fazendo alusão ao esquadrão de aviação inglesa da Segunda Guerra Mundial que teve a missão de bombardear represas alemãs, causando inundações e desestruturando econômica e moralmente a Alemanha Nazista [Dildy 2012]. Dessa associação, os diferentes ataques suportados pelo DamBuster são chamados de bombas.

2.1. Arquitetura

Do ponto de vista funcional, o DamBuster é um gerador de pacotes com controle de vazão, podendo assumir valores constantes por intervalos de tempo determinados, ou variar conforme definido pelo usuário. Assim, a arquitetura do DamBuster suporta tais funcionalidades, onde um dos requisitos do projeto consistia em facilitar a implementação de diversas formas de ataques DDoS volumétricos diretos (*flooding*). O DamBuster é composto por quatro módulos base: Interface, Commander, Arsenal e Injector, e um módulo auxiliar **Common**, contendo funções para auxiliar na sua execução. A arquitetura é mostrada na Figura 1 e os módulos descritos a seguir.

- **Interface:** Módulo responsável por implementar a entrada e a saída de dados para comunicação com o usuário, contendo dois tipos de interface, uma por linha de comando (do inglês, Command Line Interface (CLI)) e uma gráfica (do inglês, Graphical User Interface (GUI)).
- **Commander:** O módulo é dividido em dois componentes, o primeiro, commander é responsável por fazer inicialização de variáveis necessárias para a execução dos ataques, e gerenciar o manager na criação e execução dos ataques. O manager é responsavel por planejar e executar o ataque de acordo com o *draft*. Com

base nessa informação, ele requisita ao módulo Arsenal a criação de uma bomba referente ao ataque escolhido.

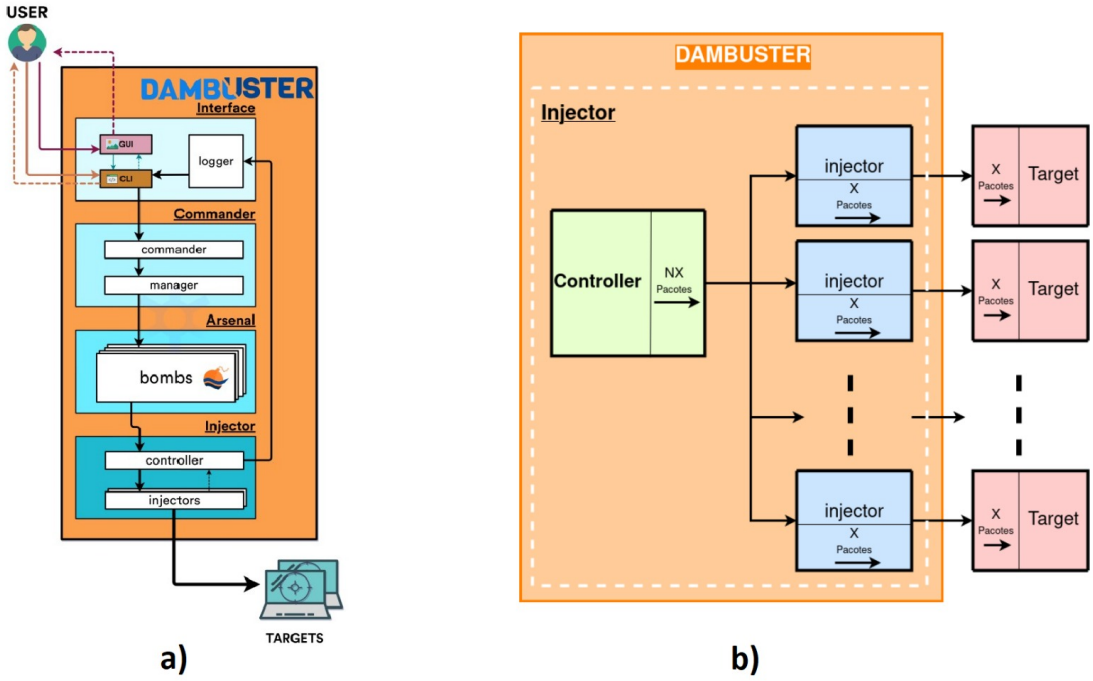

Figura 1. a) Arquitetura do DamBuster. b) Modo Padrão.

- **Arsenal:** Módulo contendo todas as "bombas" disponíveis, também responsável pela criação de pacotes que serão enviados durante o ataque, sua estrutura é modular facilitando futuras implementações de protocolos de rede ou transporte. O preenchimento dos cabeçalhos dos datagramas é feito utilizando o *blacksmith*, parte do módulo *Common*.
- **Injector:** Módulo dividido em dois componentes: **controler** e **injector**.
  - O *injector* lida com a criação de sockets e envio dos pacotes pela rede, e é a camada mais inferior da arquitetura. Cada injector é responsável por enviar pacotes para um alvo, dessa forma, cada injector ocupa uma thread da aplicação.
  - O *controller* é responsável por gerenciar os *injectors* e orquestrar o ataque. Durante o ataque, o *controller* emite feedback do ataque para o modulo interface, utilizando o *logger*.

2.2. Funcionalidades

As funcionalidades do DamBuster incluem várias formas de execução dos ataques , como: definição do alvo (endereço e porta) e da origem (*spoofing*); intensidade, duração e evolução dos ataques; e formas de condução (*carpet bombing* e *pulse wave*). Estas opções são disponibilizadas via GUI e estão detalhadas em **https://eduardoforca.gitlab.io/dambuster**. As principais funcionalidades são descritas a seguir:

- Geração de Tráfego: Os ataques do DamBuster são efetuados por meio da geração de tráfego de rede. Estes ataques são controlados por diferentes modos de ataque,

além de parâmetros gerais, compartilhados entre os modos. Os parâmetros e modos são utilizados para determinar como esses pacotes serão enviados no decorrer da execução do programa. Estão implementados quatro modos de ataques, que alteram como a taxa de envio de pacotes se comportam:

  - **Modo Padrão:** Apenas os parâmetros gerais determinarão a taxa de injeção dos pacotes e cada injector atacará um alvo com a quantidade de pacotes determinadas pelo nível do ataque. A Figura 1b representa o funcionamento.
  - **Modo Incremental:** O nível do ataque aumenta gradativamente. O valor inicial é passado pelo parâmetro de nível, o valor de i deve ser um número inteiro. A cada *i* iterações, o nível do ataque é incrementado em 1, seu limite é alcançado no nível 10.
  - **Modo Raid:** o nível do ataque é ignorado e o DamBuster tenta enviar o máximo de pacotes possível em cada iteração. A quantidade de pacotes enviada dependerá da capacidade de hardware e rede do usuário.
  - **Modo Personalizado:** o usuário pode especificar um arquivo contendo a quantidade de pacotes que serão enviados por cada injector em cada iteração. Ao chegar no final do arquivo, o DamBuster voltará para o início, gerando assim um comportamento cíclico. Com isso, o modo personalizado pode ser utilizado para criar ataques de taxa flutuante.

- Técnicas de Ataque: A ferramenta faz uso da técnica de IP Spoofing, permitindo que o usuário selecione o endereço de origem. Este endereço pode ser fixo ou aleatório. Além disso, também é possível informar múltiplos endereços, na forma de uma lista de endereços IP ou usando a notação CIDR. Esses endereços serão usados de forma aleatória ou na ordem informada, de acordo com a preferência do usuário. Os endereços informados podem ser dos protocolos IPv4 e IPv6. Além disso, o usuário pode informar um ou múltiplos endereços e também aplicar a tática de **Carpet Bombing** [Cimpanu 2019] ao informar uma lista de endereços ou um bloco CIDR. Além disso, por manipulação na linha de comando, é possível realizar **Pulse wave** [DDoS-Guard 2018] e *wave shapping*.
- Diversidade de Ataques - Bombs: Os ataques no DamBuster são denominados *bombs* e estão contidos no módulo **Arsenal**. As *bombs* são implementadas de forma independente, utilizando apenas as funções do módulo auxiliar **Common**. A versão desenvolvida neste trabalho tem suporte aos ataques: **SYN Flood**, **SYN/ACK Flood**, **RST Flood**, **UDP Flood**, e duas variações do **ICMP Flood: Echo Request** e **Echo Reply**.
- Novos Ataques: Uma das características principais da arquitetura utilizada é a modularização na implementação e inserção de novos ataques. Para a criação de novas *bombs* devem ser implementadas três funções:

  1. **Execute <Bomb>:** Responsável pela execução do ataque desejado. Essa função recebe o draft como parâmetro e invoca a função de criação do pacote padrão do ataque. Após a criação do pacote, esta função inicia o módulo Injector.
  2. **CreateAttackData:** Responsável pela criação do pacote padrão do ataque. Essa função invoca a função de criação do *payload* de dados e encapsula com os cabeçalhos dos protocolos utilizados, gerando um *packet_wrapper*.

O módulo **Common** possui diversas funções para a criação e encapsulamento de pacotes em seu submódulo *blacksmith*.

3. **Forge <Bomb>:** Responsável pela criação do payload de dados do pacote. Esta função é invocada durante a criação dos pacotes e pode utilizar argumentos passados pelo usuário para o preenchimento dos dados.

Após a implementação do ataque, é necessário integrá-lo ao programa para que possa ser executado pelo usuário. Para adicionar a bomb, esta deve ser adicionada às opções presentes no draft. Em seguida, é necessário alterar o módulo Commander para que a função de execução do ataques seja associada à opção adicionada ao draft. Deve-se então adicioná-la às opções aceitas pelo módulo Interface, associando o parâmetro de entrada com a opção inserida no draft. Dessa forma, o ataque estará disponível na ferramenta.

3. Testes durante desenvolvimento

Três cenários de testes foram projetados para avaliar as competências da ferramenta proposta em em diferentes configurações de hardware e comparada a uma ferramenta referência, no caso T50 [Lamberti 2001], esta em modo *flooding*, uma vez que esta não provê ataques incrementais. As especificações de todos os dispositivos utilizados estão na Tabela 3.

	Atacante	**Vítima**
Cenários 1 e 3		
Sistema	Manjaro 21.2.5 64-bit	Windows 11 Pro
Processador	Intel Core i5-5200U @ 2,20GHz	Intel Core i5-8400 @ 2,80GHz
Memória	8GB DDR3 @ 1600MHz	16GB DDR4 @ 2666Mhz
Placa de Rede	TP-Link UE300 USB Gigabit	Intel Gigabit Ethernet
Cenário 2		
Sistema	ClearLinux OS 36010	Manjaro 21.2.5 64-bit
Processador	Intel Core i5-8400 @ 2,80GHz	Intel Core i5-5200U @ 2,20GHz
Memória	16GB DDR4 @ 2666Mhz	8GB DDR3 @ 1600MHz
Placa de Rede	Intel Gigabit Ethernet	TP-Link UE300 USB Gigabit

Tabela 1. Especificações dos dispositivos de teste

Nos cenários 1 e 2, ataque durou 100 segundos, aumentando a taxa de envio em degraus de $10x$ a cada $10s$ para cada um dos ataques implementados. Os dados foram coletados usando o software de captura Wireshark, sendo possível identificar quando ocorreu a saturação no envio dos pacotes, i.e., o limite da ferramenta para cada cenário. As especificações dos dispositivos estão na Tabela 3. O Cenário 3 avaliou o DamBuster contra a ferramenta T50, um injetor de pacotes de código aberto utilizado como referência em ataques de *flooding* em testes de estresse de infraestruturas de rede, implementando vários protocolos (ICMP, IGMP v1 e v3, TCP, UDP, EGP, RIP v1 e v2, DCCP, RSVP, GRE, IPSec, EIGRP e OSPF) e com alta performance, chegando a enviar mais de 1.000.000 de pacotes por segundo durante um ataque do tipo SYN Flood [Lamberti 2001]. Segundo os autores, seu diferencial está em enviar todos os protocolos, sequencialmente, utilizando um único SOCKET.

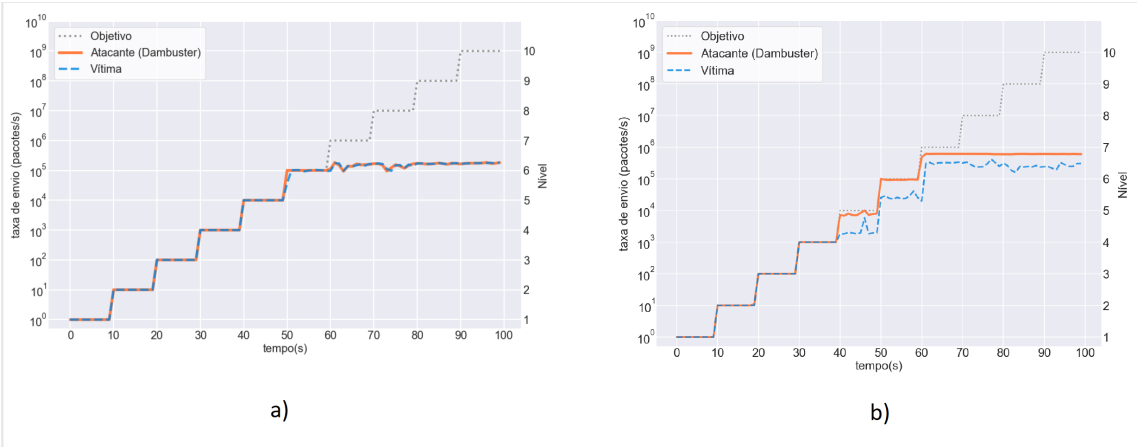

Figura 2. - SYN/Flooding: a) Cenário 1; b) cenário 2

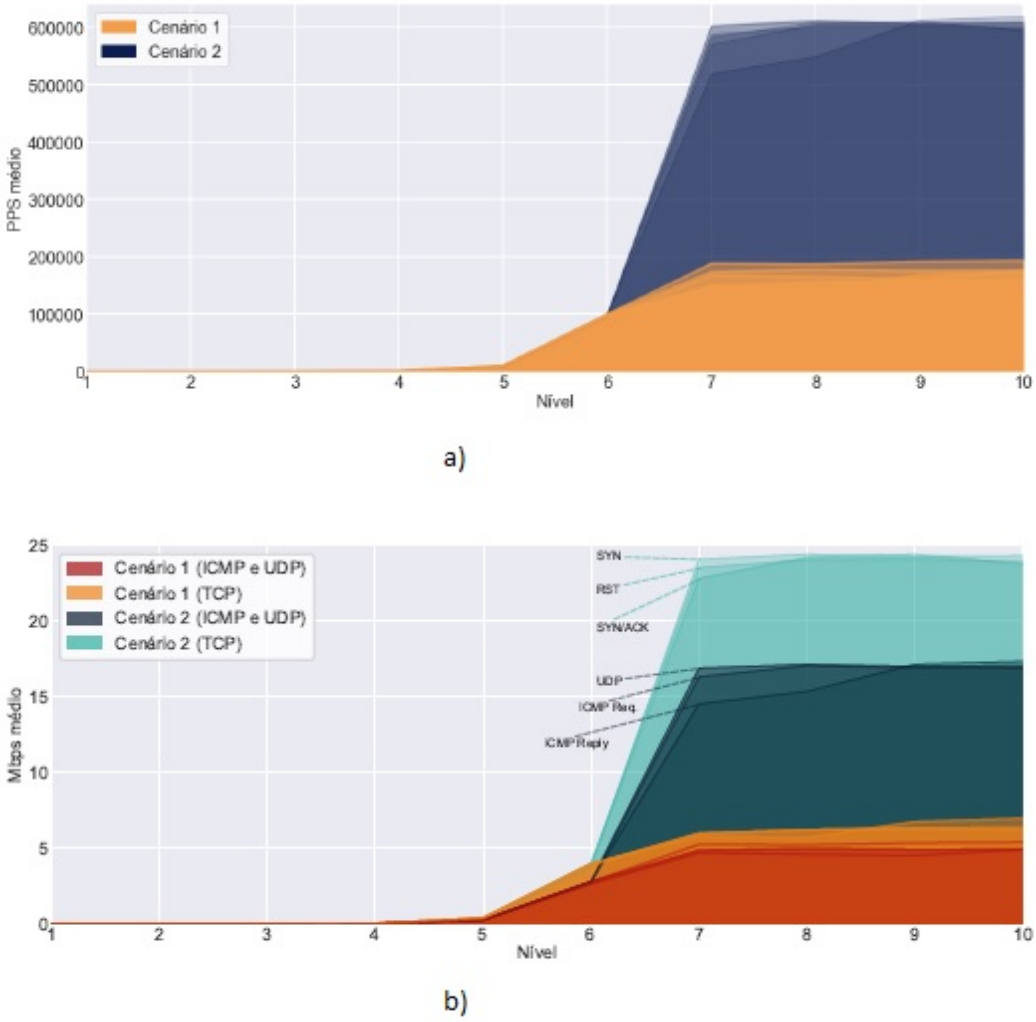

Figura 3. Cenários 1 e 2 - comparativos globais e por ataque

3.1. Avaliação dos Cenários

A Figura 2 ilustra o comportamento do DamBuster para o ataque SYNFlooding no tempo. No cenário 1 a saturação ocorreu em $\approx 10^5$ PPS enquanto no cenário 2 ocorreu em $\approx 10^6$

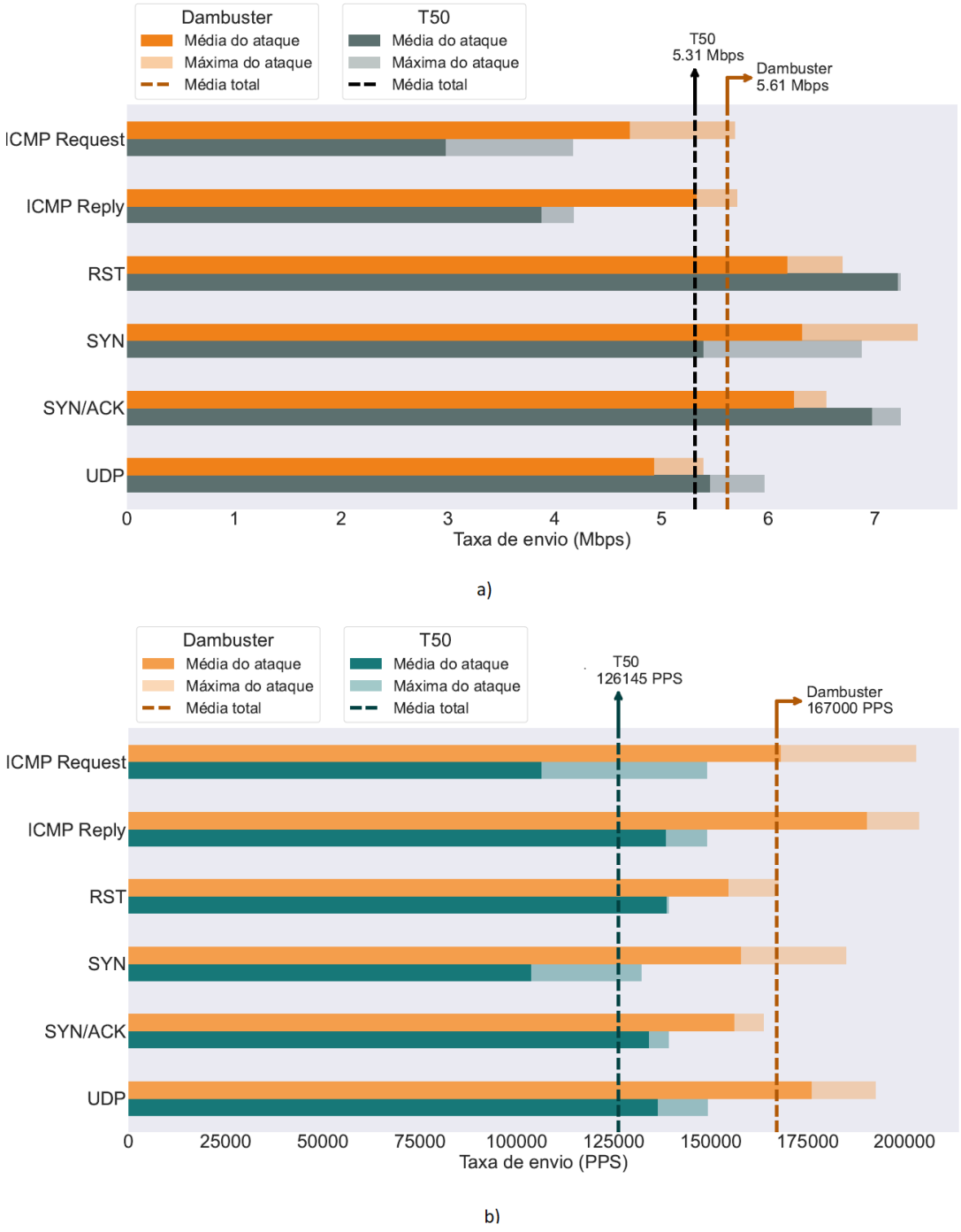

Figura 4. Cenário 3 - DamBuster vs t50: a) Mbps; b) PPS

PPS. Esse comportamento se repete nos outros ataques, que foram testados de forma idêntica. A Figura 3 (onde para o nível N a taxa é $T = 10^{N-1}$) mostra como o DamBuster utiliza os recursos, tirando vantagem do hardware mais potente, tanto globalmente quanto por protocolos utilizados nos ataques, corroborando a Figura 2, porém sem atingir a capacidade da interface de rede. Na Figura 4, temos o comparativo dos dados coletados no cenário 3. O teste foi realizado no Modo Flooding, com desempenho médio do DamBuster (5,61 Mbps e 167 kPPS) superior ao do T50 (5,31 Mbps e 126,15 kPPS),

especialmente em PPS para todos os ataques.

4. Demonstração, Códigos e Manuais

A demonstração apresentará as funcionalidades da ferramenta. Serão apresentadas também as técnicas de *carpet bombing* e *pulse wave* [DDoS-Guard 2018]. O código-fonte e a versão dockerizda estão em **https://eduardoforca.gitlab.io/dambuster/index.html**.

5. Conclusão

Neste trabalho, foram apresentados os aspectos do desenvolvimento e da avaliação do DamBuster. A sua estrutura extremamente modular permitiu a adaptação e o reuso de funções e implementações, diminuindo esforços de refatoração e otimização de código. Os testes efetuados e seus resultados evidenciaram que o DamBuster é uma ferramenta que pode ser utilizada para simular ataques DoS e avaliar sistemas de mitigação. A comparação do DamBuster com o T50 indicou sua eficiência, com resultados superiores nas métricas propostas, alcançando em torno de 40.000 PPS de diferença, e produzindo em média 132% comparado ao T50. Possíveis trabalhos futuros seriam incorporar técnicas otimizadas de implementação do T50 ao DamBuster.

Agradecimento: este trabalho recebeu apoio do projeto PROFISSA (Programmable Future Internet for Secure Software Architectures), bolsa FAPESP 2020/05152-7.

Referências

Case, D. J. D., McCloghrie, K., Rose, D. M. T., and Waldbusser, S. (1996). Introduction to Community-based SNMPv2. RFC 1901.

Cimpanu, C. (2019). 'carpet-bombing' ddos attack takes down south african isp for an entire day.

Dantas, A. L., de Oliveira Vieira, M., Vasques, A. T., and Gondim, J. J. C. (2020). Linderhof: uma ferramenta para avaliação de sistemas de mitigação de ataques reflexivos volumétricos (ddos). In *Anais Estendidos do XXXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 25–32. SBC.

DDoS-Guard (2018). Pulse Wave DDoS attacks — ddos-guard.net. https://ddos-guard.net/en/info/blog-detail/hidden-threat-of-pulse-wave-ddos-attacks. [Accessed 27-08-2023].

Dildy, D. C. (2012). *Dambusters: Operation Chastise 1943*. Bloomsbury Publishing.

Fedor, M., Schoffstall, M. L., Davin, J. R., and Case, D. J. D. (1990). Simple Network Management Protocol (SNMP). RFC 1157.

Fuller, V., Li, T., Yu, J. J. Y., and Varadhan, K. (1993). Classless inter-domain routing (cidr): an address assignment and aggregation strategy. RFC 1519, RFC Editor.

Lamberti, F. (2001). t50 · GitLab — gitlab.com. https://gitlab.com/fredericopissarra/t50/. [Accessed 24-07-2023].

Vieira, M. d. O., Dantas, A. L., Vasques, A. T., and Gondim, J. J. (2021). Linderhof v2. 0.0. In *Anais Estendidos do XXI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pages 9–17. SBC.

Zeilenga, K. (2003). Connection-less Lightweight Directory Access Protocol (CLDAP) to Historic Status. RFC 3352.