Ferramenta de Avaliação de Ataques de Negação de Serviço em uma Plataforma de Testes

Andrés Felipe Murillo Piedrahita

PEE / COPPE / UFRJ

Dissertação submetida para a obtenção do título de
Mestre em Engenheira Elétrica
ao Programa de Engenharia Elétrica/COPPE/UFRJ

FERRAMENTA DE AVALIAÇÃO DE ATAQUES DE NEGAÇÃO DE SERVIÇO EM UMA PLATAFORMA DE TESTES

Andrés Felipe Murillo Piedrahita

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Otto Carlos Muniz Bandeira Duarte

Rio de Janeiro
Maio de 2014

FERRAMENTA DE AVALIAÇÃO DE ATAQUES DE NEGAÇÃO DE SERVIÇO EM UMA PLATAFORMA DE TESTES

Andrés Felipe Murillo Piedrahita

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. Otto Carlos Muniz Bandeira Duarte, Dr. Ing.

Prof. Miguel Elias Mitre Campista, D.Sc.

Prof. Marcelo Gonçalves Rubinstein, D.Sc.

Prof. Igor Monteiro Moraes, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
MAIO DE 2014

Piedrahita, Andrés Felipe Murillo

Ferramenta de Avaliação de Ataques de Negação de Serviço em uma Plataforma de Testes/Andrés Felipe Murillo Piedrahita. – Rio de Janeiro: UFRJ/COPPE, 2014.

XVI, 81 p.: il.; 29, 7cm.

Orientador: Otto Carlos Muniz Bandeira Duarte

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2014.

Referências Bibliográficas: p. 75 – 81.

1. Segurança em Redes Informáticas. 2. Ataques de Negação de Serviço. 3. Plataformas de Testes para Redes de Computadores. 4. Mecanismos de Prevenção de Ataques de Negação de Serviço. I. Muniz Bandeira Duarte, Otto Carlos. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

À minha família, amigos e professores.

Agradecimentos

Agradeço aos meus pais, irmãos, avó e tios por todo o amor, a dedicação, apoio e compreensão durante a minha vida. Agradeço em especial a minha mãe Dignacela Piedrahita, que me ensinou nunca desistir dos meus objetivos e acreditar em mim. Agradeço a minha namorada que tem estado ao meu lado neste processo, por todo seu carinho e amor.

Agradeço aos meus amigos do laboratório: Martin Andreoni, Lyno Ferraz, Diogo Menezes, Govinda Mohini e Igor Drummond pelo apoio, ensinamentos, participação e disponibilidade durante o desenvolvimento desta dissertação de mestrado e pelos bons momentos que vivenciamos no Grupo de Teleinformática e Automação.

Agradeço, a todos os professores que participaram da minha formação. Em especial, ao meu orientador, professor Otto Carlos Duarte, por levar sua função de orientação além do básico, por todos os conselhos, dedicação e principalmente paciência, durante a orientação no mestrado. Gostaria de agradecer também aos professores Luís Henrique e Aloysio do GTA.

Agradeço aos professores Miguel Elias Mitre Campista, Marcelo Gonçalves Rubinstein e Igor Monteiro Moraes pela participação na banca examinadora.

Agradeço a todos os que de alguma forma contribuíram na minha formação e em todo o aprendizado ocorrido durante o mestrado. Por fim, agradeço a CNPq e CAPES pelo financiamento deste trabalho.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

FERRAMENTA DE AVALIAÇÃO DE ATAQUES DE NEGAÇÃO DE SERVIÇO EM UMA PLATAFORMA DE TESTES

Andrés Felipe Murillo Piedrahita

Maio/2014

Orientador: Otto Carlos Muniz Bandeira Duarte

Programa: Engenharia Elétrica

Os Ataques de Negação de Serviço (*Denial of Service Attacks* - DoS) são uma séria ameaça para a operação da Internet e nos últimos anos, o tráfego de pacotes gerado pelos ataques de DoS tem se multiplicado. O desenvolvimento tradicional de mecanismos de defesa contra esses ataques seguem uma abordagem reativa, ou seja, depois que os ataques de negação de serviço causam enormes prejuízos procura-se desenvolver novos mecanismos de defesa. Essa abordagem não é aceitável com o atual processo de integração da Internet a infraestruturas críticas. Portanto, é necessário o desenvolvimento de ferramentas para estudar e avaliar os ataques de negação de serviço e, como consequência, permitir a concepção de sistemas seguros que previnam a negação de serviços, mudando o cenário atual para uma abordagem proativa. Esta dissertação de mestrado propõe uma ferramenta de avaliação de ataques de negação de serviço usando a Plataforma FITS. A integração de ferramentas de ataques na plataforma FITS e a disponibilização de mecanismos e facilidades oferece um ambiente de testes de novas propostas de mecanismos de contramedidas a ataques de negação de serviço. As facilidades oferecidas utilizam a infraestrutura MAGI, do *testbed* DETER para o controle do experimento. Foram criados módulos para automatizar o processo de criação das redes virtuais, assim como a instalação e a configuração da MAGI. Por fim, as facilidades providas foram integradas na interface web do FITS para facilitar a criação e configuração dos experimentos. Para avaliar a funcionalidade da facilidade de testes oferecida foi elaborado o teste de um mecanismo para prevenção de ataques de negação de serviço, chamado FlowFence.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

TOOL FOR EVALUATION OF DENIAL OF SERVICE ATTACKS IN A TESTBED

Andrés Felipe Murillo Piedrahita

May/2014

Advisor: Otto Carlos Muniz Bandeira Duarte

Department: Electrical Engineering

Denial of Service Attacks (DoS) are a serious threat for the Internet and in the last years traffic generated from these attacks has multiplied. Traditional development of defense mechanisms against these attacks follows a reactive approach; in this approach, mechanisms are developed and deployed after the attacks have caused damage and financial losses. A reactive approach is not acceptable with the current integration process of Internet to critical infrastructure. Therefore, it is necessary to develop tools to study and evaluate DoS attacks and change to a proactive approach were prevention and defense mechanisms are deployed ahead of potential attacks. This dissertation proposes mechanisms and services for experimentation of Denial of Service Attacks using the FITS platform. The integration of these mechanisms in FITS platform offers a service to test new defense mechanisms proposals against DoS attacks. The proposed features use the MAGI infrastructure, developed by DETER, to control the experiments. Modules were built to automate the experimental virtual network creation and configuration of MAGI. Using this mechanisms, the prototype of a Dos prevention mechanism, called FlowFence, was developed.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

API	*Application Progamming Interface* - Interface de Programação de Aplicativos, p. 39
CCN	*Content Centric Networks* - Redes Orientadas a Conteúdo, p. 38
DETER	*Defense Technology for Experimental Research* - Tecnologia de Defesa para Pesquisa Experimental, p. 34
DHS	*Department of Homeland Security* - Departamento de Segurança Doméstica, p. 34
EPIC	*Experimentation Platform for Internet Contingencies* - Plataforma de Experimentação para Contingencias da Internet, p. 4
FITS	*Future Internet Testbed with Security* - Plataforma de Testes da Internet do Futuro com Segurança, p. 38
GRE	*Generic Routing Encapsulation* - Encapsulamento Genérico de Roteamento, p. 39
HTTP	*Hypertext Transfer Protocol* - Protocolo de Transferência de Hipertexto, p. 31
IDS	*Intrusion Detection System* - Sistema de Detecção de Intrusos, p. 26
ISEAGE	*Internet-Scale Event and Attack Generation Environment* - Entorno de Geração de Eventos e Ataques a Escala da Internet, p. 4
LOIC	*Low Orbital Ion Cannon* - Canhão de Íons de Baixa Órbita, p. 32

MAGI *Montage AGent Infrastructure* - Infraestrutura de Montagem de Agentes, p. 36

NSF *National Science Foundation* - Fundação Nacional de Ciência, p. 34

RINSE *Real Time Immersive Network Simulation Environment* - Entorno de Simulação de Redes Imersivo e de Tempo Real, p. 3

VLAN *Virtual Local Area Network* - Rede de Área Local Privada, p. 36

VPN *Virtual Private Network* - Rede Virtual Privada, p. 39

YAML *Yet Another Markup Language* - Ainda Outra Linguagem de Etiquetas, p. 37

Capítulo 1

Introdução

Um ataque de Negação de Serviço (*Denial of Service* - DoS) é um ataque que degrada ou interrompe o uso de um serviço oferecido pela vítima do ataque. A interrupção ou degradação do serviço é obtida através do consumo dos recursos disponibilizados pela vítima para atender as requisições dos usuários legítimos. O ataque pode ser realizado enviando pacotes deformados que provoquem o bloqueio de um protocolo ou de uma aplicação, ou ainda enviando um grande número de pacotes, uma inundação de pacotes, que requeira um consumo excessivo dos recursos da vítima [6]. Os ataques de negação de serviço são uma séria ameaça para a operação da Internet e nos últimos anos, o tráfego de pacotes gerado pelos ataques de DoS tem se multiplicado. Em 2006, foi reportado o primeiro ataque com 10Gb/s de tráfego [7], em 2010 foi reportado um ataque de 100Gb/s [1] e em 2013 um de 300Gb/s [8].

Os ataques de negação de serviço por inundação estão cada vez mais comuns e é cada vez mais difícil evitá-los. Como já afirmado, o objetivo da inundação é gerar uma quantidade de requisições maior que a capacidade da vítima de atendê-las, o que termina consumindo os recursos disponíveis da vítima para responder requisições legítimas. A dificuldade de evitar, bloquear ou filtrar os ataques de negação de serviço está em identificar o ataque uma vez que os pacotes de ataques usados para inundar são pacotes "válidos" que não diferem de um pacote benigno enviado por um usuário legítimo. A tentativa de defesa pela identificação da característica do tráfego de inundação gerado também tem sido contornada, pois os atacantes passaram a usar técnicas para evitar que as fontes do ataque sejam rastreadas, como a falsificação do endereço de origem do pacote. Os ataques de negação de serviço têm se tornado populares porque as características atuais da Internet facilitam a realização de esse tipo de ataques de forma anônima, enquanto propor defesas e mecanismos de prevenção é um processo complexo. Com a integração da Internet no fornecimento de múltiplos serviços de compras, redes sociais, notícias e ainda infraestruturas críticas, esse tipo de ataques representa uma ameaça que deve ser

estudada e prevenida na Internet.

De forma similar a outros tipos de ataques nas redes, os atacantes e as motivações dos ataques de negação de serviço têm evoluído. Os primeiros ataques de DoS eram feitos por indivíduos e usados como piadas ou para mostrar as fragilidades de alguns sistemas. Mas recentemente os DoS são usados como meio de manifestações políticas [9], de ações econômicas [10] ou da chamada ciberguerra [11]. O anonimato e a facilidade de se construir um ataque que resulte no sucesso da negação do serviço da vítima tem aumentado a motivação para realizar os ataques. O lucro financeiro tem motivado o crescimento dos ataques e também tem permitido a criação de grupos organizados e estruturas malignas que vendem seus "serviços." Este cenário tem permitido uma evolução nas ferramentas e tipos de ataques de negação de serviço. As primeiras ferramentas existentes só conseguiam realizar alguns tipos de ataques simples, como a inundação com pacotes UDP (*User Datagram Protocol*) e sem usar falsificação de endereços IP (*Internet Protocol*). As ferramentas têm evoluído para aumentar a capacidade do ataque e estão cada vez mais automatizadas, ocasionando impactos cada vez maiores na Internet. Atualmente, as ferramentas para realizar DoS permitem realizar vários tipos de ataques, usam falsificação de diversos campos dos pacotes enviados e usam comunicações encriptadas entre os agentes e o coordenador do ataque para dificultar a identificação e localização da fonte do ataque. Nos últimos anos, têm sido criados ataques que visam diminuir a quantidade de pacotes necessários para a negação de serviço. Alguns deles aproveitam os protocolos de rede como, por exemplo, o ataque de baixa taxa TCP, que aproveita os mecanismos de controle de congestionamento do TCP para enviar tráfego em rajadas, dificultando sua detecção. O tráfego em rajadas segue um padrão de uma onda quadrada, o pico da onda induz TCP *timeouts* na vítima o que fazm com que os emissores diminuam suas taxas de transmissão. Outros ataques são especialmente criados para atacar camadas e aplicações específicas. O ataque `Slowris` envia requisições HTTP regularmente em intervalos de vários segundos [6], mantendo ocupados os *sockets* abertos no servidor web. Depois de um tempo do ataque `Slowris`, todos os *sockets* do servidor ficam ocupados, o que impede que ele atenda as requisições de usuários legítimos.

O desenvolvimento tradicional de mecanismos de defesa contra esses ataques segue uma abordagem reativa, ou seja, depois que os ataques de negação de serviço causam enormes prejuízos procura-se desenvolver novos mecanismos de defesa. Em muitas ocasiões, esses danos são muito elevados e enquanto são desenvolvidos mecanismos de defesa, os sistemas ainda permanecem vulneráveis. Um exemplo desse tipo de situação são os ataques de reflexão de DNS (*Domain Name Server*). Esse tipo de ataque tem gerado a maior quantidade de tráfego de um DoS nos anos 2010 e 2013. Adicionalmente, o processo atual de integração da Internet com aplicações

e infraestruturas críticas faz com que a abordagem reativa represente um risco de segurança enorme. Ataques cibernéticos podem afetar perigosamente as infraestruturas críticas criando *blackout* ou provocando catástrofes. Portanto, é necessário o desenvolvimento de ferramentas para se estudar e avaliar os ataques de negação de serviço e, com consequência, permitir a concepção de sistemas seguros que previnam a negação de serviços, mudando o cenário atual para uma abordagem proativa. O uso de simulações para testar ataques em topologias de rede e protocolos é importante, porém, em muitas ocasiões não permite representar realisticamente o comportamento das ferramentas de DoS e as interações, assim como o impacto em implementações específicas de sistemas operacionais.

As plataformas de testes (*testbeds*) oferecem uma infraestrutura que representa realisticamente os protocolos e implementações analisadas em experimentos de ataques de negação de serviço. A vantagem dos *testbeds* é que representam o comportamento de protocolos e sistemas operacionais reais. Nas máquinas do *testbed* são instaladas versões reais de diversos aplicativos e ferramentas de ataque e as defesas podem ser testadas diretamente em um ambiente real, o que permite uma melhor avaliação do comportamento de ferramentas de DoS e dos mecanismos de defesa que precisam ser desenvolvidos. A plataforma de teste deve apresentar condições muito similares às condições da atual Internet. Algumas propostas de plataformas de testes para ataques de negação de serviço têm sido criadas e usadas para realizar experimentos.

O *testbed* de Ilinóis [12] usa o simulador de redes *Real Time Immersive Network Simulation Environment* (RINSE) para interconectar os equipamentos físicos com simulações de redes de comunicações. RINSE modela redes de comunicação como um conjunto de fluxos com determinada taxa entre grupos de nós, os dispositivos reais se comunicam com a plataforma usando etiquetas de *Virtual Local Area Network* (VLAN) que identificam o fluxo. No entanto, o uso de simuladores de rede depende de modelos acurados do sistema real e, em muitos casos, não existem modelos de tráfego já bem estudados e desenvolvidos. Há um risco das aproximações falharem em representar cenários complexos como, por exemplo, no caso das comunicações em redes elétricas inteligentes.

A universidade de Iowa desenvolveu o *testbed Internet-Scale Event and Attack Generation Environment* (ISEAGE) [13] para emular redes a grande escala com características similares às presentes na Internet. ISEAGE é um simulador de redes que modela um ambiente de nuvem similar à Internet. Uma topologia multi-salto pode ser construída dentro do ISEAGE. Adicionalmente, o ISEAGE pode ser conectado a equipamentos e roteadores externos. Dessa forma, se um computador externo realiza um comando *traceroute* a um servidor conectado ao ISEAGE, a emulação faz como se os pacotes tivessem atravessado múltiplos saltos até chegar ao destino.

Para criar topologias complexas, o ISEAGE pode ser instalado em diversos servidores, cada um deles representando um sistema autônomo (*Autonomous System* - AS). Adicionalmente, o ISEAGE possui um repositório de ferramentas para criar ataques de DoS, gerar tráfego e criar topologias.

O *Experimentation Platform for Internet Contingencies* (EPIC) [14] é um *testbed* desenvolvido para realizar experimentos de segurança física e cibernética. O EPIC possui dois componentes fundamentais: um componente para representar as redes de comunicações e um componente para representar sistemas físicos como processos industriais. As redes são emuladas usando as ferramentas do *emulab* [15]. Usando *emulab*, um conjunto de máquinas físicas são reservadas para o experimento e são conectadas usando uma rede comutada de alta velocidade. A topologia do experimento é criada usando etiquetas de VLAN. O *Emulab* também oferece ferramentas para instalar diversos sistemas operacionais nas máquinas, assim como software específico. O segundo componente do *testbed* é a representação de sistemas físicos como: redes elétricas e processos industriais controlados em rede. O segundo componente foi criado usando simulações em `Simulink`. Os modelos de `Simulink` são instalados em algumas máquinas usadas pelo *emulab* para os experimentos. EPIC combina elementos de simulação e emulação para criar um *testbed* para experimentos cibernéticos e físicos de segurança, mas não possui repositórios para criar ataques de negação de serviço nem ferramentas para facilitar o controle do experimento.

O *Cyber DEfense Technology for Experimental Research* (DETER) é um *testbed* desenvolvido pelas universidades de UC Berkeley e a universidade de *South California* e patrocinado pela *National Science Foundation and the Department of Homeland Security* dos Estados Unidos. O objetivo do DETER é prover uma infraestrutura robusta, que possa ser utilizada por pesquisadores e desenvolvedores de forma concorrente para testar mecanismos de segurança em ambientes reais e com topologias de média escala. No DETER, os experimentadores recebem acesso exclusivo a um conjunto de máquinas físicas para realizar seus experimentos. As máquinas físicas são ligadas segundo a topologia especificada pelo experimentador usando uma rede comutada de alta velocidade e etiquetas de VLAN. O DETER possui um conjunto de ferramentas para criar ataques de negação de serviço e controlar o experimento.

O *Future Internet Testbed with Security* (FITS)[1] é uma plataforma de teste de código aberto baseado em OpenFlow e no hipervisor Xen [16]. O FITS é uma iniciativa do Grupo de Teleinformática e Automação (GTA) da UFRJ. No FITS um conjunto de máquinas físicas é utilizado para instalar o hipervisor Xen e permitir aos usuários criarem máquinas virtuais para realizar seus experimentos. Cada instituição parti-

[1]FITS é uma rede de testes interuniversitária desenvolvida a partir da parceria de instituições brasileiras e européias. Maius informações em http://www.gta.ufrj.br/fits.

cipa com máquinas físicas nas quais são criadas e instaladas as máquinas virtuais dos experimentos. A plataforma de testes divide a rede física em redes virtuais, cada uma com sua própria topologia virtual, pilha de protocolos, regras de encaminhamento e administração. Nesse sentido, o FITS permite a criação de redes virtuais isoladas, compartilhando a mesma infraestrutura [2]. O controle de acesso para o gerenciamento e a criação de redes virtuais usa uma plataforma segura, baseada em OpenID e micro controladores seguros [17]. FITS também oferece diferenciação de qualidade de serviço, migração de máquinas virtuais [18] e controle automático de recursos dentro das máquinas físicas [19]. Na plataforma FITS têm sido avaliadas propostas para a Internet do Futuro como: Redes Orientadas a Conteúdo (*Content Centric Networks* - CCN) para distribuição de vídeo sob demanda [20], propostas de controle de acesso para redes definidas por *software* [21] e arquiteturas para prevenção de ataques de DoS usando redes definidas por *software* [22].

1.1 Objetivos da Dissertação

O objetivo desta dissertação é desenvolver mecanismos e serviços para experimentação de ataques de negação de serviço usando a Plataforma FITS. A integração desses mecanismos e facilidades na plataforma FITS oferece um serviço de testes de novas propostas de mecanismos de contramedidas a ataques de negação de serviço. As facilidades oferecidas utilizam a infraestrutura MAGI, do *testbed* DETER, para o controle do experimento. Foram criados módulos para automatizar o processo de criação das redes virtuais, assim como a instalação e a configuração da MAGI. Por fim, as facilidades providas foram integradas na interface web do FITS para facilitar a criação e configuração dos experimentos. Para avaliar a funcionalidade da facilidade de testes oferecida foi elaborado o teste de um mecanismo para prevenção de ataques de negação de serviço, chamado FlowFence.

FlowFence foi testado na plataforma FITS, usando os serviços para experimentação de ataques de negação de serviço da presente tese. O FlowFence permite prevenir ataques de negação de serviço em uma arquitetura onde os roteadores da rede monitoram o estado de suas interfaces de saída, e se comunicam com um controlador da rede. Em caso de congestionamento, o controlador envia ordens aos roteadores da rede para aplicar controle de banda aos fluxos que usam enlaces congestionados. Os resultados mostram que na ausência de um sistema de detecção de intrusão, FlowFence garante o uso equitativo aos usuários durante um DoS, evitando a inanição.

1.2 Organização da Dissertação

Este trabalho está organizado da seguinte forma. O Capítulo 2 apresenta as características e tipos de ataques de negação de serviço. São analisadas as motivações para realizar esses ataques e uma taxonomia dos ataques é apresentada. O Capítulo 3 apresenta as principais ferramentas existentes para realizar ataques de negação de serviço e as plataformas de testes DETER e FITS. Também é analisada a arquitetura das plataformas de testes e suas características. O Capítulo 4 apresenta os mecanismos e serviços desenvolvidos para a experimentação de ataques de negação de serviço. São apresentados os módulos principais do sistema e a arquitetura da plataforma de testes. Também é avaliado o desempenho dos componentes criados dentro da plataforma FITS. No final do capítulo é apresentado o mecanismo de teste FlowFence e a avaliação de seu funcionamento. Por fim, o Capítulo 5 apresenta as conclusões e direções futuras do trabalho.

Capítulo 2

Ataques de Negação de Serviço

Um ataque de Negação de Serviço (*Denial of Service - DoS*) é um ataque que degrada ou interrompe um serviço oferecido pela vítima do ataque. A interrupção ou degradação do serviço é obtida através do consumo dos recursos disponibilizados pela vítima para atender as requisições dos usuários legítimos. O ataque pode ser realizado de duas formas: i) enviando pacotes deformados que provoquem o bloqueio de um protocolo ou de uma aplicação ou ii) enviando pacotes que requeiram um consumo excessivo dos recursos da vítima [6]. Neste último caso, os pacotes enviados podem ser requisições que exijam processamento ou consumam memória em um servidor, ou ainda uma inundação de pacotes que consumam os recursos de banda passante que, desta forma, dificultam ou até inviabilizam o acesso dos usuários legítimos aos recursos para usufruir do serviço. Quando o ataque provem de múltiplas máquinas distribuídas geograficamente, ele é chamado ataque Distribuído de Negação de Serviço (*Distributed Denial of Service* - DDoS) [23].

Os ataques de negação de serviço são uma das ameaças mais sérias para a operação da Internet. A Figura 2.1 apresenta as principais ameaças identificadas por as empresas participantes do Informe Mundial de Segurança de Infraestrutura [1]. Os ataques de negação de serviço foram reportados por 76% das empresas como uma das principais ameaças. Os erros de configuração foram reportados por 60% das empresas. Ataques de DoS à infraestrutura e serviços foram reportados por 50%. Nos últimos anos, muitos ataques de negação de serviço têm obtido êxitos e impedido o funcionamento de sítios web importantes como Yahoo, Paypal, Visa, entre outras instituições financeiras e comerciais. Em 9 de fevereiro de 2000, Yahoo, eBay, Amazon.com, E*Trade, ZDNet, Buy.com e outros sítios foram vítimas de ataques de DDoS que causaram grandes prejuízos [24]. Novamente em 2010, um grupo de ativistas chamado *"Anonymous"* coordenou um ataque DDoS, utilizando simples ferramentas web, que derrubou os sítios do Mastercard, Paypal, PostFinance, Visa [9]. Além disso, o tráfego gerado durante os ataques DDoS por inundação tem aumentado e, em muitas ocasiões, têm causado efeitos colaterais afetando a operação

da Internet. Em outubro de 2002, 9 dos 13 servidores DNS raiz foram derrubados por aproximadamente uma hora como consequência de um ataque de DDoS [25]. O tráfego de pacotes gerado pelos ataques de DDoS tem se multiplicado nos últimos anos. Em 2006, foi reportado o primeiro ataque com 10Gb/s de tráfego [7], em 2010 foi reportado um ataque de 100Gb/s [1] e agora em 2013 um de 300Gb/s [8].

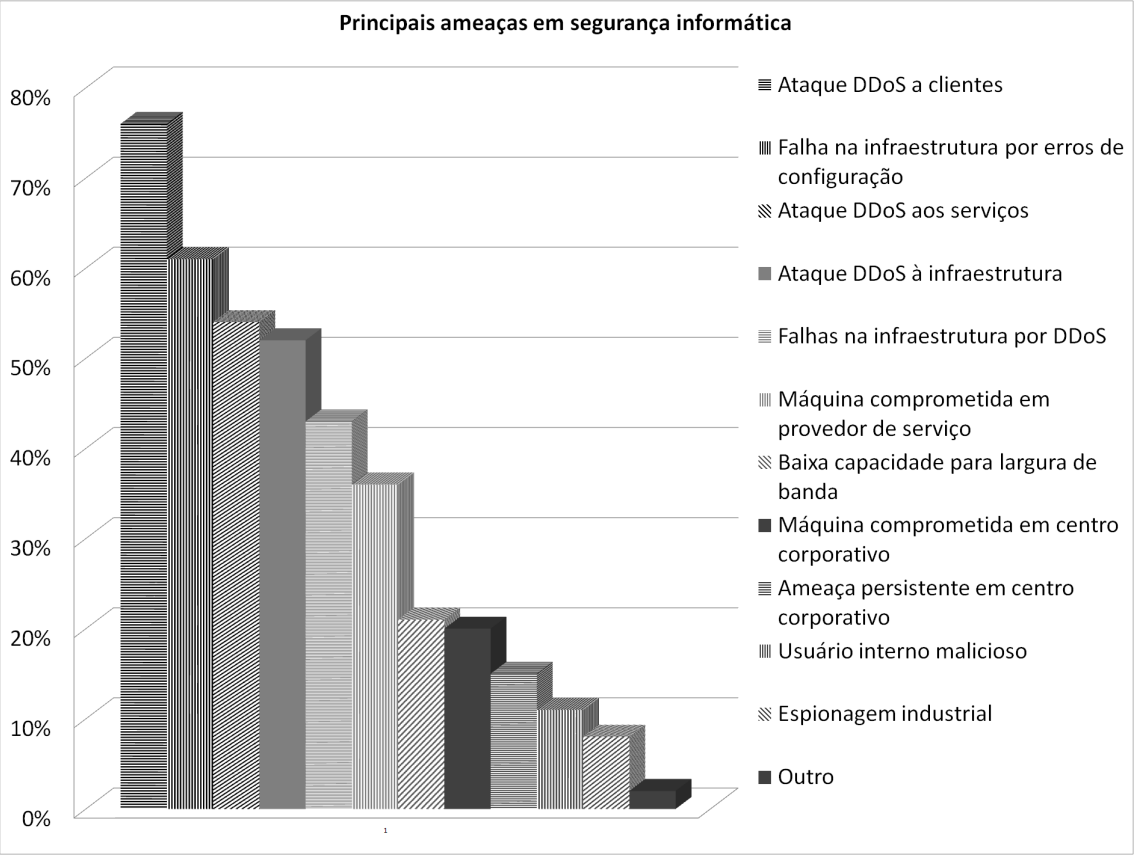

Figura 2.1: Indicação do ataque distribuído de negação de serviço como a principal ameaça operacional segundo as empresas participantes do Informe Mundial de Segurança de Infraestrutura. Fonte [1].

2.1 Princípios Gerais dos Ataques de Negação de Serviço

Os ataques de negação de serviço são facilitados pela ideia original da Internet seguir o paradigma de comunicação fim-a-fim, deixando a inteligência da rede nos extremos da comunicação [26]. Também, a Internet não foi concebida como uma rede segura e, portanto, todas as soluções de segurança são respostas às vulnerabilidades observadas ao analisar os ataques que ocasionaram danos na Internet. Por fim, a implementação das medidas de segurança é um processo lento e parcial, porque com

a atual popularidade da Internet e seus milhares de operadores provendo serviços tornou complexa a tarefa de implantar mudanças que afetem o funcionamento básico da Internet. Por exemplo, os roteadores de acesso não verificavam os endereços IPs de origem dos pacotes encaminhados, o que permitia a um atacante falsificar o endereço IP de origem durante um ataque de DoS para dificultar seu rastreamento. Algumas características da Internet que facilitam os ataques de DoS são [5, 23, 26]:

- Compartilhamento de Recursos: a Internet é uma rede de comutação de pacotes e, portanto, a banda passante de um enlace é compartilhada entre os usuários que usam esse enlace para enviar seus pacotes com a política primeiro a chegar primeiro a ser servido (*First In First Out* – FIFO). Um usuário não pode reservar uma fatia do enlace exclusivamente para seu uso. Assim, um usuário malicioso enviando uma taxa excessiva de pacotes vai prejudicar aos outros usuários;

- Núcleo Simples e Inteligência nas Extremidades: o princípio fim-a-fim da Internet indica que a função dos nós da rede entre duas entidades participantes de uma comunicação é simplesmente encaminhar pacotes o mais rápido possível. Isso significa que o núcleo de Internet são roteadores com uma alta capacidade de encaminhamento, mas que não têm informação sobre os serviços acima da camada de rede. Toda a complexidade necessária para realizar funções como autenticação, tratamento de erros entre outras é instalada nas extremidades. Isso oferece aos atacantes a possibilidade de falsificar os campos de endereço de origem dos pacotes que estão sendo encaminhados. A falsificação do endereço de origem dificulta o rastreamento do ataque, e é a base para os ataques chamados de reflexão e amplificação. Esse tipo de ataque de inundação tem mostrado uma capacidade enorme de gerar tráfego [7, 8]. A possibilidade de se falsificar o endereço de origem do pacote dificulta sobremaneira a implantação de medidas que visam impedir os ataques de negação de serviço, pois não se consegue descartar os pacotes antes que eles cheguem à vítima do ataque;

- Inexistência de Diferenciação de Serviços: o protocolo IP entrega pacotes utilizando o princípio de melhor esforço (*best effort delivery*), o que implica que todos os pacotes são tratados com a mesma prioridade [27]. Isso resulta na alta vulnerabilidade dos serviços sensíveis à latência a ataques de DoS. O melhor esforço torna muito efetivo o ataque por inundação, pois basta que os pacotes de DoS sejam em um número muito maior ou de maior tamanho que os dos pacotes dos usuários legítimos, uma vez que são encaminhados com a mesma prioridade;

- Confiabilidade dos Provedores de Serviço Internet (*Internet Service Provider*

- ISP): a Internet foi concebida como uma rede que interconecta redes, os provedores de serviço Internet e assumindo que todos os ISPs eram entidades confiáveis e com autonomia para. Portanto, a falsificação de campos dos protocolos é facilitada pois, a pilha de protocolos TCP/IP não possui mecanismos nem para assegurar a integridade dos pacotes sendo trocados e nem assegurar a autenticidade das entidades. Os pacotes são encaminhados ao destino sem nenhuma verificação da origem nem do conteúdo;

- Controle Distribuído e Autonomia: como já mencionado, a Internet consiste de um conjunto de Sistemas Autônomos (AS) interconectados entre si, onde cada sistema define suas próprias políticas de segurança. Muitas soluções para problemas de segurança dependem que todos as entidades as implementem. No entanto, as entidades são autônomas e os mecanismos de segurança que são propostos como remendos a problemas existentes não são implementados, em muitas ocasiões, por todos os provedores de serviços. Por exemplo, o efeito da falsificação dos endereços IP pode ser diminuído com os filtros de ingresso [28], mas sua efetividade depende de que os roteadores no caminho implementem o filtro.

2.2 Motivações para Realizar Ataques de Negação de Serviço

As primeiras ferramentas para realizar ataques de negação de serviço foram feitas por grupos de atacantes não profissionais que procuravam reconhecimento. Os ataques não eram lançados de forma organizada e o objetivo era simplesmente mostrar que determinado sistema era vulnerável para esse tipo de ataque. Mas nos últimos anos os ataques de DoS e DDoS usam ferramentas profissionais e as motivações para realizar os ataques têm aumentado. Atualmente, na Internet existem muitos anúncios para contratar grupo de pessoas para realizar um ataque de negação de serviço durante um determinado tempo [29]. A Figura 2.2 apresenta as principais motivações para fazer ataques de negação de serviço segundo o Informe Mundial de Segurança de Infraestrutura. As motivações políticas ocupam o primeiro lugar como motivações percebidas por 33% das empresas participantes do informe. O vandalismo foi percebido pelo 27% das empresas. As demonstrações criminais a clientes potenciais foi percebido por 24%.

As principais motivações dos atacantes para realizar ataques de negação de serviço são:

- Crenças Ideológicas: atualmente é a principal motivação para realizar ataques

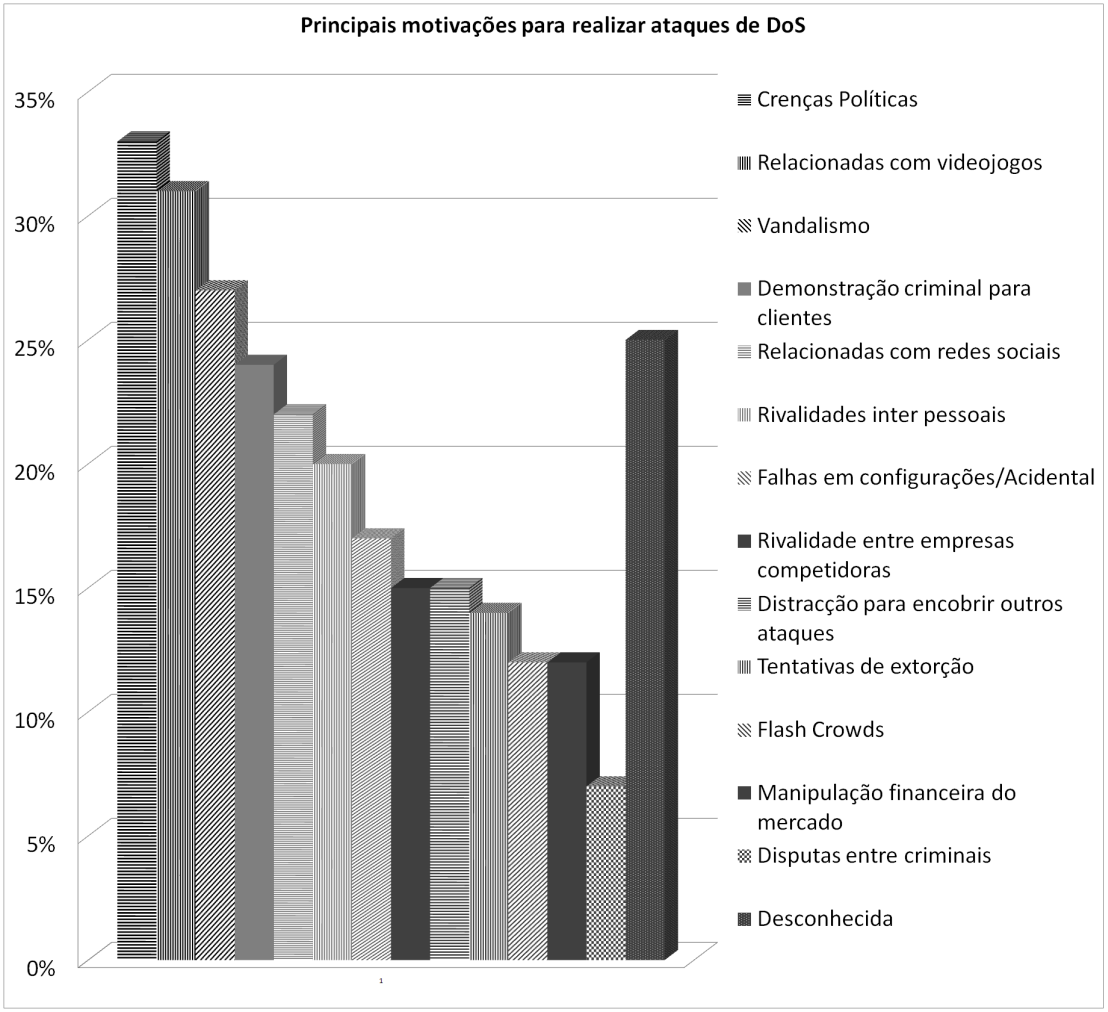

Figura 2.2: Motivações mais comuns dos ataques de negação de serviço. A principal motivação reportada é ideologia ou motivos políticos com um 33% seguida de jogos de vídeo, enquanto para um 25% dos ataques não foi identificada uma motivação específica. Fonte [1].

de negação de serviço [30]. Os ataques têm como alvo serviços e sítios relacionados com discussões políticas e são usados como forma de protesto para atrair a atenção do público geral. Os ataques de 2010 do grupo *Anonymous* foram motivados por discussões ideológicas sobre direitos de autor e Internet [9];

- Lucros Econômicos: empresas contratam ou até mesmo realizam os ataques de negação de serviço contra as empresas concorrentes para obter vantagem em determinadas situações críticas. Este tipo de atividade remunerada incentiva grupos malfeitores a investir em melhorar suas capacidades para obter êxitos nos ataques DoS para aumentar o lucro financeiro;

- Guerra Cibernética: os atacantes que têm este tipo de motivação pertencem a organizações militares ou terroristas e, portanto, são considerados tecnica-

mente habilitados e treinados, além de possuírem acesso a amplos recursos para realizar os ataques. As motivações do ataque podem ser afetar a moral de uma nação inimiga atacando serviços como empresas jornalísticas [31], sítios web de líderes políticos [11], afetar serviços financeiros [10] entre outros.

- Motivos Pessoais: ataques de negação de serviço são lançados por pessoas frustradas ou ex-empregados de companhias. Usualmente estes têm poucas capacidades técnicas.

2.3 Ataques de Negação de Serviço Distribuídos

Um atacante pode realizar um ataque de negação de serviço usando uma máquina ou um conjunto de máquinas. No segundo caso, o ataque chama-se Ataque de Negação de Serviço Distribuído (*Distributed Denial of Service* - DDoS) e seu efeito pode ser avassalador, uma vez que a capacidade do ataque passa a ser proporcional ao número de máquinas utilizadas para lançar o ataque. O primeiro passo é recrutar vítimas secundárias ou *zombies* ou *bots*, depois do recrutamento o atacante passa a ter o controle sobre elas. O recrutamento é feito usando *malware* que infeta a máquina ou explorando vulnerabilidades para abrir *backdoors* e enviar comandos à vítima. Ainda existem ataques não precisam infetar as vítimas, simplesmente requerem que ela acesse um site que execute um *script* malicioso no explorador para inundar a outra vítima [32]. O segundo passo é ordenar um ataque de todo seu exército de *bots*, denominado *botnet*, contra a máquina alvo, que é chamada *vítima primária*. As vítimas secundárias podem estar distribuídas pela Internet, o que dificulta a detecção do atacante porque os ataques provem de endereços IPs de diferentes regiões e podem estar amplamente distribuídas. O procedimento completo de um ataque de DDoS possui as seguintes etapas [26]:

Varredura: nesta etapa um conjunto de máquinas realiza uma varredura de máquinas em diferentes redes para encontrar portas abertas ou serviços que tenham alguma vulnerabilidade para contaminarem as máquinas. Para a varredura são utilizadas ferramentas como `netcat` [1]. Dependendo da sofisticação das ferramentas do atacante, pode-se procurar uma vulnerabilidade fixa ou uma base de dados de vulnerabilidades conhecidas nos serviços encontrados. O processo é muito similar ao comportamento de um verme (*worm*), já que consiste em encontrar e contaminar vítimas em determinado domínio. O processo de varredura pode ser [26]:

- Aleatório: as máquinas aleatoriamente escolhem um endereço IP em determinado espaço de endereços. Esse tipo de varredura gera altas taxas de tráfego

[1] Netcat é uma ferramenta de código aberto, disponível em: http://netcat.sourceforge.net/

entre redes porque é muito provável que os endereços estejam em redes diferentes;

- Lista de Alvos: as máquinas possuem uma lista de endereços IPs que muito provavelmente possuem as vulnerabilidades que a ferramenta pode explorar para infetar outras máquinas. A desvantagem de esse método é que a lista tem que ser criada anteriormente;

- Padrões de Tráfego: utiliza os padrões de tráfego de algumas máquinas para varrer vítimas secundárias. Por exemplo, um servidor web pode ter código malicioso que faça varreduras das máquinas que acessam a determinado conteúdo web, também pode ser embutido um código malicioso em *emails* distribuídos pela Internet;

- Subrede local: as máquinas só realizam varreduras dentro de sua subrede. A vantagem é que gera pouco tráfego entre redes.

Exploração de Vulnerabilidade: a exploração da vulnerabilidade consiste em enviar um comando, ou requisição à vítima secundária ou *bot* que gere uma condição inesperada é que permita a execução de código na vítima. Uma ferramenta popular utilizada para explorar vulnerabilidades é `metasploit` [2];

Propagação: a propagação consiste em instalar o código malicioso para realizar o ataque, abrir algum canal de comunicação para coordenar o ataque e possivelmente o código da ferramenta de varredura para continuar infetando outras vítimas. Existem várias formas de propagação:

- Propagação a partir da Fonte Central: depois de comprometer uma vítima secundária o código malicioso é descarregado de um servidor central. O `li0n` [33] funcionava de esta forma;

- Propagação em Cadeia: o código é descarregado desde a máquina que realizou o processo de varredura. Esta forma tem maiores possibilidades de sobreviver porque não possui um ponto único de falha. A *worm The Ramen [34]* usava este mecanismo;

- Propagação Autônoma: o código malicioso é injetado diretamente durante a fase de exploração. O *worm* `Code Red` [35] assim como muitos outros vermes (*worms*) distribuídos em *emails* utilizam este mecanismo. Este mecanismo reduz a quantidade de tráfego necessário para contaminar outras máquinas e dificulta sua detecção.

[2]Metasploit é uma ferramenta de código aberto, disponível em: http://www.metasploit.com/

2.4 Características dos Ataques de Negação de Serviço

Os ataques de negação de serviço têm evoluído desde ferramentas que realizavam inundações simples para ferramentas complexas que aproveitam as características de diversas aplicações e protocolos para acabar com os recursos das vítimas. Porém, muitos ataques possuem características similares que permitem realizar uma classificação para compreender melhor seu funcionamento. Os dois tipos de ataques de negação de serviço são os ataques por vulnerabilidade e os ataques de inundação. Esta dissertação foca os ataques por inundação e os ataques por vulnerabilidades são mencionados brevemente. As principais características dos ataques de negação de serviço são apresentadas na Figura 2.3. As características são agrupadas em categorías que descrevem os componentes principais dos ataques, as categorias são: comunicação entre atacante e *botnet*, o tipo de inundação, o tipo de carga e a camada objetivo do ataque.

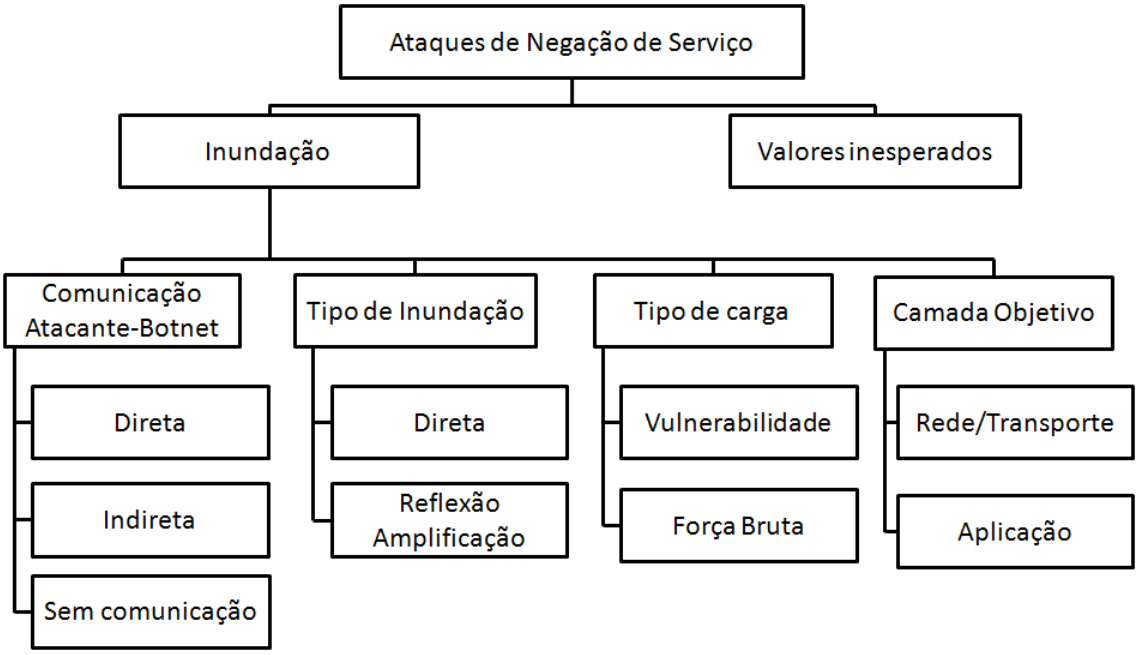

Figura 2.3: Características dos ataques de negação de serviço de inundação.

2.4.1 Ataques por vulnerabilidade

Este tipo de ataques é realizado enviando um pacote falsificado com o objetivo de bloquear uma aplicação ou protocolo da vítima. Isso é obtido aproveitando uma vulnerabilidade do protocolo ou aplicação para criar um laço (*loop*) ou um estado de bloqueio. Ao contrário dos ataques por inundação, um único pacote é suficiente

para provocar a negação de serviço. A solução para este tipo de ataques consiste em uma atualização de software para corrigir a vulnerabilidade. O ataque LAND(*Local Area Network Denial*) envia um pacote TCP SYN a uma porta aberta, o pacote é falsificado para que o endereço de origem e destino do pacote seja o endereço IP da vítima [36]. Assim, em sistemas vulneráveis, isto resulta em um laço no qual a vítima continue respondendo a si mesma indefinidamente. A maioria de sistemas atuais não é vulnerável a este tipo de ataques. O ataque *Ping of Death* consiste em enviar um pacote ICMP com um tamanho maior a 65535 bytes, o pacote é enviado fragmentado e uma vítima vulnerável seria derrubada ao tentar reconstituir o pacote [37]. Atualmente, quase nenhuma máquina é vulnerável a este tipo de ataque.

2.4.2 Ataques de inundação

Os ataques de inundação têm como objetivo consumir os recursos da vítima para impedir que atenda requisições de usuários legítimos. Essa condição pode ser obtida enviando pacotes que causem operações no servidor que consumam processamento ou memória, ou esgotem a largura de banda disponível para acessar o serviço. Nesse sentido, para que o ataque seja efetivo o atacante deve ter a capacidade de gerar grandes cargas no servidor ou no enlace sem utilizar muitos recursos. Isso é possível através do uso de *botnets* onde o atacante usa poucos pacotes para coordenar um ataque de milhares de máquinas atacando um objetivo, usando ataques de reflexão/amplificação onde o atacante use poucos pacotes para criar muitos pacotes de grande tamanho ou aproveitando vulnerabilidades que permitam, utilizando um número reduzido de pacotes, gerarem altas cargas no servidor. Em todos os casos é necessário estabelecer uma comunicação entre o atacante e as máquinas participantes do ataque:

Forma de comunicação

Durante um ataque de negação de serviço, um atacante requer enviar comandos às máquinas participantes do ataque para especificar a vítima do ataque, diversos parâmetros do tipo de tráfego a ser enviado, e coordenar o inicio e finalização do ataque, em esta categoria existem três tipos de comunicação:

Comunicação Direta: Este tipo de comunicação também é chamado de Modelo Agente-Mestre. A Figura 2.4 apresenta este tipo de comunicação. Nesse modelo, existem uma jerarquia na qual no primeiro nível está o atacante que coordena tudo o ataque. No segundo modelo, um conjunto de máquinas chamados de *handlers* recebem ordens do atacante e a comunicação entre eles é normalmente encriptada. O terceiro nível corresponde a um conjunto de agentes que são as máquinas que enviam

os pacotes do ataque para a vítima, a comunicação entre os *masters* e os agentes usa diversos protocolos como UDP ou ICMP. Quando uma máquina é infetada pode participar como *master* ou agente. O atacante mantém unicamente uma lista com os *masters* e quando são agregados novos agentes eles se reportam a um ou mais *masters*. O modelo apresenta três níveis porque o código malicioso está instalado nos *masters* e handlers e o atacante não precisa sempre estar conectado à *botnet*. Para realizar um ataque, o atacante se conecta os *masters* e envia ordens que são encaminhadas a seus agentes. A desvantagem de esse modelo de comunicação é que se um agente é descoberto, a lista de *masters* pode ser comprometida e se um *master* é comprometido o atacante pode ser descoberto se tenta se conectar a esse *master*.

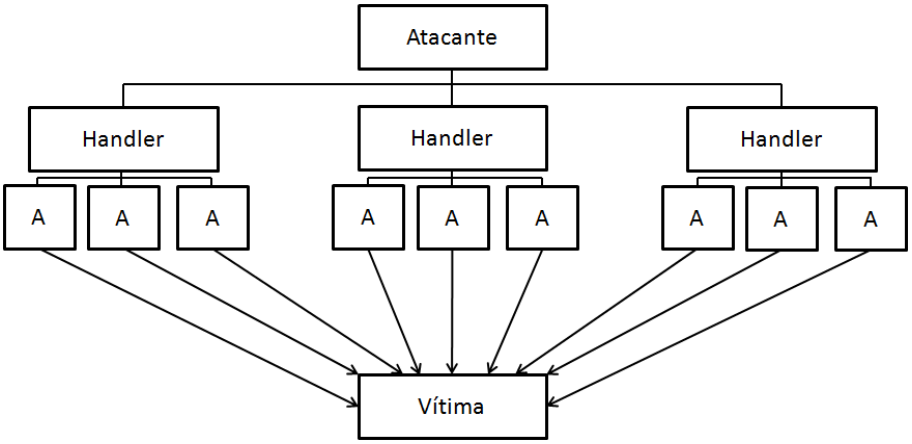

Figura 2.4: Modelo de comunicação direta. Os agentes recebem ordens dos *handlers* e se comunicam com periodicidade com eles utilizando pacotes de diversos protocolos.

Comunicação Indireta: A comunicação indireta utiliza algum mecanismo que não requer manter uma lista de máquinas como *masters*. Essa forma de comunicação utiliza ferramentas como os canais IRC (*Internet Relay Channel*) para estabelecer a comunicação entre o atacante e os agentes. A Figura 2.5 apresenta um exemplo. Quando um agente é agregado à *botnet* ele se une a um canal IRC específico e fica aguardando ordens do atacante. A vantagem da comunicação indireta é que o compromisso de um agente não necessariamente revela a identidade do atacante, e que os canais IRC são fáceis de criar.

Sem comunicação: Em este modelo, os parâmetros do ataque, o momento de inicio e finalização e as vítimas do ataque são armazenadas no código malicioso. A vantagem de esse modelo é que não requer nenhuma infraestrutura de comunicação, o que dificulta localizar ao atacante. A desvantagem é que não resulta possível modificar as condições do ataque, e o código só pode ser utilizado para uma vítima predeterminada.

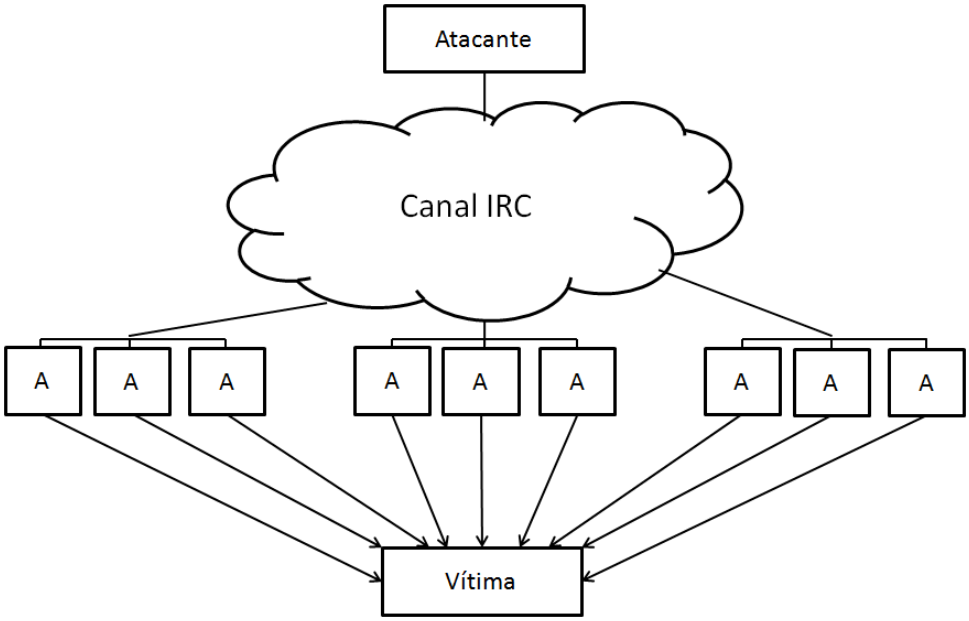

Figura 2.5: Modelo de comunicação indireta. Os agentes se conectam a um canal IRC preestabelecido. O atacante utiliza o canal IRC para coordenar os ataques.

Tipos de inundação

O tipo de inundação se refere à forma como o tráfego é enviado pelas máquinas pertencentes à botnet, existem dos tipos de inundações:

Inundação direta: Na inundação direta, a botnet dirige diretamente seu ataque às vítimas, possivelmente falsificando a IP de origem para dificultar o rastreamento. O impacto de este tipo de ataques depende da capacidade de gerar tráfego das *botnet*, a capacidade tem que ser maior que a capacidade de processamento ou a largura de banda do servidor. Embora os endereços IPs dos pacotes estejam falsificados é possível que os roteadores no caminho até a vítima identifiquem a localização dos *bots*.

Ataques de Reflexão/Amplificação: Em este tipo de ataque os *bots* falsificam o endereço IP de origem com o endereço IP da vítima e enviam seus pacotes a alguma entidade que possa responder a esse tipo de pacote, por exemplo, um servidor aberto de DNS. Assim, a entidade responde o pacote recebido à vítima. Dependendo do tipo de ataque é possível que a resposta do pacote seja muito maior que o pacote inicial, o que permite amplificar o tráfego enviado à vítima. Nos primeiros tipos de ataque de reflexão/amplificação eram usados pacotes com destino *broadcast* para que um número enorme de máquinas respondessem à vítima. Atualmente, a maioria de roteadores não encaminham pacotes com endereço de destino broadcast. As vantagens de este tipo de ataque são: Um atacante pode aproveitar outra infraestrutura para amplificar sua capacidade para gerar tráfego, resulta mais difícil rastrear este

tipo de ataques e em muitos casos este tipo de ataques geram efeitos colaterais nas entidades que são usadas como refletores.

Tipos de carga

Como o objetivo dos ataques de inundação é gerar uma carga ou congestionamento na vítima, um atacante tentara utilizar um tipo de carga nos pacotes que maximize o uso de recursos na vítima ou utilizara pacotes simples que visem a inundar à vítima [26]

Exploração vulnerabilidade: O tipo de carga selecionada procura aproveitar uma vulnerabilidade no protocolo ou aplicação atacada. Por exemplo os ataques TCP SYN aproveitam que no processo de estabelecimento de uma conexão, o servidor aloca recursos de memória para as tentativas de conexão até que elas são estabelecidas ou canceladas. Nesse caso, o ataque envia múltiplas requisições de apertura de conexão, mas nunca completa o processo, esgotando a memória da vítima.

Força bruta: Outros ataques podem simplesmente realizar uma inundação utilizando pacotes normais que vicem consumir os recursos da máquina, este segundo tipo de ataque é menos efetivo e requer uma capacidade maior do atacante para gerar tráfego. É importante ressaltar que se um tipo de ataque explora uma vulnerabilidade a qual é corrigida na vítima, o ataque vira uma inundação por força bruta.

Camada objetivo do ataque

Os ataques de negação de serviço podem ser dirigidos a consumir a largura de banda de uma vítima, os recursos alocados para o funcionamento dos protocolos de comunicação ou a determinadas aplicações específicas rodando no servidor. Dependendo de isso, existem duas camadas objetivos dos ataques de negação de serviço:

Ataques à camada de rede/transporte: Os ataques são criados utilizando pacotes dos protocolos UDP, TCP ou ICMP e tem como objetivo inundar consumir a largura de banda da vítima ou consumir os recursos da vítima para estes protocolos. Como o objetivo é a camada de rede/transporte, este tipo de ataques pode afeitar várias aplicações que sejam oferecidas pela vítima. Adicionalmente, as defesas para esse tipo de ataque podem ser instaladas, na fonte, no destino ou na rede, já que os roteadores só têm acesso à informação da camada de rede.

Ataques à camada de aplicação: Os ataques a esta camada procuram acabar com os recursos de processamento e memória dos servidores oferecendo aplicações a usuários legítimos. Em termos gerais, esse tipo de ataque consume menos largura de banda e são muito difíceis de detectar por entidades da rede, porque normalmente não tem acesso à informação trocada nessa camada. Por isso, as defesas contra esse

tipo de ataque normalmente são instaladas no servidor. Outra característica de esse tipo de ataque é que dependendo da aplicação atacada, pode não ser possível utilizar falsificação de IP, porque primeiro tem que ser estabelecida uma conexão TCP.

2.4.3 Exemplos de Ataques de Negação de Serviço

Inundação UDP: É uma inundação direta à camada de transporte. Este tipo de ataque consiste no envio de uma grande quantidade de pacotes UDP à vítima do ataque. É um ataque de inundação direta por força bruta, já que os pacotes UDP não possuem nenhum estado e não geram respostas da vítima. Um tráfego muito grande chegando à vítima termina por ocupar toda a largura de banda disponível e impede o acesso a usuários legítimos. Embora este tipo de ataques requeira uma grande capacidade do atacante para gerar tráfego, podem ser difíceis de prevenir e detectar porque os pacotes UDP de um ataque não levam nenhuma característica especial que permita diferenciá-los do tráfego legítimo. A única característica do fluxo malicioso é seu grande volume, mas se o atacante usa um número muito grande de máquinas, a vítima vai receber pequenos fluxos provenientes de sítios muito diversos.

Inundação ICMP: É um tipo de ataque similar à inundação UDP, mas utiliza pacotes ICMP para criar as inundações. É usado este tipo de pacotes porque muitos *firewalls* não bloqueiam este tipo de tráfego.

Inundação TCP SYN: É um tipo de inundação direta à camada de transporte que explora a vulnerabilidade do protocolo TCP de alocar memória para as tentativas de conexão até que sejam completadas ou canceladas e não verificar a quantidade de pacotes TCP SYN recebidos por um cliente em um intervalo de tempo. O atacante envia um pacote de TCP SYN com um endereço IP falso á vítima, a vítima responde com um pacote TCP SYN ACK e aloca memória para a conexão. O atacante nunca envia um pacote ACK para confirmar o estabelecimento da conexão, o que ocasiona que a conexão fique em um estado SYN_RECV e os recursos de memória não são liberados. Uma inundação de este tipo de ataques pode consumir rapidamente os recursos disponíveis em um servidor para atender novas conexões.

Ataque NAPTHA: O ataque é similar ao TCP SYN Flood, mas ele envia pacotes para manter conexões nos servidores em diferentes estados, gastando recursos no servidor. O atacante não mantém nenhuma informação sobre o estado da conexão, simplesmente responde à vítima utilizando os valores dos campos recebidos no último pacote. A diferencia da inundação SYN, se as conexões são mantidas em um estado diferente de SYN_RECV não é possível falsificar a IP de origem.

Ataque *Smurf*: Este ataque é uma inundação de reflexão/amplificação à camada de transporte e utiliza pacotes ICMP. O atacante envia pacotes falsificados onde endereço de origem é o endereço da vítima. Os pacotes são enviados com endereço

de destino *broadcast*, o que ocasiona que um grande número de máquinas respondam os pacotes ICMP à vítima. Atualmente, o ataque não é efetivo porque a maioria de roteadores não encaminham pacotes com endereço de destino em *broadcast*

Ataque Fraggle: O ataque explora dois protocolos UDP, o protocolo *Echo* e o *CharGen*. O atacante envia um pacote a uma máquina com a porta do serviço *CharGen* aberta. O endereço de origem do pacote é falsificado com o endereço da vítima e a porta de origem com a porta de *Echo*, criando um loop entre as duas entidades.

Ataque de baixa taxa TCP: O ataque é uma inundação direta por vulnerabilidade que explora o mecanismo de retransmissão por *timeout* de TCP [38]. O objetivo do ataque é causar negação de serviço utilizando tráfego em rajadas para dificultar sua detecção. O tráfego em rajadas segue um padrão de uma onda quadrada, o pico da onda induz TCP *timeouts* na vítima o que causa que os emissores diminuam sua taxa de transmissão. Depois do pico, a taxa do tráfego malicioso diminuiu e novamente se incrementa logo depois que os emissores saem do estado de *timeout*. Isso ocasiona que os fluxos TCP legítimos fiquem com uma taxa de transmissão mínima, sem que o atacante tenha que altos volumes de tráfego de forma constante.

Ataque de reflexão/amplificação DNS: Esse tipo de ataques utilizam aos servidores DNS como entidades refletoras e amplificadoras [6]. A Figura 2.6 apresenta um exemplo de um ataque de reflexão por DNS. Um atacante pode utilizar um servidor DNS comprometido para inserir entradas grandes no DNS. Posteriormente uma *botnet* lança uma inundação de requisições DNS ao servidor comprometido e outros servidores com o endereço IP de origem falsificado com o endereço da vítima. Assim, as repostas são enviadas de volta a vítima que resulta inundada. A amplificação é obtida porque as respostas DNS têm um tamanho maior às requisições. Adicionalmente, o tráfego dos servidores DNS normalmente não é bloqueado como malicioso e se um *firewall* bloqueia a porta 53 do serviço DNS, o servidor não poderá resolver nomes. Assim, a solução para esse tipo de ataques não resulta trivial. O maior ataque do 2013 foi um ataque de reflexão DNS, dirigido contra uma companhia anti-spam, o ataque conseguiu atingir uma taxa de 300Gb/s [8].

Ataque de inundação HTTP: Esse tipo de inundações, dirigidas à camada de aplicação pretendem gerar uma carga de processamento muito grande no servidor Web, o que impede ao servidor atender as requisições de usuários legítimos. Uma inundação HTTP simples utiliza um conjunto de máquinas para gerar uma grande quantidade de requisições GET/POST a um servidor web. O conteúdo solicitado por ser grande e cada atacante pode enviar varias requisições por segundo. Outro tipo de inundação pode utilizar o fato que HTTP 1.1 permite realizar várias requisições utilizando uma mesma sessão. Esse tipo de ataques impede ao servidor atender as requisições HTTP de usuários legítimos.

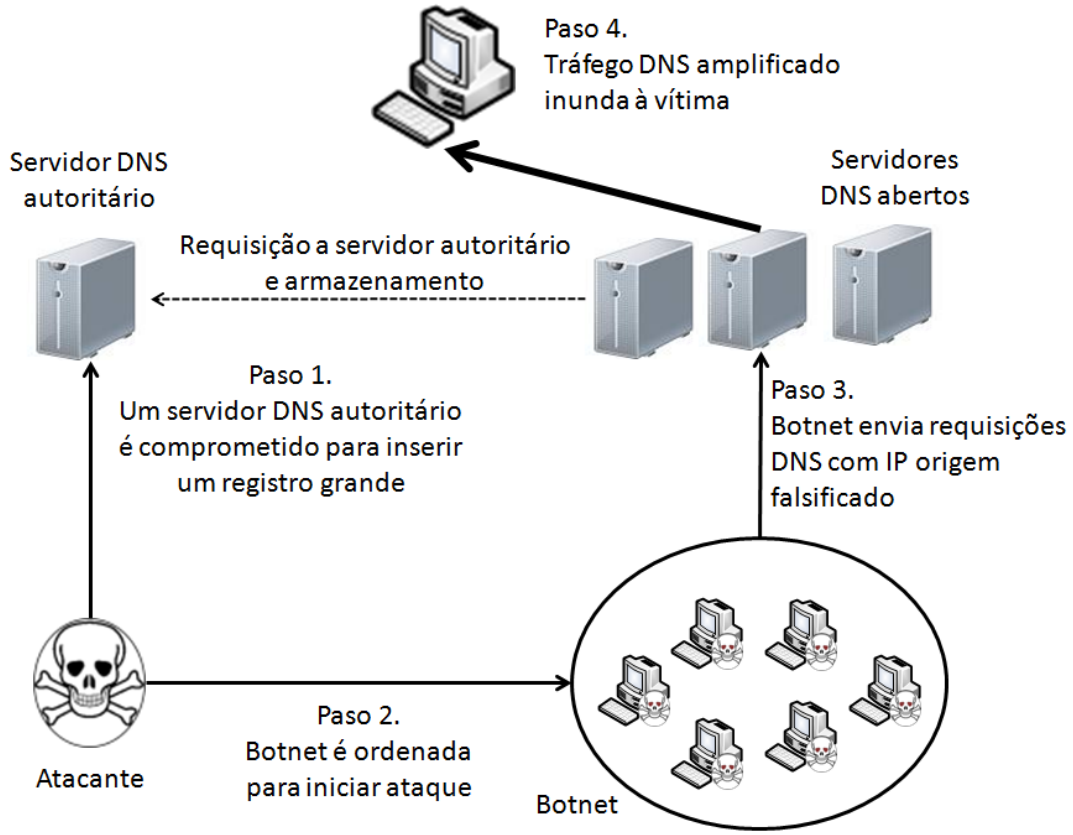

Figura 2.6: Ataque DoS por reflexão/amplificação DNS. Um atacante pode comprometer um servidor DNS autoritário para instalar nele um registro de tamanho grande. Posteriormente a botnet envia requisições DNS a um conjunto de servidores DNS. O endereço IP das requisições é falsificado com o endereço da vítima. Os servidores DNS respondem à vítima e provocam a inundação.

Ataques de requisição/resposta lenta: O objetivo de essa inundação à camada de aplicação é gerar a mínima quantidade de pacotes possíveis à vítima do ataque. Em um ataque de requisição lenta, um atacante envia requisições parciais de cabeçalhos HTTP. As requisições são enviadas lentamente e nunca são completadas. À medida que chegam novas requisições são ocupados mais recursos no servidor até que finalmente ele não possui recursos para atender novas requisições de usuários legítimos. Já no caso de resposta lenta, um atacante envia um cabeçalho HTTP indicando que vá a enviar uma resposta de um tamanho enorme. Depois, o atacante envia uns poucos bytes da resposta muito lentamente. De novo, os recursos são esgotados até que o servidor não consegue atender novas requisições.

Ataques assimétricos: O objetivo de esse tipo de ataques é enviar requisições que geram altas cargas de processamento no servidor vítima, como acessos a bases de dados e operações de disco.

2.5 Observações Finais

Os ataques de negação de serviço são facilitados pelas características no projeto da Internet. O compartilhamento de recursos cria um risco que um grupo de usuários mal comportados, abusando dos recursos compartilhados, possa afetar o desempenho de outros usuários. O principio fim-a-fim da Internet facilita situações donde os extremos finais da comunicação são saturados de pacotes encaminhados por entidades intermediarias que possuem pouco conhecimento da informação encaminhada. A falta de mecanismos para assegurar qualidade de serviço na Internet permite que pacotes simples, que não pertencem a aplicações críticas, sejam tratados com a mesma prioridade que outros pacotes, o que possibilita inundações com tudo tipo de pacote. A falta de enforcamento de responsabilidade na Internet permite que muitos usuários façam ataques de forma anônima, o que permite a proliferação de tudo tipo de atacantes, e finalmente o gerenciamento distribuído da Internet pode dificultar a implantação de mecanismos de segurança entre sistemas autônomos, o que dificulta a prevenção de ataques de negação de serviço distribuído.

Atualmente, as principais motivações para fazer ataques de negação de serviço são políticas e econômicas. Na política, os DoS viraram uma ferramenta de protesta e de atrair a atenção. Nesse sentido, são derrubados sítios de organizações econômicas e políticas participantes de discussões o que promovem leis impopulares para determinados setores da população. No caso do ataque de *Anonymous* no 2010 e 2012 o ataque foi coordenado pelo grupo, mas os o maior volume de tráfego foi gerado por cidadãos de todas as partes do mundo participando voluntariamente do ataque e usando ferramentas *web* simples desenvolvidas por *Anonymous* [39]. Os DoS também são utilizados como uma ferramenta comercial para causar prejuízo a empresas da competência, em muitas ocasiones uma empresa contrata a terceiros para realizar DoS programados durante momentos importantes para a competência. Isso tem ajudado a criar um mercado de ferramentas de ataques e defesas [40].

As motivações para criar ataques de negação de serviço e as características da Internet têm facilitado a criação de muitos tipos de DoS. Em todos os casos, o objetivo é impedir que a vítima ofereça seus serviços a usuários legítimos sem necessidade de invadir a máquina da vítima. Nos ataques de negação de serviço por inundação, isso é obtido consumindo os recursos disponíveis da vítima. O atacante coordena as máquinas disponíveis para seus ataques para explorar as vulnerabilidades de algum protocolo usado na comunicação ou a aplicação específica. Os ataques que geram padrões de tráfego muito similares aos legítimos e conseguem consumir muitos recursos são os mais efetivos. Se não é explorada nenhuma vulnerabilidade, a inundação pode ser obtida usando a força bruta, enviando uma grande quantidade de pacotes com o objetivo de consumir a largura de banda da vítima, dependendo da

distribuição de máquinas usadas no ataque e a possibilidade de falsificar endereços IPs, os fluxos da inundação pode ser difícil de identificar.

Capítulo 3

Ferramentas de Ataques de Negação de Serviço

O capítulo anterior apresentou uma análise das características, motivações e tipos de ataques de negação de serviço. Foi analisada a relação das características da Internet com os ataques de negação de serviço e foi apresentada uma taxonomia desses ataques. Este capítulo analisa as diferentes ferramentas disponíveis para realizar ataques de negação de serviço. São apresentadas ferramentas para realizar inundações nas camadas de rede e transporte assim como na camada de aplicação. Também é apresentado o estado da arte nos ambientes de testes para ataques de DoS.

3.1 Ferramentas para DoS na camada de rede e transporte

A maioria de ferramentas para realizar a inundação para um ataque de negação de serviços distribuído possui dois tipos de códigos, o primeiro é o código usado pelo atacante para controlar o ataque e o segundo tipo são os códigos dos agentes e dos *handlers*. As ferramentas possuem mecanismos para esconder a execução do código nos *handlers* e agentes, uma vez que estes rodam em máquinas contaminadas e deve ser evitado qualquer tipo de ação que denuncie que a máquina está sob conrole. Os mecanismos que visam esconder a execução do código utilizam nomes diferentes para os processos criados na pilha de processos, não gerando nenhuma mensagem na *stdout*, e encriptam os dados importantes como as listas dos participantes dos ataques que seriam fáceis de serem detectados. A comunicação entre o *master* e os *handlers* é realizada por uma conexão `Telnet` ou uma conexão cifrada por chaves simétricas. No caso das comunicações cifradas, as senhas para gerar o par de chaves podem estar armazenadas no código ou serem geradas no momento da compilação

da ferramenta. A comunicação entre os *handlers* e os agentes pode utilizar quadros ICMP ou UDP com valores predeterminados em seu *payload* para transmitir as ordens que permitem coordenar o ataque. Algumas ferramentas permitem criar uma *shell* no *handler* ou agente para atualizar a ferramenta. Estes tipos de ferramentas só possuem código para gerar e coordenar as inundações. Assume-me o exército de *bots* já foi criado e como foi mencionado na Seção 2.3, um atacante utilizaria algum tipo de ferramenta de exploração de vulnerabilidades ou *worms* para a criação da *botnet*. As principais ferramentas nesta categoria são:

Tabela 3.1: Principais características das ferramentas para realizar inundações na camada de rede. Fonte [4].

Nome	**Comunicação Atacante Handler**	**Comunicação Handler Agente**	**Inundações**	**Capacidade Atualização**
Trin00	TCP	Pacotes UDP	UDP	Não
TFN	TCP	Pacotes ICMP	UDP TCP SYN ICMP *Smurf*	Não
TFN2K	TCP encriptada Pacotes ICMP, UDP encriptados	Pacotes ICMP, UDP encriptados	UDP TCP SYN ICMP *Smurf* `Targa3`	Sim
Stacheldraht	TCP encriptada	Pacotes ICMP	UDP TCP SYN ICMP *Smurf*	Sim
Trinity	Canal IRC	Canal IRC	UDP TCP SYN ICMP	Não

3.1.1 Trin00

A Trin00 foi uma das primeiras ferramentas criadas para realizar ataques de negação de serviço. Ela utiliza o modelo de comunicação direta com o atacante, *handlers* e os agentes. Uma sessão `Telnet` é estabelecida entre o atacante e o *handler* para coordenar o ataque. No caso de uma segunda tentativa de conexão ocorrer em um *handler* enquanto uma conexão estiver ativa, um alarme é enviado para a primeira conexão. Este procedimento alerta o atacante que o *handler* foi comprometido [4]. O *handler* mantém um arquivo em claro com as IPs de seus agentes.

Quanto aos agentes, eles podem lançar inundações UDP, podem inundar uma máquina com endereço IP específico ou máquinas relacionadas a um conjunto de IPs. Os endereços IP de origem dos pacotes da inundação não estão falsificados e, portanto, é possível rastrear o ataque até o agente que executa a inundação. Também é possível configurar o tempo, em segundos, de duração da inundação, o que dificulta os procedimentos de rastreamento. Trin00 utiliza algumas portas predeterminadas para a comunicação entre os participantes do ataque, mas em outras versões do código podem variar, na versão obtida para esta dissertação as portas são apresentadas na Tabela 3.2:

Tabela 3.2: Portas usadas pela ferramenta Trin00. Fonte [4].

Origem	**Destino**	**Porta**
Atacante	Handler	27665/TCP
Handler	Agente	27444/UDP
Agente	Handler	31335/UDP

Usando a ferramenta Trin00, o atacante envia comandos ao *handler*, descritos na tabela 3.3, e cada *handler* envia o respectivo comando a seus agentes. Trin00 utiliza pacotes UDP, enviados sobre várias portas aleatórias, para enviar os comandos aos agentes, o que dificulta rastrear esse tipo de comunicação. Resulta mais fácil tentar bloquear a comunicação entre o *handler* e o atacante que devem usar uma porta predeterminada.

3.1.2 *Tribe Flood Network* (TFN)

A ferramenta *Tribe Flood Network* (TFN) possui um funcionamento similar a ferramenta Trin00 e também usa o modelo de comunicação direta. A lista de agentes mantida pelos *handlers* está encriptada usando o algoritmo *blowfish*, a chave simétrica para encriptar a lista de agentes está armazenada no código. Os *handlers* se comunicam com os agentes utilizando pacotes ICMP_ECHOREPLY [5]. Isso pode dificultar a detecção da comunicação dos agentes, mas sistemas de *Intrusion Detection System* (IDS) monitorando os fluxos de uma rede podem procurar por pacotes ICMP_ECHOREPLY chegando ou saindo da rede sem que exista um ICMP_ECHOREQUEST nesse fluxo anteriormente. Os comandos para os agentes são enviados utilizando o campo de 16 bits do ICMP_ECHOREPLY. A Tabela 3.4 apresenta os comandos da ferramenta. TFN possui uma capacidade maior que Trin00 para gerar ataques, os ataques disponíveis são: Inundação UDP, inundação TCP SYN, inundação ICMP e *Smurf*.

Tabela 3.3: Comandos da ferramenta Trin00 que pode realizar inundações UDP. Fonte [4].

Comando	**Função**
die	Desligar o *handler*
quit	Terminar a conexão com *handler*
mtimer N	Estabelecer a duração do ataque de DoS em segundos
dos IP	Endereço IP da vítima
mdie pass	Desativar todos agentes, requer enviar uma senha
mping	Enviar um *ping* a todos os agentes
mdos <ip1:ip2:ip3>	Especificar múltiplos endereços como vítimas do DoS
info	Mostrar na tela a versão do código e informação de compilação
msize	Estabelecer o tamanho do *buffer* para os pacotes durante o DoS
nslookup host	Realizar um *nslookup* para o *host* especificado
killdead	Criar uma lista com os agentes que responderam com o *string "hello"*, eliminando os agentes inativos que não responderam.
bcast	Listar todos os agentes ativos
help[cmd]	Mostrar informações de uso (ajuda) do comando especificado
mstop	Terminar o ataque de DoS

Tabela 3.4: Comandos da ferramenta TFN que permite os ataques de inundação TCP SYN, ICMP e *Smurf* com a possiblidade de falsificar os endereços IP de origem. Fonte [5].

Comando	**Função**
-2<bytes>	Estabelecer o tamanho do pacote das inundações
-1<mask>	Máscara para falsificar os endereços IP de origem que aleatórios. Os parâmetros 1 referem-se a endereços classe A, 2 classe B, 3 classe C
0	Trocar o tamanho dos pacotes de inundação
<1 targets>	Inundar com pacotes UDP
<2 targets> <port>	Inundar com pacotes TCP SYN
3 <targets>	Inundar com ICMP
4 <port>	Criar um console que escuta os comandos na porta especificada
5 <target@bcast>	Provocar uma inundação *Smurf*. Os *bcast* representam os amplificadores

3.1.3 *Tribe Flood Network* 2000 (TFN2K)

A ferramenta *Tribe Flood Network* 2000 (TFN2K) é uma versão aprimorada da TFN, na qual a comunicação entre todos os participantes do ataque é encriptada, o que dificulta a detecção do ataque. A chave simétrica é definida pelo atacante quando realiza a compilação do código. A comunicação entre os participantes do ataque é encriptada usando o algoritmo CAST-256. Os comandos entre o *handler* e os agentes podem utilizar tanto o protocolo TCP, quanto o UDP ou quanto o ICMP, o que também dificulta a detecção do ataque. A ferramenta pode realizar as mesmas inundações que a ferramenta TFN e acrescentou-se a possibilidade de realizar um novo ataque conhecido como `targa3`. Este ataque cria pacotes malformados de maneira a também realizar ataques por vulnerabilidades. Com TFN2K podem ser lançadas inundações que combinem pacotes TCP, UDP e ICMP. Com TFN2K é possível criar um *shell* de comandos nos agentes, o que permite atualizar o código da ferramenta. Os comandos que podem ser transmitidos usando TFN2K são apresentados na Tabela 3.5. Para enviar argumentos junto com os parâmetros é usado o carácter $-i$. Adicionalmente, a ferramenta TFN2K pode enviar pacotes com endereços de origem IP aleatórios para simular tráfego normal da rede e, assim, dificultar o rastreamento do atacante.

Tabela 3.5: Comandos da ferramenta TFN2K. Fonte [4].

Comando	**Função**
a	Criar um console de comandos no agente
b	Trocar o tamanho dos pacotes da inundação
c	Trocar o modo de falsificação de IPs
d	Interromper a inundação
e	Inundar com pacotes UDP
f	Inundar com pacotes SYN
g	Estabelecer a porta da inundação
h	Inundar com pacotes ICMP
i	Efetuar o ataque *Smurf*
j	Efetuar o ataque `Targa3`
k	Definir os intervalos de inundações UDP/TCP/ICMP
l	Executar um comando no agente

3.1.4 *Shaft*

Shaft é também uma ferramenta baseada na ferramenta Trin00 [5]. O atacante se comunica com o *handler* utilizando uma conexão `Telnet` e os agentes recebem ordens através de comunicações usando o UDP. *Shaft* utiliza falsificação de endereço IP para

enviar os pacotes de inundação. Uma característica da ferramenta é a capacidade de trocar as portas usadas pelos *handlers* para enviar e receber comandos. Isso dificulta a detecção da comunicação entre os agentes e os *handlers* e entre o atacante e os *handlers*. Adicionalmente o *handler* pode enviar comandos aos agentes para reunir estatísticas sobre o estado do ataque. As estatísticas podem ser usadas pelo atacante para estimar a quantidade de agentes necessária para gerar um tráfego de inundação que consiga derrubar à vítima.

3.1.5 *Stacheldraht*

A ferramenta combina as características de Trin00 e TFN2K, a comunicação entre o atacante e os *handlers* é encriptada usando uma chave simétrica definida no momento de compilação da ferramenta. O algoritmo utilizado para encriptar é *blowfish*. O agente ao ser ativado procura por *handlers* armazenados numa lista encriptada que é criada ao momento de compilar o código. No caso da lista não ser criada, uma lista de endereços IPs predeterminada é utilizada. A comunicação entre os agentes e os *handlers* é feita em claro, usando pacotes ICMP_ECHOREPLY. Os comandos para os agentes são enviados utilizando pacotes ICMP_ECHOREPLY, nos quais o campo de ID é utilizado para identificar o comando. O *handler* atualiza sua lista de agentes ativos usando um mecanismo de *heartbeat*. O mecanismo funciona com os agentes enviando periodicamente pacotes ICMP_ECHOREPLY com um valor de ID igual a 666 e no campo de dados com um *string* de caracteres correspondente à sequência de letras “ficken”.

O *handler* ao receber esse pacote responde com outro ICMP_ECHOREPLY com o ID de 667 é o *string* correspondente à seqüência de letras "ficken". A Tabela 3.6 apresenta os comandos que podem ser enviados aos agentes. O comando `.killall` termina o processo que está rodando nos agentes, isso pode ser útil para eliminar os agentes que foram descobertos. Adicional aos comandos mencionados na tabela, a ferramenta também possui a capacidade de atualizar o código de todos os participantes do ataque, utilizando o comando Berkeley `rcp` na porta 514/TCP [4].

3.1.6 *Trinity*

A ferramenta *Trinity* de ataque de negação de serviço por inundação é uma ferramenta que, ao contrário das descritas anteriormente, usa comunicação indireta através de um canal IRC. A vantagem da comunicação indireta é que o compromisso de um agente não necessariamente revela a identidade do coordenador do ataque. Esta ferramenta também realiza inundações UDP, TCP SYN, ICMP e utiliza falsificação de endereços IP de origem. Quando o código dos agentes é executado, ele se conecta com um canal IRC, configurado no momento da compilação da ferramenta.

Tabela 3.6: Comandos da ferramenta *Stacheldraht*. Fonte [4].

Comando	**Função**
.distro user server	Instalar uma nova cópia do agente no servidor usando a conta indicada
.help	Mostrar na tela os comandos suportados
.killall	Destruir todos os agentes
Madd ip1[:ip1] ip2[:ip2]	Adicionar estes IPs à lista de vítimas
.mlist	Listar os endereços IPs sendo atacados atualmente
.mping	Enviar um PING a todos os agentes para verificar se estão ativos
.msadd	Adicionar um *handler* a lista de *handlers* no agente
.msort	Retirar da lista de agentes os agentes inativos
.mstop ip1[:ip1] ip2[:ip2]	Interromper a inundação nas vítimas especificadas
.msrem	Retirar um *handler* a lista de *handlers* no agente
.msyn ip1[:ip1] ip2[:ip2]	Inundar com TCP SYN as vítimas especificadas
mtimer segundos	Estabelecer a duração da inundação
.mudp ip1[:ip1] ip2[:ip2]	Inundar com UDP nas vítimas especificadas
.micmp ip1[:ip1] ip2[:ip2]	Inundar ICMP nas vítimas especificadas
.settisize	Estabelecer o tamanho dos pacotes ICMP utilizados
.setusize	Estabelecer o tamanho dos pacotes UDP utilizados

O nome de usuário usado no canal está composto pelas primeiras 6 letras do *host-name* da máquina infectada e três letras aleatórias [41]. O agente posteriormente aguarda os comandos enviados pelo atacante, que podem ser enviados a agentes individuais ou a tudo o canal IRC.

3.2 Ferramentas de Ataques de DoS na Camada de Aplicação

Os ataques de negação de serviço por inundação que visam consumir o recurso de banda passante requerem que seja enviada uma grande quantidade de pacotes por segundo para consumir a banda passante do enlace. Os ataques de negação de serviço na camada aplicação visam exaurir outro recurso diferente da banda passante e, como isto, não exigir uma grande infraestrutura de ataque, dificultando assim a detecção do ataque. Busca-se com esse tipo de ataques reduzir a necessidade de um grande número de máquinas comprometidas para conseguir derrubar servidores

que não estejam preparados para este tipo de ataques. Outra característica destas ferramentas é que não usam falsificação de IP, porque os ataques à camada de aplicação precisam primeiro estabelecer uma conexão TCP e a falsificação do IP de origem naõ conseguiria estabelecer uma conexão. Igualmente aos ataques anteriormente descritos, o atacante primeiro precisa explorar vulnerabilidades em algumas máquinas e infetá-las com o código que realizará o ataque.

3.2.1 *Slowris*

O objetivo da ferramenta *Slowris* é derrubar um servidor web através da manutenção de conexões ativas ao enviar requisições HTTP regularmente em intervalos de vários segundos [6]. Este procedimento mantém os *sockets* abertos no servidor web e depois de um tempo de ataque, todos os *sockets* do servidor ficam ocupados com as requisições de *Slowris*, o que impede que ele atenda as requisições de usuários legítimos. Uma característica do *Slowris* é que durante o ataque, o arquivo de *logs* do servidor web não criará entradas correspondentes a erros, porque nenhuma sessão HTTP é concluída, mas ao finalizar o ataque, muitas entradas de erros 400 serão criadas no log. Este ataque, ao contrário dos ataques de inundação de pacotes IP, utiliza pouca largura de banda passante.

3.2.2 *RUDY*

A ferramenta *RUDY*, do acrônimo *aRe U Dead Yet?*, é uma ferramenta baseada em um princípio similar ao *Slowris*, que é o de ocupar os recursos de um servidor utilizando transações que nunca concluem [6]. RUDY utiliza o envio de dados com uma frequência muito baixa. O funcionamento de RUDY está baseado na premissa que muitos servidores web aguardam um longo tempo para o envio de dados dos formulários pelo usuário para poder receber informações de usuários não adaptados a determinados dispositivos de acesso e também devido a conexões lentas à Internet. Logo, RUDY envia um cabeçalho HTTP indicando uma resposta POST a um formulário disponível no servidor web e posteriormente envia um byte da resposta com um intervalo de alguns segundos entre cada envio. RUDY tenta criar algumas conexões simultâneas para ocupar todos os recursos disponíveis no servidor.

3.2.3 *Low Orbital Ion Cannon* (LOIC)

O LOIC foi a ferramenta principal utilizada pelo grupo de ativistas *Anonymous* durante os ataques de DDoS que têm realizado desde 2010. É uma ferramenta para realizar inundação de HTTP, enviando uma grande quantidade de requisições `HTTP GET` [42]. A característica fundamental do LOIC é que foi projetada para ter um

uso muito simples, e ainda possui uma interface gráfica. Estas características dos ataques estão relacionadas ao tipo de atividade do grupo *Anonymous*, que é um grupo de ativistas que coordena o ataque formado de milhares de pessoas no mundo como voluntários. Assim, o LOIC máquinas atacantes de voluntários e, portanto, não possui nenhum mecanismo para invadir e contaminar máquinas para fazer um exército de *bots* e também não precisa esconder seu funcionamento. O ataque é formado então por um exército de voluntários que executam o código de ataque a vítima, coordenados por um canal IRC.

3.3 Plataformas de Testes para Ataques de Negação de Serviço

Os ataques de negação de serviço têm se servido de ferramentas que têm evoluído para aumentar a capacidade do ataque e cada vez mais automatizadas, ocasionando impactos cada vez maiores na Internet. O desenvolvimento tradicional de mecanismos de defesa contra esses ataques seguem uma abordagem reativa, ou seja, depois que os ataques de negação de serviço causam enormes prejuízos procura-se desenvolver novos mecanismos de defesa. Logo, os mecanismos de segurança são desenvolvidos como resposta aos ataques que ocasionaram danos. No entanto, há dois fatores fundamentais que requerem uma mudança de abordagem para prover segurança. Primeiro, a crescente qualificação técnica dos atacantes e dos tipo de ataque. Já existe o interesse "comercial e financeiro" em se fazer ataques. Isto muda radicalmente o cenário de atacantes que são adolescentes, ingênuos ou frustrados para atacantes profissionais e quadrilhas especializadas com recursos para investir em tecnologia, formação de recursos humanos e em novas formas de ataque. O segundo fator fundamental de mudança de cenário consiste no processo atual de integração da Internet com aplicações e infraestruturas críticas. Ataques cibernéticos podem afetar perigosamente as infraestruturas críticas criando *blackout* ou provocando catástrofes. Portanto, é necessário o desenvolvimento de ferramentas para se estudar e avaliar os ataques de negação de serviço e, com consequência, permitir a concepção de sistemas seguros que previnam a negação de serviços, mudando o cenário atual para uma abordagem proativa.

O uso de simulações para testar ataques em topologias de rede e protocolos é muito importante, porém, em muitas ocasiões não permite representar realisticamente o comportamento das ferramentas de DoS e as interações, assim como seu impacto em implementações específicas de sistemas operacionais [43]. Com o uso de simulações, diversos parâmetros dos modelos dos roteadores devem ser configurados adequadamente para imitar o comportamento de roteadores reais; alguns desses

parâmetros são: tamanho dos *buffers*, políticas e taxas de encaminhamento [44].

As plataformas de testes (*testbeds*) oferecem uma infraestrutura que representa realisticamente os protocolos e implementações analisadas em experimentos de ataques de negação de serviço [45]. A vantagem dos *testbeds* é que representam o comportamento de protocolos e sistemas operacionais reais. Nas máquinas do *testbed* são instaladas versões reais de diversos aplicativos e ferramentas de ataque e as defesas podem ser testadas diretamente em um ambiente real, o que permite uma melhor avaliação do comportamento de ferramentas de DoS e dos mecanismos de defesa que precisam ser desenvolvidos. A plataforma de teste deve apresentar condições muito similares às condições da atual Internet. Na presente seção é apresentado o *testbed Cyber DEfense Technology for Experimental Research)* (DETER), um *testbed* desenvolvido pelas universidades de UC Berkeley e a universidade de *South California* e patrocinado pela *National Science Foundation and the Department of Homeland Security* dos Estados Unidos. Também é apresentado o *Future Internet Testbed with Security - FITS* um *testbed* desenvolvido pelo Grupo de Teleinformática e Automação (GTA), em colaboração com outras instituições, um *testbed* de código aberto baseado em OpenFlow e tecnologias de virtualização de Xen.

3.3.1 *Cyber DEfense Technology for Experimental Research* - DETER

O DETER [46] é um *testbed* de pesquisa em segurança cibernética desenvolvido pelas universidades de UC Berkeley e a universidade de *South California* e patrocinado pela *National Science Foundation* (NSF) e o *Department of Homeland Security* (DHS) dos Estados Unidos. O DETER foi criado para responder à falta de uma infraestrutura que permitisse aos pesquisadores e desenvolvedores de mecanismos de segurança testar suas propostas em condições que permitam avaliar o efeito de escala e comportamento agregado. A maioria dos profissionais da área de segurança não possui recursos para criar uma grande infraestrutura para testes e terminam avaliando suas propostas em ambientes pouco realistas, o que não permite avaliar a eficácia da solução e dificulta o processo de implementação. Adicionalmente, as plataformas criadas por empresas particulares não possuíam elementos comuns ou reutilizáveis por outras equipes, o que dificulta o processo de experimentação e validação dos resultados por outras equipes. Assim, o DETER é uma iniciativa nacional para criar um *Testbed* de Segurança [46]. O objetivo do DETER é prover uma infraestrutura robusta, que possa ser utilizada por pesquisadores e desenvolvedores de forma concorrente para testar mecanismos de segurança em ambientes reais e com topologias de média escala. Um dos objetivos primordiais do DETER é permitir a "repetibilidade" dos experimentos e a obtenção dos mesmos resultados. Adicio-

nalmente, como o DETER é um *testbed* de segurança, seu projeto foi feito para ser um *testbed* seguro que não permita a propagação de código malicioso à Internet, ou permita ser utilizado para fazer DoS às redes externas.

Arquitetura do DETER

Os usuários da plataforma DETER recebem acesso a um conjunto de máquinas físicas conectadas em uma topologia especificada pelos próprios usuários. Em cada máquina física da plataforma é instalada uma imagem do sistema operacional especificado pelo usuário e podem ser instaladas diversas aplicações e ferramentas para iniciar diversos ataques como, por exemplo, os ataques de negação de serviço. Logo, é possível testar propostas de defesas e coletar estatísticas dos experimentos. A topologia experimental criada, chamada rede experimental, é isolada da Internet, mas os experimentadores podem acessar as máquinas utilizando sessões `ssh` através de uma rede especial chamada rede de controle. A arquitetura do DETER é apresentada na Figura 3.1. As máquinas para experimentos do DETER estão agrupadas em dois aglomerados (*clusters*), um localizado na UC Berkeley e outro na ISI. Em total o DETER possui 488 máquinas experimentais. Os aglomerados (*clusters*) estão conectados pela Rede da Califórnia de Pesquisa de Alto Desempenho, com capacidade de 1Gb/s [47]. Todas as máquinas experimentais possuem quatro interfaces de rede, uma delas é usada pela rede de controle. A rede experimental é interconectada através de comutadores de alto desempenho e são usadas etiquetas de VLAN para criar a topologia do experimento e isolar o tráfego entre experimentos. Todo o processo de criação de experimento, assim como a execução de diversas aplicações do DETER é controlado por um servidor principal chamado *"Boss Server"*. Todo o processo de criação e controle do experimento, assim como o acesso as máquinas experimentais é feito através da rede de controle. A rede de controle está isolada fisicamente da rede experimental, e são instalados múltiplos *firewalls* para impedir que seja encaminhado tráfego da rede experimental para a rede de controle, assim como evitar que tráfego da rede de controle seja encaminhado para a Internet, além das sessões `ssh` dos usuários remotos do DETER. O servidor chamado *"User"* possui as pastas de armazenamento de arquivos dos usuários. Quando um usuário é aceito para realizar experimentos no DETER é criada uma conta nesse servidor com um diretório com 10 GB de espaço. Um diretório é compartilhado via NFS entre o diretório do usuário e os nós experimentais. Finalmente, a plataforma DETER é acessado remotamente através de um *firewall* chamado *"Gatekeeper"*.

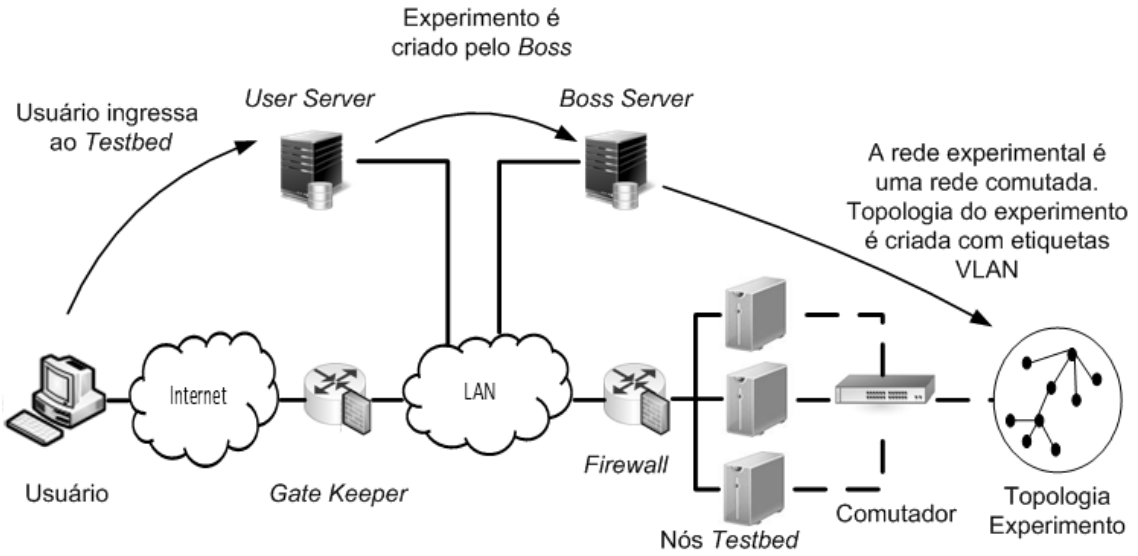

Figura 3.1: Arquitetura do *testbed* DETER. O *testbed* pode ser acessado através da Internet. Os nós experimentais são conectados segundo a topologia definida pelo usuário, usando uma rede comutada e etiquetas de VLAN.

Projetos e Experimentos do DETER

O modelo do funcionamento do DETER está baseado no conceito de Projetos e Experimentos. Um usuário desejando utilizar o *testbed* submete uma descrição do Projeto pretende realizar no *testbed*. A descrição deve indicar os riscos identificados do Projeto, a Universidade ou Instituição realizando-o e um chefe do Projeto. O chefe do Projeto normalmente é um professor e recebe permissões para criar contas para outras pessoas dentro do Projeto. Um Projeto está composto por um conjunto de experimentos, que são as topologias e o conjunto de testes que realizam os pesquisadores. A descrição de um experimento pode ser armazenada para ser executado ou modificado em futuras ocasiões. A descrição de um experimento é a topologia da rede, as imagens que são carregadas nas máquinas experimentais e os *scripts* de instalação de *software* e execução dos testes. O processo de criação de experimento tem duas fases: A descrição do experimento e a execução do experimento. Na fase de descrição do experimento é utilizada a linguagem `Tcl` para descrever a topologia da rede e indicar a imagem do sistema operacional a ser carregada na máquina experimental. Depois, o arquivo de descrição da topologia é submetido ao Servidor *Boss*. Posteriormente é executado um processo conhecido em DETER como *swap in*. Durante o *swap in* o Servidor realiza a alocação de recursos, reservando o número de máquinas necessárias para criar a topologia e configurando a rede descrita usando etiquetas de VLAN na rede experimental. Por motivos de segurança, as máquinas são formatadas entre experimentos, pelo que durante cada *swap in* é carregada a imagem especificada pelo usuário no disco da máquina física. Depois, as máqui-

nas são ligadas e são executados os scripts de inicialização que tenha especificado o usuário no arquivo de descrição do experimento. Nesse momento o controle das máquinas é entregue ao usuário que pode executar seu experimento. Em cada máquina experimental é criado um diretório com o nome do experimento, esse diretório é compartilhado via NFS com a pasta do usuário no servidor *Users*, o que facilita o processo de aquisição de dados do experimento.

*M*ontage AGent Infrastructure - MAGI

A infraestrutura MAGI foi criada no DETER para ajudar o processo de descrição dos experimentos e também para criar uma ferramenta que controle a execução do experimento. Em MAGI são construídos modelos de comportamento que serão representados por um conjunto de máquinas participantes do experimento e executarão diversas ações de forma sincronizada. Cada experimento possui um *script* que descreve os conjuntos de máquinas e seus comportamentos, assim como a seqüência de ações e eventos que compõem o experimento. Isso é obtido através de um elemento conhecido como Orquestrador que controla a execução do experimento através da rede de controle do DETER. O orquestrador se comunica com as outras máquinas participantes do experimento para receber notificações e enviar comandos [48].

Um experimento na MAGI possui três componentes principais: Grupos, Agentes e Fluxos de eventos. Os grupos são um conjunto de máquinas que tem uma funcionalidade similar, por exemplo, um conjunto de máquinas clientes HTTP ou um conjunto de *bots* participando de uma *botnet*. A vantagem do uso de grupos, é que resulta fácil variar a escala do experimento, incluindo o excluindo elementos dos grupos participantes do experimento. Na criação de um experimento, um grupo é definido por uma lista de máquinas e um nome para o grupo. O segundo componente são os agentes. Um agente é um código que é executado por um grupo. Um exemplo de um agente é o código de um servidor *web*. Supondo que seja utilizado o servidor web Apache, o código do agente é um código escrito para controlar o início e detecção da execução do serviço, e possivelmente um código para modificar os arquivos de configuração do Apache. Todos os agentes possuem dois tipos de arquivos, o primeiro é um código escrito em *Python* que é utilizado pelo orquestrador MAGI para a ativação do agente e a execução de comandos. O segundo arquivo é um arquivo .IDL que define a interface do agente. Quando é configurado um experimento, deve ser definido o nome do agente, o grupo que executará seu código e a rota do código do agente. O terceiro componente são os fluxos de eventos, os fluxos de eventos definem a seqüência de eventos e comandos do experimento. Para ilustrar com um exemplo simples, considere um conjunto de atacantes que realizam uma inundação a um servidor durante cinco minutos. Nesse caso, o fluxo de eventos é: a configuração dos parâmetros dos atacantes, o início da inundação, um tempori-

zador de cinco minutos e a finalização da inundação. É definido um fluxo para cada grupo definido no experimento. Também é possível configurar o início de determinada ação no momento de finalização de uma ação por parte de outro agente, por exemplo, que os atacantes iniciem a inundação só depois que o servidor web está disponível. Nesse caso, são usados `triggers` para realizar essa sincronização. Cada experimento possui um fluxo especial, chamado de `cleanup`. Esse fluxo é utilizado pelo MAGI para finalizar o experimento.

Os modelos de comportamento são conhecidos como agentes, e podem representar, por exemplo, um servidor web ou um ataque de DoS. Um conjunto de máquinas pode utilizar o mesmo agente, pertencendo a um grupo, no caso do DoS, isso representaria o comportamento de uma *botnet*. Os agentes executam comandos, que são métodos de código e reportam eventos ao orquestrador. O orquestrador instala uns pontos de sincronização conhecidos como disparadores e os disparadores indicam a um agente quando deve executar determinado comando. Assim, existem vários tipos de disparadores como *timeouts* ou eventos que são reportados por outros agentes. De essa forma, o orquestrador pode controlar a execução do experimento e todo o processo é descrito em um script em uma linguagem baseada em YAML (*Yet Another Markup Language*) conhecida como *YAML Agent Activation Language*. A Figura 3.2 apresenta a arquitetura de MAGI. O orquestrador executa um serviço que é o núcleo do MAGI, e abre *sockets* para receber notificações das outras máquinas e enviar comandos. Cada máquina participante do experimento roda um serviço de cliente MAGI e estabelece uma conexão com o orquestrador para receber comandos e enviar notificações. O processo de comunicação do MAGI utiliza a rede de controle do DETER.

3.3.2 *Future Internet Testbed with Security* - FITS

FITS[1] é uma plataforma de teste de código aberto baseado em OpenFlow e no hipervisor Xen [16].

No FITS um conjunto de máquinas físicas é utilizado para instalar o hipervisor Xen e permitir aos usuários experimentadores criarem máquinas virtuais para realizar seus experimentos. Cada instituição participa com máquinas físicas onde são criadas e instaladas as máquinas virtuais dos experimentos. A plataforma de testes divide a rede física em redes virtuais, cada uma com sua própria topologia virtual, pilha de protocolos, regras de encaminhamento e administração. Nesse sentido, o FITS permite a criação de redes virtuais isoladas, compartilhando a mesma infraestrutura [2]. O controle de acesso para o gerenciamento e a criação de redes

[1]FITS é uma rede de testes interuniversitária desenvolvida a partir da parceria de instituições brasileiras e européias. Maiores informações em http://www.gta.ufrj./fits

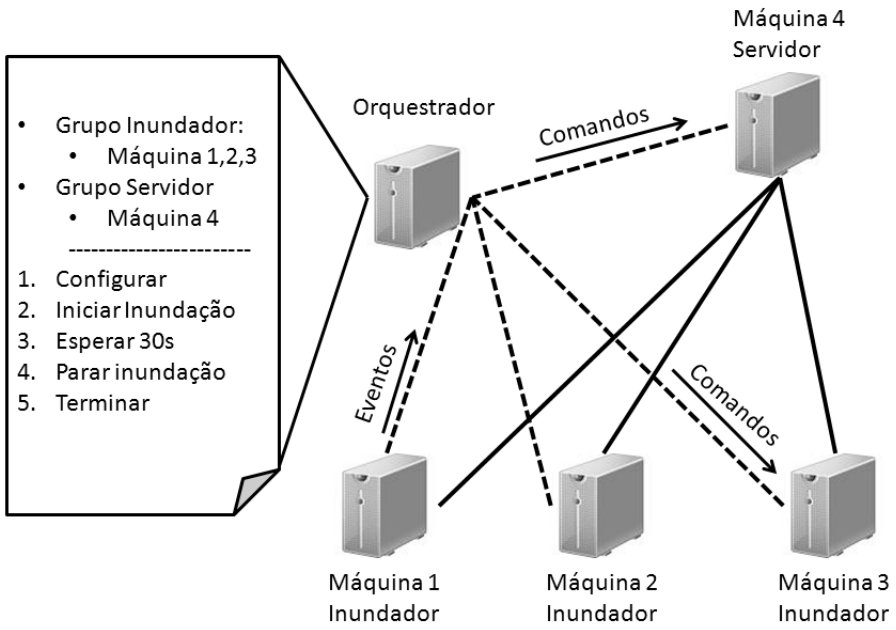

Figura 3.2: Arquitetura da MAGI. O orquestrador possui o *script* que descreve o experimento.

virtuais usa uma plataforma segura, baseada em OpenID e micro controladores seguros [17]. FITS também oferece diferenciação de qualidade de serviço, migração de máquinas virtuais [18] e controle automático de recursos dentro das máquinas físicas [19]. Na plataforma FITS têm sido avaliadas propostas para a Internet do Futuro como: Redes Orientadas a Conteúdo (*Content Centric Networks* - CCN) para distribuição de vídeo sob demanda [20], propostas de controle de acesso para redes definidas por *software* [21] e arquiteturas para prevenção de ataques de DoS usando redes definidas por *software* [22].

FITS permite a participação de instituições como ilhas, onde cada ilha possui suas políticas para experimentação. As máquinas físicas estão ligadas usando túneis *Generic Routing Encapsulation* (GRE) e as ilhas estão conectadas usando túneis *Virtual Private Network* (VPN), o que emula uma Camada 2 entre as máquinas físicas. Essa emulação permite que as máquinas virtuais usem diferentes protocolos de encaminhamento. A arquitetura necessária para que isso seja possível é baseada no funcionamento de redes com o hipervisor Xen e o OpenFlow. A Figura 3.3 apresenta a arquitetura geral do FITS. Uma ilha é formada por um conjunto de máquinas físicas que pertencem a uma instituição. Em cada máquina física são criadas e instaladas máquinas virtuais. Uma facilidade importante oferecida pela plataforma FITS é a migração, pois as máquinas virtuais podem ser migradas dentro de uma ilha assim como entre ilhas. Uma instituição pode criar e gerenciar suas máquinas físicas localizadas em máquinas físicas de outra instituição usando a fer-

ramenta `libvirt` do hipervisor Xen. Entre ilhas é criada uma conexão VPN para isolar o tráfego da plataforma do resto da Internet.

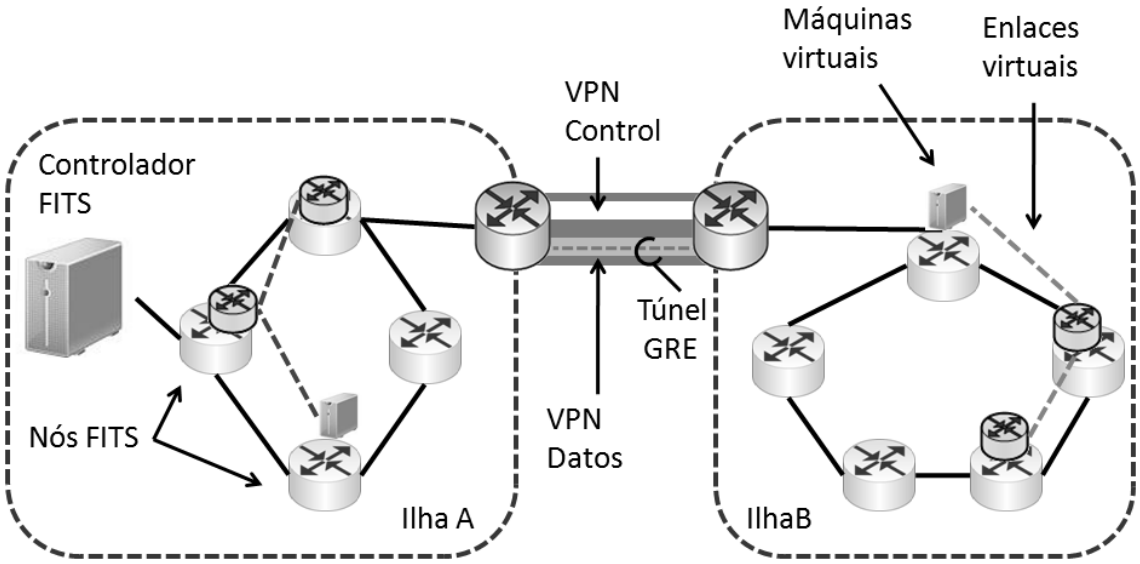

Figura 3.3: Arquitetura de interconexão das ilhas no FITS, na qual são usados dois túneis VPN: um para dados e outra para controle. Os dados são passados em túneis GRE que são encapsulados em VPNs para atravessar a Internet. Fonte [2].

A Figura 3.4 apresenta os componentes do FITS. Uma máquina física em FITS é chamada de Nó FITS e possui o hipervisor Xen para hospedar as máquinas virtuais do experimento, a comunicação entre máquinas virtuais é realizada através do comutador por *software Open vSwitch*. A comunicação entre ilhas é feita usando conexões VPN. A máquina central é chamada controlador do FITS e possui os serviços de gerenciamento da plataforma e também um controlador OpenFlow. Na máquina central, está instalado o Flowvisor, o que permite que outros usuários experimentadores criem e testem seus próprios controladores OpenFlow. O OpenFlow é uma API para o controle de plataformas de encaminhamento por software. O encaminhamento em OpenFlow está baseado na definição de tabelas de regras para os fluxos [49], no qual um fluxo é um conjunto de pacotes que possui características iguais em seus campos de endereço MAC, IP de origem e destino, número de porta da camada de transporte entre outros. Para cada fluxo é definida uma interface de saída. Em uma rede OpenFlow, os comutadores recebem essas regras de um controlador que roda uma aplicação de controle. Assim, os operadores de redes podem instalar proativamente algumas regras padrão nos comutadores e quando chega um pacote que não possui uma regra, o pacote é enviado para o controlador que instala uma nova regra no comutador e o pacote é encaminhado. No FITS, o comutador por *software Open vSwitch* implementa a API OpenFlow nos comutadores e o controlador OpenFlow utilizado é POX. Em seguida é explicado em detalhes o

funcionamento do hipervisor Xen e do OpenFlow no FITS.

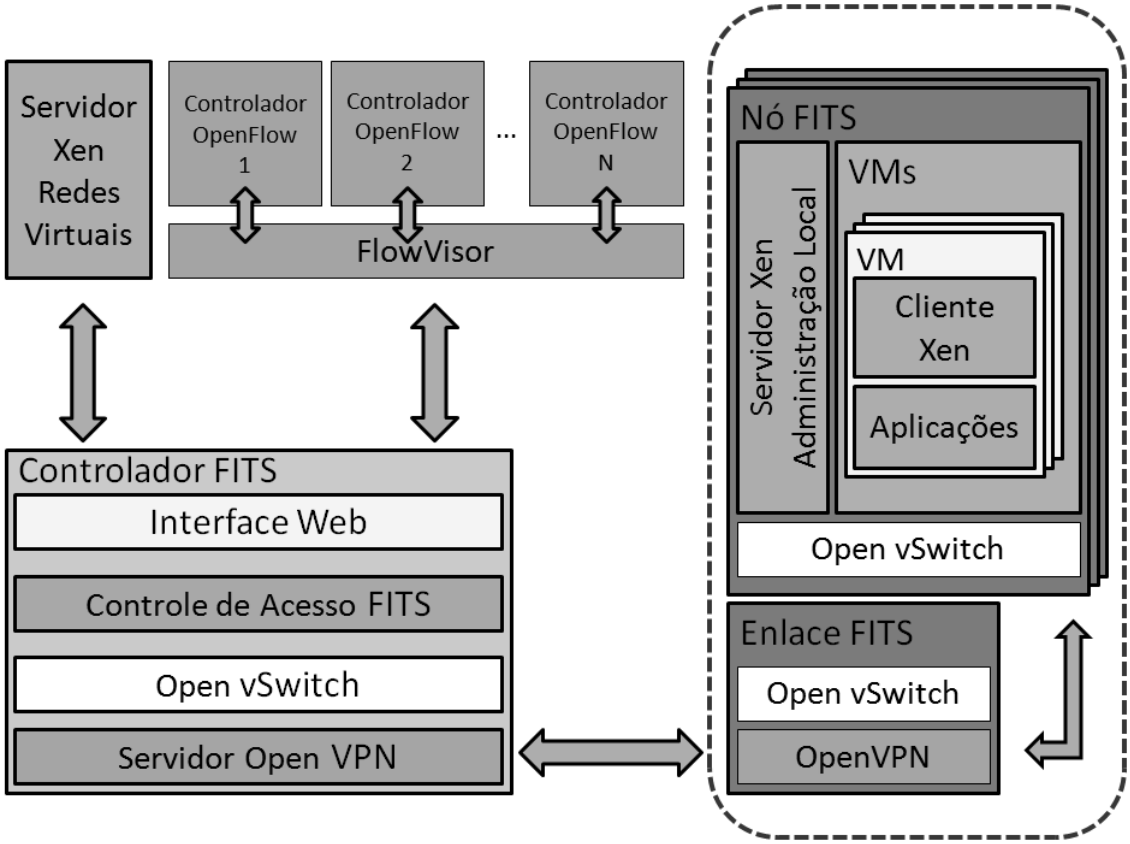

Figura 3.4: Arquitetura do FITS. As máquinas físicas possuem um servidor Xen local para criar máquinas virtuais e a interconexão de máquinas físicas através da Internet é feita usando *Open vSwitch* e OpenVPN. Fonte [2].

O hipervisor Xen no FITS

O Xen no FITS está configurado no modo *bridge*, que é apresentado na Figura 3.5. A figura mostra dois roteadores físicos (Roteador Físico 1 e Roteador Físico 2) com duas interfaces físicas Ethernet cada um (peth0 e peth1). Um túnel GRE interconecta a interface física Ethernet peth1 do Roteador Físico 1 com a interface física Ethernet peth0 do Roteador Físico 2. O enlace virtual mostrado na figura é estabelecido com os mecanismos de entrada/saída do Xen que faz a associação do Roteador Virtual 11 do Roteador Físico 1 com o Roteador Virtual 21 do Roteador Físico 2. O modo *bridge* define uma ponte Ethernet, *bridge* Ethernet, que interconecta as interfaces da máquina física às interfaces das máquinas virtuais [3]. Nesse caso, as interfaces físicas são representadas por interfaces virtuais que são ligadas ponto-a-ponto com as interfaces físicas. Já dentro da máquina física é utilizado um comutador por *software* para realizar o encaminhamento dos pacotes, em FITS isso é feito com *Open vSwitch* que faz o encaminhamento dos pacotes de acordo com o

endereço físico da interface virtual. Quando um pacote chega, a primeira ação tomada é gravar em uma tabela de aprendizagem a porta do comutador e o endereço de origem do pacote, marcando que tal endereço é acessível pela porta de entrada do pacote [3]. Para determinar a rota para o pacote, o *Open vSwitch* consulta ao controlador OpenFlow, que agrega uma entrada na tabela de encaminhamento de fluxos do *Open vSwitch* é o pacote é encaminhado à seguinte máquina física. O processo é repetido até que o pacote chega à máquina física que possui a máquina virtual e o pacote é entregue usando a interface virtual do Xen.

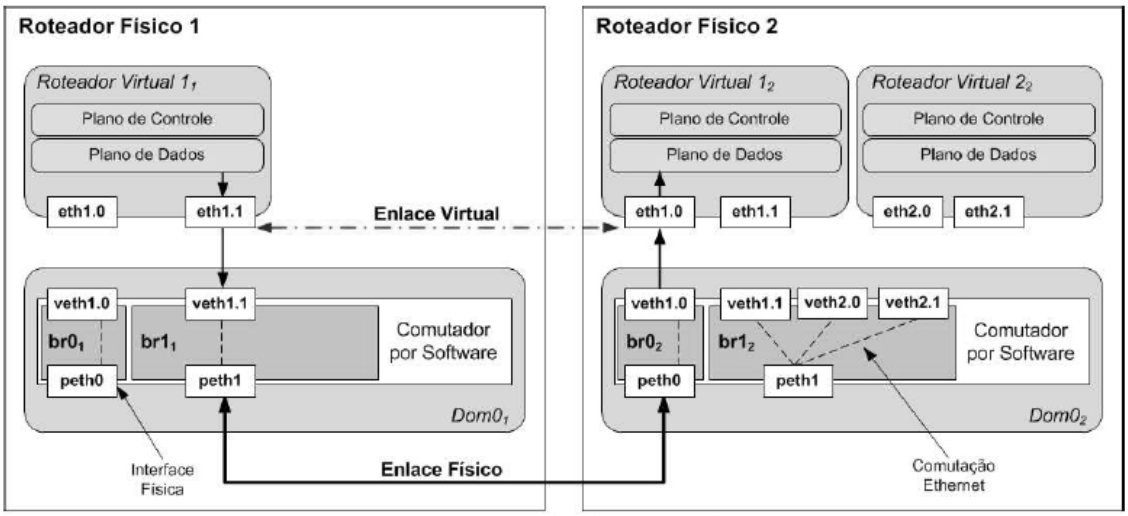

Figura 3.5: Encaminhamento de pacotes de acordo com o modo bridge do Xen. As interfaces virtuais são conectadas ponto a ponto com interfaces de um comutador por *software*. Fonte [3].

A Figura 3.6 apresenta a configuração com o comutador *Open vSwitch*. O *Open vSwitch* agrega a cada pacote uma etiqueta de VLAN para isolar as redes virtuais, dessa forma, cada rede virtual possui uma etiqueta de VLAN diferente e única. Em resumo, um pacote saindo de uma interface virtual passa pelo mecanismo de entrada/saída do Xen, o *Open vSwitch* agrega a etiqueta de VLAN é a rota é consultada com o controlador OpenFlow, finalmente o pacote é encaminhado para o próximo salto usando o túnel GRE. Na máquina física de destino, o *Open vSwitch* retira a etiqueta VLAN e o Xen entrega o pacote para a máquina virtual correspondente.

3.4 Observações Finais

As ferramentas para realizar ataques de negação de serviço na camada de rede e transporte têm evoluído desde simples ferramentas que não realizavam todos os tipos de inundações e que a comunicação entre os coordenadores dos ataques e a *botnet* era em claro, a ferramentas que podem realizar uma grande variedade de inundações e

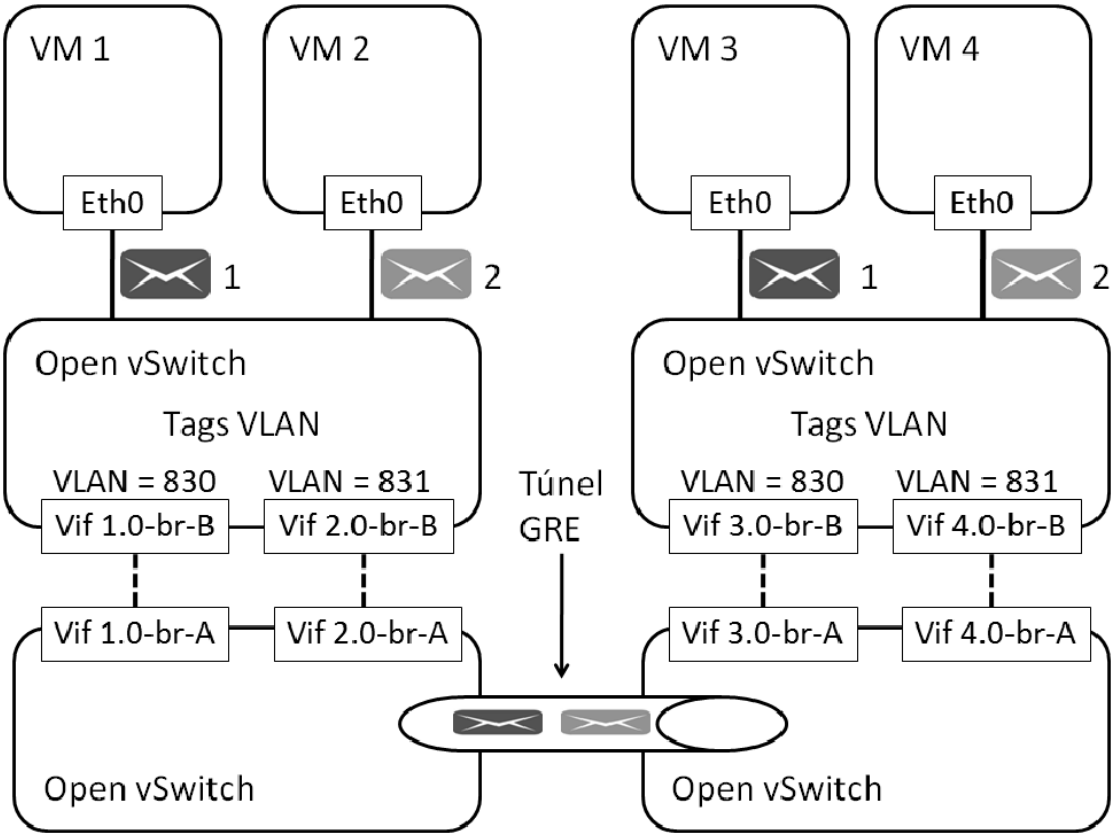

Figura 3.6: Configuração das pontes (*bridges*) e *Open vSwitch* no FITS. A primeira ponte (*bridge*) tem a função de colocar uma etiqueta de VLAN para identificar de maneira única cada rede virtual e a segunda ponte (*bridge*) tem a função de comutar o pacote. Usam-se dois *Open vSwitch* para as duas funções.

que usam canais encriptados para coordenar os ataques. As ferramentas para realizar DoS na camada de aplicação usam menos largura de banda, porque requerem enviar uma quantidade menor de pacotes, mas não usam falsificação de IPs porque primeiro tem que ser estabelecida uma conexão TCP válida.

DETER é um *testbed* para testes de segurança cibernética que é uma iniciativa nacional dos Estados Unidos. Seu objetivo é oferecer uma plataforma de testes para que pesquisadores em segurança possam testar ataques e mecanismos de defesa em um ambiente realista, observando a influencia da escala e o comportamento agregado. Em DETER, os experimentadores fazem uso exclusivo e ilimitado de determinada faixa dos recursos físicos disponíveis do *testbed*. Recebem acesso a um conjunto de máquinas físicas que são conectadas segundo a topologia especificada e podem utilizar as ferramentas do DETER para criar ataques e instalar defesas. Como o *testbed* foi projetado para experimentar com códigos maliciosos, as máquinas experimentais são formatadas entre testes.

FITS é um *testbed* interuniversitário baseado em virtualização que usa o hipervisor Xen e OpenFlow. Em FITS, os experimentadores criam máquinas virtuais para

realizar seus experimentos. A comunicação entre as máquinas físicas que compõem o experimento é feita usando túneis GRE e VPNs e o roteamento é feito utilizando OpenFlow. As instituições participam do FITS provendo máquinas físicas para hospedar as máquinas virtuais dos experimentos. Cada instituição em FITS é considerada uma ilha com suas próprias políticas de administração. Um experimentador pode hospedar e controlar suas máquinas virtuais em outras ilhas usando a biblioteca `libvirt` do Xen. O nó central do FITS está na ilha do GTA e possui os serviços de administração da plataforma, assim como o principal controlador OpenFlow. As outras solicitudes podem criar seus próprios controladores OpenFlow e criar fatias da rede virtual para serem controladas por ele, usando o *FlowVisor*.

Capítulo 4

Mecanismos e Serviços para Experimentação de Ataques de Negação de Serviço

O capítulo anterior apresentou as principais ferramentas para criar ataques de negação de serviço. Foi analisado seu funcionamento, e foi ressaltada a grande evolução das ferramentas que, no início, eram simples e realizavam inundações com opções limitadas e passaram a oferecer facilidades avançadas como, por exemplo: falsificação de endereço IP de origem, diferentes tipos de inundações, uso de vulnerabilidades como a de reflexão de DNS. Foram apresentadas duas plataformas de testes, o DeterLab, a plataforma de maior escala que existe hoje para realizar testes de segurança, financiada pela *National Science Foundation* (NSF) e o *Department of Homeland Security* (DHS) dos Estados Unidos e o*Future Internet Testbed with Security* (FITS), uma plataforma de testes desenvolvida pelo Grupo de Teleinformática e Automação em parceira com outras instituições brasileiras e européias, que usa redes virtuais para realizar experimentos de propostas para a Internet do Futuro. Este capítulo apresenta mecanismos e serviços para experimentação de ataques de negação de serviço usando a Plataforma FITS. A ideia básica é propor e desenvolver facilidades a serem integradas na plataforma FITS de forma a oferecer um serviço de testes de novas propostas de mecanismos de contramedidas a ataques de negação de serviço. Assim, são criadas redes virtuais na Plataforma FITS para configurar e executar experimentos de negação de serviço. As facilidades oferecidas utilizam a infraestrutura MAGI, do DETER para o controle do experimento. Portanto, foram criados módulos para automatizar o processo de criação das redes virtuais, assim como a instalação e a configuração da MAGI. Por fim, as facilidades providas foram integradas na interface web do FITS para facilitar a criação e configuração dos experimentos. Para avaliar a funcionalidade da facilidade de testes oferecida

foi elaborado o teste de um mecanismo para prevenção de ataques de negação de serviço.

4.1 Princípios Gerais do Serviço de Testes de Negação de Serviço

As plataformas de testes oferecem um ambiente mais próximo do real para os experimentos, que os modelos ou simulações, no que se refere a protocolos de comunicação, sistemas operacionais e tráfego. Isso porque utilizam como nós da plataforma de teste os mesmos dispositivos e *software* usados no ambiente real. No entanto, uma importante limitação das plataformas de testes é o tamanho e o custo da sua infraestrutura física. Para experimentos de grande escala seria necessária uma infraestrutura enorme. A plataforma de testes FITS, aborda a limitação da infraestrutura utilizando a técnica de virtualização, que permite compartilhar um nó físico com diferentes nós virtuais. Assim, na plataforma de testes FITS, os usuários experimentadores criam redes virtuais para realizar experimentos de propostas para a Internet do Futuro. A vantagem de usar redes virtuais para esse tipo de experimentos é um melhor aproveitamento da infraestrutura física da plataforma. O serviço de experimentação aqui desenvolvido segue esta ideia e, portanto, são usadas redes virtuais para representar a rede dos experimentos. O uso de MAGI permite trazer para a plataforma FITS a facilidade na criação e controle dos experimentos de negação de serviço.

O processo de configuração e execução de experimentos de negação de serviço possui duas etapas: a configuração do ambiente de teste e a configuração do experimento a ser executado. Na etapa de configuração do ambiente de teste é criada a rede virtual do experimento na plataforma FITS e instanciada a infraestrutura MAGI. A criação da rede virtual inclui a geração e a configuração de imagens virtuais, onde cada imagem representa um nó da rede virtual. Em seguida são configurados diversos arquivos dentro de cada imagem para estabelecer a topologia da rede. Nesta etapa também ocorre a configuração da infraestrutura MAGI do DETER integrada ao FITS para oferecer um controle do experimento mais apurado. Assim, copia-se uma imagem de um disco com o MAGI instalado e modificam-se os arquivos necessários para a correta comunicação dos serviços de cliente e servidor do MAGI para a troca de comandos e eventos. A configuração é dependente da topologia e os nomes das máquinas criadas na rede virtual. Essa primeira etapa é feita pelos *scripts* do serviço de testes de negação de serviço. Os *scripts* recebem como entrada um arquivo feito pelo usuário que tem como parâmetros: os nomes das máquinas usadas, a rota onde são armazenados os discos das máquinas e os parâmetros das

interfaces de rede das máquinas

A segunda etapa é a configuração do experimento a ser executado. Nesta etapa, são configurados os atores do experimento e é definido o fluxo de eventos do experimento de maneira similar a como é feita na plataforma DETER.

O fluxo do experimento contém os eventos que controlam o experimento, tais como: ativação de um servidor web, ativação dos nós atacantes do DoS e ativação de coleta de medidas por alguns clientes usando ferramentas como *Ping*, *Httperf*, entre outras. Essa última etapa, é feita usando a interface web do FITS.

4.1.1 Arquitetura da Plataforma de Testes

A plataforma de testes possui componentes que estão alocados na máquina principal da plataforma FITS e ao momento da criação da rede virtual, são configuradas as máquinas virtuais com a infraestrutura MAGI para o controle do experimento. A Figura 4.1 apresenta a arquitetura da plataforma de testes.

Controlador do Experimento

O controlador do experimento é instalado na máquina principal do FITS. O controlador é responsável pela criação da rede virtual, a configuração das máquinas virtuais com o MAGI e oferece a interface web para a configuração do experimento. A criação da rede virtual recebe como entrada um arquivo .AAL criado pelo usuário que especifica os parâmetros de cada máquina do experimento. Os parâmetros que o usuário deve especificar são: nome da máquina, rota onde é armazenado o disco da máquina, parâmetros das interfaces de rede da máquina e a máquina física do FITS na qual é alocada a máquina. O módulo de criação recebe o arquivo de entrada e cria o arquivo de configuração da máquina virtual do Xen. Para a criação do disco, é copiada uma imagem base que já possui instalado o MAGI. Posteriormente, são copiados ao disco os arquivos necessários para o funcionamento da máquina. Os discos são armazenados em uma pasta compartilhada por *Network File System* (NFS), essa pasta é accessível pelas outras máquinas físicas do FITS. Como foi indicado na Seção 3.3.2, a conectividade entre as máquinas virtuais que estão alocadas em máquinas físicas diferentes é proporcionada pelos *bridges* entre máquinas físicas e o *OpenFlow*. O controle do experimento usa túneis `ssh` para enviar comandos às máquinas virtuais para iniciar o parar os serviços do MAGI e iniciar o experimento definido pelo usuário, também são utilizados para incluir novos agentes criados pelos experimentadores na imagem base do MAGI. A interface web permite a configuração do experimento, também permite criar, editar e eliminar grupos e agentes, assim como manipular os fluxos de eventos. A interface web recebe como entrada, um arquivo de topologia que é criado depois da criação da rede virtual.

O orquestrador é um nó participante do experimento que roda o serviço do servidor do MAGI, qualquer nó pode ser indicado como orquestrador. Quando um nó é ligado, ele envia mensagens a um *socket* específico em cada máquina na topologia para perguntar se já existe um orquestrador. Caso negativo, o orquestrador inicia o serviço de servidor e então o serviço abre *sockets* para escutar as mensagens dos outros nós participando do experimento. O nó orquestrador possui o *script* que descreve o experimento e que pode ser configurado usando a interface web. Quando um experimento é iniciado por um usuário, o controlador do experimento carrega esse arquivo no orquestrador.

Os nós do experimento rodam o serviço de cliente de MAGI. O serviço é iniciado quando um nó é ligado e recebe uma resposta por parte de algum nó da topologia, indicando que ele é o orquestrador. Os clientes do MAGI ficam aguardando os comandos descritos no *script* do experimento e que são enviados pelo orquestrador. Os clientes também notificam ao servidor MAGI os eventos que são gerados durante o experimento. Essas notificações são utilizadas pelo servidor para o sistema de *triggers* descrito na Seção 3.3.1. Com a arquitetura desenvolvida, um usuário usa o módulo de criação de redes para criar ou modificar as redes virtuais a serem usada e usa a interface web para configurar e controlar a execução do experimento.

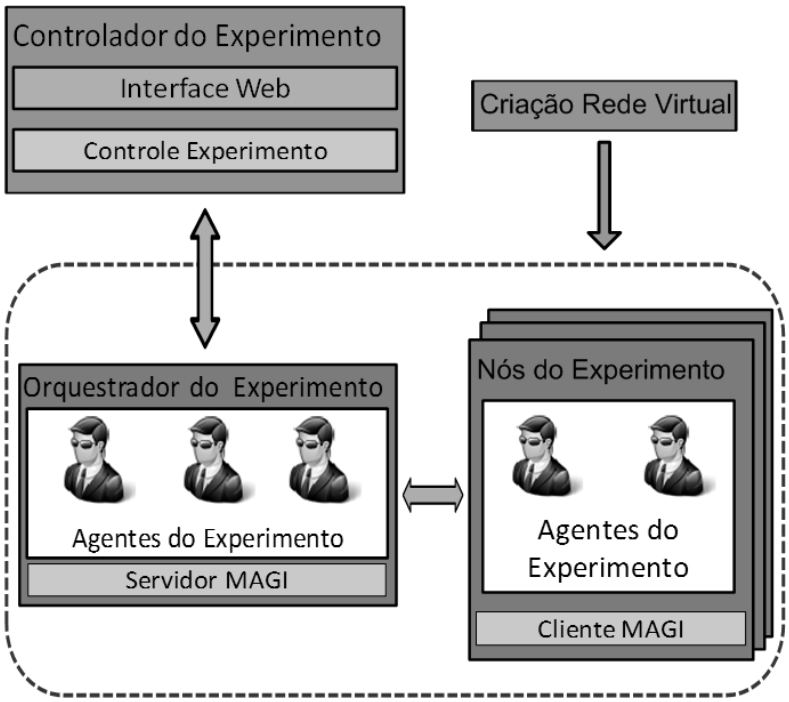

Figura 4.1: Arquitetura da Plataforma de Testes para Experimentação de Ataques de Negação de Serviço. O módulo de criação de rede virtual cria e configura os nós MAGI. O gerenciador do experimento controla a configuração e execução do experimento.

4.1.2 Avaliação de Ferramentas para Ataques de Negação de Serviço em FITS

Esta seção apresenta os resultados das ferramentas e serviços desenvolvidos para realizar ataques de negação de serviço na plataforma FITS com dois objetivos. O primeiro objetivo é de compreender melhor o funcionamento das ferramentas e entender as dificuldades e limitações das ferramentas no ambiente FITS. O segundo objetivo é avaliar a capacidade do ambiente FITS para representar a dinâmica dos ataques de negação de serviço.

A Figura 4.2 apresenta a topologia utilizada para os testes. Todas as máquinas virtuais foram criadas dentro de um servidor com um processador Intel(R) Xeon(R) CPU X5690 3.47 GHz com 16 núcleos e 48 GB de memória RAM, rodando Debian Linux 3.2.0-4-amd64. Todos os elementos apresentados na figura são máquinas virtuais. No caso dos nós, as máquinas possuem uma CPU virtual e 256MB de RAM. No caso do roteador e o servidor as máquinas possuem uma CPU virtual e 512 MB de RAM. Todas as máquinas rodam Debian Linux 3.2.0-4-amd64.

Foram utilizadas duas ferramentas para criar as inundações: a ferramenta *Stacheldraht* e o agente *Flooder* desenvolvido pela equipe do DETER. Foi escolhida a ferramenta *Stacheldraht* porque permite realizar inundações TCP SYN, UDP e ICMP; adicionalmente, a ferramenta permite ajustar o tamanho dos pacotes da inundação. A ferramenta *Flooder* do DETER foi escolhida porque permite realizar inundações TCP SYN e UDP e permite ajustar o tamanho dos pacotes da inundação e a taxa de geração de pacotes. O *Flooder* não realiza falsificação do endereço IP de origem dos pacotes da inundação.

A ferramenta Stacheldraht, descrita na Seção 3.1.5, foi adaptada para funcionar como um agente do MAGI. A adaptação foi feita criando um programa em Python que controla a configuração e execução da inundação. Também foram criados os arquivos que são usados como interface para interagir com o MAGI.

Usando o Nó 1 como atacante e *handler* e o Nó 2 como agente. A segunda ferramenta utilizada foi o agente *Flooder* desenvolvido pela equipe do DETER. O *Flooder* pode gerar inundações TCP SYN, UDP eICMP e que também pode controlar a forma da inundação de três tipos: plana, trem de pulsos, e rampa. Também é possível limitar a taxa de pacotes por segundo que envia cada máquina e a faixa de portas de origem e destino dos. O *Flooder* não utiliza falsificação de IP de origem nos pacotes da inundação.

O objetivo do primeiro teste realizado é determinar a capacidade das máquinas virtuais para gerar um tráfego de ataque de negação de serviço, determinar a capacidade do roteador de encaminhar os pacotes e do servidor em receber uma grande quantidade de pacotes, correspondentes à inundação do ataque de negação

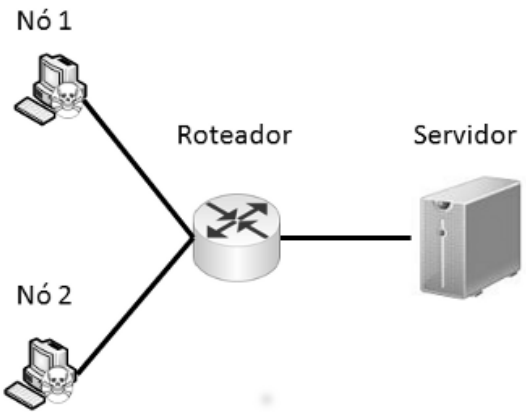

Figura 4.2: Topologia de testes das ferramentas de DoS. Os nós atacantes realizam as inundações ao servidor, através do roteador.

de serviço.

Foram realizados testes usando inundações TCP SYN e inundações UDP

usando dois tamanhos de pacotes, 100 bytes e 1024 bytes. Foram realizadas essas inundações porque são as inundações mais comuns nos ataques de negação de serviço.

No caso das inundações UDP, foram feitos testes com tamanhos distintos de pacotes para avaliar a capacidade do hipervisor Xen de enviar pacotes com tamanhos pequenos e grandes. Em cada teste foi contada a quantidade de pacotes transmitidos, recebidos pelo roteador e recebidos por um servidor em outra rede. A duração dos testes foi de 30 segundos. A duração de cada teste foi de 30 segundos para obter várias medidas da quantidade de pacotes transmitido, recebido pelo roteador e recebido pelo servidor

Os resultados são apresentados usando um tipo de gráfico chamado de *boxplot*. A linha no meio da caixa representa a mediana dos dados obtidos durante os 30 segundos de duração dos testes. As linhas superiores e inferiores da caixa representam os percentis 75 e 25 dos dados. As linhas por fora das caixas representam os dados extremos que não foram considerados isolados, os dados isolados são desenhados individualmente com cruzes. A Figura 4.3 apresenta os resultados de uma inundação SYN realizada com a ferramenta Stacheldraht. Na inundação SYN foram gerados 117000 pacotes, foram recebidos pelo roteador 116000 e foram recebidos 115000 pacotes pelo servidor.

Quase a totalidade dos pacotes foi recebida apropriadamente pelo servidor, o que indica que neste caso o FITS pode representar realisticamente um DoS.

As Figuras 4.4, 4.5 e 4.6 apresenta os resultados de inundações UDP feitas com *Stacheldraht* usando pacotes de 1 byte, 100 bytes e 1024 bytes de *payload* respectivamente, nesse caso a perda é muito maior.

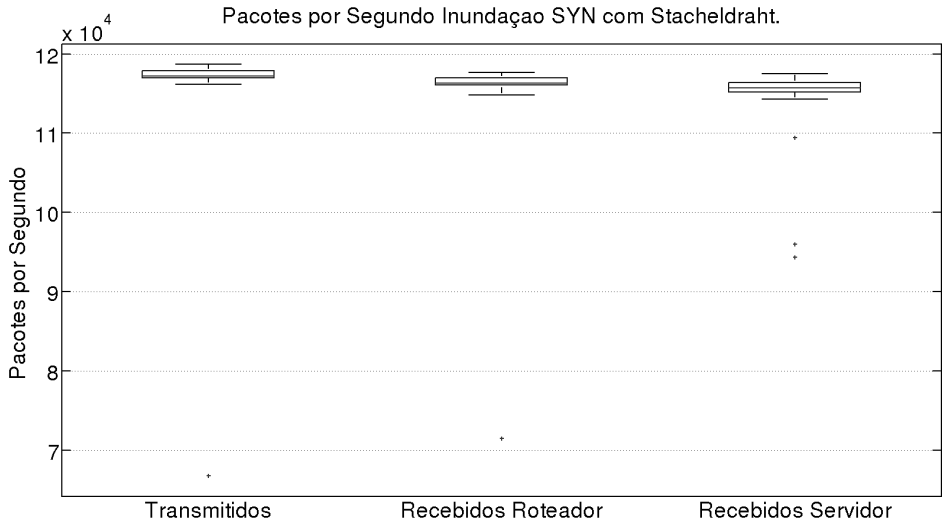

Figura 4.3: Taxa em pacotes por segundo de inundação SYN usando a ferramenta Stacheldraht.

No caso de 1 byte foram transmitidos 130000 pacotes e somente foram recebidos 12000 pacotes. No caso de 100 bytes foram transmitidos 128000 pacotes e somente foram recebidos 10000 pacotes no roteador. No caso de 1024 bytes foram transmitidos 125000 pacotes e recebidos 9900 pacotes no roteador.

Analisando o procedimento de envio de pacotes do FITS, descrito na Seção 3.3.2, vários fatores podem ocasionar a perda de pacotes: i) o canal de entrada/saída do hipervisor Xen, que recebe os pacotes gerados pelas máquinas virtuais e copia para que a máquina física transmita o pacote, ii) o processo de agregar etiquetas de VLAN, usando em FITS para isolar as redes virtuais, iii) o processamento do *open vSwitch* para realizar a comutação de pacotes, iv) o desempenho do controlador OpenFlow, para encaminhar o pacote até a máquina física onde esta alocada a máquina virtual de destino.

A última causa é descartada porque todos os experimentos foram realizados dentro da mesma máquina física. O *Open vSwitch* possui entradas para todas as máquinas virtuais que estão alocadas na mesma máquina física. Nesse caso, um pacote enviado desde uma máquina virtual a outra máquina virtual na mesma máquina física, não precisa passar pelo controlador do FITS é simplesmente é comutado pelo *Open vSwitch*. Já no caso de pacotes enviados entre diferentes máquinas físicas, o primeiro pacote do fluxo chega até o controlador que estabelece a rota no comutador *Open vSwitch*.

O canal de entrada/saída do Xen é um procedimento que copia os pacotes do espaço de memória de cada interface virtual para o espaço de memória da placa de rede. Como é um processo de cópia de memória, seu desempenho é influenciado

pelo tamanho dos pacotes sendo transmitidos.

Os pacotes SYN utilizados não possuem *payload* e apresentam um bom desempenho, mas a diferença entre a inundação UDP de 100 bytes e 1024 bytes é muito pequena para indicar que essa seja a causa na queda de desempenho, embora o tamanho do pacote gere uma pequena diferença. Analisando os *traces* gerados com a inundação TCP SYN foi comprovado que o faixa de portas em que a inundação ocorre é muito pequeno. O que indica que o número de fluxos novos gerados durante uma inundação SYN com Stacheldraht é menor às inundações UDP.

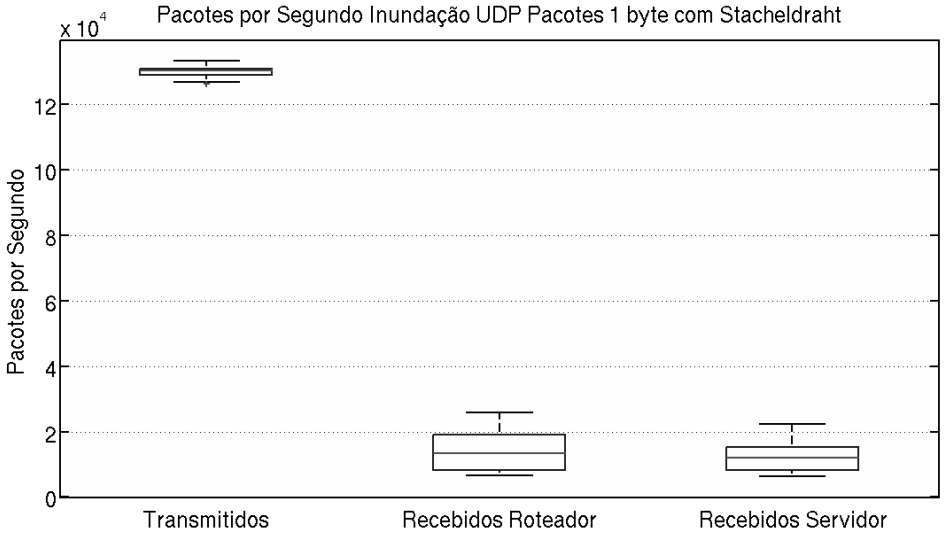

Figura 4.4: Taxa em pacotes por segundo de inundação UDP usando *Stacheldraht*, os pacotes da inundação possuem uma carga útil de 1 byte. Aproximadamente o 90% dos pacotes não é recebido pelo servidor

Para comparar o funcionamento das ferramentas *Stacheldraht* e *Flooder*, os testes de inundações TCP SYN e UDP com 1 e 1024 bytes foram realizados com o *Flooder* e seus resultados são apresentados nas Figuras 4.7, 4.8 e 4.9. Na inundação SYN foram gerados 128000 pacotes e foram recebidos 6000 pacotes. Estudando os *traces* do *Flooder* foi achada uma diferencia no funcionamento da *Stacheldraht*. Observamos que o *Flooder* gera inundações com pacotes que tem uma faixa de portas de origem muito maior, nos testes realizados entre as portas 80 e 10080. O tamanho dos pacotes SYN neste caso foi igual ao tamanho do *Stacheldraht*, o que parece indicar que o problema de desempenho está relacionado com o número de fluxos criados e não o tamanho dos pacotes. No caso da inundação com pacotes de 1 byte e 1024 bytes são obtidos resultados similares à inundação com *Stacheldraht*, com 130000 transmitidos e 10000 recebidos no roteador para o caso de pacotes com 1 byte de *payload* e com 125000 transmitidos e recebidos 9000 pacotes para o caso de 1024 bytes

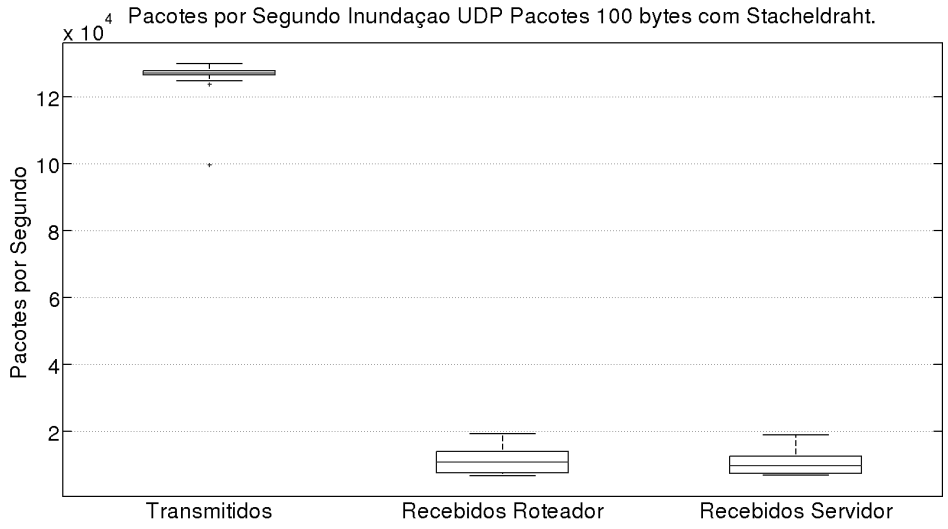

Figura 4.5: Taxa em pacotes por segundo de inundação UDP usando Stacheldraht, os pacotes da inundação possuem uma carga útil de 100 bytes. Aproximadamente o 90% dos pacotes não é recebido pelo servidor

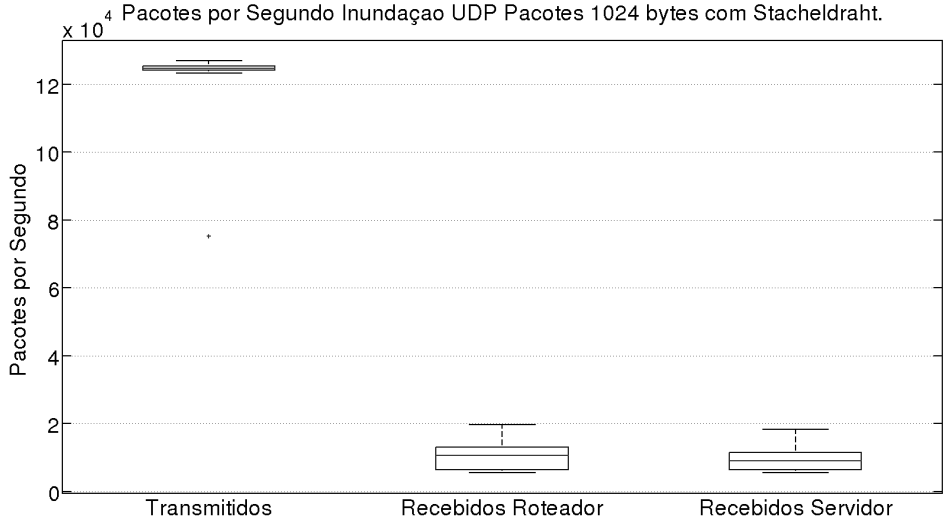

Figura 4.6: Taxa em pacotes por segundo de inundação UDP usando Stacheldraht, os pacotes da inundação possuem uma carga útil de 1024 bytes. Aproximadamente o 90% dos pacotes não é recebido pelo servidor

A ferramenta DETER *Flooder* foi utilizada para analisar com maior detalhe o efeito do número de fluxos gerados durante um ataque de DoS no FITS. A faixa de portas de origem dos pacotes da inundação foi controlado. A vantagem de usar o *Flooder* é que não falsifica o endereço IP de origem o que permite controlar a quantidade de fluxos que são criados durante o DoS. Os resultados são apresentados nas Figuras 4.10, 4.11, 4.12. Nas figuras 4.10 e 4.11 em que a faixa de portas de origem

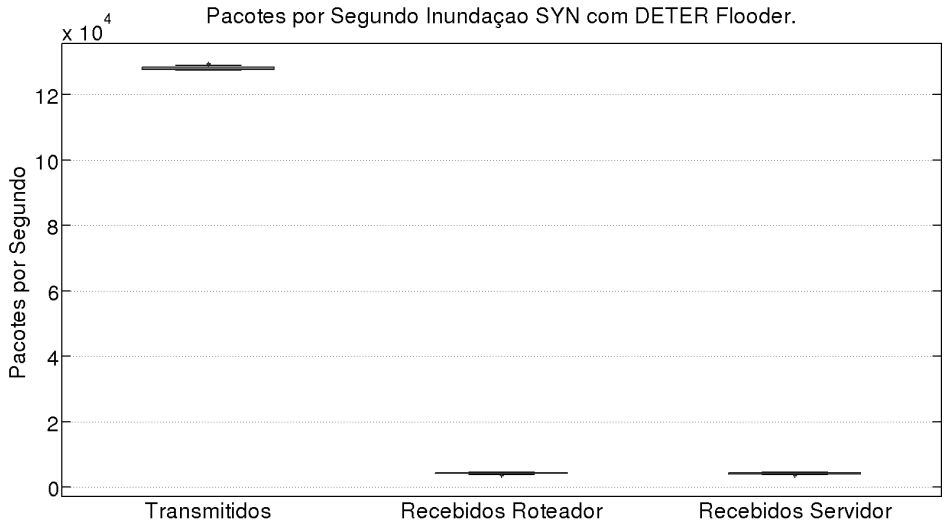

Figura 4.7: Taxa em pacotes por segundo de inundação SYN usando a ferramenta textit DETER Flooderconfigurado para usar umafaixa de 10000 portas na porta de origem dos pacotes.

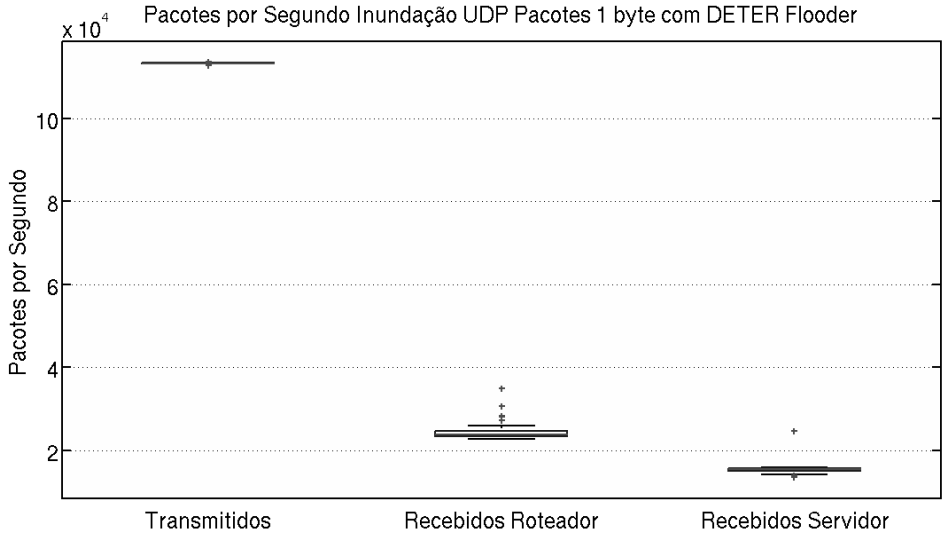

Figura 4.8: Taxa em pacotes por segundo de inundação UDP usando DETER *Flooder*, os pacotes da inundação possuem uma carga útil de 1 byte. Aproximadamente o 90% dos pacotes não é recebido pelo servidor

é de 1 e 100 portas, a quantidade de pacotes recebida pelo roteador e o servidor é muito similar à quantidade de pacotes transmitidos. Já no caso de 10000 portas, apresentado na Figura 4.12 a diferencia é muito maior. Nos três casos foram usados pacotes de 1024 bytes, o que reforça o argumento que o canal de entrada/saída do Xen não ocasiona a queda significativa no desempenho. Os resultados mostram que

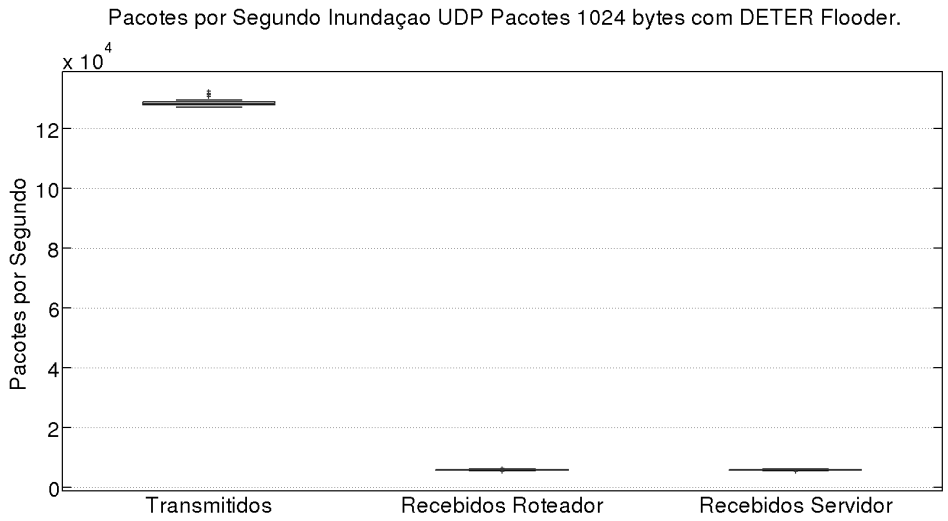

Figura 4.9: Taxa em pacotes por segundo de inundação UDP usando *D*ETER Flooder com uma carga útil de 1024 bytes. Aproximadamente o 90% dos pacotes não é recebido pelo servidor.

o maior efeito no desempenho é a quantidade de novos fluxos criados durante uma inundação, esse processo pode estar influenciado pelo processamento do comutador *open vSwitch* para processar novos fluxos, durante o passo pelas *bridges* do FITS ou no processo de agregar etiquetas de VLAN. O comutador *open vSwitch* quando processa novos fluxos, cria novas entradas em sua base de dados de fluxos, já quando são transmitidos pacotes do mesmo fluxo, simplesmente é incrementado um contador. O primeiro processo é muito mais custoso em termos computacionais e ocasiona as quedas no desempenho, quando é reduzido o número de *bridges* utilizado no envio dos pacotes, é reduzido o número de vezes que essa operação é realizada, o que melhora o desempenho.

A faixa de portas de origem dos pacotes de inundação tem influencia no número de fluxos criados, e esse número de fluxos criados afeta a quantidade de pacotes que podem ser entregados ao roteador e o servidor. A ferramenta DETER *Flooder* foi utilizada para avaliar esse impacto, usando diferentes faixas de portas e diferente quantidade de pacotes transmitidos por segundo. Os resultados são apresentados na Tabela 4.1. Em todos os testes o tamanho dos pacotes foi de 1024 bytes. Se as portas de origem dos pacotes da inundação variam em uma faixa de 10000 portas, é possível entregar pacotes com uma taxa de 8000 pacotes por segundo. Existem perdas na entrega de pacotes quando são usados valores superiores a 8000 pacotes por segundo e 1000 portas. Mesmo usando uma faixa de 5000 portas não resulta possível entregar 16000 pacotes por segundo. Para conseguir essa taxa e taxas superiores foi necessário limitar a faixa de portas a 2500 portas.

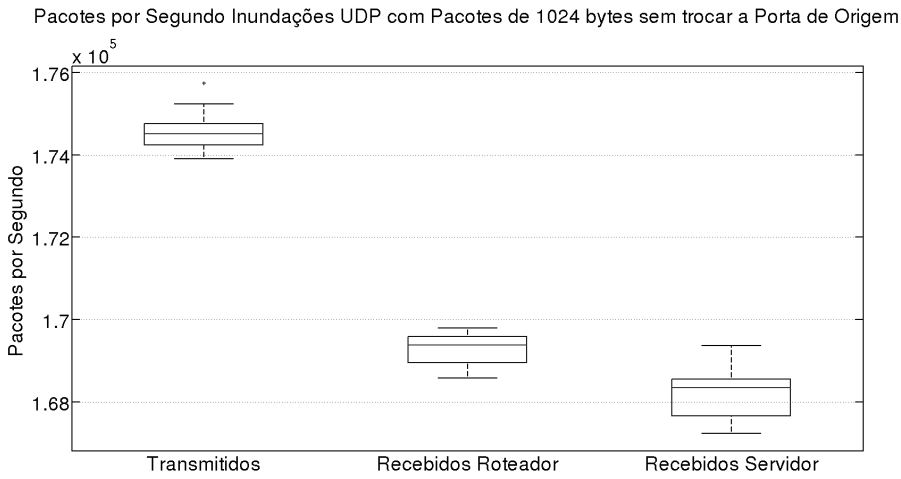

Figura 4.10: Taxa em pacotes por segundo em inundações UDP com carga útil de 1024 bytes usando a ferramenta DETER *Flooder*. Para todos os pacotes é usada a mesma porta de origem. Não há perda significativa de pacotes.

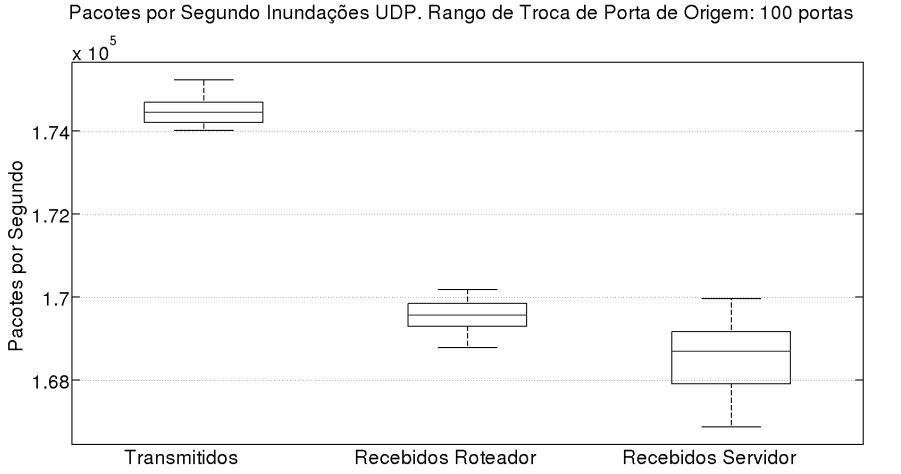

Figura 4.11: Taxa em pacotes por segundo em inundações UDP com carga de 1024 bytes usando a ferramenta DETER *Flooder*. O endereço da porta de origem dos pacotes é alternada em uma faixa de 100 portas. Não há perda significativa de pacotes.

As Figuras 4.13, 4.14, 4.15 apresentam os resultados de testes de inundação com Stacheldraht durante diferentes configurações das etiquetas de VLAN e os *bridges* do FITS. Em todos os casos, foram transmitidos 128000 pacotes. Na Figura 4.13 as duas redes foram configuradas para utilizar a mesma etiqueta de VLAN, a quantidade de pacotes recebidos foi similar aos casos anteriores, com 8400 pacotes recebidos. Já no caso em que não foi utilizada uma etiqueta de VLAN, apresentado na Figura 4.14 o desempenho é melhorado e são recebidos cerca de 21000 pacotes. Se bem a quantidade de pacotes recebidos é muito inferior a quantidade de pacotes recebidos,

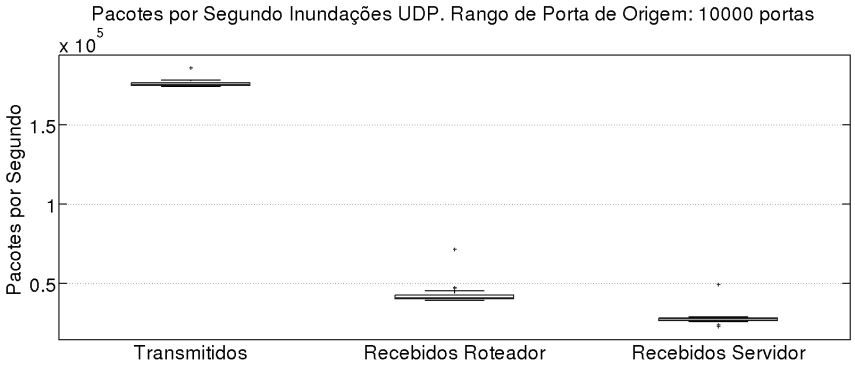

Figura 4.12: Taxa em pacotes por Segundo em inundações UDP com com carga de 1024 bytes usando a ferramenta DETER *Flooder*. A porta de origem dos pacotes é alternada em uma faixa de 10000 portas. A perda de pacotes é de aproximadamente 90%

Tabela 4.1: Taxa de de pacotes transmitidos e recebidos quando é usado o DETER *Flooder* para diferentes faixas de portas de origem dos pacotes da inundação

Faixa de portas de origem	Pacotes Transmitidos	Pacotes Recebidos
10000	16000	10000
10000	8000	8000
5000	16000	9200
2500	16000	16000
2500	32000	32000
2500	64000	64000
2500	125000	125000

representa uma melhora de 250%. Quando não são usadas etiquetas de VLAN, os *scripts* de criação da máquina virtual, não criam *bridges* adicionais para as interfaces virtuais e a interface virtual é diretamente conectada na*bridge* do *open vSwitch*. Nos testes da Figura 4.15, foram eliminados os bridges mostrados na Figura 3.6 e foram manualmente criados *bridges* para conectar diretamente as máquinas virtuais. Os resultados apresentam um melhor desempenho que no caso de não usar etiquetas de VLAN. O que indica que uma causa significativa de desempenho é o processamento de um número grande de fluxos novos a cada *bridge* utilizado em FITS.

Para determinar em qual dos bridges a queda de desempenho é mais significativa foram feitos testes contando o número de pacotes entre os pontos dos mecanismos de virtualização e comutação usados pelas máquinas virtuais para transmitir pacotes segundo foi apresentado na figura 3.6. A Figura 4.16 apresenta os resultados dos testes. A primeira caixa representa o número de pacotes transmitido na máquina virtual, as seguintes caixas correspondem aos pacotes recebidos em cada ponto da

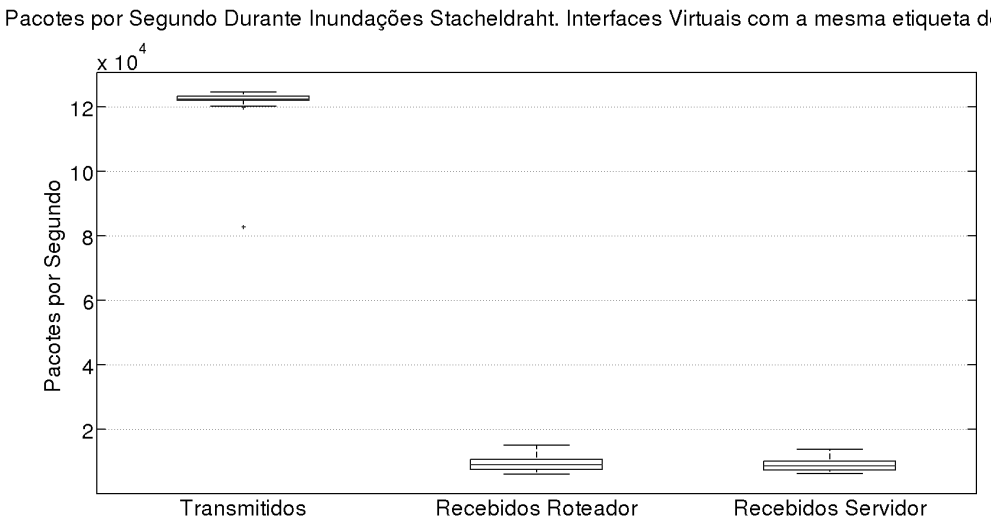

Figura 4.13: Taxa de pacotes por segundo em inundações UDP com carga de 1024 bytes usando a ferramenta Stacheldraht configurado para as redes virtuais usarem a mesma etiqueta de VLAN.

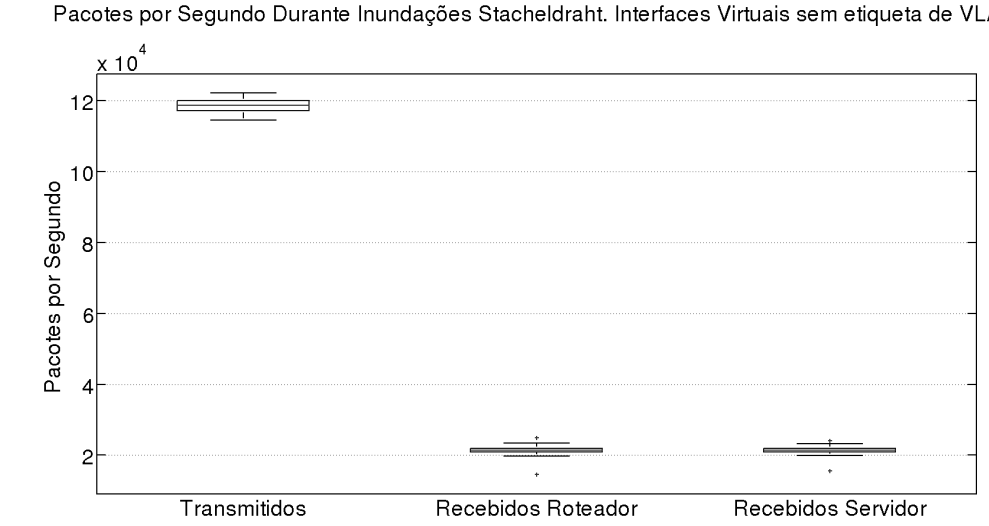

Figura 4.14: Taxa de pacotes por segundo em inundações UDP com carga de 1024 bytes usando a ferramenta Stacheldraht configurado para as redes virtuais não usarem a mesma etiqueta de VLAN. A taxa de pacotes entregues aumenta para 20000 pacotes

comunicação. A segunda caixa chamada `vif` representa o número de pacotes recebido pela interface de *backend* do Xen na máquina física. Pode ser visto que não existe uma queda significativa de pacotes, essa interface de *backend* corresponde ao canal de comunicação I/O do Xen. A terceira caixa representa o número de pacotes processados pelo primeiro *bridge* que tem a função de marcar com etiquetas de

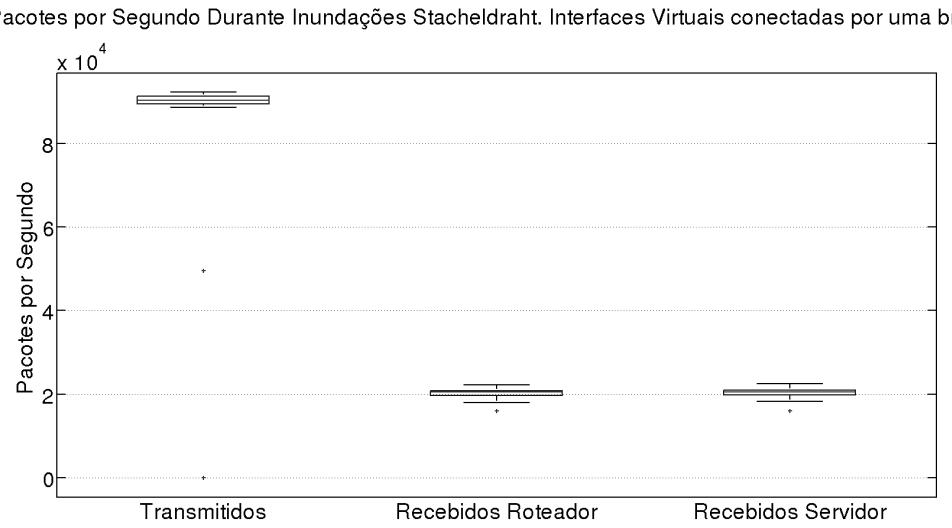

Figura 4.15: Taxa de pacotes por segundo em inundações UDP com carga de 1024 bytes usando a ferramenta Stacheldraht configurado para as redes virtuais se conectarem diretamente por uma única bridge. A taxa de pacotes entregues aumenta para 20000 pacotes

VLAN os pacotes da máquina virtual. Pode ser apreciado que a queda de desempenho mais significativa é presente em esse ponto. Como foi explicado anteriormente quando são criados novos fluxos, o comutador *open vSwitch* tem que realizar um processamento maior à transmissão de novos pacotes do mesmo fluxo, essa carga maior de processamento leva a queda no desempenho. A seguinte caixa corresponde à *bridge* 2 tem a função de comutar o pacote à máquina virtual de destino. Por último as seguintes caixas representam os pacotes contados nas *bridges* e interfaces da máquina virtual receptora.

Para explicar a perda de pacotes quando não é trocada a porta de origem durante o DoS foi realizado um teste similar ao anterior no qual a porta de origem dos pacotes do DoS não foi alterada e foram medidos os pacotes recebidos em cada ponto usado pelas máquinas virtuais para transmitir pacotes. A Figura 4.17 apresenta o resultado dos testes. A primeira caixa representa os pacotes transmitidos pela máquina virtual. A segunda caixa, chamada de `vif` corresponde aos pacotes recebidos pela interface de *backend* do canal de comunicação de I/O do Xen. O número de pacotes recebidos nesse ponto é menor, o que indica uma pequena perda de pacotes por que causa do canal de comunicação do Xen que não consegue transmitir todos os pacotes na taxa que os recebe, mesmo sem criar fluxos novos. Nas caixas correspondentes às *bridges* pode ser observado que não existe maior perda de pacotes. Já no lado do receptor, existe uma nova perda de pacotes com o canal de comunicação de I/O do Xen o que indica que o procedimento de recepção é mais lento que o de transmissão.

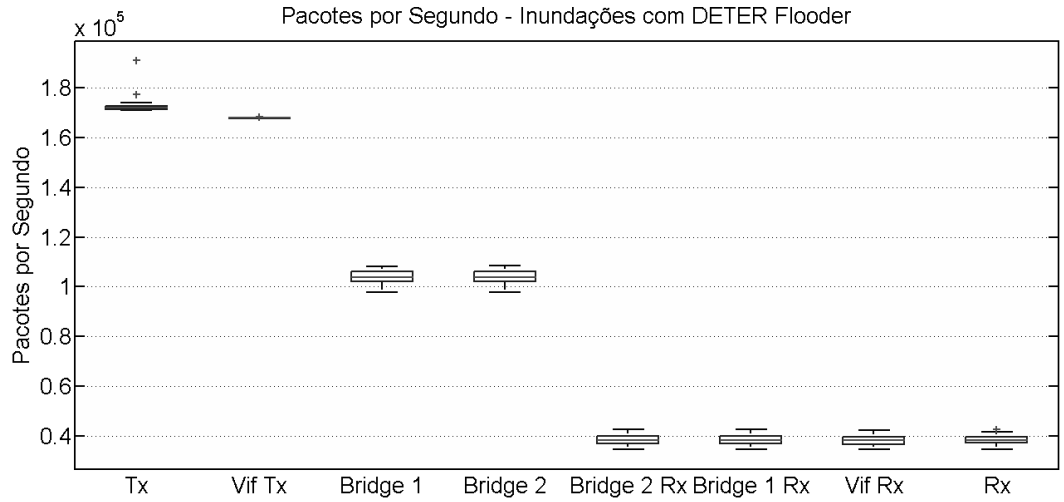

Figura 4.16: Taxa em pacotes por segundo em inundações usando a ferramentatextit Flooder em diferentes pontos do caminho na comunicação entre duas máquinas virtuais. O endereço da porta de origem dos pacotes da inundação é alternado. A maior perda de pacotes ocorre na *bridge* que marca os pacotes com etiquetas de VLAN.

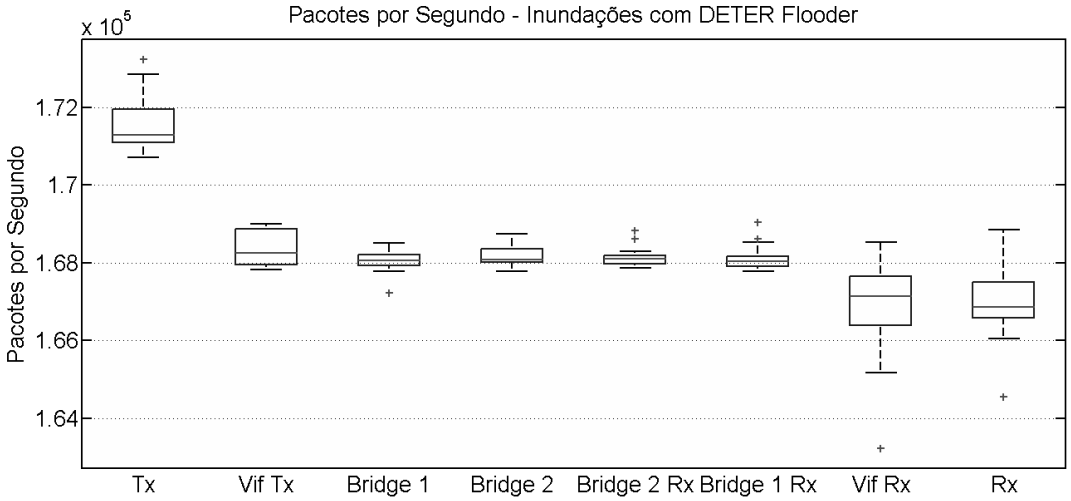

Figura 4.17: Taxa em pacotes por segundo em inundações usando a ferramenta*F*looder em diferentes pontos da comunicação entre duas máquinas virtuais sem trocar o endereço da porta de origem dos pacotes da inundação. Nem todos os pacotes transmitidos pela máquina virtual conseguem ser processados na interface de *backend* (`vif`) da máquina física.

4.2 Caso de Estudo: FlowFence - Um Sistema para Prevenção de Ataques de Negação de Serviço usando Controle de Banda

Esta seção apresenta o FlowFence que é um sistema para prevenção de ataques de negação de serviço usando controle de banda. O FlowFence é usado como estudo de caso para testar o funcionamento do serviço de teste de ferramentas de negação de serviço proposto. O FlowFence é configurado como um agente do MAGI e executado pelos roteadores da rede e um controlador. A seção seguinte detalha o funcionamento do FlowFence e os testes feitos na plataforma.

Os ataques de negação de serviço distribuídos usam um conjunto de máquinas espalhadas geograficamente para gerar requisições falsas ao seu alvo. O objetivo é conseguir enviar umconjunto de mensagensao alvo de forma a consumir recursos da vítima suficientes para obter a negação de serviço. Assim, a detecção perto da origem do ataque não é uma tarefa trivial, porque o número de requisições geradas por cada máquina pode não ser elevado o suficiente para gerar uma anomalia e ser detectada como ataque. Algumas defesas contra negação de serviço usam mecanismos para detectar e bloquear pacotes no destino. Porém, o uso de defesas unicamente no alvo não evita o consumo de recursos na rede causado pelas inundações. Outros tipos de defesas, chamadas híbridas, combinam a detecção do ataque perto do alvo (destino) e também mecanismos para bloquear o tráfego nos roteadores da rede. Logo, é possível evitar a concentração de requisições no alvo e também interromper o consumo de recursos na rede bloqueando os pacotes de ataque [6]. Alguns desses mecanismos podem funcionar mesmo sem conseguir diferenciar os fluxos maliciosos dos legítimos e visam a evitar a inanição. Contudo, essas propostas usam mecanismos que atuam de forma distribuída, ou requerema introduçãode informações adicionais no cabeçalho dos pacotes atravessando a rede, o que diminui seu desempenho.

O sistema FlowFence não requer introduzir um cabeçalho adicional aos pacotes atravessando a rede. FlowFence possui uma arquitetura com roteadores e um controlador centralizado que coordena as ações de defesa à negação de serviço. Os roteadores da rede monitoram o estado de ocupação de suas interfaces de saída e quando detectam um congestionamento notificam o controlador da rede. O controlador envia comandos de controle de banda para todos os roteadores que possuem fluxos encaminhados à interface congestionada. Em resposta aos comandos recebidos, os roteadores limitam a banda de suas interfaces de saída utilizando um sistema de filas. O controle de banda é utilizado para proporcionar uso equitativo dos recursos e diminuir os gargalos na rede. Um protótipo do FlowFence foi implementado na plataforma de testes para experimentação de ataques de negação de serviço que

faz parte do *Future Internet Testbed with Security* (FITS)

4.2.1 Defesas contra Ataques de Negação de Serviço por Inundação

Para dificultar a detecção de um ataque distribuído de negação de serviço um atacante pode falsificar alguns campos dos pacotes enviados uma vez que a Internet só requer o enderço IP de destino para que o pacote chegue ao destino. Porém, esses pacotes são encaminhados por roteadores e nesse processo os roteadores podem marcar os pacotes. Assim, estas marcas no pacote poder ser para rastrear o caminho até a origem dos pacotes que são fontes de um ataque de negação de serviço (*Denial of Service Attack* - DoS). Chen *et al.* propõem um sistema de prevenção de DoS baseado na marcação determinística de pacotes [50]. A marcação determinística marca todos os pacotes encaminhados por um roteador. Usando um Sistema de Detecção de Intrusão (IDS) (*Intrusion Detection System* - IDS) um hospedeiro (*host*) detecta um DoS e envia uma requisição ao seu roteador para marcar deterministicamente os pacotes pertencentes ao ataque. Os pacotes marcados são usados pelo IDS para construir uma árvore que tem como raiz a vítima e as folhas são cada fonte do ataque. A marcação determinística evita que um pacote com marca falsificada percorra todo o caminho do ataque, porque os roteadores no caminho remarcariam o pacote. Contudo, o sistema é vulnerável a ataques que enviem inundações em rajadas com uma duração menor do que o tempo de criação da árvore. Nesse caso, a fonte do ataque não é detectada. O sistema FlowFence proposto pode usar um IDS para identificar tráfego malicioso e enviar uma ordem para descartar esse tráfego, mas se não é possível identificar o tráfego malicioso, garante um uso equitativo entre todos os usuários, evitando a inanição.

Para evitar ataques em rajadas, Laufer *et al.* propõem um sistema de rastreamento que reconstrói o caminho do ataque usando somente um pacote [51]. O sistema usa uma generalização de um filtro de Bloom para criar um *hash.* que inclui todas as assinaturas dos roteadores em um caminho. A generalização é usada para reduzir a probabilidade de falsos positivos.

Um roteador pode dar capacidades, medidas em *tokens*, aos nós da rede para controlar o tráfego recebido. Os nós da rede utilizam certo número de *tokens* para enviar pacotes. Yang *et al.* propõem TVA, uma arquitetura que usa *tokens* para prevenir DoS [52]. Um IDS é utilizado para detectar tráfego malicioso e o roteador não renova os *tokens* dos nós enviando esse tráfego. Em TVA, os receptores devem armazenar informação de estado para cada fluxo e as fontes devem modelar o uso das capacidades para solicitar a renovação. Esse armazenamento de estado e modelagem de capacidades limita a escalabilidade da proposta. A ação de controle da TVA é a

não renovação de *tokens*. Assim, um atacante identificado pode continuar criando inundações até que seus *tokens* acabem. O sistema FlowFence opta por usar o controle de congestionamento pra evitar a negação de serviço e desta forma evita o uso de *tokens* que requerem procedimentos de negociação e renovação.

NetFence [53] é uma arquitetura que cria uma malha fechada de controle de congestionamento para reduzir o impacto de um DoS. Nesta arquitetura, roteadores detectam congestionamento em suas interfaces de saída e enviam informação autenticada de retroalimentação aos roteadores pertos da fonte do ataque para reduzir o impacto do DoS. Para trocar essas informações de uma forma segura os autores propõem adicionar um campo entre oscampos das camadas IP e TCP [54]. Embora a proposta evite a inanição dos usuários legítimos, a arquitetura propõe a modificação da pilha TCP/IP, o que limita sua aplicabilidade. Além disso, a informação de retroalimentação está presente em cada pacote enviado e usa um *hash* criptográfico. O *hash* tem que ser processado para cada pacote enviado. O FlowFence não requer modificações na pilha de protocolos e evita a sobrecarga da inclusão de um novo cabeçalho, com o uso de um controlador que se comunica com os comutadores da rede para controlar a banda.

Mattos e Duarte propõem QFlow [49], um mecanismo para garantir qualidade de serviço em redes virtuais usando controle de recursos em redes OpenFlow. O mecanismo é baseado em Xen e OpenFlow, o controle de recursos de rede é feito usando um sistema de filas em OpenFlow. Cada máquina virtual configura seus parâmetros de QoS e eles são lidos por uma aplicação no hipervisor, essa aplicação associa esses parâmetros a um conjunto de filas para tentar garantir o mínimo solicitado por cada máquina virtual. Em caso de ainda dispor de recursos físicos disponíveis, os recursos são distribuídos de acordo aos parâmetros configurados por cada máquina virtual. O sistema FlowFence proposto provê um mecanismo para detectar congestionamento nas interfaces de saída dos roteadores e também provêum uso equitativo para todos os fluxos sendo encaminhados no roteador.

4.2.2 Objetivos do Sistema FlowFence

Esta seção define o modelo do atacante e as condições do ataque e do tráfego de ataque. Para estas condições especifica-se os objetivos a serem atingidos.

Modelo do Atacante

Ataques de inundação: Assume-se que o atacante gera ataques de inundação de pacotes que visam esgotar recursos debanda passante, comoa banda passante dos enlaces ou a capacidade de encaminhamento dos roteadores. No modelo, não são considerados atacantes que geram negação de serviço por exploração de vulnerabili-

dades da camada de aplicação ou mensagens com valores inesperados no cabeçalho. Assume-se também que o atacante não consegue comprometer o controlador ou os comutadores.

Tráfego malicioso idêntico ao legítimo: Assume-se que o atacante pode gerar inundações com tráfego que imita tráfego legítimo e, portanto, não é possível diferenciar os fluxos maliciosos dos fluxos legítimos.

Os objetivos do sistema de defesa

Uso equitativo de recursos: na ausência de identificação de tráfego malicioso, a proposta garante o uso equitativo dos recursos da rede para os usuários, o que reduz o impacto do ataque de negação de serviço. Algumas propostas visam punir fluxos que apresentam grande consumo de banda, para garantir recursos para os fluxos pequenos [55]. Mas na ausência de um IDS, é possível que alguns de esses fluxos grandes sejam de usuários legítimos.

Não requer modificações nos protocolos usados: o sistema não deve requer modificações nas pilhas de protocolos usadas nas redes dos usuários. O sistema não deve incluir campos adicionais no cabeçalho nos pacotes encaminhados na rede.

Rápida resposta: o sistema deve reagir rapidamente às condições de negação de serviço detectadas.

4.2.3 Arquitetura do FlowFence

O sistema FlowFence proposto consiste de roteadores que monitoram suas interfaces de saída para detectar condições de congestionamento e um controlador centralizado que mantém informações sobre o estado global da rede, para enviar comandos de controle de banda a todos os roteadores para limitarem a banda dos fluxos que estão no enlace congestionado. Os roteadores ao receber comandos do controlador criam filas para cada fluxo sendo encaminhado. Nesse caso, um fluxo é definido como o conjunto de mensagens que possui a mesma dupla de IP de origem e destino. O roteador divide equitativamente a banda disponível na interface de saída entre as filas criadas e cada fluxo ocupa uma fila. Quando termina o ataque, o roteador reporta o novo estado ao controlador e termina o controle de banda. Para prevenir que um atacante gere falsos alarmes a comunicação entre o FlowFence e o controlador usa um canal seguro. A arquitetura do FlowFence é apresentada na Figura 4.18. Os componentes do funcionamento do FlowFence são apresentados na Figura 4.19. Dentro da plataforma de testes, o código do FlowFence foi implementado como agentes do MAGI que rodam nos roteadores e no controlador da rede. De essa forma, é possível controlar a execução do experimento usando as ferramentas descritas anteriormente, o que facilita os testes realizados.

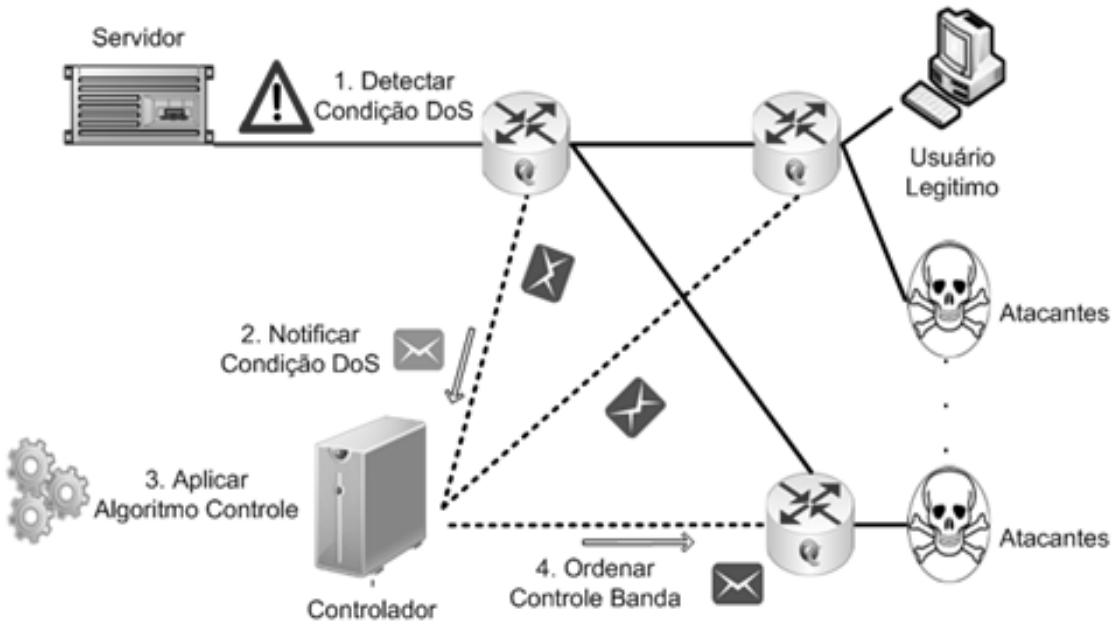

Figura 4.18: Arquitetura de FlowFence: sensor de ataque, comutadores configurados com filas e controlador cetralizado

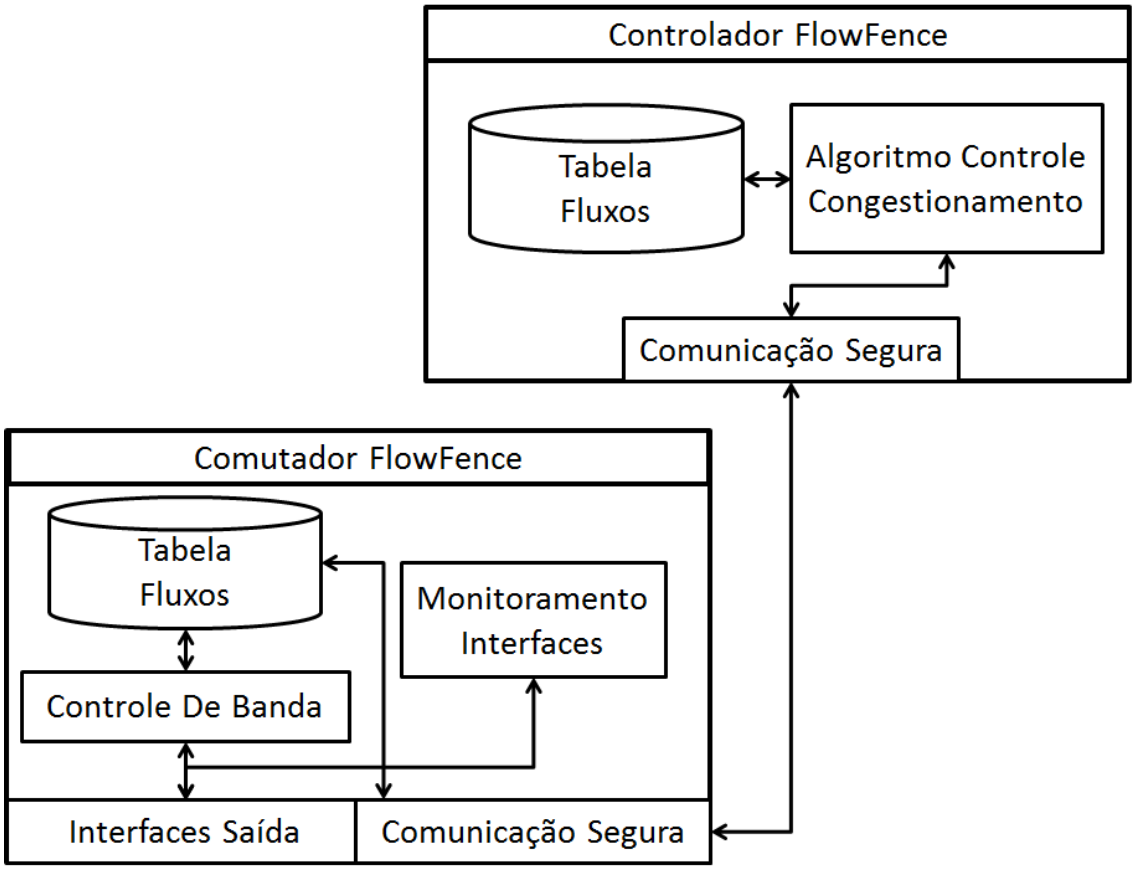

Figura 4.19: Componentes do FlowFence.

Detecção de Condição de Negação de Serviço - Os ataques de negação de serviço por inundação geram altas taxas de ocupação das interfaces de saída dos roteadores. Em muitos casos os roteadores possuem enlaces bem provisionados e em condições normais a ocupação média da interface é menor ao 95% [53]. Os roteadores configurados com FlowFence periodicamente capturam amostras da quantidade de bytes transmitida por segundo e utilizam uma janela deslizante exponencial para medir a ocupação media do canal. Quando a ocupação do canal supera o 95% da capacidade total, uma notificação de congestionamento é enviada ao controlador. O uso de uma janela deslizante permite evitar alarmes falsos por possíveis picos de curta duração que podem ser condições normais da rede. Foi utilizada uma janela exponencial, por dar um peso maior às últimas amostras tomadas, detectando rapidamente uma condição de DoS.

Notificação ao Controlador - O controlador da rede mantém conexões seguras com os roteadores utilizando uma interface diferente da utilizada para o encaminhamento dos dados dos usuários. Quando uma condição de DoS é detectada o roteador envia uma mensagem que contém: O identificador do roteador, o endereço de rede do enlace congestionado e uma marcação de tempo. A marcação de tempo é usada para evitar possíveis ataques de repetição.

Algoritmo de Controle - Quando o controlador recebe uma notificação de congestionamento, verifica em sua tabela de fluxos, quais roteadores possuem fluxos que usam o enlace congestionado e envia um comando para encaminhar os fluxos às filas de controle de banda criadas pelo roteador. O controle de banda brinda a mesma quantidade de banda para cada fluxo encaminhado.

Controle de Banda - Um roteador cria filas para cada um dos fluxos encaminhados e aloca uma fatia igual da banda total para cada fila. A banda é controlada até que a ocupação da interface de saída é menor aos 80 % da capacidade total da interface. Quando a ocupação do canal é menor que 80% uma nova notificação é enviada ao controlador e o controle banda é terminado.

4.2.4 Implementação e Avaliação

O protótipo de FlowFence foi implementado dentro da plataforma de testes para experimentar ataques de negação de serviço que faz parte do *Future Internet Testbed with Security* (FITS) [2]. Usando o módulo de criação de redes virtuais foi criada uma rede virtual com máquinas virtuais gerenciadas pelo hipervisor Xen 4.1.4. Os roteadores da rede rodam o agente *Application Switch* criado para o MAGI, o agente utiliza o roteador programável Open vSwitch[1], que provê o mecanismo para implementar as filas para o controle de banda. O encaminhamento dos fluxos para as

[1]http://www.openvswitch.org/.

filas é controlado por OpenFlow. O controlador da rede roda o agente *FlowFence* que utiliza o controlador de OpenFlow POX. Os agentes *Application Switch* monitoram as interfaces de saída, notificam ao controlador sobre o estado do congestionamento e aplicam o controle de banda. O controlador roda o agente *FlowFence* que mantém informações sobre o estado da rede, recebe as notificações de congestionamento e modifica os fluxos dos roteadores para encaminhá-los nas filas de controle de banda. Para criar os ataques de negação de serviço foi usado o agente *Flooder* do MAGI.Adicionalmente, foram criados agentes para rodar as ferramentas `Iperf`[2], *Httperf*[3] usadas pelos clientes para realizar as medidas de avaliação de desempenho do protótipo. Um servidor foi utilizado para hospedar as máquinas virtuais usadas nos testes. O servidor possui um processador Intel(R) Xeon(R) CPU X5690 3.47GHz com 16 núcleos e 48GB de memória RAM e executa Debian Linux 3.2.0-4-amd64. As máquinas virtuais são configuradas com um CPU virtual, 256MB de ARAM e executam Debian Linux 3.2.0-4-amd64, a banda das interfaces virtuais foi limitada a 100Mb/s.

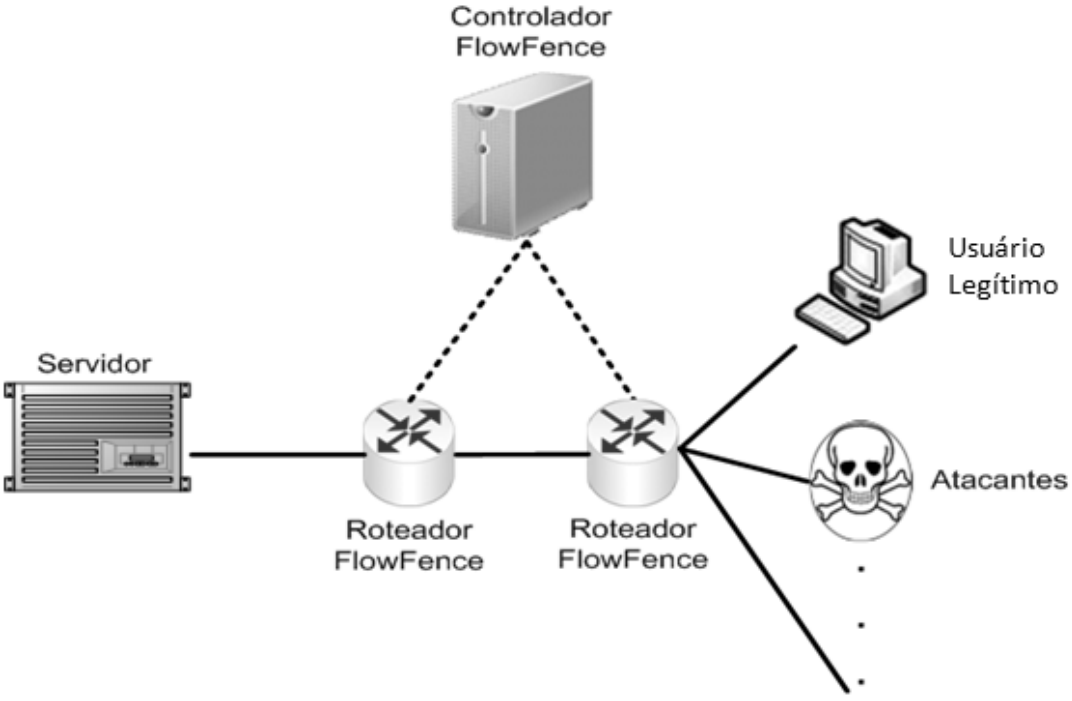

Figura 4.20: Topologia utilizada durante os testes. A rede virtual foi criada usando o módulo de criação de redes da plataforma.

Foi utilizada uma topologia *Dumbbell* apresentada na Figura 4.20, a topologia está composta por 2 roteadores, 1 controlador, 10 dispositivos finais e 1 servidor. Os 10 dispositivos finais pertencem a uma mesma LAN e se conectam a um dos

[2]http://iperf.sourceforge.com/
[3]http://www.hpl.hp.com/research/linux/httperf/

roteadores. O servidor está conectado ao segundo roteador e os dois roteadores são conectados usando uma interface virtual de 100Mb/s. Uma rede separada é usada para comunicar os roteadores com o controlador. Dos 10 dispositivos finais, um deles é usado como cliente do servidor e o número de atacantes é alterado segundo os testes a continuação. Tanto o cliente como os atacantes encaminham suas requisições e inundações ao servidor.

O primeiro teste avalia a eficácia do protótipo em assegurar o uso equitativo dos recursos da rede para o cliente. Um cliente utiliza `Iperf` para avaliar a banda disponível até o servidor é ao mesmo tempo um número variável de atacantes inicia uma inundação. O número de atacantes é controlado modificando o *script* do experimento para incluir ou excluir nós do grupo de atacantes. Cada atacante gera uma inundação de 300 Mb/s, usando pacotes UDP de 1024 bytes destinadas à porta 80 do servidor. O `Iperf` foi executado durante 30 segundos. A Figura 4.21 mostra os resultados do teste. É possível perceber que o cliente, mesmo durante uma inundação que supera várias vezes a capacidade dos enlaces, consegue atender uma taxa muito perto da taxa esperada.

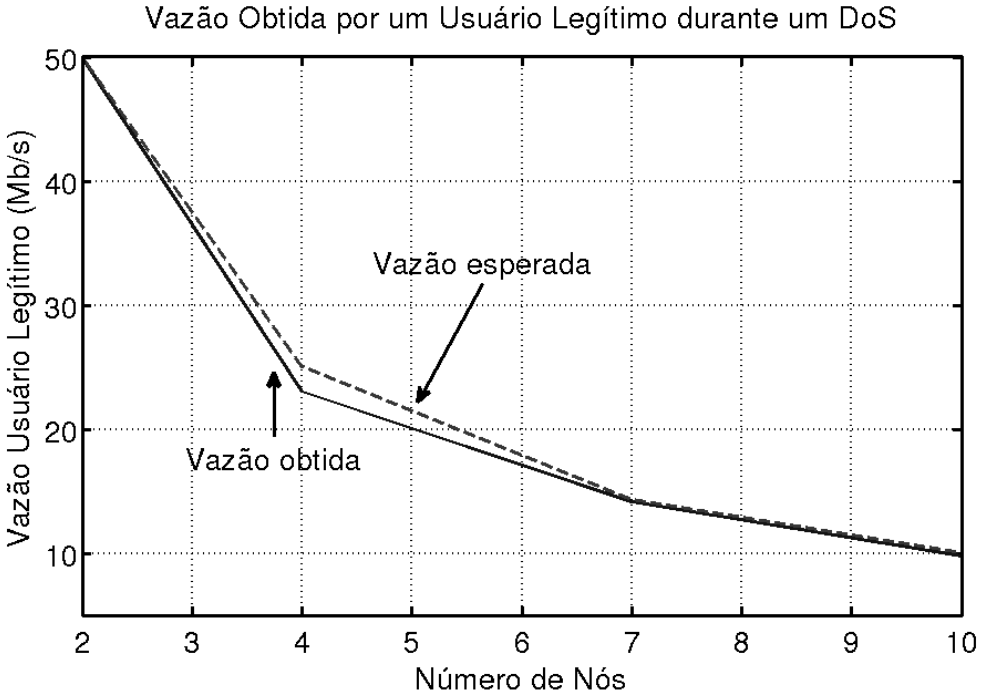

Figura 4.21: Vazão de um usuário legítimo durante um ataque de negação de serviço. A vazão de um usuário legítimo diminui proporcionalmente com a quantidade de usuários. A diminuição proporcional é o resultado da alocação equitativa dos recursos disponíveis da rede estabelecida pelo algoritmo de controle proposto.

O segundo teste avalia a capacidade do FlowFence para evitar a inanição dos usuários legítimos na presença de um ataque de DoS. Neste teste o cliente executa

o agente de `Httperf` para testar o tempo de resposta de um servidor HTTP. O cliente tenta realizar conexões a uma taxa de 100 conexões por segundo, durante 30 segundos. O parâmetro *timeout* da tentativa de conexão foi estabelecido em 7 segundos. Durante os 30 segundos os atacantes realizam uma inundação com pacotes UDP de 1024 bytes na porta 80, que é a porta onde o servidor escuta as requisições HTTP. Cada atacante cria 40000 pacotes por segundo, gerando um fluxo de 320Mb/s, e o número de atacantes foi variado a cada teste até alcançar um volumo de inundação de 3 Gb/s. A Figura 4.22 mostra o resultado do teste. Sem o uso de FlowFence o cliente não consegue estabelecer uma conexão para volumes maiores aos 1500 Mb/s, já que o tempo de tentativa de conexão excede os 7 segundos. Com FlowFence, os resultados mostram um crescimento baixo no tempo de resposta do servidor com o incremento no número de clientes. O tempo de resposta do servidor é de 1 segundo com um volume de inundação de 3 Gb/s, o que comprova que a inanição é evitada.

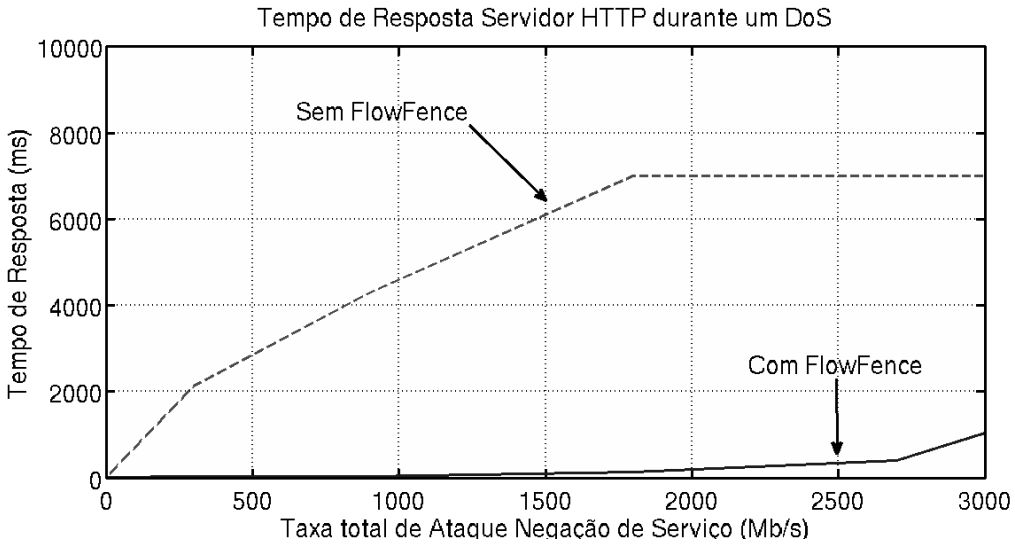

Figura 4.22: Tempo de resposta de um servidor HTTP durante um ataque de negação de serviço.

O terceiro teste apresenta o tempo de reação do FlowFence, entendido como o tempo que tarda o sistema em dar um uso equitativo do enlace a um usuário legítimo depois do inicio de uma inundação. Foram usados *triggers* temporizadores para controlar a execução do experimento. Novamente, foi utilizado o `Iperf` para medir a banda disponível para um usuário legítimo, quando 9 atacantes geram uma inundação de 3Gb/s. A duração do `Httperf` foi de 30 segundos e da inundação de 35 segundos. A Figura 4.23 apresenta os resultados do teste. A inundação é iniciada no segundo 2 do teste, o usuário legítimo perde sua banda disponível no segundo 4, e 4 segundos depois FlowFence consegue dar uma fatia equitativa da banda a ele.

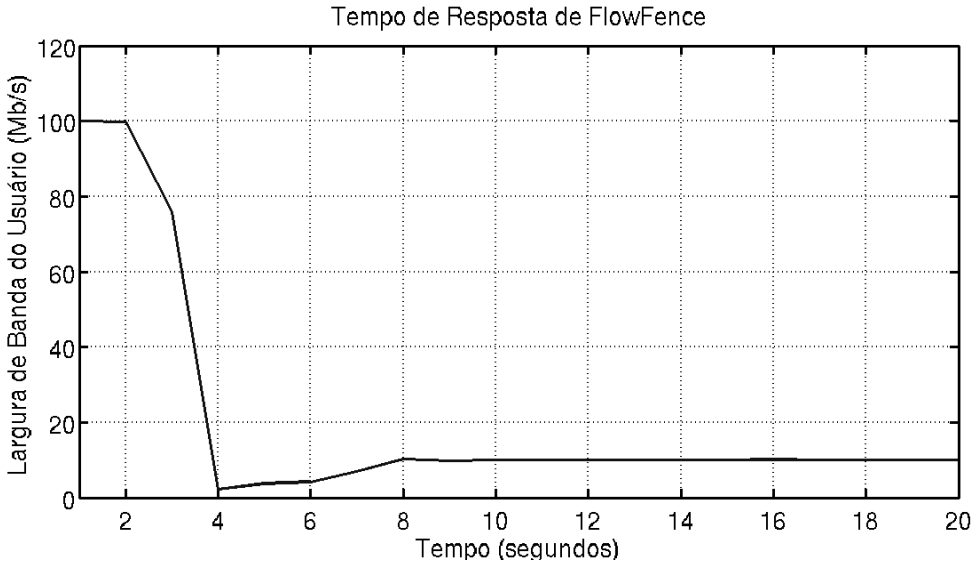

Figura 4.23: Tempo de reação do FlowFence para limitar a banda das interfaces congestionadas. `Httperf` foi usado para medir a banda disponível para um usuário.

4.2.5 Análise do FlowFence

A segurança do mecanismo FlowFence é proporcionada pela comunicação segura entre os roteadores e o controlador, que é feita com uma rede distinta à rede utilizada para encaminhar pacotes dos usuários. Desta forma, um atacante não consegue falsificar uma notificação de congestionamento que poderia prejudicar aos usuários legítimos. Se um roteador fosse comprometido, ele poderia notificar ao controlador de um possível DoS, o que dispararia erroneamente o mecanismo de controle de banda em vários roteadores. Se o controlador fosse comprometido, toda a rede estaria comprometida.

O sistema FlowFence pode oferecer, na ausência de um sistema de Detecção de Intrusão, uso equitativo dos recursos da rede por parte de usuários maliciosos ou legítimos. Se um IDS fosse utilizado, o FlowFence poderia evitar a inanição dos usuários legítimos. Nesse caso, o IDS detecta o fluxo malicioso e controlador ordena seu bloqueio.

No entanto, o FlowFence sozinho não é um mecanismo de defesa eficiente, pois ele não consegue limitar um ataque de negação de serviço distribuído quando um atacante tem a capacidade de criar centenas de fluxos maliciosos, levaria ao usuário a usar uma banda muito pequena da rede, o que poderia ocasionar que muitas transações não fossem concluídas com sucesso. Os resultados dos testes indicam que o FlowFence é muito ágil em reagiar a um ataque. Mesmo em inundações com um alto volume de tráfego, FlowFence consegue reagir em poucos segundos e evitar a inanição dos usuários legítimos. Uma proposta que usasse um mecanismo de controle ponto a ponto, com comunicação entre os roteadores poderia tardar

mais tempo até que o controle de banda chegasse perto da origem do ataque. A arquitetura de comunicação entre os roteadores e o controlador evita a necessidade de incluir informação de retroalimentação para o controle de banda, incluir essa informação implicaria uma modificação dos protocolos convencionais utilizados nas redes de dados. Adicionalmente, essa informação deve estar autenticada para evitar sua falsificação, o que representaria realizar operações criptográficas salto, a salto, para um determinado conjunto de pacotes.

Uma arquitetura com um controlador com informações sobre o estado da rede permite a implantação de algoritmos de controle fim-a-fim para prevenir ataques de negação de serviço e outras situações de congestionamento, FlowFence, usa um algoritmo para dar alocação equitativa a todos os fluxos, mas alternativas podem ser utilizadas [56]

Para a implementação do sistema FlowFence, é necessária a inclusão de um controlador na rede, e de enlaces adicionais que comuniquem ao controlador com os roteadores, adicionalmente, os roteadores devem ser modificados para implementar o monitoramento, as notificações e o controle de banda. Se forem usados roteadores programáveis como o *Open vSwitch*, essas modificações são simples de implementar.

A rede virtual do experimento foi criada usando o módulo de criação de rede virtual da plataforma FITS, seu funcionamento foi testado usando o sistema FlowFence. O sistema FlowFence foi implementado como um conjunto de agentes do MAGI e sua execução foi controlada usando o *script* do experimento, o que prova a funcionalidade da plataforma de testes como uma facilidade para a criação de ataques de negação de serviço e a avaliação de mecanismos de defesa.

4.3 Observações Finais

A plataforma para experimentação de ataques de negação de serviço permite a criação de cenários de redes virtuais e o teste de ataques de negação de serviço assim como defesas. O processo de criação de redes virtuais pode ser facilitado usando o módulo de criação de redes virtuais, que permite usando um único arquivo inicial, gerar todos os arquivos de configuração e os discos necessários para uma rede virtual funcional dentro da plataforma FITS. A plataforma de testes também oferece ferramentas para o controle de experimentos que estão baseadas no MAGI, desenvolvido pelo DeterLab. Agentes executando código de ataques, defesas, ou para tomar medidas dos testes podem ser incluídos dentro do experimento e sua execução pode ser controlada usando um único *script*. A configuração do experimento pode ser feita usando a interface web do FITS.

Segundo os testes realizados com as ferramentas de ataques de DoS, Stacheldraht e *Flooder* do DETER, o FITS não consegue encaminhar e entregar todos os pacotes

da inundação quando o número de fluxos novos criados é muito grande. Durante um DoS, a falsificação de IPs de origem e portas de origem e destino cria uma grande quantidade de fluxos que tem que ser processados pelas *bridges* do FITS e o comutador por *software open vSwitch*.

Como caso de estudo da plataforma de testes, foi apresentado o FlowFence, um sistema de prevenção de ataques de negação de serviço usando comunicação segura entre roteadores e controlador e controle de banda. O sistema identifica condições de negação de serviço monitorando a ocupação das interfaces de saída dos roteadores e um controlador para implementar uma aplicação de controle que garante uso equitativo de recursos a todos os fluxos encaminhados a um enlace congestionado. Os resultados mostram que na ausência de um sistema de detecção de intrusão, FlowFence garante o uso equitativo aos usuários durante um DoS, evitando a inanição. Os resultados também evidenciam as funcionalidades oferecidas pela plataforma de testes.

Capítulo 5

Conclusão

Os ataques de negação de serviço ainda são um problema persistente da Internet porque obtêm vantagem de suas características básicas. O compartilhamento de recursos cria cenários, que permitem que um grupo de usuários mal comportados afete o desempenho da comunicação e serviços de outros usuários. O principio fim-a-fim da Internet facilita situações nas quais os extremos finais da comunicação são saturados de pacotes encaminhados por entidades intermediárias que possuem pouco conhecimento da informação encaminhada. A falta de mecanismos para assegurar qualidade de serviço na Internet permite que pacotes simples, que não pertencem a aplicações críticas, sejam tratados com a mesma prioridade que outros pacotes, o que possibilita inundações com tudo tipo de pacote. A ausência de atenticação do usuário de origem na Internet permite que muitos usuários façam ataques de forma anônima, o que permite a proliferação de tudo tipo de atacantes, e finalmente o gerenciamento distribuído da Internet pode dificultar a implantação de mecanismos de segurança entre sistemas autônomos, o que dificulta a prevenção de ataques de negação de serviço distribuídos. Assim, uma arquitetura da Internet resistente aos ataques de DoS deve contar com mecanismos para que os nós intermediários realizem controle de recursos para permitir que pacotes considerados críticos sejam entregues aos destinatários finais. De forma similar, tal arquitetura deve permitir o rastreamento de atacantes e acordos têm que ser estabelecidos entre os provedores de infraestrutura e serviços para facilitar a instalação de medidas preventivas de DoS.

As principais motivações para fazer ataques de negação de serviço são políticas e econômicas. Na política, os DoS viraram uma ferramenta de protesta e de atrair a atenção. Nesse sentido, são derrubados sítios de organizações econômicas e políticas participantes de discussões o que promovem leis impopulares para determinados setores da população. Não entanto, os DoS não somente são usados como ferramenta política, em ocasiões uma empresa contrata a terceiros para realizar DoS programados durante momentos importantes para a empresa. Esse mercado para os

ataques de DoS tem beneficiado a proliferação de novas ferramentas para realizar os ataques, o que faz necessário o desenvolvimento de plataformas de testes para testar mecanismos de defesa.

As ferramentas para realizar ataques de negação de serviço têm evoluído desde simples ferramentas que não realizavam vários tipos de inundações e que a comunicação entre os componentes dos ataques e a *botnet* era em claro, para ferramentas que podem realizar uma grande variedade de inundações e que usam canais encriptados para coordenar os ataques. As ferramentas visam a criar uma infraestrutura de comunicações onde um atacante consiga enviar ordens a um número grande de *bots* usando pacotes que parecem legítimos e inofensivos para evitar a detecção por parte dos IDS. No caso de ferramentas para realizar DoS na camada de aplicação, a inspeção dos pacotes até a camada de transporte não revelaria informação que permita identificar um ataque, porque esse tipo de ataques não precisa gerar uma inundação enorme de pacotes. Os DoS na camada de aplicação precisam de defesas instaladas nos servidores o objetivos do ataque já que só eles possuem acesso ao conteúdo total dos pacotes, permitindo obter informações sobre transações que causem prejuízo.

DETER é um *testbed* para testes de segurança cibernética que é uma iniciativa nacional dos Estados Unidos. Em DETER, os usuários recebem uso exclusivo de um número de máquinas virtuais na topologia de rede especificada por eles. O *testbed* oferece um entorno seguro e ferramentas para realizar testes de ataques de negação de serviço. As redes experimentais criadas estão isoladas da Internet e os discos das máquinas usadas são formatados entre experimentos, o que permite realizar testes com códigos maliciosos de DoS e *worms*. O DETER possui uma infraestrutura chamada MAGI que permite controlar a execução de experimentos e desenvolver código que pode ser utilizado por outros pesquisadores posteriormente. Não entanto, experimentos de grande escala em DETER requerem uma quantidade grande de máquinas que pode não ser alocada num momento específico.

Os mecanismos de virtualização e isolamento de redes virtuais no testbed FITS garantem que várias redes virtuais compartilhem a infraestrutura física do testbed. Esta facilidade permite um melhor aproveitamento da infraestrutura física sem afetar o desempenho dos experimentos realizados. Não entanto, esses mecanismos de virtualização diminuem o desempenho na criação de novos fluxos, o que limita a taxa de pacotes que conseguem ser enviados entre as máquinas virtuais. Os resultados experimentais provam que um uso menor de *bridges* de *Open vSwitch* pode melhorar até em um 250% o desempenho no envio de pacotes de novos fluxos frente ao uso de várias etiquetas de VLAN e vários *bridges*. Esse comportamento do encaminhamento dos pacotes entre máquinas virtuais deve ser considerado durante os experimentos de DoS que usem IP *spoofing* ou umafaixa muito grande de portas de

origem.

O serviço proposto oferece um ambiente para experimentação de ataques de negação de serviço que permite a criação de cenários de redes virtuais, assim como a experimentação de ataques de negação de serviço e defesas. A criação de redes virtuais é facilitada usando o módulo de criação de redes virtuais. O módulo utiliza um único arquivo inicial e gera todos os arquivos de configuração e os discos necessários para uma rede virtual funcional dentro da plataforma FITS. O serviço também oferece ferramentas para o controle de experimentos, essas ferramentas estão baseadas no MAGI, desenvolvido pelo DeterLab. Assim, os experimentadores podem criar e incluir seus próprios agentes executando código de ataques, defesas, ou para tomar medidas dos testes. O controle do experimento pode ser controlado usando um único *script*, o que facilita os testes dentro do testbed onde várias máquinas virtuais interagem. A configuração do experimento pode ser feita usando a interface web do FITS.

Para usar as facilidades oferecidas pelo serviço de experimentação de ataques de negação de serviço, foi proposto, desenvolvido e avaliado o sistema FlowFence. O sistema FlowFence permite prevenir ataques de negação de serviço usando uma arquitetura na qual os roteadores da rede monitoram o estado de suas interfaces de saída, e se comunicam com um controlador da rede, em caso de congestionamento, para aplicar controle de banda aos fluxos que usam enlaces congestionados. Os resultados mostram que o FlowFence garante o uso equitativo aos usuários durante um ataque de negação de serviços, evitando a inanição. O FlowFence foi desenvolvido como um agente MAGI e a topologia de testes foi feita usando o módulo de criação de redes virtuais, seu funcionamento evidencia as funcionalidades oferecidas pela plataforma de testes para experimentação de ataques de negação de serviço.

Como trabalhos futuros podem ser desenvolvidas estratégias de virtualização de rede que permitam a rápida criação e encaminhamento de números grandes de fluxos para conseguir realizar experimentos de ataques de negação de serviço que criem muitos fluxos novos. Essas estratégias de virtualização teriam que garantir o isolamento entre redes virtuais que é garantido pelas estratégias atuais. Também podem ser desenvolvidos algoritmos de alocação de máquinas virtuais que considerem os requisitos das máquinas virtuais participantes do experimento para realizar uma alocação que permita um desempenho ótimo em função do encaminhamento de pacotes e processamento.

Referências Bibliográficas

[1] ARBOR NETWORKS. *Worldwide Infrastructure Security Report: 2012 Report.* Arbor Networks, http://www.arbornetworks.com/resources/infrastructure-security-report, 2013. Acessado em abril de 2014.

[2] GUIMARÃES, P. H. V., FERRAZ, L. H. G., TORRES, J. V., et al. "Experimenting Content-Centric Networks in the Future Internet Testbed Environment", *IEEE International Conference on Communications (ICC)-Workshop on Cloud Convergence*, junho 2013.

[3] MATTOS, D. M. F. *Uma Arquitetura de Virtualização de Redes Orientada à Migração com Qualidade de Serviço.* Dissertação de mestrado, COPPE / Universidade Federal do Rio de Janeiro, 2012.

[4] CRISCUOLO, P. J. *"Distributed Denial of Service Trin00, Tribe Flood Network 200, And Stacheldraht".* Relatório técnico, http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA396999, fevereiro 2000. Acessado em abril de 2014.

[5] DOULIGERIS, C., MITROKOTSA, A. "DDoS Attacks and Defense Mechanisms: Classification and State-of-the-art", *Computer Networks*, v. 44, n. 5, pp. 643–666, 2004.

[6] ZARGAR, S., JOSHI, J., TIPPER, D. "A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks", *IEEE Communications Surveys Tutorials*, v. 15, n. 4, pp. 2046–2069, 2013.

[7] FRANK SCALZO. *Recent DNS Reflector Attacks From the Victim and the Reflector Point Of View.* VeriSign, https://www.nanog.org/meetings/nanog37/presentations/frank-scalzo.pdf, 2006. Acessado em abril de 2014.

[8] THE HUFFINGTON POST. *Spamhaus Hit With 'Largest Publicly Announced DDoS Attack' Ever, Affecting Internet Users Worldwide.* The Huffington

Post Inc, http://www.huffingtonpost.com/2013/03/27/spamhaus-cyber-attack_n_2963632.html, 2013. Acessado em abril de 2014.

[9] THE GUARDIAN. *Operation Payback cripples MasterCard site in revenge for WikiLeaks ban.* Guardian News and Media, http://www.theguardian.com/media/2010/dec/08/operation-payback-mastercard-website-wikileaks, 2010. Acessado em abril de 2014.

[10] THE NEW YORK TIMES. *Bank Hacking Was the Work of Iranians, Officials Say.* The New York Times Company, http://www.nytimes.com/2013/01/09/technology/online-banking-attacks-were-work-of-iran-us-officials-say.html?_r=0, 2013. Acessado em abril de 2014.

[11] THE NEW YORK TIMES. *Before the Gunfire, Cyberattacks.* The New York Times Company, http://www.nytimes.com/2008/08/13/technology/13cyber.html, 2008. Acessado em abril de 2014.

[12] NICOL, D. M., DAVIS, C. M., OVERBYE, T. "A Testbed for Power System Security Evaluation", *International Journal of Information and Computer Security*, v. 3, n. 2, pp. 114–131, outubro 2009.

[13] RURSCH, J., JACOBSON, D. "When a testbed does more than testing: The Internet-Scale Event Attack and Generation Environment (ISEAGE) - providing learning and synthesizing experiences for cyber security students." In: *2013 IEEE Frontiers in Education Conference*, pp. 1267–1272, outubro 2013.

[14] SIATERLIS, C., GENGE, B., HOHENADEL, M. "EPIC: A Testbed for Scientifically Rigorous Cyber-Physical Security Experimentation", *IEEE Transactions on Emerging Topics in Computing*, v. 1, n. 2, pp. 319–330, dezembro 2013.

[15] SIATERLIS, C., GARCIA, A., GENGE, B. "On the Use of Emulab Testbeds for Scientifically Rigorous Experiments", *IEEE Communications Surveys Tutorials*, v. 15, n. 2, pp. 929–942, 2013.

[16] MATTOS, D. M. F., MAURICIO, L. H., CARDOSO, L. P., et al. "Uma Rede de Testes Interuniversitária a com Técnicas de Virtualizacao Híbridas", *Salão de Ferramentas do XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos-SBRC*, maio 2012.

[17] RECORDON, D., REED, D. "OpenID 2.0: A Platform for User-centric Identity Management". In: *Proceedings of the Second ACM Workshop on Digital Identity Management*, pp. 11–16, 2006.

[18] PISA, P. S., FERNANDES, N. C., CARVALHO, H. E. T., et al. "OpenFlow and Xen-Based Virtual Network Migration". In: Pont, A., Pujolle, G., Raghavan, S. (Eds.), *Communications: Wireless in Developing Countries and Networks of the Future*, v. 327, *IFIP Advances in Information and Communication Technology*, Springer Berlin Heidelberg, pp. 170–181, 2010.

[19] GONZALEZ BEZERRA, G. M. *Um Sistema Automatizado de Gerência de Recursos para Ambientes Virtualizados*. Relatório técnico, 2013.

[20] RAMOS, F. B., ALVARENGA, I. D., CALDAS, P. M., et al. "Distribuição de Vídeo sob Demanda Baseada em Redes de Distribuição de Conteúdo utilizando Redes Orientadas a Conteúdo", *Aceito para publicação em XXXII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos-SBRC*, maio 2014.

[21] MATTOS, D. M. F., DUARTE, O. C. M. B. "AuthFlow: Um Mecanismo de Autenticação e Controle de Acesso para Redes Definidas por Software", *Aceito para publicação em XXXII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos-SBRC*, maio 2014.

[22] LOBATO, A. G., FIGUEIREDO, U. R., ANDREONI, M., et al. "Uma Arquitetura Elástica para Prevenção de Intrusão em Redes Virtuais usando Redes Definidas por Software", *Aceito para publicação em XXXII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos-SBRC*, maio 2014.

[23] PENG, T., LECKIE, C., RAMAMOHANARAO, K. "Survey of Network-based Defense Mechanisms Countering the DoS and DDoS Problems", *ACM Computing Surveys*, v. 39, n. 1, 2007.

[24] GARBER, L. "Denial-of-service attacks rip the internet", *Computer*, v. 33, n. 4, pp. 12–17, abril 2000.

[25] THE NEW YORK TIMES. *Powerful Attack Upset Global Internet Traffic*. The New York Times Company, http://www.nytimes.com/2002/10/23/technology/23INTE.html, 2002. Acessado em abril de 2014.

[26] MIRKOVIC, J., REIHER, P. “A Taxonomy of DDoS Attack and DDoS Defense Mechanisms”, *SIGCOMM Computer Communication Review*, v. 34, n. 2, pp. 39–53, 2004.

[27] CAMPISTA, M. E. M., FERRAZ, L. H. G., MORAES, I. M., et al. “Interconexão de Redes na Internet do Futuro: Desafios e Soluções”. In: *SBRC 2010*, Gramado, RS, Brasil, maio 2010.

[28] FERGUSON, P., SENIE, D. *"Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing"*. Relatório técnico, http://www.ietf.org/rfc/rfc2827.txt, maio 2000.

[29] ARBOR NETWORKS. *Learn about DoS attacks*. Arbor Networks Inc., http://www.arbornetworks.com/attack-ddos, 2013. Acessado em abril de 2014.

[30] FULTZ, N., GROSSKLAGS, J. “Blue versus Red: Towards a Model of Distributed Security Attacks”. In: *Financial Cryptography and Data Security*, v. 5628, *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 167–183, 2009.

[31] THE HUFFINGTON POST. *North Korea Cyber Attacks: Pyongyang Accuses South, U.S. Of 'Persistent And Intensive' Cyber Attack*. TheHuffingtonPost.com, Inc., http://www.huffingtonpost.com/2013/03/15/north-korea-cyber-attacks_n_2881767.html, 2013. Acessado em abril de 2014.

[32] GEVA, M., HERZBERG, A., GEV, Y. “Bandwidth Distributed Denial of Service: Attacks and Defenses”, *IEEE Security Privacy*, v. 12, n. 1, pp. 54–61, janeiro 2014.

[33] MAX VISION. *Lion Internet Worm Analysis*. http://www.ouah.org/lionw.htm, 2001. Acessado em abril de 2014.

[34] SYMANTEC SECURITY. *Linux Ramen Worm*. http://service1.symantec.com/sarc/sarc.nsf/html/linux.ramen.worm.html, 2001. Acessado em abril de 2014.

[35] CERT. *Advisory CA-2001-19 "Code Red"Worm Exploiting Buffer Overflow In IIS Indexing Service DLL*. Computer Emergency Response Team, http://www.cert.org/advisories/CA-2001-19.html, 2001. Acessado em abril de 2014.

[36] INTERNET SECURITY SYSTEMS. *Land denial of service (Land attack)*. IBM Corporation, http://www.iss.net/security_center/reference/vuln/Land_Attack.htm, 2001. Acessado em abril de 2014.

[37] INTERNET SECURITY SYSTEMS. *Ping of Death*. IBM Corporation, http://www.iss.net/security_center/advice/Intrusions/2000012/default.htm, 2001. Acessado em abril de 2014.

[38] KUZMANOVIC, A., KNIGHTLY, E. W. "Low-rate TCP-targeted Denial of Service Attacks: The Shrew vs. The Mice and Elephants". In: *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, pp. 75–86, 2003.

[39] CNN MONEY. *Anonymous strikes back after feds shut down piracy hub Megaupload*. CNN - Cable News Network, http://money.cnn.com/2012/01/19/technology/megaupload_shutdown/, 2012. Acessado em abril de 2014.

[40] WEBROOT. *DDoS for hire services offering to 'take down your competitor's web sites' going mainstream*. Webroot Inc, http://www.webroot.com/blog/2012/06/06/ddos-for-hire-services-offering-to-take-down-your-competitors-web-sites-going-mainstream/, 2012. Acessado em abril de 2014.

[41] MARCHESSEAU, M. *"Trinity- distributed denial-of-service attack tool"*. Relatório técnico, http://www.giac.org/paper/gsec/123/trinity-distributed-denial-service-attack-tool/100510, setembro 2000. Acessado em abril de 2014.

[42] TECHNOLOGIES, P. *"Low Orbit Ion Cannon - LOIC"*. Relatório técnico, https://github.com/neweracracker/loic, fevereiro 2010. Acessado em abril de 2014.

[43] MIRKOVIC, J., FAHMY, S., REIHER, P., et al. "How to Test DoS Defenses". In: *Conference For Homeland Security, 2009. CATCH '09. Cybersecurity Applications Technology*, pp. 103–117, março 2009.

[44] FLOYD, S., KOHLER, E. "Internet Research Needs Better Models", *SIGCOMM Computing Communication Review*, v. 33, n. 1, pp. 29–34, janeiro 2003.

[45] HARDAKER, W. AND KINDRED, D. AND OSTRENGA, R. AND STERNE, D. AND THOMAS, R. *Justification and Requirements for a National DDoS Defense Technology Evaluation Facility.* Network Associates Laboratories, http://isi.edu/deter/docs/02-052DDoS_Defense_Evaluation_Facility.pdf, julho 2002. Acessado em abril de 2014.

[46] MIRKOVIC, J., BENZEL, T., FABER, T., et al. "The DETER Project: Advancing the Science of Cyber Security Experimentation and Test". In: *IEEE International Conference on Technologies for Homeland Security (HST)*, 2010.

[47] BENZEL, T., BRADEN, R., KIM, D., et al. "Experience with DETER: a testbed for security research". In: *2nd International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities, 2006. TRIDENTCOM 2006.*, pp. 10 pp.–388, 2006.

[48] DETER. *MAGI (Montage AGent Infrastructure).* University of Southern California, http://montage.deterlab.net/magi/magi.html, 2013. Acessado em abril de 2014.

[49] MATTOS, D. M. F., DUARTE, O. C. M. B. "QFlow: Um Sistema com Garantia de Isolamento e Oferta de Qualidade de Serviço para Redes Virtualizadas", *XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos-SBRC*, maio 2012.

[50] CHEN, R., PARK, J.-M., MARCHANY, R. "A Divide-and-Conquer Strategy for Thwarting Distributed Denial-of-Service Attacks", *IEEE Transactions on Parallel and Distributed Systems*, v. 18, n. 5, pp. 577–588, 2007.

[51] LAUFER, R. P., VELLOSO, P. B., DUARTE, O. C. M. B. "Um Novo Sistema de Rastreamento de Pacotes IP contra Ataques de Negação de Serviço", *XXIII Simpósio Brasileiro de Redes de Computadores - SBRC 2005*, maio 2005.

[52] YANG, X., WETHERALL, D., ANDERSON, T. "TVA: A DoS-Limiting Network Architecture", *IEEE/ACM Transactions on Networking*, v. 16, n. 6, pp. 1267–1280, 2008.

[53] LIU, X., YANG, X., XIA, Y. "NetFence: preventing internet denial of service from inside out". In: *Proceedings of the ACM SIGCOMM 2010 conference*, SIGCOMM '10, pp. 255–266. ACM, 2010.

[54] LIU, X., YANG, X., WETHERALL, D., et al. "Efficient and Secure Source Authentication with Packet Passports". In: *Proceedings of the 2Nd Conference on Steps to Reducing Unwanted Traffic on the Internet - Volume 2*, pp. 2–2. USENIX Association, 2006.

[55] MAHAJAN, R., BELLOVIN, S. M., FLOYD, S., et al. "Controlling High Bandwidth Aggregates in the Network", *ACM SIGCOMM Computer Communication Review*, v. 32, n. 3, pp. 62–73, 2002.

[56] GHOBADI, M., YEGANEH, S. H., GANJALI, Y. "Rethinking End-to-end Congestion Control in Software-defined Networks". In: *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, pp. 61–66. ACM, 2012.