UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
ESCOLA POLITÉCNICA
DEPARTAMENTO DE ELETRÔNICA E DE COMPUTAÇÃO

Implementação e Avaliação de um
Sistema de Rastreamento de Pacotes IP

Autor:

Marcelo Duffles Donato Moreira

Orientador:

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

Co-orientador:

Rafael Pinaud Laufer, M.Sc.

Examinadores:

Prof. Flávio Luis de Mello, D.Sc.

Prof. Luís Henrique Maciel Kosmalski Costa, Dr.

Miguel Elias Mitre Campista, D.Sc.

DEL

Fevereiro de 2009

À minha família e à Flávia, amor da minha vida.

Agradecimentos

Agradeço em primeiro lugar a Deus pelo dom da vida e por todos os outros dons que Ele me concedeu e que hoje coloco à disposição da sociedade.

Aos meus pais, Julio Cezar e Marli, por terem escolhido abrir mão de conforto e comodidade em função da educação e do bem-estar dos filhos; pelo exemplo de vida, pelo amor e pelo incentivo e orientação passados em todos os anos da minha vida.

À minha noiva Flávia, pelo amor, carinho e companheirismo; por compartilhar as alegrias, os sonhos e as tristezas; e por fazer parecer 900 km uma distância pequena.

À minha comunidade do Caminho Neocatecumenal de Copacabana, por me ajudar a encontrar o sentido da vida, por todos os momentos vividos, pelas experiências de fé, pela ajuda nos momentos difíceis, pelos cansativos, mas profícuos, trabalhos para levar os jovens à Jornada Mundial da Juventude. Ao Pe. Michele Ferrara, pelo exemplo de fé e dedicação. A Dom Lourenço de Almeida Prado (*in memoriam*), pelo exemplo de vida e por todas as vezes que não hesitou em renovar minha matrícula e dos meus irmãos no Colégio de São Bento, mesmo quando meus pais não tinham como pagar. Ficam a certeza de felicidade no céu e as boas lembranças na terra.

Aos meus avós, em especial ao meu avô materno Jorge Henrique, pelo exemplo de vida, pela dedicação, pelas piadas e pelas férias divertidas em Conceição da Barra/ES.

Ao meu padrinho André, pela amizade e por estar presente nos momentos importantes da minha vida.

Ao professor e orientador Otto, por seus conselhos e orientação, por ensinar-me a ter espírito crítico, a escrever de forma científica, a fazer exposições em público e pelas longas horas de discussões interessantes.

Aos amigos e colegas de graduação, Carlo Fragni, André Luiz, Flávio Raposo e Sérgio Francisco, pela amizade e pelos bons momentos. Aos amigos e colegas do GTA, Danilo, Natalia, Bruno Taft, Mariângela, Pedro Esposito, Reinaldo, Carina, Daniel Cunha, Pedro Velloso, Miguel e Igor. Agradeço em especial ao colega e co-orientador Rafael Laufer pela contribuição fundamental neste trabalho e por se colocar sempre disponível mesmo atarefado e à distância.

A todos os meus professores, desde a pré-escola, em particular aos professores da UFRJ, pela dedicação e excelência. Agradeço em especial aos professores do Departamento de Eletrônica e de Computação, Osvaldo Pereira, Jomar Gozzi, Sérgio Lima Netto, Luiz Wagner e Fernando Barúqui, do Instituto de Matemática, Cassio Neri, Felipe Acker e Severino Collier, e do Instituto de Física, Carlos Farina. Agradeço em particular ao doutor Miguel Elias Mitre Campista e aos professores Otto Carlos Muniz Bandeira Duarte, Luís Henrique Maciel Kosmalski Costa e Flávio Luis de Mello, pela presença na banca examinadora e contribuição neste trabalho.

A todos que me incentivaram, contribuindo de forma direta ou indireta, para a minha formação profissional.

Ao CNPq, CAPES, RNP/FUNTTEL, FAPERJ, FINEP e UOL pelo financiamento da pesquisa.

Resumo do Projeto Final apresentado ao DEL/UFRJ como parte dos requisitos necessários para a obtenção do grau de Engenheiro Eletrônico e de Computação.

Implementação e Avaliação de um Sistema de Rastreamento de Pacotes IP

Marcelo Duffles Donato Moreira

Fevereiro de 2009

Orientador: Otto Carlos Muniz Bandeira Duarte
Departamento: Engenharia Eletrônica e de Computação

Os ataques de negação de serviço distribuídos são hoje uma das maiores ameaças de segurança da Internet. Os atacantes utilizam redes de ataque, as *botnets*, da ordem de milhões de computadores. Tipicamente, cada estação de ataque gera certa quantidade de tráfego em direção à vítima e o tráfego agregado é então responsável por prejudicar ou interromper completamente uma atividade legítima. Com tamanho poder de ataque, os resultados dos ataques são catastróficos. Um relatório recente do FBI indica que os prejuízos financeiros e de imagem das empresas e instituições atacadas são enormes.

Apesar do avanço em pesquisa e desenvolvimento de técnicas de defesa contra ataques de negação de serviço, as ferramentas comerciais de proteção e reação a ataques ainda são incipientes e/ou incompletas e os verdadeiros autores dos ataques em geral não são descobertos, permanecendo anônimos e impunes. Uma solução promissora é tornar a rede capaz de rastrear os atacantes, a fim de interromper os ataques e aplicar punições legais. Esse trabalho apresenta uma nova estrutura de marcação de pacotes que tem como características principais a robustez à interferência do atacante e uma elevada capacidade de compactação. São apresentadas análises teóricas e de simulação que comprovam a eficácia do sistema de rastreamento proposto. O sistema é comparado a outras propostas da literatura e apresenta melhor desempenho em termos robustez, acurácia e economia de espaço no cabeçalho dos pacotes.

Palavras-Chave

Redes de Computadores

Internet

Negação de Serviço

Rastreamento de Pacotes

Filtro de Bloom

Lista de Acrônimos

BGP : *Border Gateway Protocol*;

CAIDA : *Cooperative Association for Internet Data Analysis*;

CERT : *Computer Emergency Response Team*;

CSI : *Computer Security Institute*;

DoS : *Denial of Service*;

DDoS : *Distributed Denial of Service*;

DNS : *Domain Name System*;

FBC : Filtro de Bloom Concatenado;

FBG : Filtro de Bloom Generalizado;

FBI : *Federal Bureau of Investigation*;

ICMP : *Internet Control Message Protocol*;

IP : *Internet Protocol*;

IPv4 : *Internet Protocol*, versão 4;

IPv6 : *Internet Protocol*, versão 6;

IRC : *Internet Relay Chat*;

ISP : *Internet Service Provider*;

NAT : *Network Address Translation*;

TCP : *Transmission Control Protocol*;

TTL : *Time-To-Live*;

UDP : *User Datagram Protocol*;

Sumário

Resumo **v**

Lista de Figuras **xi**

I Introdução **1**

I.1 Histórico dos Ataques de Negação de Serviço 2

I.2 A Raiz do Problema 3

I.3 Abordagens de Defesa 5

I.4 Objetivos desse Trabalho 6

I.5 Organização do Texto 7

II Ataques de Negação de Serviço **8**

II.1 Ataques Distribuídos 8

II.2 Por Que DDoS é um Problema Difícil? 11

II.3 Rastreamento de Pacotes como Solução 13

III Sistemas de Rastreamento **15**

III.1 Sistemas Propostos na Literatura 16

III.2 O Sistema de Rastreamento Proposto 18

III.2.1 Procedimento de Marcação de Pacotes 18

III.2.2 Procedimento de Reconstrução de Rota 20

III.2.3 Características Importantes 20

IV Filtros de Bloom Robustos a Ataques **22**

IV.1 Filtro de Bloom 25

IV.1.1 Algoritmo de Inserção de Elemento 25

IV.1.2 Algoritmo de Teste de Pertinência 26

IV.1.3 Falsos Positivos 26

IV.2 Extensões do Filtro de Bloom 27

IV.2.1 Filtro de Bloom Generalizado 28

IV.3 O Ataque do Ajuste da Condição Inicial 32

IV.4 O Filtro de Bloom Concatenado 35

IV.4.1 Definição 35

IV.4.2 Algoritmo de Inserção de Elemento 36

IV.4.3 Algoritmo de Teste de Pertinência 38

V Resultados Analíticos e de Simulação **41**

V.1 Resultados Analíticos 41

V.1.1 Falsos Positivos 41

V.1.2 Robustez 44

V.2 Resultados de Simulação 46

SUMÁRIO

VI Conclusão **49**

Referências Bibliográficas **51**

Lista de Figuras

II.1 Ataque por inundação. Recursos essenciais da rede ou da vítima são sobrecarregados pelo atacante. 9

II.2 Ataque de negação de serviço distribuído exemplificando uma hierarquia que dificulta a detecção do atacante. 10

II.3 A topologia assimétrica favorece ataques distribuídos, pois a vítima pode ser inundada mesmo que cada zumbi gere apenas uma pequena parcela do tráfego agregado. 13

III.1 O procedimento de marcação de pacotes que insere rastros no pacote em cada roteador. 19

III.2 O procedimento de reconstrução de rota é feito de forma recursiva a partir da vítima. 20

IV.1 Compromisso entre robustez e perda de informação para um FBG. 31

IV.2 Esquema mostrando a informação disponível à vítima após a marcação do pacote pelos roteadores da rota de ataque. 32

IV.3 O ataque do ajuste da condição inicial com a inserção forjada de alguns elementos antes de enviar o pacote. 33

LISTA DE FIGURAS

IV.4 Criação de sufixos arbitrários forjados pelo atacante como resultado do ataque do ajuste da condição inicial. 34

IV.5 Formato do FBC com a concatenação de N subfiltros de m/N bits cada. . 36

IV.6 Esquema de marcação 1 com subfiltros inicialmente zerados e aplicação do algoritmo de inserção do Filtro de Bloom convencional. 38

IV.7 A inserção de elemento no FBC onde o conteúdo original do subfiltro é substituído pela marcação do elemento a ser inserido. 39

IV.8 Teste de pertinência no FBC. A marcação do elemento a ser testado é comparada ao conteúdo do subfiltro indicado pelo contador. 40

V.1 Probabilidade de falso positivo em função da fração de bits em zero. 43

V.2 Probabilidade de falso positivo para o esquema de marcação 1. 44

V.3 Ordem de inserção dos elementos no filtro. 45

V.4 Número de atacantes rastreados em função do número de bits por elemento. 47

Capítulo I

Introdução

DISPONIBILIDADE é um requisito de segurança fundamental para a Internet. A disponibilidade trata de garantir que os serviços e recursos da rede estarão disponíveis quando forem necessários [1], isto é, trata de garantir que, no momento em que um usuário tentar acessar um dado serviço, o servidor será capaz de atendê-lo da forma esperada e haverá recursos suficientes na rede para que o usuário seja servido. O serviço pode ser, por exemplo, o uso de um buscador de páginas, a compra de um determinado produto ou simplesmente a troca de mensagens entre duas entidades. Portanto, não se pode imaginar o funcionamento de um sistema complexo e distribuído como a Internet sem que o requisito da disponibilidade seja satisfeito. Não obstante, ataques de negação de serviço (*Denial of Service* - DoS) ameaçam hoje a disponibilidade dos serviços da Internet e são um dos problemas de segurança mais difíceis da atualidade [2]. A maioria destes ataques é originada pelas chamadas *botnets*, que são redes de máquinas subvertidas que, em conjunto, tornaram-se mais poderosas que centenas de supercomputadores [3]. Tipicamente, cada estação de ataque gera certa quantidade de tráfego em direção à vítima e o tráfego agregado é então responsável por prejudicar ou interromper completamente uma atividade legítima [4]. Há diversas formas de se realizar um ataque de negação de serviço, mas a ideia básica é consumir recursos essenciais da rede, como banda passante ou fila nos roteadores, ou recursos da própria vítima, como memória ou capacidade de processamento. Dessa forma, o serviço provido pela vítima se torna indisponível quando

um usuário tenta acessá-lo.

I.1 Histórico dos Ataques de Negação de Serviço

As estatísticas mostram que os ataques de negação de serviço na Internet são bastante frequentes e os prejuízos financeiros e de imagem são gigantescos, conforme aponta o relatório de 2008 do CSI/FBI (*Computer Security Institute/Federal Bureau of Investigation*) sobre crimes na área de computação [5]. Além disso, o histórico dos ataques mostra que eles estão crescendo em intensidade, frequencia, gravidade e alcance. Em janeiro de 2001, um ataque de negação de serviço impediu o acesso de 98% dos usuários a quaisquer servidores da Microsoft [6]. Ainda em 2001, pesquisadores da CAIDA (*Cooperative Association for Internet Data Analysis*) usaram a técnica *backscatter* para inferir o comportamento das atividades de negação de serviço [7]. Um dos resultados é uma subestimativa de que a frequencia dos ataques de negação de serviço na Internet é de 4.000 ocorrências por semana. Em outubro de 2002, houve um ataque aos 13 servidores-raiz do DNS (*Domain Name System*) [8]. O serviço de DNS é crucial para o funcionamento da Internet, pois é responsável pela tradução de nomes, que são facilmente memorizados por seres humanos, em endereços IP (*Internet Protocol*), que são usados para identificar e endereçar as estações na Internet. Como o DNS é um sistema que faz uso pesado de técnicas de cacheamento e o ataque durou apenas uma hora, não houve grande abalo nas atividades da Internet. Entretanto, 9 dos 13 servidores foram seriamente afetados. Se o ataque tivesse durado mais algum tempo, provavelmente os serviços da Internet em todo o globo seriam afetados. Em 2004, os prejuízos financeiros causados por ataques nos Estados Unidos foram da ordem de 26 milhões de dólares, o que levou o CSI/FBI a classificar a negação de serviço como um dos incidentes de segurança que mais causaram prejuízos às instituições americanas naquele ano [9]. Juntamente com os crimes na Internet, os ataques de negação de serviço evoluíram e passaram a ser usados como forma de extorsão e as redes de ataque, as *botnets*, passaram a ser vendidas no mercado negro da Internet através das redes IRC (*Internet Relay Chat*), conforme um estudo de

2007 [10]. Um exemplo desse tipo de atividade criminosa foi relatado pelos jornais em janeiro de 2006, quando um jovem admitiu controlar cerca de 500.000 computadores que eram alugados a outros criminosos. As reportagens da época relatam que o jovem havia auferido cerca de 60 mil dólares em dinheiro, um computador e um carro da marca BMW com a atividade criminosa [11]. Recentemente, o crime cibernético passou a fazer parte dos conflitos militares internacionais. Em 2007 e 2008 foram relatados ataques de negação de serviço durante os conflitos da Rússia com a Geórgia e a Estônia [12, 13].

I.2 A Raiz do Problema

Esse contundente histórico de ataques demonstra que, apesar do avanço em pesquisa e desenvolvimento de técnicas de defesa contra ataques de negação de serviço, há ainda muito que se fazer nessa área. As ferramentas comerciais de proteção e reação a ataques são ainda incipientes e/ou incompletas e os verdadeiros autores dos ataques em geral não são descobertos, permanecendo anônimos e impunes. Mas, se os ataques de negação de serviço causam tantos prejuízos e são tão graves, por que não se combatem os atacantes? Qual é o verdadeiro problema que impede a prevenção, detecção e reação a esses ataques? Para chegar à raiz do problema, é necessário um aprofundamento nos princípios da arquitetura da Internet, que facilitam a proliferação de ataques e dificultam a identificação dos atacantes. São dois os princípios fundamentais que são explorados pelos ataques de negação de serviço: a comutação de pacotes e o paradigma fim-a-fim. No projeto da Internet na década de 1970 optou-se por uma rede de comutação de pacotes, em oposição à comutação de circuitos empregada na telefonia [14]. O objetivo principal era um melhor aproveitamento do meio de transmissão, que poderia ser compartilhado por diferentes estações. No entanto, uma consequencia direta da comutação de pacotes é a inexistência de reserva de recursos. É exatamente esse ponto que é explorado pelos ataques de negação de serviço: uma vez ocupados pelos atacantes, os recursos da rede ficam indisponíveis para os usuários corretos. Além disso, nenhum estado, como a rota seguida pelos pacotes, é armazenado na infraestrutura de rede e o roteamento na Internet é dinâmico, sendo

portanto difícil filtrar um pacote de ataque baseando-se no seu endereço de origem. O segundo princípio explorado pelos ataques de negação de serviço é o paradigma fim-a-fim, que apregoa que o núcleo da rede deve ser simples e a inteligência deve ficar nas extremidades da rede, isto é, nas estações finais. Uma consequencia do paradigma fim-a-fim é que os requisitos de cada aplicação devem ser garantidos pelas estações finais. Em particular, a garantia de requisitos de segurança, como a autenticação, é de responsabilidade das estações finais. Dessa maneira, o protocolo da camada de rede da Internet, o IP (*Internet Protocol*), não provê autenticação da fonte e, portanto, podem-se injetar na rede pacotes com endereço de origem forjado. Essa vulnerabilidade do IP é explorada pelos atacantes para garantir o seu anonimato. Além disso, o envio de pacotes com endereço de origem forjado evita a adoção de contramedidas aos ataques, já que mecanismos de defesa que usam o endereço de origem para diferenciar um pacote de ataque de um pacote legítimo são ineficazes nesse caso.

Quando os primeiros ataques de negação de serviço surgiram na Internet, os defensores do paradigma fim-a-fim diziam que os problemas de segurança deveriam ser tratados pelas estações finais. No entanto, o enorme crescimento dos ataques de negação de serviço distribuídos (*Distributed Denial-of-Service - DDoS - attacks*) indicou que pelo menos alguns mecanismos de segurança devem ser providos pelo núcleo da rede. O argumento é simples: uma vez que o ataque tornou o serviço de rede indisponível, o problema não pode ser resolvido pelas extremidades. Além disso, não há atualmente nenhum tipo de proteção contra ataques aos próprios elementos de rede, o que dificulta uma possível reação aos ataques. Portanto, conclui-se que o requisito da disponibilidade deve ser garantido também pelo núcleo da rede. Propõe-se uma cooperação entre as estações finais e os elementos de rede como forma de tornar possível o combate aos ataques de negação de serviço.

I.3 Abordagens de Defesa

Tradicionalmente há duas abordagens para combater os ataques de negação de serviço. A primeira abordagem é a prevenção. A abordagem preventiva consiste em evitar que o ataque aconteça. Isso pode ser feito através da técnica de filtragem de ingresso (*ingress filtering*) [15], que consiste em garantir que os pacotes que entram em um roteador são de fato oriundos das redes especificadas pelo endereço de origem no cabeçalho do pacote. No entanto, essa técnica requer uma implementação global para ser efetiva. Além disso, para a adoção de uma solução na Internet é necessário que a proposta não apenas seja tecnicamente possível, mas também seja economicamente viável. Como não há nenhuma vantagem econômica para um provedor de serviço da Internet (*Internet Service Provider* - ISP) adotar a filtragem de ingresso, essa técnica necessita do altruísmo dos ISPs para funcionar corretamente, o que não é uma suposição prática. Além disso, mecanismos importantes como a tradução de endereço de rede (*Network Address Translation* - NAT) e o IP móvel usam endereços de origem forjados de forma legítima e, portanto, seriam prejudicados no caso de adoção da filtragem de ingresso. Uma segunda abordagem para combater os ataques de negação de serviço é reagir ao ataque. Seguindo essa abordagem reativa, uma solução promissora para o problema é tornar a rede capaz de reconstruir a rota atravessada por cada pacote e, assim, identificar a sua verdadeira origem. Isso é feito através da adoção de um sistema de rastreamento de pacotes [16, 17]. Uma vez rastreado o atacante, medidas legais podem ser tomadas. É sabido, no entanto, que os atacantes adicionam camadas extras de proteção iniciando o ataque não pelos seus próprios computadores, mas por estações intermediárias previamente comprometidas. O problema de encontrar o verdadeiro autor dos ataques a partir da identificação das estações originadoras do tráfego de ataque é conhecido na literatura técnica como o *stepping-stone problem* [18]. Portanto, pode-se afirmar que uma solução completa para o problema da identificação do verdadeiro autor de um ataque de negação de serviço necessita de um esforço conjunto de diversos mecanismos de defesa e o primeiro passo é o rastreamento de pacotes. O escopo desse trabalho é o rastreamento de pacotes, isto é, a identificação das estações que geram diretamente o tráfego de ataque. Ainda que não haja uma solução

para o problema do *stepping-stone* e, consequentemente, o verdadeiro autor dos ataques não seja identificado, podem-se utilizar as informações parciais da rota de ataque obtidas pelo sistema de rastreamento para filtrar o tráfego de ataque. Além disso, só o fato do atacante saber que ele pode ser rastreado já é um fator inibidor de futuros ataques.

I.4 Objetivos desse Trabalho

O objetivo desse trabalho é a implementação e avaliação de um sistema de rastreamento proposto pelo autor [19]. O sistema de rastreamento proposto utiliza a técnica de marcação de pacotes, na qual cada roteador insere uma informação de rastreamento no cabeçalho do pacote no momento do seu encaminhamento. Dessa forma, após a travessia do pacote pela rede, a vítima recebe um pacote em cujo cabeçalho se encontram informações de rastreamento suficientes para reconstruir a rota de ataque e, então, rastrear o atacante. Propõe-se a utilização de uma estrutura de dados compacta para o armazenamento da rota atravessada por cada pacote IP. A estrutura de dados escolhida é denominada Filtro de Bloom [20], que permite um armazenamento eficiente de elementos ao custo de uma pequena probabilidade de falso positivo. O Filtro de Bloom permite a inserção de elementos e a verificação posterior se um dado elemento foi ou não inserido no filtro. Ao encaminhar um pacote, cada roteador insere seu endereço IP no cabeçalho do pacote, onde está embutido o Filtro de Bloom. Assim, após receber o pacote de ataque, a vítima utiliza o Filtro de Bloom embutido no cabeçalho do pacote para iniciar a reconstrução da rota de ataque. O filtro contém a informação de quais roteadores foram atravessados pelo pacote. A partir dessa informação, utiliza-se um procedimento de reconstrução de rota para chegar até a estação que enviou o pacote.

Apesar de sua comprovada eficiência, o uso do Filtro de Bloom em aplicações distribuídas como o rastreamento de pacotes IP é limitado por questões de segurança. Uma análise de falhas de segurança de Filtros de Bloom [19] mostrou que a questão principal gira em torno da condição inicial do filtro, que é controlada pelo atacante no caso dos ataques na Internet. Pode-se mostrar que o atacante é capaz de ajustar a condição inicial

do pacote de forma a dificultar ou até impedir o seu rastreamento. Para a solução desse problema, propõe-se uma nova estrutura de dados denominada Filtro de Bloom Concatenado (FBC), cujo objetivo é prover robustez à interferência do atacante. É garantida a robustez à interferência do atacante através do apagamento das informações iniciais inseridas pelo atacante. A ideia chave da proposta é concatenar diversos subfiltros, cada qual admitindo somente um elemento. Essa construção específica garante a robustez desejada sem que haja perda de informação legítima. Como somente um elemento é inserido por subfiltro, podem-se sobrescrever as informações iniciais inseridas pelo atacante sem perda de informação legítima. Resultados analíticos mostram que o filtro proposto é bem eficaz e que a probabilidade de sucesso do atacante decresce exponencialmente com o tamanho de cada subfiltro. Não foi encontrada na literatura nenhuma proposta que provesse tal robustez sem o inconveniente da perda de informação legítima.

O presente trabalho apresenta análises teóricas e de simulação do sistema de rastreamento proposto. O sistema proposto é implementado e avaliado através de um simulador desenvolvido com a linguagem de programação C++. Os resultados comprovam a eficácia do mecanismo proposto. Comparado às outras propostas encontradas na literatura, o filtro proposto possui como principais vantagens a ausência de falsos negativos, a maior capacidade de armazenamento e a maior robustez à interferência do atacante.

I.5 Organização do Texto

Este trabalho está organizado da seguinte forma. No Capítulo II são apresentados os ataques de negação de serviço e as razões que justificam ser esse um dos grandes desafios de segurança da atualidade. No capítulo III é feita uma descrição completa do funcionamento de um sistema de rastreamento e uma revisão da literatura. O Capítulo IV apresenta o Filtro de Bloom proposto nesse trabalho, assim como uma análise comparativa do filtro proposto nesse trabalho com outros filtros propostos na literatura. Resultados analíticos e de simulação são apresentados no Capítulo V. Por fim, no Capítulo VI serão apresentadas as conclusões sobre este trabalho.

Capítulo II

Ataques de Negação de Serviço

OS ataques de negação de serviço visam comprometer recursos essenciais da vítima de forma a tornar indisponível o serviço oferecido. Laufer *et al.* apresentam uma taxonomia dos principais tipos de ataques de negação de serviço de acordo com o tipo de recurso explorado [21]. Os ataques são classificados em ataque por inundação, por refletor, à infraestrutura de rede ou por vulnerabilidade. O caso mais comum é o ataque por inundação, cujo objetivo é impedir o acesso a um determinado serviço através da sobrecarga de recursos da rede ou da vítima. Os recursos da vítima podem ser memória, poder de processamento, espaço em disco, etc. Tais recursos são sobrecarregados através do envio de pacotes a uma taxa superior àquela que o servidor pode suportar, fazendo com que requisições legítimas não sejam atendidas. A Figura II.1 ilustra esse tipo de ataque. Na figura a vítima é representada por V, o atacante por A e o usuário por U.

II.1 Ataques Distribuídos

Uma dificuldade para a execução de ataques de negação de serviço está em gerar mensagens a uma taxa suficientemente alta, de forma que a vítima ou sua infraestrutura de rede não consiga atendê-las em tempo hábil. Se a vítima tiver poucos recursos, é possível a realização de ataques de inundação usando apenas o poder de fogo de um único ata-

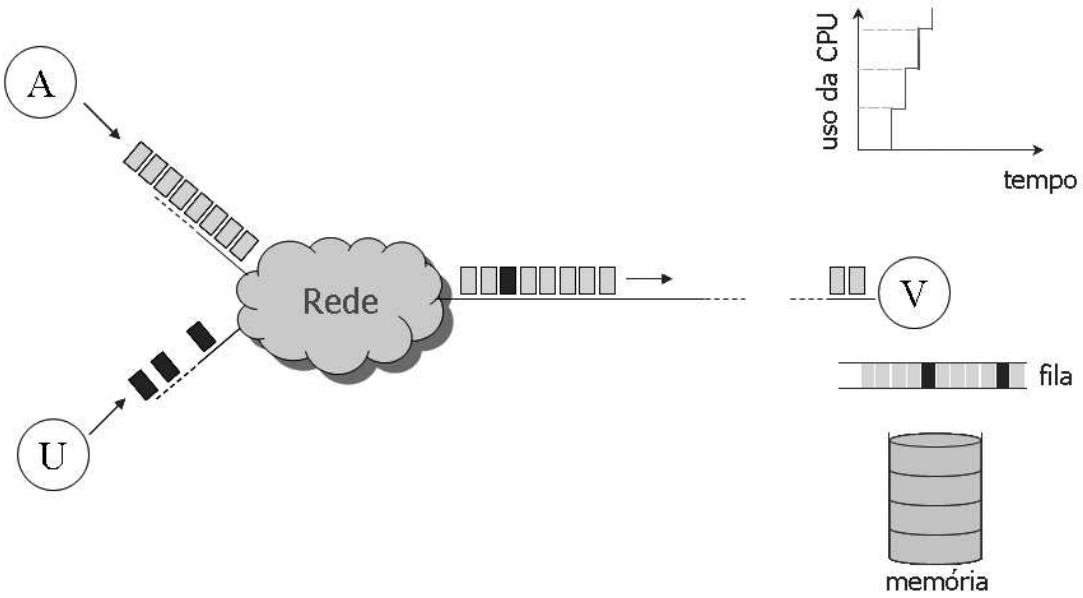

Figura II.1: Ataque por inundação. Recursos essenciais da rede ou da vítima são sobrecarregados pelo atacante.

cante. Porém, no caso da vítima ser um servidor superdimensionado, o efeito do tráfego gerado por um único atacante pode ser insuficiente para exaurir os recursos da vítima. Por essa razão, ataques a servidores de grandes empresas são geralmente realizados através de uma ação conjunta de diversas estações controladas e coordenadas pelo atacante, caracterizando um ataque de negação de serviço distribuído (*Distributed* DoS - DDoS). A vantagem dos ataques distribuídos é que, desde que seja utilizado um número suficiente de estações, sempre é possível inundar a vítima e negar o seu serviço. Consequentemente, mesmo servidores superdimensionados podem sofrer prejuízos consideráveis. A prova disso é que diversos servidores famosos como Yahoo, Ebay, Amazon.com e CNN.com já foram atacados [22, 23].

As estações controladas pelo atacante são conhecidas como agentes, escravos ou zumbis [21]. Tais estações geralmente não pertencem ao atacante e são invadidas, recrutadas e controladas através de ferramentas automatizadas [24, 25, 26]. Essas ferramentas possuem métodos automáticos para o recrutamento e o controle dos zumbis. O método mais comum para recrutamento dos zumbis é a varredura de uma larga faixa de endereços IP na tentativa de encontrar máquinas com vulnerabilidades conhecidas. Esse método

se revelou bastante eficiente, principalmente quando as ferramentas são atualizadas com vulnerabilidades recentemente descobertas, que possuem maior probabilidade de ainda não terem sido resolvidas pelos administradores das máquinas a serem invadidas. Outra maneira de recrutar facilmente um grande número de zumbis é através de vermes (*worms*), que são pragas digitais que se propagam rapidamente pela Internet. Os vermes são programas que procuram por máquinas vulneráveis e as infectam com uma cópia do seu código [21]. Alguns vermes usam as máquinas infestadas especialmente para realizar ataques de negação de serviço distribuído. Exemplos de vermes desse tipo incluem o *Code Red*, o *W32/Blaster* e o *W32/Sobig.F* [4]. A Figura II.2 (extraída de [27]) ilustra a operação de um ataque distribuído.

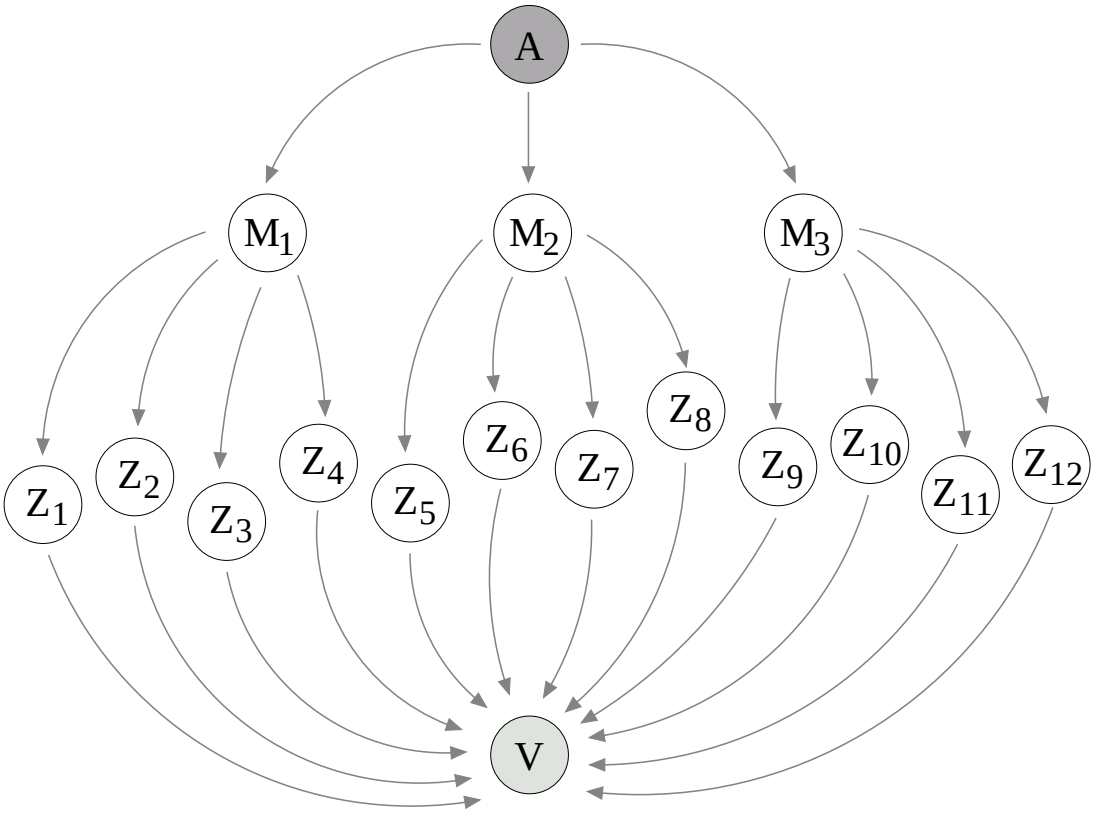

Figura II.2: Ataque de negação de serviço distribuído exemplificando uma hierarquia que dificulta a detecção do atacante.

A fim de controlar uma grande quantidade de zumbis, o atacante faz uso de uma rede hierárquica. As estações intermediárias entre o atacante e os zumbis são denominadas

mestres. O atacante controla diretamente as estações-mestre, que são responsáveis por repassar os comandos aos seus respectivos zumbis, conforme ilustrado na figura acima.

II.2 Por Que DDoS é um Problema Difícil?

Os ataques de negação de serviço distribuídos possuem características que tornam o desenvolvimento de contramedidas uma tarefa desafiadora. As principais características dos ataques de negação de serviço distribuídos que dificultam a ação dos mecanismos de defesa são apresentadas a seguir.

Em primeiro lugar, os atacantes controlam um grande número de máquinas de ataque. Há relatos de ataques distribuídos com milhões de zumbis [28, 29, 30]. Isso garante um poder de ataque impressionante. Segundo Matt Sergeant, especialista de segurança da empresa americana MessageLabs, a rede de zumbis formada pelo verme *Storm* é capaz de superar o poder de processamento de 500 dos mais poderosos supercomputadores existentes [3]. A afirmação considera uma rede de zumbis de cerca de 2 milhões de computadores pessoais infestados pelo verme, sendo que há estimativas de que a rede do *Storm* pode chegar a 50 milhões de zumbis [31].

Uma segunda característica importante dos ataques de negação de serviço é que o tráfego de ataque é muito similar ao tráfego dos usuários legítimos. Em geral, o tráfego de ataque é composto por mensagens com características típicas da comunicação do serviço atacado. Assim, não é possível distinguir o tráfego de ataque do tráfego de usuários legítimos. Portanto, a adoção de contramedidas ao ataque é bastante dificultada, pois ações que priorizem o tráfego legítimo em detrimento do tráfego de ataque são difíceis de serem tomadas.

O terceiro ponto é o fato dos atacantes usarem diversos artifícios para ocultar sua identidade. O primeiro artifício é consequencia do uso de redes hierárquicas para a execução de ataques distribuídos. Com o uso de estações intermediárias para o controle dos zumbis o atacante acaba por adicionar uma camada extra de proteção, pois para se ras-

trear o atacante, é necessário primeiro identificar os zumbis para depois descobrir quem são os mestres e, por fim, chegar até o atacante. Quanto mais camadas existirem nesta hierarquia, mais protegido está o atacante. O segundo artifício usado pelos atacantes para ocultar sua identidade é o uso de endereços IP de origem forjados. Isso possibilita o envio dos pacotes de ataque para a vítima sem deixar nenhum rastro que identifique a estação de origem. Embora dificulte muito a identificação do atacante, o uso de endereços de origem forjados não é necessário para o sucesso de ataques de negação de serviço. Os efeitos dos ataques seriam os mesmos caso fossem usados endereços de origem legítimos. Entretanto, o uso de endereços de origem legítimos possibilitaria a interrupção do ataque em andamento e a identificação das estações de origem dos pacotes de ataque. Dessa forma, seria ingênuo demais acreditar que os atacantes não usam todos os artifícios possíveis para ocultar sua identidade e impedir a adoção de contramedidas ao ataque. Por essa razão, pacotes com endereço de origem forjados são amplamente utilizados em ataques de negação de serviço, conforme afirma o estudo do grupo de segurança CERT (*Computer Emergency Response Team*), da Universidade Carnegie Mellon [32]. A conclusão foi obtida através da análise do comportamento das *botnets* e também do código-fonte de ferramentas automatizadas de ataque.

O quarto ponto é que os mecanismos de segurança convencionais não funcionam contra ataques de negação de serviço. Os mecanismos de segurança tradicionais são *firewalls*, sistemas de detecção de intrusão e sistemas de atualização e correção de *software*. Infelizmente, esses mecanismos são ineficazes no combate a ataques de negação de serviço. Em geral, o tráfego de ataque possui características similares ao tráfego legítimo e, por essa razão, não são bloqueados pelos *firewalls*. Diferentemente de invasões ou roubos, os ataques de negação de serviço não precisam ser detectados, pois a interrupção do serviço já é um alerta do ataque. Com isso, o uso de sistemas de detecção de intrusão contra esse tipo de ataque perde o sentido. Também a atualização e a correção de *software* oferecem pouco suporte ao combate aos ataques de negação de serviço, visto que servidores sem nenhum tipo de vulnerabilidade podem ser atacados. O fator determinante para o sucesso do ataque é a indisponibilidade do serviço oferecido, que pode ser alcançada sem a exploração de vulnerabilidades da vítima.

Uma quinta razão para o sucesso dos ataques de negação de serviço é a topologia assimétrica da Internet. Os enlaces da Internet são assimétricos: no núcleo há enlaces com altas taxas de transmissão, mas na borda, os enlaces são de baixa capacidade. Uma consequencia dessa assimetria é a possibilidade das estações localizadas nas bordas serem inundadas pelo tráfego agregado dos zumbis num ataque distribuído. A assimetria permite que cada zumbi gere apenas um pequena parcela de tráfego e, ainda assim, a vítima seja inundada pelo tráfego agregado dos zumbis, conforme ilustrado na Figura II.3. O pequeno tráfego gerado por cada zumbi pode nem ser detectado como uma anomalia e passar despercebido pelos roteadores.

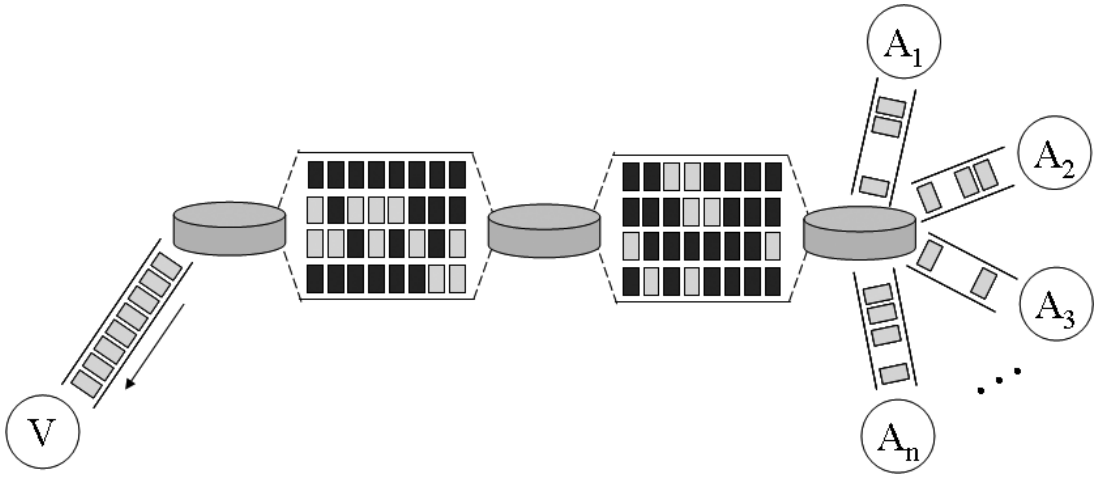

Figura II.3: A topologia assimétrica favorece ataques distribuídos, pois a vítima pode ser inundada mesmo que cada zumbi gere apenas uma pequena parcela do tráfego agregado.

II.3 Rastreamento de Pacotes como Solução

Conforme argumentado no Capítulo I, os princípios de projeto da Internet acabaram por facilitar o crescimento e o sucesso dos ataques de negação de serviço. Por isso defende-se a inclusão de mecanismos de defesa no núcleo da rede para garantir a disponibilidade dos serviços da Internet. Um dos principais fatores para o sucesso dos ataques de negação de serviço é a impunidade que, por sua vez, é consequencia do anonimato. O rastreamento de pacotes surge como o primeiro passo para a identificação dos autores dos ataques de

negação. O rastreamento de pacotes é o foco do próximo capítulo.

Capítulo III

Sistemas de Rastreamento

O rastreamento de pacotes tem dois objetivos principais. O primeiro objetivo é identificar as estações originadoras do tráfego de ataque na tentativa de chegar mais perto e encontrar o criminoso que está controlando a rede de ataque. O segundo objetivo é obter informações parciais da rota de ataque para interromper o tráfego de ataque ou aplicar outras medidas reativas. Dessa maneira, redes que possuem esse tipo de sistema observam um incremento significativo de segurança.

Quando sob ataque, a solução atualmente empregada pelas empresas é bastante precária, pois se serve de procedimentos que requerem a intervenção humana. Em geral, quando ocorre um ataque, a vítima entra em contato com o administrador da rede do seu provedor de serviço (*Internet Service Provider* - ISP). Então, a vítima ou o administrador do ISP procura uma característica dos pacotes usados no ataque ou faz-se uma análise do tráfego no roteador associado à vítima. É feito um teste no roteador mais próximo da vítima para determinar por qual enlace o tráfego do ataque chega a esse último roteador. Não é incomum o bloqueio total do enlace suspeito, o que por si só já é uma negação de serviço ao tráfego legítimo que trafegava pelo enlace bloqueado. Quando possível, continua-se o procedimento de forma a realizar o bloqueio de tráfego somente no local mais próximo da origem do ataque. Assim, uma vez determinado o enlace correto, o procedimento é repetido no roteador localizado na outra extremidade do enlace. Depois, salto-a-salto,

este processo é repetido até que seja determinado o roteador de onde parte o ataque. Em alguns casos, é necessária a interação entre provedores de serviço, o que torna o processo lento. Além disso, esse procedimento, por não ser automatizado, depende da cooperação de outros operadores de rede e o ataque precisa ser suficientemente longo para permitir o rastreamento completo, não sendo possível realizá-lo após o término do ataque.

III.1 Sistemas Propostos na Literatura

Devido a essa dificuldade dos administradores de rede em descobrir a origem de pacotes IP, sistemas de rastreamento automático foram propostos. Os sistemas de rastreamento de pacotes IP podem ser divididos em duas categorias: sistemas de rastreamento sem estado e sistemas de rastreamento baseados em auditoria [33]. Os sistemas de rastreamento sem estado não armazenam nenhum tipo de informação sobre a origem dos pacotes na infraestrutura da rede. Neste caso, toda a atividade de rastreamento se baseia no estado da rede no momento em que o rastreamento é feito ou são usadas as informações contidas no cabeçalho dos pacotes. Por outro lado, os sistemas baseados em auditoria armazenam de alguma forma, durante o processo de envio/encaminhamento/recepção dos dados, informações que podem ser úteis para reconstruir o caminho real percorrido pelos pacotes.

Burch e Cheswick propõem um sistema de rastreamento sem estado baseado em testes de enlaces [34]. Seu funcionamento é baseado na observação do fluxo de ataque recebido pela vítima à medida que cada enlace é inundado por grandes rajadas de tráfego. Apesar de não necessitar de nenhuma intervenção por parte dos operadores da rede, esta técnica exige um mapa topológico apurado assim como possíveis nós dispostos a sofrerem inundações por um período curto de tempo. Neste sistema, é possível que as verdadeiras fontes de um ataque nem cheguem a ser reveladas se o ataque for subitamente interrompido no meio do processo de rastreamento. Além disso, a variação observada na vítima para o caso de ataques distribuídos ou ainda para ataques com pequenos fluxos pode ser imperceptível.

III.1 Sistemas Propostos na Literatura

Savage *et al.* introduzem um sistema probabilístico de marcação de pacotes [17]. Os roteadores informam à vítima sobre sua presença na rota, escrevendo informações de rastreamento em campos pouco usados do cabeçalho IP. Basicamente, os campos usados para fragmentação são sobrecarregados com a informação sobre a rota. Visando reduzir o processamento no roteador e o espaço necessário em cada pacote, técnicas de codificação e amostragem são empregadas. Posteriormente, a vítima consegue reconstruir a rota percorrida por um fluxo de ataque depois de recebido um número suficiente de pacotes deste fluxo. Embora inovadora, a proposta necessita de um alto poder computacional durante a reconstrução da rota de ataque pela vítima e requer um ataque com grande número de pacotes, pois, caso contrário, gera diversos falsos positivos mesmo em ataques distribuídos de pequeno porte [2]. Análises realizadas por Park e Lee mostram ainda a vulnerabilidade do sistema para o caso do atacante forjar as marcações do cabeçalho IP do pacote [35].

Outro método probabilístico foi proposto por Bellovin *et al.* [16]. Ao rotear um pacote, cada roteador probabilisticamente envia para a vítima um pacote ICMP (*Internet Control Message Protocol*) com informações sobre si mesmo e sobre seus roteadores adjacentes. Para um fluxo suficientemente longo, a vítima usa estes dados recebidos para reconstruir a rota de ataque. Entretanto, como as informações de auditoria são enviadas em pacotes separados, a autenticação das mensagens é necessária de forma a evitar mensagens forjadas pelo atacante. Logo, a adoção de uma infraestrutura de chave pública se torna inevitável. Em uma extensão desse trabalho [36], novos conceitos como utilidade e valor das mensagens de rastreamento são introduzidos, assim como uma proposta para melhorá-los.

Os sistemas de rastreamento baseados em auditoria armazenam informações na própria infraestrutura de rede. Sob esta perspectiva, a maneira mais simples de recolher rastros de auditoria é cada roteador registrar todos os pacotes que o atravessam [37]. Porém, recursos excessivos são requisitados tanto para o armazenamento quanto para a mineração dos dados. Além disso, a invasão de um roteador acarretaria ainda em problemas de privacidade, uma vez que ele contém informações sobre todos os pacotes roteados.

Uma alternativa para reduzir o armazenamento de um grande volume de informações é utilizar um Filtro de Bloom [20]. Snoeren et al. propõem um mecanismo que possui a vantagem de rastrear um único pacote IP que tenha passado na rede sem a necessidade de se armazenar todo o tráfego roteado [38]. Para isso, são usados Filtros de Bloom em dispositivos acoplados aos roteadores que armazenam os pacotes roteados de forma compacta. Periodicamente, os filtros saturados são armazenados para futuras requisições e trocados por novos filtros vazios. Para mais tarde determinar se um pacote passou pelo roteador, o seu filtro simplesmente é verificado. Um processo repetitivo pode ser feito por cada roteador para reconstruir o caminho do pacote até a sua verdadeira origem. Porém, mesmo com o uso de Filtros de Bloom, tal sistema exige uma alta capacidade de armazenamento. Melhorias propostas por Li *et al.* diminuem o espaço necessário para o armazenamento, embora a capacidade de se rastrear um único pacote seja comprometida [39].

III.2 O Sistema de Rastreamento Proposto

A proposta desse trabalho é um sistema de rastreamento baseado na marcação de pacotes. Um sistema deste tipo é composto por dois procedimentos distintos: a marcação de pacotes e a reconstrução de rota. A marcação de pacotes é o procedimento no qual cada roteador insere uma informação no pacote para identificar a sua presença na rota de ataque. Esta tarefa é realizada por cada roteador que o pacote atravessa. Dessa forma, ao receber um pacote, a vítima dispõe de informações suficientes para reconstruir a rota de ataque. Este último procedimento é denominado reconstrução de rota e é realizado pela vítima em conjunto com os roteadores da rede. A seguir, os procedimentos de marcação de pacotes e reconstrução de rota do sistema proposto são brevemente explicados.

III.2.1 Procedimento de Marcação de Pacotes

A maneira mais simples de se rastrear a origem de um pacote é fazer com que cada roteador insira o seu próprio endereço IP no pacote roteado. Entretanto, a adição de dados

ao pacote durante o roteamento implica um acréscimo significativo de processamento e um aumento do tamanho do pacote a cada salto. Tal fato pode acarretar em fragmentações desnecessárias dos pacotes, o que pode sobrecarregar os roteadores. No sistema proposto, cada pacote IP possui um campo para armazenar a rota percorrida. Este campo tem tamanho fixo de forma a evitar a fragmentação do pacote. O procedimento de marcação é ilustrado na Figura III.1, onde A representa o atacante, V representa a vítima e R_i representa os roteadores. Pouco antes de encaminhar um pacote, cada roteador insere no pacote alguma informação que possa identificá-lo posteriormente. Na ilustração a rota de ataque é formada pelos roteadores (R_7, R_5, R_2, R_1). O primeiro roteador a marcar o pacote é o R_7. Antes de encaminhar o pacote para o próximo roteador, R_7 insere o seu rastro no cabeçalho do pacote. Depois, R_5 também insere o seu rastro e encaminha o pacote para R_2. O mesmo procedimento é realizado por R_2 e, finalmente, por R_1. Desta forma, ao receber um pacote, a vítima dispõe de informações de todos os roteadores atravessados por aquele pacote. A partir desse ponto, a vítima inicia o procedimento de reconstrução de rota, explicado na Seção III.2.2. O sistema proposto não armazena o endereço IP de cada roteador nos pacotes. De forma a economizar espaço e reduzir o processamento nos roteadores, um Filtro de Bloom Concatenado (FBC), descrito no Capítulo IV, é usado em cada pacote para armazenar a rota atravessada.

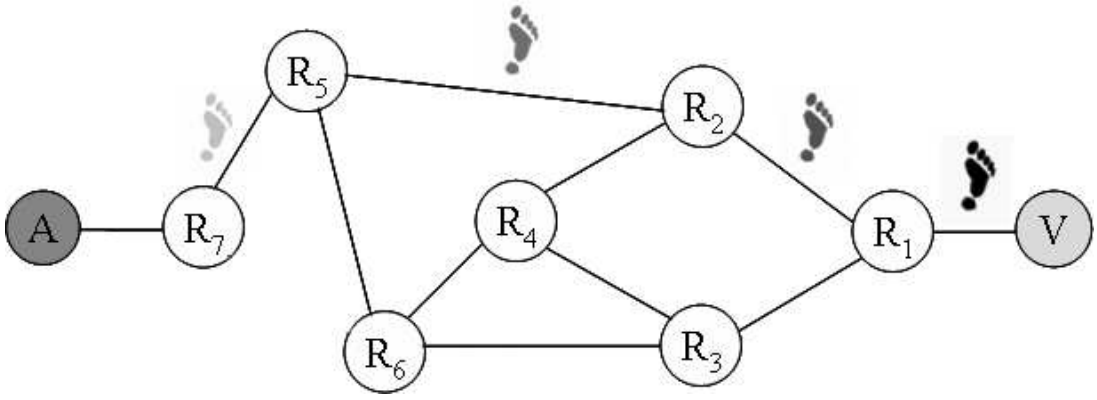

Figura III.1: O procedimento de marcação de pacotes que insere rastros no pacote em cada roteador.

III.2.2 Procedimento de Reconstrução de Rota

Conforme mencionado na seção anterior, a vítima recebe um Filtro de Bloom Concatenado (FBC) que representa a rota atravessada por cada pacote recebido. Entretanto, os endereços dos roteadores não podem ser determinados diretamente pela vítima. Para verificar se um determinado roteador está presente na rota, é preciso testar se o seu endereço IP está contido no FBC. A Figura III.2 ilustra a reconstrução de rota iniciada pela vítima V em direção ao atacante A. No exemplo, a rota de ataque a ser reconstruída é (R_7, R_5, R_2, R_1). Ao receber o pacote de ataque, a vítima inicia o procedimento de reconstrução, testando a presença de R_1 no filtro do pacote recebido. Como R_1 é reconhecido, ele recebe o filtro de V e continua o procedimento. Por sua vez, R_1 verifica a presença dos seus vizinhos R_2 e R_3 no filtro. Como somente R_2 é reconhecido, o filtro é então repassado somente para R_2. O roteador R_2 verifica qual dos seus dois vizinhos R_4 e R_5 é reconhecido pelo filtro; somente R_5 é reconhecido. Finalmente, R_5 testa a presença de R_7 no filtro. R_7 é reconhecido pelo filtro e incorporado ao grafo de reconstrução e o procedimento termina.

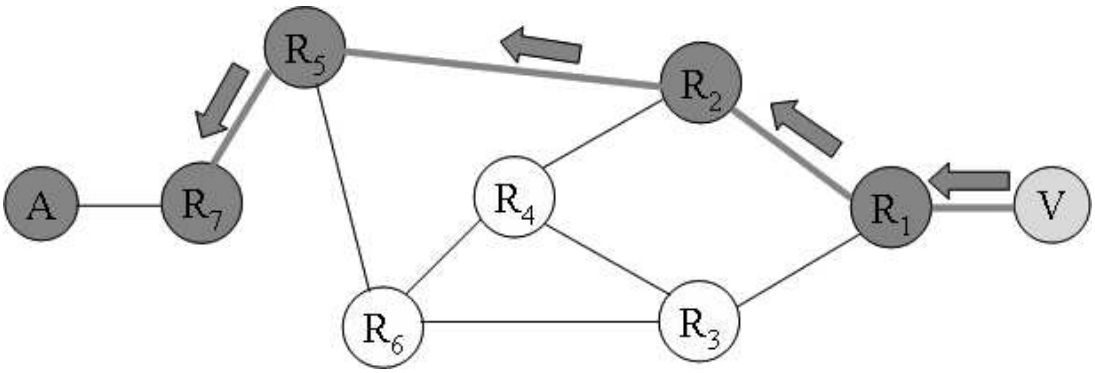

Figura III.2: O procedimento de reconstrução de rota é feito de forma recursiva a partir da vítima.

III.2.3 Características Importantes

O sistema proposto possui duas características importantes: pode rastrear ataques de até um único pacote e não armazena estado na infraestrutura de rede. As vantagens do

rastreamento por um único pacote são a escalabilidade e a robustez à burla do atacante. Conforme mencionado no Capítulo II, há relatos de ataques distribuídos com milhões de zumbis. Dessa forma, a fim de passar despercebido pela rede, é possível que cada zumbi envie uma pequena quantidade de pacotes. Esse artifício consegue burlar os sistemas de rastreamento que precisam de múltiplos pacotes para iniciar a reconstrução de rota. Por essa razão, o sistema de rastreamento ideal deve ser capaz de rastrear ataques de até 1 pacote por zumbi. A segunda característica importante da proposta é o rastreamento sem estado. Isso permite a implantação do sistema proposto mesmo em redes de altíssima velocidade.

Capítulo IV

Filtros de Bloom Robustos a Ataques

FILTRO de Bloom é uma estrutura de dados capaz de representar um conjunto de forma compacta [20]. Ele permite o armazenamento de apenas um vetor de bits, ao invés de todos os elementos do conjunto a ser representado. Para isso, são aplicadas técnicas de sumarização, através do uso de funções *hash*. O custo dessa compacidade é a possibilidade de um elemento externo ao conjunto ser reconhecido como um elemento legítimo do conjunto representado. Tal evento é conhecido como um falso positivo.

O Filtro de Bloom consiste em um vetor de bits, inicialmente zerados, ao qual podem ser aplicados dois algoritmos. O primeiro algoritmo é o de inserção de elemento, que permite a inclusão de um dado elemento no conjunto representado pelo filtro. O outro algoritmo é denominado teste de pertinência. Trata-se de uma verificação se um dado elemento pertence ou não ao conjunto representado pelo filtro. Se, previamente, o elemento tiver sido submetido ao algoritmo de inserção, então certamente o teste de pertinência é positivo, isto é, o elemento é reconhecido pelo filtro como pertencente ao conjunto de elementos inseridos. Porém, devido ao esquema de sumarização utilizado, existe uma probabilidade não nula de que ocorra um falso positivo, ou seja, que um elemento externo seja reconhecido como pertencente ao conjunto de elementos inseridos no filtro. Além disso, à medida que novos elementos vão sendo inseridos, ocorre um processo de saturação do Filtro de Bloom, acarretando altas taxas de falso positivo. Entretanto,

a economia de espaço justifica o uso de Filtros de Bloom, desde que a probabilidade de falso positivo seja mantida pequena.

Filtros de Bloom foram inicialmente utilizados nas áreas de verificação ortográfica e banco de dados. Recentemente, verificou-se que eles podem garantir uma troca eficiente de informações em aplicações distribuídas. Desde então surgiram novas propostas de aplicações na área de redes de computadores, como colaboração em redes *overlay* e *peer-to-peer* (P2P), localização de recursos, protocolos de roteamento e infraestruturas de medição [40]. Por exemplo, nas aplicações de redes *peer-to-peer* (P2P) e localização de recursos, cada nó envia periodicamente em difusão (*broadcast*) um Filtro de Bloom representando os objetos daquele nó ou que podem ser alcançados através dele. Dessa maneira, todos os nós possuem conhecimento dos objetos que podem ser alcançados e, ao invés de usar, por exemplo, um identificador de 64 bits para cada objeto, são necessários apenas 8 ou 16 bits por objeto. Seguindo a mesma ideia de enviar o Filtro de Bloom no lugar de uma lista explícita de elementos, na aplicação de distribuição de *Web cache* servidores *proxy* trocam Filtros de Bloom a fim de compartilhar o conteúdo do *cache* de cada servidor. No caso de um servidor não possuir uma página requisitada, ele primeiro verifica os filtros dos outros servidores para determinar se eles a possuem em *cache* ou não. No caso afirmativo, a requisição da página é repassada àquele servidor. A pequena probabilidade de falso positivo adicional introduzida pelo uso do Filtro de Bloom é amplamente justificada pela significativa redução do tráfego de rede devido à sumarização obtida com o uso do Filtro de Bloom.

Apesar de sua comprovada eficiência, o uso de Filtros de Bloom nessas aplicações distribuídas é limitado por aspectos de segurança. A questão principal gira em torno da condição inicial do filtro. Nesse capítulo é apresentado um ataque no qual a condição inicial do filtro é cuidadosamente ajustada no intuito de comprometer aplicações distribuídas nas quais o filtro é enviado pela rede. É mostrado que o atacante pode maximizar a probabilidade de falso positivo dos elementos desejados ou até mesmo saturar o filtro. Um filtro saturado torna impossível distinguir elementos legítimos de falsos positivos em um teste de pertinência, apresentando taxas de falso positivo de até 100%. Os efeitos

de ataques desse tipo nas aplicações mencionadas anteriormente podem ser catastróficos. Nas aplicações de compartilhamento de *Web cache* e de redes P2P, os demais nós acreditarão que o atacante possui todas as páginas *Web* ou objetos procurados. Os pedidos serão encaminhados ao atacante, que poderá substituir os objetos procurados por outros ao seu bel prazer.

No caso do rastreamento de pacotes IP, o atacante pode comprometer a acurácia do sistema de rastreamento. Ao maximizar a probabilidade de falso positivo de roteadores que não pertencem à rota de ataque, o atacante faz com que sejam rastreadas outras rotas além da verdadeira. Mostra-se que, num caso extremo, o atacante pode saturar o Filtro de Bloom, fazendo com que todas as rotas da Internet sejam rastreadas. Consequentemente, a reconstrução de rota torna-se completamente ineficaz.

Apesar das falhas de segurança, pouco se tem estudado sobre mecanismos eficientes de segurança para Filtros de Bloom. Nesse trabalho, é proposto um Filtro de Bloom robusto à interferência da condição inicial do filtro. A estrutura de dados proposta é denominada Filtro de Bloom Concatenado (FBC). A ideia chave é sobrescrever as informações iniciais, a fim de reduzir a interferência da condição inicial do filtro. Uma proposta existente na literatura que visa tal objetivo é o Filtro de Bloom Generalizado (FBG) [41]. O FBG é uma proposta pioneira de um filtro seguro, porém possui a limitação de perda de informação, devido à possibilidade de informações inseridas pelos primeiros elementos serem sobrescritas por inserções posteriores. Para contornar esse problema, propõe-se dividir o filtro original em diversos subfiltros com somente um elemento. O Filtro de Bloom Concatenado (FBC) é o resultado da concatenação desses subfiltros. Comparado ao FBG, o filtro proposto possui como principais vantagens a ausência de falsos negativos, a maior capacidade de armazenamento e a maior robustez à interferência da condição inicial.

IV.1 Filtro de Bloom

O Filtro de Bloom é uma estrutura de dados capaz de representar eficientemente um conjunto $S = \{s_1, s_2, \ldots, s_n\}$ de n elementos. Ele é constituído por um vetor de m bits, inicialmente todos zerados. São admitidos dois algoritmos: inserção de elemento e teste de pertinência. O Filtro de Bloom associado a S é o vetor resultante da aplicação do algoritmo de inserção aos n elementos de S. A aplicação do teste de pertinência a um elemento x informa se x pertence ou não a S, com uma certa probabilidade de falso positivo associada.

IV.1.1 Algoritmo de Inserção de Elemento

Para inserir um elemento no filtro, são utilizadas k funções *hash* independentes $h_1, h_2, \ldots, h_k$, cujas saídas variam uniformemente no espaço discreto $\{0, 1, \ldots, m-1\}$. O Algoritmo 1 recebe como argumentos de entrada um elemento $s_i \in S$ $(1 \leq i \leq n)$ e o Filtro de Bloom B. O passo seguinte é o preenchimento com 1 dos bits do vetor B correspondentes às posições $h_1(s_i), h_2(s_i), \ldots, h_k(s_i)$. Ao final desse processo, o elemento s_i está inserido no Filtro de Bloom.

Algoritmo 1: Inserção de Elemento no Filtro de Bloom

Entrada: $s_i \in S$ e o vetor B de m bits.

Resultado: Inserção do elemenho s_i no filtro B

início

  para $j = 1$ **até** k **faça**
    $B[h_j(s_i)] \leftarrow 1$

fim

IV.1.2 Algoritmo de Teste de Pertinência

Para determinar se um elemento x pertence ou não ao conjunto S, é necessário aplicar o algoritmo de teste de pertinência. O Algoritmo 2 recebe como argumentos de entrada o elemento x e o Filtro de Bloom B. Verificar a pertinência de x significa checar se x já foi inserido no Filtro de Bloom ou não. Assim, o objetivo é verificar se os bits do vetor B correspondentes às posições $h_1(x), h_2(x), \ldots, h_k(x)$ estão preenchidos com 1. Se pelo menos um bit estiver zerado, então certamente $x \notin S$, pois se x já tivesse sido inserido no Filtro de Bloom anteriormente, então todos os bits indicados pelas funções *hash* deveriam estar preenchidos com 1. Nesse caso, o algoritmo retorna 0, indicando que o teste de pertinência foi negativo. Por outro lado, se todos os bits estão preenchidos com 1, então é assumido que $x \in S$ e o algoritmo retorna 1, indicando que o teste de pertinência foi positivo.

Algoritmo 2: Teste de Pertinência no Filtro de Bloom

Entrada: Elemento x e o vetor B de m bits.

Saída: 1, se $x \in S$; 0, caso contrário.

```
início
    para j = 1 até k faça
        se B[h_j(x)] = 0 então retorna 0
    retorna 1
fim
```

Cabe aqui fazer uma observação. Se, de fato, $x \in S$, então o teste de pertinência é sempre positivo. Isso ocorre porque nenhuma inserção posterior pode zerar um bit preenchido com 1. Dessa maneira, todos os bits preenchidos na inserção de x estão sempre em 1 e, portanto, o teste de pertinência é sempre positivo.

IV.1.3 Falsos Positivos

É importante notar que um elemento externo $x \notin S$ pode ser reconhecido como um elemento legítimo do conjunto, evento denominado falso positivo. Tal evento ocorre quando

todos os bits $h_1(x), h_2(x), \ldots, h_k(x)$ foram preenchidos com 1 por elementos de S previamente inseridos no filtro. A probabilidade de se encontrar um falso positivo para um elemento $x \notin S$ é calculada da seguinte maneira. Como as funções *hash* usadas são uniformes e independentes, a probabilidade p de um determinado bit permanecer em zero mesmo depois de inseridos os n elementos é

$$p = \left(1 - \frac{1}{m}\right)^{kn}. \tag{IV.1}$$

Como o mesmo cálculo se aplica a todos os bits do vetor B, na média, uma fração p dos bits permanece zerada após as inserções [42]. Dessa forma, a fração média de bits preenchidos com 1 depois de n inserções é $1 - p$. A probabilidade de falso positivo f_p é a probabilidade de se encontrar um bit em 1 para todas as k posições indicadas, ou seja,

$$f_p = \left(1 - p\right)^k = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k. \tag{IV.2}$$

IV.2 Extensões do Filtro de Bloom

A literatura acerca dos Filtros de Bloom é bastante vasta. Encontram-se também diversas variações do Filtro de Bloom convencional. Dentre elas, destacam-se o Filtro de Bloom Contador (*Counting Bloom Filter*), que admite não somente a inserção, mas também a retirada de elementos [43]; o Filtro de Bloom Paralelo, usado para inspeção de cadeias de octetos dentro de pacotes [44]; o Filtro de Bloom Comprimido (*Compressed Bloom Filter*), que permite a transmissão eficiente de um filtro entre duas estações [42]; o Filtro de Bloom Hierárquico (*Hierarchical Bloom Filter*) que possibilita a busca de subcadeias de caracteres [45]; o Filtro de Bloom Atenuado (*Attenuated Bloom Filter*), que melhora a localização e o roteamento de documentos em redes *peer-to-peer* (P2P); e o Filtro de Bloom Espectral (*Spectral Bloom Filter*), que retorna uma estimativa do número de ocorrências de um dado elemento no filtro [46]. Outra variação, proposta por Estan e Varghese, é um Filtro de Bloom com suporte a multiconjuntos usado para identificar

fluxos intensos de dados em um roteador [47]. Seguindo a abordagem de representação de multiconjuntos, foi proposto também o *Space-Code Bloom Filter* [48].

Apesar de serem eficientes para as aplicações a que se destinam, as propostas aqui mencionadas não levam em consideração aspectos de segurança. Dessa maneira, essas aplicações podem estar sujeitas a ataques que explorem as vulnerabilidades dos respectivos Filtros de Bloom. Uma proposta pioneira de um filtro seguro é o chamado Filtro de Bloom Generalizado (FBG), que visa deixar o Filtro de Bloom menos dependente da sua condição inicial [41]. Como o FBG tem o mesmo objetivo do Filtro de Bloom Concatenado proposto nesse trabalho, ele será brevemente descrito a seguir.

IV.2.1 Filtro de Bloom Generalizado

Assim como o Filtro de Bloom original, o Filtro de Bloom Generalizado (FBG) é uma estrutura de dados capaz de representar um dado conjunto de forma compacta. A originalidade é a introdução de funções *hash* que zeram bits, além daquelas que preenchem bits com 1. Os autores mostram que isso permite ao filtro ser menos dependente da sua condição inicial, já que agora a probabilidade de falso positivo é limitada, independentemente do estado inicial do filtro [27]. O custo dessa vantagem é a introdução de falsos negativos, isto é, agora um elemento pertencente ao conjunto representado pelo filtro pode não ser reconhecido como tal. O FBG também admite os algoritmos de inserção de elemento e de teste de pertinência. Entretanto, não há mais a restrição de que os bits do filtro estejam inicialmente zerados. No FBG, os bits do filtro podem assumir qualquer valor inicial.

O algoritmo de inserção de elemento é semelhante ao do caso convencional. São utilizadas $k_0 + k_1$ funções *hash* independentes $g_1, g_2, \ldots, g_{k_0}, h_1, h_2, \ldots, h_{k_1}$ cujas saídas variam uniformemente no espaço discreto $\{0, 1, \ldots, m-1\}$. Os bits correspondentes às posições $h_1(s_i), h_2(s_i), \ldots, h_{k_1}(s_i)$ são preenchidos com 1 e os bits correspondentes às posições $g_1(s_i), g_2(s_i), \ldots, g_{k_0}(s_i)$ são zerados.

Para determinar se um elemento x pertence ou não ao conjunto S, é necessário aplicar o algoritmo de teste de pertinência. O objetivo é verificar se os bits correspondentes às

posições $g_1(x), g_2(x), \ldots, g_{k_0}(x)$ estão zerados e se os bits $h_1(x), h_2(x), \ldots, h_{k_1}(x)$ estão preenchidos com 1. Se pelo menos um bit está invertido, então é assumido que $x \notin S$ e o teste de pertinência é negativo. Diferentemente do Filtro de Bloom convencional, agora há uma possibilidade de um elemento $x \in S$ não ser reconhecido pelo filtro, evento denominado falso negativo. Por outro lado, se nenhum bit está invertido, então é assumido que $x \in S$ e o teste de pertinência é positivo.

Uma análise detalhada das probabilidades de falso positivo e falso negativo do FBG pode ser obtida em [27]. Um resultado interessante é que, quando $m \gg k_0$ e $m \gg k_1$, a probabilidade de falso positivo do FBG é limitada e seu máximo vale

$$\left(\frac{k_0}{k_0 + k_1}\right)^{k_0} \left(\frac{k_1}{k_0 + k_1}\right)^{k_1}. \tag{IV.3}$$

No entanto, os cálculos realizados consideram uma escolha aleatória da condição inicial do filtro, na qual todos os bits têm a mesma probabilidade p_0 de estar em 0. Portanto, escolhendo cuidadosamente os bits iniciais do filtro, um atacante fazer com que a probabilidade de ser positivo o teste de pertinência de determinados elementos que não pertencem ao filtro seja maior do que o limite superior aqui mostrado. Esse ataque e sua respectiva probabilidade de sucesso são apresentados em detalhe mais adiante, nas Seções IV.3 e V.1.2, respectivamente. É mostrado ainda que a probabilidade de sucesso do atacante é bem menor quando o Filtro de Bloom Concatenado é utilizado.

Outra consideração que deve ser feita é que os falsos negativos introduzidos com o FBG são devidos, essencialmente, à possibilidade de inversão de um bit de um elemento durante a inserção de elementos posteriores no filtro. Apesar de serem responsáveis pela independência da condição inicial do filtro, os demais efeitos dos falsos negativos são bastante indesejáveis. Se o Filtro de Bloom for encarado como uma espécie de *buffer* de armazenamento dos elementos inseridos, um efeito bastante maléfico dos falsos negativos é a perda de informação, interpretada como um fenômeno de "esquecimento" dos elementos antigos. Em outras palavras, a introdução de falsos negativos causa perda de informação, ou seja, a limitação de memória desse *buffer*. A memória aqui não se refere a espaço

ou número de bits, mas se refere a deixar de "lembrar" ou representar a pertinência de um elemento do conjunto de elementos inseridos. O Filtro de Bloom original possui memória ilimitada, isto é, não há perda de informação e, portanto, o filtro não "esquece" nenhum dos elementos inseridos por mais que se insiram novos elementos, o que equivale a dizer que não existem falsos negativos. Já o FBG não pode aceitar a inserção de tantos elementos quanto se queira, pois invariavelmente existirá um elemento que irá inverter um dos bits de um elemento inserido anteriormente, ocasionando o "esquecimento" do elemento mais antigo. Por exemplo, esse fenômeno de "esquecimento" dos elementos antigos foi observado nas simulações do sistema de rastreamento proposto por Laufer *et al.*, nas quais somente cerca de 7 roteadores foram rastreados, para um filtro de 256 bits e uma rota de 15 saltos [49]. Isso significa que a probabilidade de inversão de bits foi grande o suficiente para que os 8 primeiros roteadores inseridos no filtro fossem "esquecidos" pelo filtro com a inserção do endereço de novos roteadores no filtro. Conclui-se que a inversão de bits tem como consequência direta a perda das informações mais antigas, a fim de que novas informações possam ser inseridas. Por outro lado, quanto mais bits forem invertidos, maior é a probabilidade das informações iniciais do filtro serem apagadas. Consequentemente, maior é a robustez à interferência da condição inicial do filtro. Assim, uma medida da robustez do filtro é a probabilidade de falso negativo, após n inserções subsequentes, de um elemento que era reconhecido pelo filtro devido à condição inicial. A Figura IV.1 ilustra, para um FBG de 128 bits, o compromisso entre a robustez e a perda de informação legítima, medida pela probabilidade de "esquecimento" do primeiro elemento legítimo inserido.

Da figura, pode-se observar que a robustez do FBG cresce com o aumento do número de elementos inseridos no filtro. Nota-se que, mesmo com poucos elementos inseridos no filtro, a robustez já é bastante elevada. Porém, pode-se notar que a probabilidade de perda de informação acompanha a curva da robustez. Por exemplo, na curva onde $k = k_0 = k_1 = 8$, com apenas 5 inserções já se tem uma robustez de $\approx 98{,}2\%$, mas isso significa uma probabilidade de "esquecimento" do primeiro elemento legítimo de $\approx 96{,}5\%$, ou seja, a chance de que haja perda de informação é grande. Logo, não se podem usar muitas funções *hash*, pois isso causa um indesejável fenômeno de "esquecimento" dos

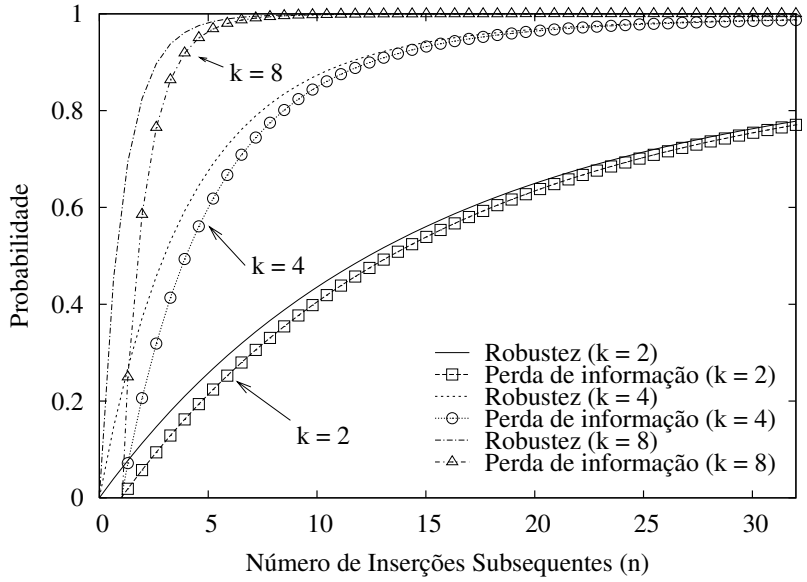

Figura IV.1: Compromisso entre robustez e perda de informação para um FBG.

elementos mais antigos. Assim, deve-se escolher uma outra curva, na qual $k = k_0 = k_1$ seja menor. Por exemplo, na curva onde $k = 2$, a perda de informação cai para $\approx 22\%$, porém agora a robustez agora é de apenas 27%. Nesse caso, como $k = k_0 = k_1 = 2$, da Equação IV.3, a probabilidade máxima de falso positivo do FBG é 6,25%. Será mostrado que o novo filtro proposto neste trabalho, o FBC, tem uma probabilidade de falso positivo equivalente quando são usados 4 bits por elemento, o que corresponde a uma robustez de $93,75\%$, que é bem maior do que os 27% do FBG. Nota-se que no FBG foram usados $128/5 = 25,6$ bits por elemento, enquanto que no FBC foram necessários apenas 4 bits por elemento, para uma mesma probabilidade de falso positivo. Portanto, conclui-se que o filtro concatenado proposto alcança maior robustez usando menos bits. A ideia chave do filtro concatenado é a utilização subfiltros que admitem somente a inserção de um elemento cada. Dessa maneira, não existem inserções posteriores nos subfiltros que possam inverter bits dos elementos mais antigos e, portanto, elimina-se a possibilidade de "esquecimento" dos elementos mais antigos. Assim, a probabilidade de falso negativo é sempre nula como é a do Filtro de Bloom convencional. Resumindo, para uma mesma probabilidade de falso positivo, a proposta desse trabalho supera o FBG em robustez e compacidade. Essas e outras vantagens do FBC são detalhadas na Seção IV.4.

IV.3 O Ataque do Ajuste da Condição Inicial

O ataque descrito nessa seção é bem simples. Ele se inspira na aplicação do Filtro de Bloom ao rastreamento de pacotes. Nessa aplicação, cada roteador insere seu endereço no Filtro de Bloom contido no cabeçalho do pacote IP. Desta forma, ao receber um pacote de ataque, a vítima dispõe de um filtro cujos elementos são todos os roteadores componentes da rota atravessada. Com isso, a vítima pode reconstruir a rota de ataque, seguindo o procedimento descrito na Seção III.2.2. Essa situação é esquematizada na Figura IV.2.

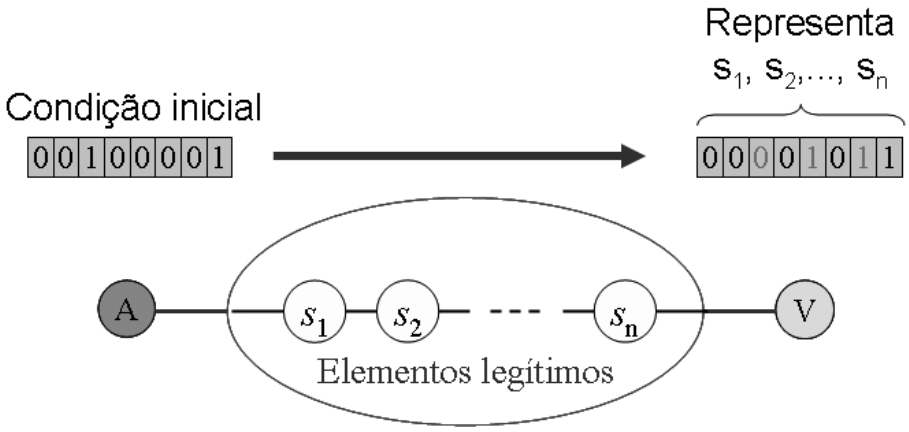

Figura IV.2: Esquema mostrando a informação disponível à vítima após a marcação do pacote pelos roteadores da rota de ataque.

É importante notar que é o atacante quem inicializa o filtro de cada pacote enviado e esse filtro é depois modificado pelos roteadores atravessados. Assim, como o atacante detém o controle sobre a condição inicial do filtro, se a marcação dos pacotes não for robusta à interferência da condição inicial, então o rastreamento pode ser burlado, conforme mostrado a seguir.

Sejam U o conjunto de todos os roteadores da Internet, $S = \{s_1, s_2, \ldots, s_n\}$ o conjunto dos roteadores integrantes da rota de ataque e $A \subset (U \setminus S)$ um conjunto de roteadores que não pertencem à rota de ataque. O objetivo do ataque proposto é maximizar a probabilidade de falso positivo dos elementos de A, somente ajustando a condição inicial do filtro. Uma maneira simples de implementar tal ataque é simplesmente inserir um conjunto $B \subset U$ de roteadores no filtro antes de enviá-lo aos outros nós da rede. Essa

situação é esquematizada na Figura IV.3.

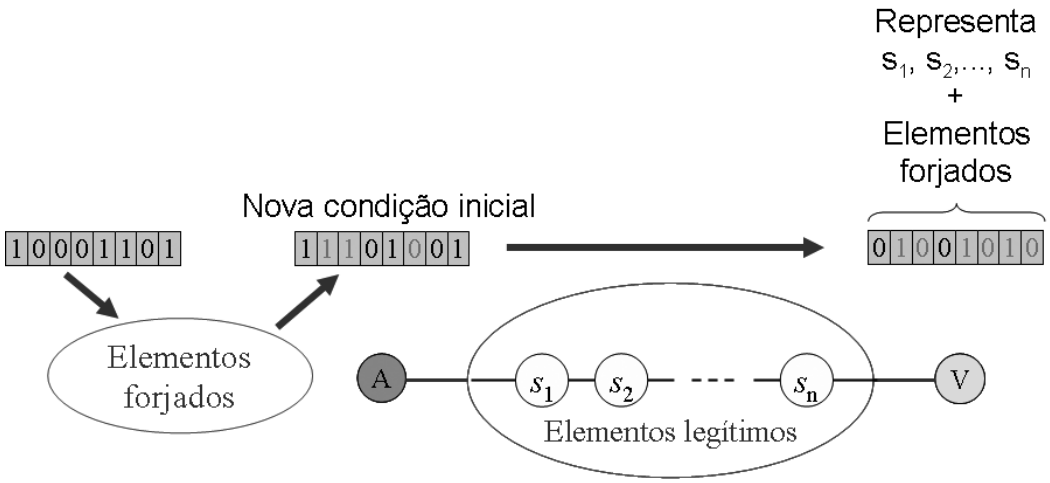

Figura IV.3: O ataque do ajuste da condição inicial com a inserção forjada de alguns elementos antes de enviar o pacote.

Caso a marcação dos pacotes não seja robusta à interferência da condição inicial e, portanto, não haja uma maneira de retirar esses elementos do filtro, invariavelmente eles terão teste de pertinência positivo. E ainda, se $B \subset A$, então esses elementos serão falsos positivos, pois certamente eles não pertencem a S. A consequencia do ataque é a criação de sufixos arbitrários forjados pelo atacante durante o processo de reconstrução de rota. A Figura IV.4 mostra esse efeito da criação de sufixos forjados pelo atacante. A rota real de ataque é (R_7, R_5, R_2, R_1). Mas, no exemplo, o atacante inseriu os endereços dos roteadores R_4 e R_6 no filtro antes de enviá-lo à vítima. Com isso, durante a reconstrução de rota esses roteadores foram incorporados ao grafo de reconstrução, que passou a conter o sufixo forjado (R_6, R_4).

Um caso particular interessante desse ataque ocorre quando $B = U$, ou seja, quando a inicialização satura o filtro com a inserção de todos os elementos do conjunto universo. Isso pode ser facilmente implementado preenchendo-se todos os bits do Filtro de Bloom original com 1. Uma consequência direta desse ataque é o comprometimento da acurácia do sistema de rastreamento, já que são rastreadas outras rotas além da verdadeira rota de ataque. No caso extremo em que $B = U$, o grafo de reconstrução é formado pelo conjunto de todas as rotas da Internet, quando o ideal seria que ele fosse composto apenas pela rota

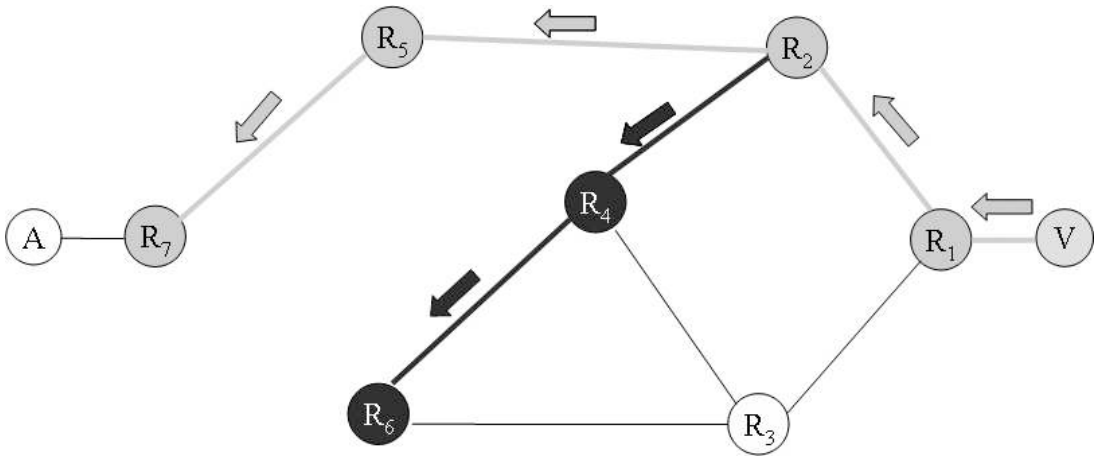

Figura IV.4: Criação de sufixos arbitrários forjados pelo atacante como resultado do ataque do ajuste da condição inicial.

de ataque. Logo, percebe-se que o Filtro de Bloom original é um alvo fácil para ataques desse tipo. Por outro lado, o Filtro de Bloom Generalizado (FBG) foi projetado para ser robusto a ataques que visem a saturação do filtro. Com o uso do FBG, a probabilidade máxima de falso positivo, expressa pela Equação IV.3, é independente do estado do filtro e só depende do número de funções *hash* usadas. No entanto, os cálculos realizados supunham uma escolha aleatória dos bits iniciais. O problema principal é que o FBG não apaga os bits inicializados pelo atacante. Dessa forma, os bits iniciais podem influenciar os falsos positivos. Portanto, apesar de ser bem mais robusto do que o filtro original, o FBG está sujeito a ataques nos quais haja um ajuste cuidadoso da condição inicial do filtro. O ataque descrito nessa seção se encaixa perfeitamente nesse perfil e é bem simples de ser realizado. Basta que o atacante conheça o algoritmo de inserção do filtro.

Para o FBG, uma medida da robustez do filtro é a probabilidade de falso negativo dos elementos mais antigos. Quanto maior essa probabilidade, maior é a probabilidade de um elemento inserido pelo atacante não ser reconhecido em um teste de pertinência. No entanto, foi mostrado na Seção IV.2.1 que um aumento da probabilidade de falso negativo implica redução da capacidade de armazenamento do filtro. Há, portanto, um compromisso entre robustez e capacidade de armazenamento. A ideia fundamental do Filtro de Bloom Concatenado é justamente quebrar o compromisso da robustez com a

capacidade de armazenamento, permitindo a coexistência de uma alta robustez com uma alta capacidade de armazenamento. A construção do FBC é feita de forma a desacoplar a probabilidade de falso negativo da robustez do filtro: os falsos negativos foram eliminados e a robustez passou a ser garantida pelo apagamento das informações iniciais. Esses resultados ficarão mais evidentes na próxima seção. Na Seção V.1.2 é analisada a probabilidade de sucesso do atacante para cada um dos filtros apresentados nesse trabalho.

IV.4 O Filtro de Bloom Concatenado

Nessa seção é apresentada a proposta desse trabalho. Assim, como o Filtro de Bloom convencional, o Filtro de Bloom Concatenado (FBC) é uma estrutura de dados capaz de representar um conjunto de forma compacta. O FBC, porém, possui características adicionais que o tornam mais adequado para aplicações distribuídas nas quais a questão da segurança é crucial. A ideia central do filtro proposto é concatenar diversos filtros pequenos, denominados subfiltros. Somente um elemento é inserido em cada subfiltro. Duas vantagens principais advêm dessa construção. A primeira delas, intrinsecamente relacionada à questão da segurança, é que se podem sobrescrever as informações iniciais contidas no filtro, garantindo a robustez à interferência da condição inicial. A segunda vantagem é a garantia de uma probabilidade de falso negativo nula, já que um bit de um elemento inserido no filtro jamais será invertido por uma futura inserção.

IV.4.1 Definição

Sejam m e N inteiros positivos tais que m é múltiplo de N. O FBC é definido como a concatenação de N vetores, denominados subfiltros, de m/N bits cada. O FBC é, portanto, constituído por um vetor de $N * (m/N) = m$ bits, que podem assumir qualquer valor inicial. A Figura IV.5 ilustra a definição do FBC.

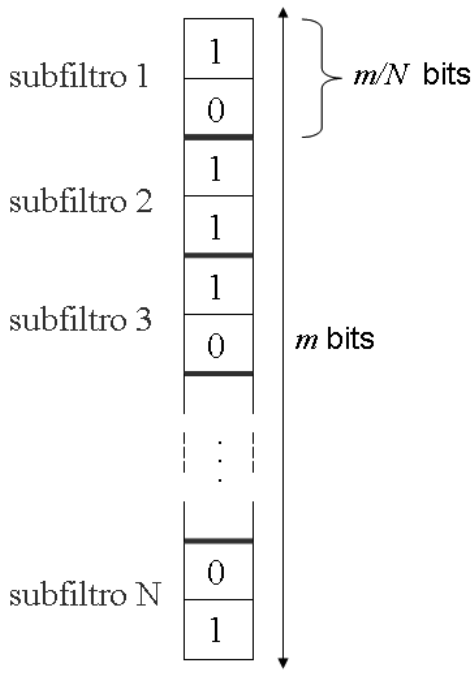

Figura IV.5: Formato do FBC com a concatenação de N subfiltros de m/N bits cada.

IV.4.2 Algoritmo de Inserção de Elemento

Seja um conjunto $S = \{s_1, s_2, \ldots, s_n\}$ de n elementos ($n \leq N$). A inserção dos elementos de S é feita sequencialmente de acordo com o índice de cada elemento. Assim, o elemento s_i ($1 \leq i \leq n$) deve ser inserido no i-ésimo subfiltro. O número máximo de elementos admitidos pelo FBC é N, que é arbitrado pelo projetista do filtro. A escolha do valor de N é uma decisão de projeto e varia de acordo com a aplicação. Em geral, N deve ser a cota superior para a estimativa do número de elementos que são inseridos, de forma que todos os elementos sejam admitidos pelo filtro, mesmo no pior caso. Para algumas aplicações, como é o caso do rastreamento de pacotes IP, é vantajoso que N seja fixo. Porém, se a aplicação permitir, o valor de N pode ser dinâmico, bastando alocar os subfiltros à medida que novos elementos são inseridos. Nesse último caso, a vantagem é o não desperdício de espaço com subfiltros inutilizados. Em todo caso, nesse trabalho assume-se N fixo e suficientemente grande para que todas as inserções necessárias possam ser efetuadas.

O Algoritmo 3 realiza a inserção de elemento no FBC. O algoritmo recebe como

argumentos de entrada um elemento $s_i \in S$ e o Filtro de Bloom Concatenado C.

Algoritmo 3: Inserção de Elemento no Filtro de Bloom Concatenado

Entrada: $s_i \in S$ e o vetor C de m bits.

Resultado: Inserção do elemenho s_i no filtro C

início

  selecione *esquema* **faça**

    caso *esquema 1*

      para $j = 0$ **até** $(m/N) - 1$ **faça**

        $C[i * (m/N) + j] \leftarrow 0$

      $inserirElementoFB(s_i, C[i * (m/N)])$

    caso *esquema 2*

      $C[i * (m/N)] \leftarrow H(s_i)$

fim

O algoritmo aplica ao elemento s_i um dos dois esquemas heurísticos de marcação diferentes propostos nesse trabalho. O esquema de marcação pode ser interpretado como a forma de "assinatura" de cada elemento. O primeiro esquema é ilustrado na Figura IV.6. Esse esquema é uma adaptação do algoritmo de inserção de elemento do Filtro de Bloom original (Seção IV.1) na qual os zeros também representam informação. Essa adaptação tem como objetivo inverter, o máximo possível, os bits iniciais do filtro, no intuito de aumentar a robustez à interferência da condição inicial. O resultado dessa adaptação é analisado na Seção V.1.2. O segundo esquema de marcação é bem simples. O vetor de marcação é simplesmente o resultado da aplicação de uma função *hash* $H : S \rightarrow \{0, 1, \ldots, 2^{m/N} - 1\}$ ao elemento s_i. Esse esquema é similar ao utilizado em [50].

A Figura IV.7 mostra um exemplo no qual o elemento s_2 é inserido no FBC. O subfiltro no qual o elemento vai ser inserido é indicado por um contador. No exemplo, o contador vale 2 e, por isso, o elemento é inserido no segundo subfiltro. É aplicado o esquema de marcação sobre o elemento s_2 e o resultado é colocado no segundo subfiltro, substituindo o conteúdo original desse subfiltro. A principal vantagem do algoritmo de inserção proposto para o FBC é a garantia do apagamento da informação original contida nos subfiltros onde

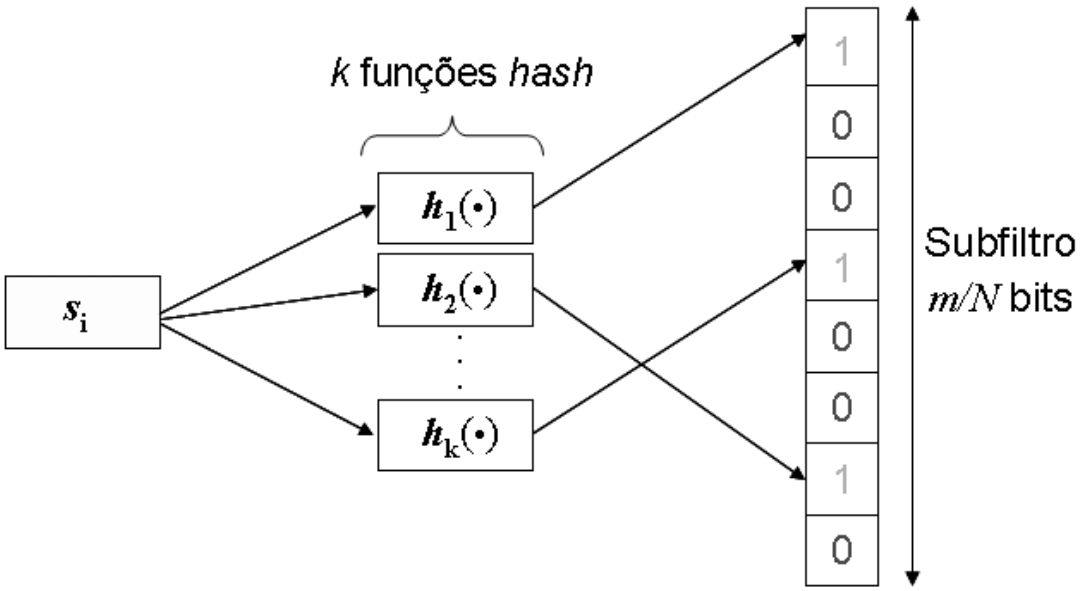

Figura IV.6: Esquema de marcação 1 com subfiltros inicialmente zerados e aplicação do algoritmo de inserção do Filtro de Bloom convencional.

houve uma inserção. Outra vantagem é que, ao contrário do Filtro de Bloom convencional e do FBG, se pode paralelizar o acesso ao vetor de bits.

IV.4.3 Algoritmo de Teste de Pertinência

Propõem-se dois algoritmos de teste de pertinência para o FBC, que diferem basicamente nas suposições que são realizadas. Seja U o conjunto universo de todos os elementos. O primeiro algoritmo supõe que existe um mecanismo auxiliar, como um contador, que informa o índice do subfiltro no qual qualquer elemento $x \in U$ foi inserido ou, caso não tenha sido inserido, o mecanismo auxiliar informa o único subfiltro no qual x teria alguma chance de ser inserido. Dessa maneira, não é necessário checar todos os subfiltros para determinar a pertinência de x, bastando somente checar o subfiltro indicado pelo mecanismo auxiliar. O segundo algoritmo não faz nenhuma suposição adicional, sendo necessária a verificação de todos os subfiltros. Caso o elemento seja reconhecido por pelo menos um dos subfiltros, então o elemento é considerado membro do grupo. Pode-se questionar se a hipótese do primeiro algoritmo é viável ou não. A viabilidade dessa hipó-

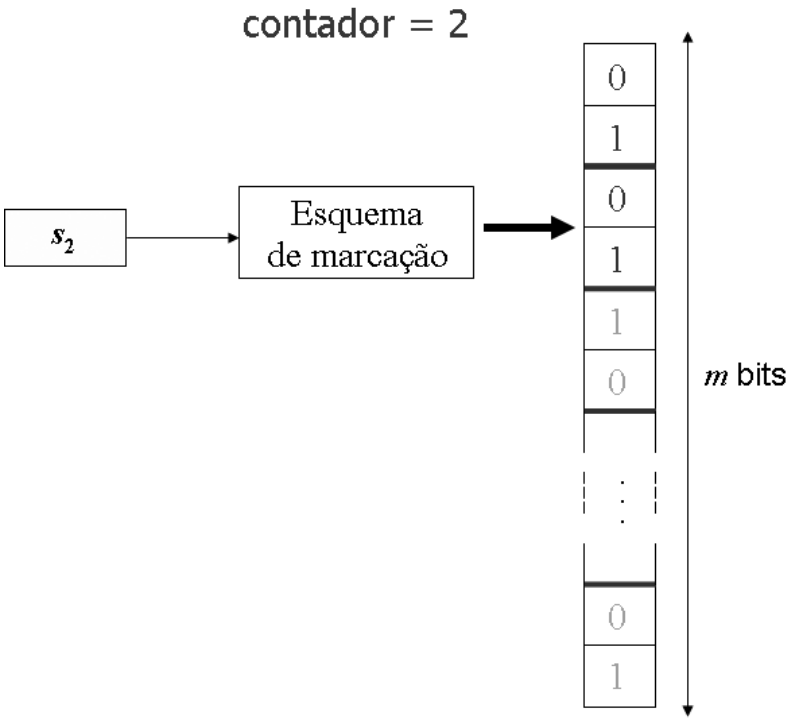

Figura IV.7: A inserção de elemento no FBC onde o conteúdo original do subfiltro é substituído pela marcação do elemento a ser inserido.

tese depende da aplicação em questão. Evidentemente, a probabilidade de ser positivo o resultado do segundo algoritmo é maior. Como o teste de pertinência é positivo para todo elemento $s_i \in S$ nos dois algoritmos, então a probabilidade de falso positivo do segundo algoritmo é maior. Logo, se na aplicação em questão a hipótese do primeiro algoritmo for viável, deve-se optar por esse algoritmo, em detrimento do segundo. No caso na aplicação do rastreamento IP, pode-se utilizar o campo TTL (*time-to-live*) do cabeçalho IP como o contador que indicará o índice do subfiltro em que o endereço de cada roteador tem chance de ter sido inserido. Por essa razão, nesse trabalho é analisado apenas o primeiro algoritmo. Uma análise do segundo algoritmo será deixada para um trabalho futuro. O primeiro algoritmo é apresentado a seguir.

Algoritmo 4: Teste de Pertinência no Filtro de Bloom Concatenado

Entrada: Elemento x e o vetor C de m bits.

Saída: 1, se $x \in S$; 0, caso contrário.

início

  declare $mascaraMarcacao[m/N]$ vetor de bits

  $esquemaMarcacao(x, mascaraMarcacao)$

  $i \leftarrow mecanismoAuxiliar(x)$

  para $j = 0$ **até** $(m/N) - 1$ **faça**

    se $C[i * (m/N) + j] \neq mascaraMarcacao[j]$ **então retorna** *0*

  retorna *1*

fim

O Algoritmo 4 recebe como argumentos de entrada um elemento x e o Filtro de Bloom Concatenado C. É chamado o procedimento $esquemaMarcacao()$, que atualiza o vetor $mascaraMarcacao$ com a "assinatura" do elemento x. Se o vetor $mascaraMarcacao$ for exatamente igual ao i-ésimo subfiltro, então o algoritmo retorna 1, indicando teste de pertinência positivo. Caso contrário, o algoritmo retorna 0, indicando teste de pertinência negativo. A Figura IV.8 ilustra esse procedimento.

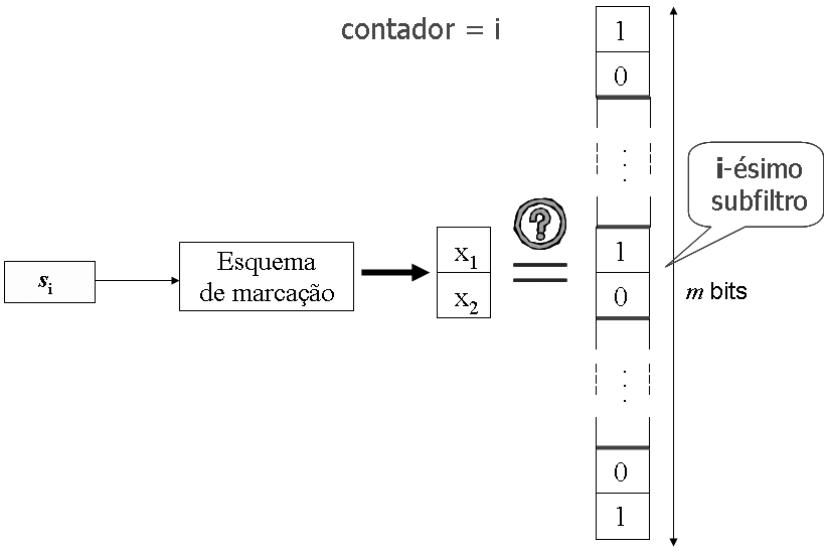

Figura IV.8: Teste de pertinência no FBC. A marcação do elemento a ser testado é comparada ao conteúdo do subfiltro indicado pelo contador.

Capítulo V

Resultados Analíticos e de Simulação

NESSE capítulo são apresentados os resultados obtidos no projeto. São derivadas expressões analíticas para a probabilidade de falso positivo do filtro proposto, do número ótimo de funções *hash* que minimiza a probabilidade de falso positivo e da probabilidade de sucesso do atacante para o ataque do ajuste da condição inicial, descrito na Seção IV.3. Isso permite comparar a robustez do filtro proposto com o Filtro de Bloom original. Por motivos de clareza de redação, optou-se colocar a análise da robustez do Filtro de Bloom Generalizado na Seção IV.2.1, a fim de ilustrar melhor o compromisso entre robustez e perda de informação para um FBG. Por fim, o sistema de rastreamento proposto é avaliado através de resultados de simulação.

V.1 Resultados Analíticos

V.1.1 Falsos Positivos

Nessa seção é feita uma análise da probabilidade de falso positivo da proposta deste trabalho, o Filtro de Bloom Concatenado (FBC). Um falso positivo ocorre quando um elemento externo ao filtro possui uma "assinatura" idêntica à do elemento que foi inserido no subfiltro em questão. Vale ressaltar que apenas um elemento é inserido em cada

subfiltro e assume-se que é conhecido o único subfiltro no qual um dado elemento tem chance de ter sido inserido. Como cada esquema de marcação possui uma forma de "assinatura" diferente, cada um deles tem uma probabilidade de falso positivo diferente.

Seja p a probabilidade de um bit específico ser zerado. Vale notar que p é igual à probabilidade de um bit estar zerado após a inserção do elemento no subfiltro. Além disso, na média, $p(m/N)$ bits são zerados e $(1-p)(m/N)$ bits são preenchidos com 1. Assim, em todos os esquemas de marcação, a probabilidade f_p de falso positivo é

$$f_p = p^{p(m/N)}(1-p)^{(1-p)(m/N)}. \tag{V.1}$$

A diferença é que, para cada esquema de marcação, p pode assumir valores distintos.

No esquema de marcação 1, p é a probabilidade de nenhuma das k funções *hash* preencher um bit com 1. Como as funções *hash* são uniformes e independentes, então

$$p = \left(1 - \frac{1}{(m/N)}\right)^k. \tag{V.2}$$

No esquema de marcação 2, como a saída da função *hash* é uniformemente distribuída ao longo do seu conjunto-imagem, cada bit tem a mesma probabilidade de ser preenchido com 1 ou 0. Logo, p é simplesmente 1/2.

A Figura V.1 mostra o gráfico da probabilidade de falso positivo, dada pela Equação V.1, em função da fração de bits em zero, p. Em cada uma das curvas apresentadas tem-se um tamanho de subfiltro m/N diferente. Para a compreensão desse gráfico, é importante ter em mente que p representa tanto a fração de bits em zero como a probabilidade de um bit ser zerado numa inserção. Observa-se que, quando p tende a 1, f_p tende a 1. Isso ocorre porque, nessa situação, todos os bits do filtro estão em 0 e a probabilidade de um bit ser zerado é de 100%. Logo, a probabilidade de falso positivo tenderá a 1. Uma situação análoga ocorre quando p tende a 0. Outra observação é que f_p diminui com o aumento da fração de bits por subfiltro m/N e, à medida que m/N aumenta, se alarga o intervalo no qual f_p é próxima de zero.

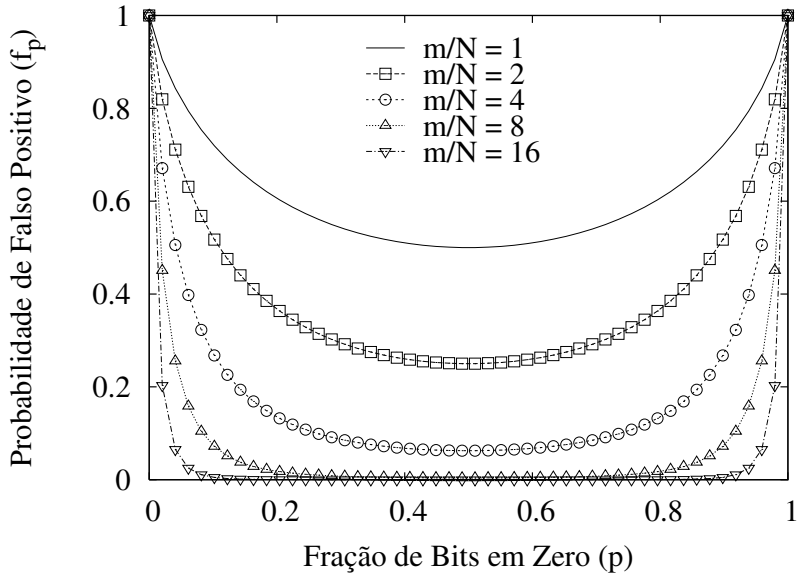

Figura V.1: Probabilidade de falso positivo em função da fração de bits em zero.

É importante notar que o gráfico possui eixo de simetria em $p = 1/2$. Isso facilita a demonstração que esse é o ponto de mínimo de f_p. Observa-se que a equação V.1 seria expressa da mesma forma, caso fosse dada em função do termo $1 - p$. Assim, por razões de simetria, $p^{(min)} = 1/2$. Substituindo esse valor na Equação V.1, determina-se a probabilidade mínima de falso positivo $f_p^{(min)}$, dada por

$$f_p^{(min)} = \left(\frac{1}{2}\right)^{m/N}. \tag{V.3}$$

Pode-se observar que o decaimento de f_p com o aumento número de bits por subfiltro m/N é exponencial, de forma que f_p atinge valores bem próximos a zero mesmo quando a razão m/N ainda assume valores razoavelmente pequenos. Nota-se que, no esquema de marcação 2, p é sempre igual a 1/2. Logo, nesse caso, a probabilidade de falso positivo é sempre mínima, sendo expressa pela Equação V.3. Já no esquema de marcação 1, p depende também do número k de funções *hash*. Porém, o gráfico da Figura V.2 mostra que existe um valor ótimo de k que minimiza f_p. Apenas graficamente, pode-se observar que o valor ótimo ocorre quando k é ligeiramente menor do que m/N. Analiticamente, o valor ótimo de k é dado por

$$k^{(min)} = -\frac{\ln(2)}{\ln\left(1 - \frac{1}{(m/N)}\right)}. \tag{V.4}$$

Para se chegar a essa expressão, basta igualar a 1/2 a expressão de p na Equação V.2. Esse resultado mostra que os dois esquemas de marcação propostos possuem a mesma probabilidade de falso positivo mínima.

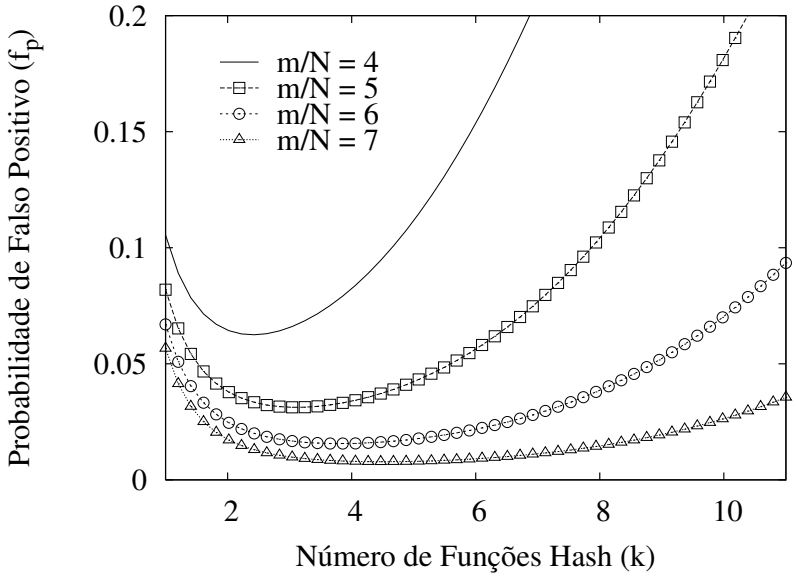

Figura V.2: Probabilidade de falso positivo para o esquema de marcação 1.

V.1.2 Robustez

Nessa seção, é analisada a robustez à interferência do atacante do Filtro de Bloom original e do filtro proposto nesse trabalho, o FBC. É usada como métrica de robustez a probabilidade p_a de sucesso do atacante ao executar o ataque descrito na Seção IV.3. Quanto menor a probabilidade p_a, maior é a robustez do filtro.

Para avaliar a probabilidade p_a de sucesso do atacante considera-se a probabilidade do último elemento inserido pelo atacante ser reconhecido pelo filtro. Assim, sem perda de generalidade, pode-se modelar o ataque como sendo a inserção de um único elemento forjado e_f no filtro. Conforme a descrição do ataque na Seção IV.3, após a inserção do

elemento forjado e_f, o pacote é enviado pela rede e, então, os endereços dos roteadores da rota de ataque são inseridos no filtro. Os endereços dos roteadores da rota de ataque são os elementos legítimos $s_1, s_2, \ldots, s_n$. A Figura V.3 deixa clara a ordem com que os elementos são inseridos no filtro.

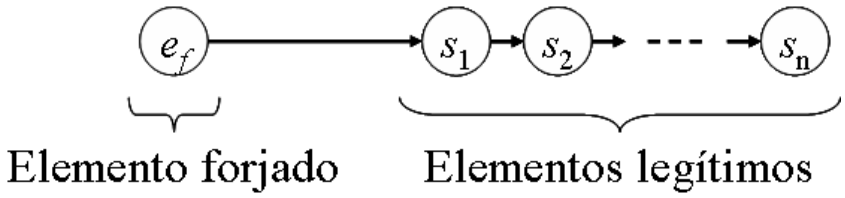

Figura V.3: Ordem de inserção dos elementos no filtro.

É considerado que o atacante teve sucesso se o elemento e_f por ele inserido tiver teste de pertinência positivo após a inserção de n elementos legítimos. Se pelo menos um bit inserido pelo atacante for invertido pelas n inserções posteriores, então o ataque não terá sucesso. Assim, p_a é a probabilidade de nenhum bit inserido pelo atacante ser invertido.

No FBC, para os dois esquemas de marcação, p_a é a probabilidade da "assinatura" do elemento inserido pelo atacante ser igual à "assinatura" do elemento legítimo inserido posteriormente no mesmo subfiltro. Isso é equivalente à probabilidade de falso positivo. Portanto,

$$p_a = f_p^{(min)} = \left(\frac{1}{2}\right)^{m/N}. \tag{V.5}$$

Uma observação importante é que, no esquema de marcação 1, essa probabilidade seria maior, caso não fosse feita a adaptação apresentada na Seção IV.4, que visa inverter ao máximo os bits iniciais do filtro. Sem essa adaptação, os bits inseridos pelo atacante que permanecessem intocados pela inserção do elemento legítimo não seriam invertidos e, portanto, a probabilidade de sucesso do atacante seria maior. Agora, basta que apenas um desses bits seja invertido pela adaptação que o ataque irá falhar.

Já o Filtro de Bloom original não possui nenhum mecanismo que garanta robustez à interferência do atacante. Assim, como nenhum bit inserido pelo atacante é invertido,

então a probabilidade de sucesso do atacante $p_a = 1$. Isso significa que um nó malicioso pode comprometer completamente uma aplicação distribuída na qual o filtro original é utilizado. No caso do rastreamento de pacotes, o atacante consegue burlar o sistema de rastreamento e garantir o seu anonimato.

V.2 Resultados de Simulação

A fim de avaliar o desempenho do sistema de rastreamento proposto foi feita uma implementação do FBC e dos procedimentos de marcação de pacotes e reconstrução de rota, descritos na Seção III.2. Foi então desenvolvido um simulador de ataques de negação de serviço distribuídos usando a linguagem de programação C++. Da mesma forma que em [27], para criar a topologia usada na simulação, foi usado o gerador de topologia *nem* [51], baseado em amostragem do mapa da Internet. O gerador *nem* extrai aleatoriamente um subgrafo de um mapa de rede, mantendo suas propriedades originais, como grau de vizinhos, distância média e diâmetro da topologia. A topologia empregada consiste de 10.000 roteadores com uma média de 3,15 vizinhos por roteador e é amostrada de uma topologia real da Internet. O atacante é aleatoriamente escolhido dentre o conjunto dos nós do grafo e, então, é simulado um ataque de negação de serviço definindo-se um caminho de ataque a partir do nó escolhido. A seguir, é simulado o envio do pacote de ataque realizando-se a marcação do pacote de acordo com o procedimento de marcação definido pelo sistema proposto. Após a marcação do pacote, é realizada a reconstrução da rota de ataque a partir da vítima, que é o último nó do caminho de ataque escolhido. Os resultados são obtidos levando-se em consideração um intervalo de confiança de 95% em relação à média das amostras, representado nos gráficos por barras de erro verticais.

Uma observação importante a ser feita é que, como o Filtro de Bloom Concatenado não admite falsos negativos, então o sistema de rastreamento proposto sempre consegue rastrear o atacante. Mas, a existência de uma probabilidade de falso positivo não nula implica a possibilidade do procedimento de reconstrução incorporar ao grafo de reconstrução alguns roteadores pelos quais o pacote de ataque não passou. Assim, para medir

a acurácia do sistema de rastreamento proposto foram feitas rodadas de simulação nas quais era medido o número de atacantes rastreados. A cada rodada um novo par de nós (atacante, vítima) é escolhido e a topologia também varia. Idealmente, espera-se que o número de atacantes rastreados seja 1, ou seja, somente o verdadeiro atacante é rastreado. Então, um número de atacantes rastreados muito maior do que 1 indica uma baixa acurácia. A Figura V.4 mostra o resultado do número de atacantes rastreados em função do número de bits por elemento (m/n) para diferentes tamanhos de rota de ataque.

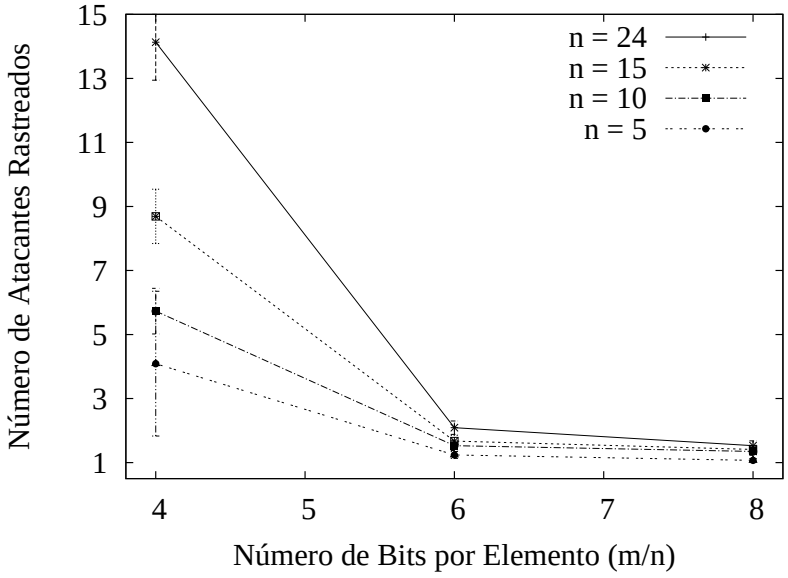

Figura V.4: Número de atacantes rastreados em função do número de bits por elemento.

Pela figura, pode-se observar o compromisso entre sobrecarga de cabeçalho (*header overhead*), medida em número de bits por elemento, e a acurácia do sistema, medida em número de atacantes rastreados. O sistema proposto sempre encontra o verdadeiro atacante, mesmo quando a rota de ataque é grande. Porém, a acurácia diminui com a diminuição do número de bits por elemento. Isso está de acordo com a Equação V.3, pois uma redução da razão m/N implica um aumento da probabilidade de falso positivo. De acordo com estatísticas da Internet [52], a distribuição dos tamanhos das rotas possui média 15,3 e desvio padrão 4,2. Assim, supondo uma distribuição Gaussiana, a probabilidade do tamanho de uma rota ser menor ou igual a 24 é de aproximadamente 98%. Consequentemente, observando a curva onde $n = 24$, pode-se concluir que o sistema

proposto consegue rastrear 98% das rotas da Internet com uma acurácia de 2,1 usando apenas 6 bits por elemento. Isso representa uma economia de espaço de cabeçalho de 81,3%, considerando endereços IPv4 de 32 bits; e uma economia de 95,3%, considerando endereços IPv6 de 128 bits. Além disso, a acurácia pode ser ainda melhor, ao custo de menor economia de espaço no cabeçalho. Nota-se também que, para um dado número de bits por elemento, as rotas menores são rastreadas com melhor acurácia.

Capítulo VI

Conclusão

Esse trabalho apresenta um novo sistema de rastreamento de pacotes baseado na técnica de marcação de pacotes. O sistema utiliza como estrutura de marcação um Filtro de Bloom robusto à interferência da condição inicial denominado Filtro de Bloom Concatenado (FBC). O FBC possui a propriedade de ser robusto a ataques, além da característica do Filtro de Bloom original de representar um conjunto de forma compacta. A ideia chave da proposta é concatenar diversos subfiltros, cada qual representando um único elemento. Esta estrutura permite uma alta robustez à interferência da condição inicial, obtida com a ação de sobrescrever o conteúdo original de cada subfiltro. Assim, é garantido que a informação inicial originada pelo atacante não influencia a probabilidade de falso positivo. Além disso, essa técnica não introduz falsos negativos, que poderiam causar perda de informações legítimas e limitação da capacidade de armazenamento. Como somente um elemento é inserido por subfiltro, podem-se sobrescrever as informações iniciais sem perda de informação legítima.

São obtidas expressões analíticas para os falsos positivos e os resultados mostram que a probabilidade de sucesso do atacante decresce exponencialmente com o tamanho de cada subfiltro. É mostrado que o Filtro de Bloom convencional não é robusto a ataques, pois ao saturar o filtro a probabilidade de sucesso do atacante é de 100%. Quando comparado ao Filtro de Bloom Generalizado (FBG), o filtro proposto não apresenta falsos negativos

e mostra-se bem mais robusto para o mesmo tamanho do filtro. Os falsos negativos são eliminados pela construção específica da representação de um único elemento por subfiltro. A maior robustez é obtida pelo apagamendo da condição inicial dos subfiltros no momento da inserção de novos elementos. No Filtro de Bloom Generalizado, existe um compromisso entre a robustez e a probabilidade de "esquecimento" dos elementos mais antigos. Quanto maior a robustez, maiores são as taxas de "esquecimento" do FBG, interpretadas como uma forma de limitação de memória do filtro.

Os resultados indicam que o Filtro de Bloom Concatenado proposto se apresenta como uma excelente opção para representação de elementos em aplicações distribuídas eficientes e seguras devido a sua compacidade e robustez. Com apenas 4 bits por subfiltro, a probabilidade de sucesso do atacante é de 6,25% e, com 8 bits por subfiltro, essa probabilidade cai para aproximadamente 0,39%.

O sistema de rastreamento proposto foi implementado e avaliado através de um simulador desenvolvido com a linguagem de programação C++. Os resultados de simulação confirmam as expressões analíticas obtidas. As simulações mostram que o sistema proposto sempre encontra o verdadeiro atacante, mesmo quando a rota de ataque é grande. Além disso, a acurácia do rastreamento é excelente. Usando apenas 6 bits por elemento, o sistema proposto consegue rastrear 98% das rotas da Internet com uma acurácia de 2,1 atacantes encontrados por rota rastreada. Isso representa uma economia de espaço de cabeçalho de 81,3%, considerando endereços IPv4 de 32 bits; e uma economia de 95,3%, considerando endereços IPv6 de 128 bits.

Referências Bibliográficas

[1] W. Stallings, *Cryptography and Network Security - Principles and Practice, 4th Edition.* Prentice Hall, 2006.

[2] D. X. Song e A. Perrig, "Advanced and Authenticated Marking Schemes for IP Traceback", abril de 2001.

[3] S. Gaudin, *Storm Worm Botnet More Powerful Than Top Supercomputers.* InformationWeek, setembro de 2007.

[4] J. Mirkovic, S. Dietrich, D. Dittrich e P. Reiher, *Internet Denial of Service: Attack and Defense Mechanisms.* Prentice Hall PTR, 2004.

[5] R. Richardson, *2008 CSI/FBI Computer Crime and Security Survey.* http://www.gocsi.com/forms/csi_survey.jhtml.

[6] R. Lemos, *Attack knocks out Microsoft Web sites.* CNET News, janeiro de 2001.

[7] D. Moore, G. Voelker e S. Savage, "Inferring Internet Denial of Service Activity", in *Proceedings of the USENIX Security Symposium*, Washington, DC, EUA, pp. 9–22, agosto de 2001.

[8] A. Press, *Powerful Attack Upset Global Internet Traffic.* New York Times, outubro de 2002.

[9] L. A. Gordon, M. P. Loeb, W. Lucyshyn e R. Richardson, *CSI/FBI Computer Crime and Security Survey*, 2004.

[10] J. Franklin, V. Paxson, A. Perrig e S. Savage, "An inquiry into the nature and causes of the wealth of internet miscreants", in *CCS'07: Proceedings of the 14th ACM conference on Computer and communications security*, New York, NY, USA, pp. 375–388, ACM, 2007.

[11] Reuters, *'Botmaster' pleads guilty to computer crimes*. ZDNet News, janeiro de 2006.

[12] I. Traynor, *Russia accused of unleashing cyberwar to disable Estonia*. The Guardian, maio de 2007.

[13] J. Markoff, *Before the Gunfire, Cyberattacks*. New York Times, agosto de 2008.

[14] V. G. Cerf e R. E. Icahn, "A Protocol for Packet Network Intercommunication", *SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 2, no. 2, pp. 71–82, 2005.

[15] P. Ferguson e D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", 2000.

[16] S. M. Bellovin, M. D. Leech e T. Taylor, "ICMP Traceback Messages", *Internet Draft: draft-ietf-itrace-04.txt*, agosto de 2003.

[17] S. Savage, D. Wetherall, A. Karlin e T. Anderson, "Network Support for IP Traceback", *IEEE/ACM Transactions on Networking*, vol. 9, no. 3, pp. 226–237, junho de 2001.

[18] Y. Zhang e V. Paxson, "Detecting Stepping Stones", in *Proceedings of the 9th USENIX Security Symposium*, pp. 171–184, 2000.

[19] M. D. D. Moreira, R. P. Laufer, P. B. Velloso e O. C. M. B. Duarte, "Uma Proposta de Marcação de Pacotes para Ratreamento Robusto a Ataques", in *Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais - SBSeg'2007*, Rio de Janeiro, RJ, Brasil, agosto de 2007.

[20] B. H. Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors", *Communications of the ACM*, vol. 7, no. 13, pp. 442–426, julho de 1970.

[21] R. P. Laufer, I. M. Moraes, P. B. Velloso, M. D. D. Bicudo, M. E. M. Campista, D. de O. Cunha, L. H. M. K. Costa e O. C. M. B. Duarte, "Negação de Serviço: Ataques e Contramedidas", in *Minicursos do V Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais - SBSeg'2005*, ch. 1, setembro de 2005.

[22] L. Garber, "Denial-of-Service Attacks Rip the Internet", *IEEE Computer*, vol. 4, no. 33, pp. 12–17, abril de 2000.

[23] CNN.com, *Denial of service hackers take on new targets*, fevereiro de 2000. http://www.cnn.com/2000/TECH/computing/02/09/denial.of.service.03.

[24] D. Dittrich, *The DoS Project's 'trinoo' distributed denial of service attack tool*, outubro de 1999. http://staff.washington.edu/dittrich/misc/trinoo.analysis.txt.

[25] D. Dittrich, *The 'Tribe Flood Network' distributed denial of service attack tool*, outubro de 1999. http://staff.washington.edu/dittrich/misc/tfn.analysis.txt.

[26] D. Dittrich, *The 'stacheldraht' distributed denial of service attack tool*, dezembro de 1999. http://staff.washington.edu/dittrich/misc/stacheldraht.analysis.txt.

[27] R. P. Laufer, "Rastreamento de Pacotes IP contra Ataques de Negação de Serviço", Tese de Mestrado, PEE/COPPE, Universidade Federal do Rio de Janeiro, Rio de Janeiro, RJ, Brasil, setembro de 2005.

[28] G. Keizer, *Dutch Botnet Bigger Than Expected*. InformationWeek, outubro de 2005.

[29] R. Lemos, *Worm worries grow with release of Windows hacks*. ZDnet News, abril de 2004.

[30] B. Krebs, *'Sasser' Worm Tip of the PC Bug Invasion*. Washington Post, maio de 2004.

[31] K. Spiess, *Worm 'Storm' gathers strength*. Neoseeker, setembro de 2007.

[32] CERT, *CERT Advisory CA-1999-17 Denial-of-Service Tools*, dezembro de 1999. http://www.cert.org/advisories/CA-1999-17.html.

[33] R. P. Laufer, P. B. Velloso e O. C. M. B. Duarte, "Defeating DoS Attacks with IP Traceback", in *IFIP Open Conference on Metropolitan Area Networks - MAN'2005*, Ho Chi Minh, Vietnã, pp. 131–148, abril de 2005.

[34] H. Burch e B. Cheswick, "Tracing Anonymous Packets to their Approximate Source", in *USENIX LISA'00*, Nova Orleans, LA, EUA, pp. 319–327, dezembro de 2000.

[35] K. Park e H. Lee, "On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attack", in *Proceedings of the IEEE INFOCOM 2001 Conference*, Anchorage, AK, EUA, abril de 2001.

[36] A. Mankin, D. Massey, C.-L. Wu, S. F. Wu e L. Zhang, "On Design and Evaluation of "Intention-Driven" ICMP Traceback", in *Proceedings of the IEEE ICCCN 2001 Conference*, Scottsdale, AZ, EUA, outubro de 2001.

[37] R. Stone, "CenterTrack: An IP Overlay Network for Tracking DoS Floods", in *9th USENIX Security Symposium*, Denver, CO, EUA, pp. 199–212, agosto de 2000.

[38] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, B. Schwartz, S. T. Kent e W. T. Strayer, "Single-Packet IP Traceback", *IEEE/ACM Transactions on Networking*, vol. 10, no. 6, pp. 721–734, dezembro de 2002.

[39] J. Li, M. Sung, J. Xu e L. Li, "Large-Scale IP Traceback in High-Speed Internet: Practical Techniques and Theoretical Foundation", in *Proceedings of the 25th IEEE Symposium on Security and Privacy*, Oakland, CA, EUA, maio de 2004.

[40] A. Broder e M. Mitzenmacher, "Network Applications of Bloom Filters: A Survey", *Internet Mathematics*, vol. 1, no. 4, no. 4, pp. 485–509, 2003.

[41] R. P. Laufer, P. B. Velloso, D. de O. Cunha, I. M. Moraes, M. D. D. Bicudo e O. C. M. B. Duarte, "A New IP Traceback System against Denial-of-Service Attacks", in *12th International Conference on Telecommunications - ICT'2005*, Cidade do Cabo, África do Sul, maio de 2005.

[42] M. Mitzenmacher, "Compressed Bloom Filters", *IEEE/ACM Transactions on Networking*, vol. 10, no. 5, pp. 604–612, outubro de 2002.

[43] L. Fan, P. Cao, J. Almeida e A. Z. Broder, "Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol", *IEEE/ACM Transactions on Networking*, vol. 8, no. 3, pp. 281–293, junho de 2000.

[44] S. Dharmapurikar, P. Krishnamurthy, T. S. Sproull e J. W. Lockwood, "Deep Packet Inspection Using Bloom Filters", *IEEE Micro*, vol. 24, no. 1, pp. 52–61, janeiro de 2004.

[45] K. Shanmugasundaram, H. Brönnimann e N. Memon, "Payload Attribution via Hierarchical Bloom Filters", in *Proceedings of the 11th ACM conference on Computer and Communications Security*, Washington, DC, EUA, pp. 31–41, outubro de 2004.

[46] S. Cohen e Y. Matias, "Spectral Bloom Filters", in *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, San Diego, CA, EUA, pp. 241–252, junho de 2003.

[47] C. Estan e G. Varghese, "New Directions in Traffic Measurement and Accounting: Focusing on the Elephants, Ignoring the Mice", *ACM Transactions on Computer Systems*, vol. 21, no. 3, no. 3, pp. 270–313, 2003.

[48] A. Kumar, J. Xu, J. Wang, O. Spatschek e L. Li, "Space-Code Bloom Filter for Efficient Per-Flow Traffic Measurement", in *Proceedings of the IEEE INFOCOM 2004 Conference*, Hong Kong, China, pp. 1762–1773, março de 2004.

[49] R. P. Laufer, P. B. Velloso, D. de O. Cunha, I. M. Moraes, M. D. D. Bicudo, M. D. D. Moreira e O. C. M. B. Duarte, "Towards Stateless Single-Packet IP Traceback", in *LCN'07: Proceedings of the 32nd IEEE Conference on Local Computer Networks*, Washington, DC, USA, pp. 548–555, IEEE Computer Society, 2007.

[50] A. Yaar, A. Perrig e D. Song, "Pi: A Path Identification Mechanism to Defend against DDoS Attacks", in *IEEE Symposium on Security and Privacy*, maio de 2003.

[51] D. Magoni e J.-J. Pansiot, "Internet Topology Modeler Based on Map Sampling", in *Proceedings of the 7th IEEE International Symposium on Computer Communications*, Washington, DC, EUA, p. 1021, julho de 2002.

[52] P. Mahadevan, D. Krioukov, M. Fomenkov, X. Dimitropoulos, K. C. Claffy e A. Vahdat, "The Internet AS-level Topology: Three Data Sources and One Definitive Metric", *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, no. 1, pp. 17–26, 2006.