Rastreamento de Pacotes em Redes IP

Maycon Ferraz Braga

JUIZ DE FORA - MG
JULHO, 2009

Maycon Ferraz Braga

Rastreamento de Pacotes em Redes IP

Monografia submetida ao corpo docente do Departamento de Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora, como parte integrante dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Eduardo Pagani Julio

JUIZ DE FORA - MG

JULHO, 2009

Rastreamento de Pacotes em Redes IP

Maycon Ferraz Braga

Monografia submetida ao corpo docente do Departamento de Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora, como parte integrante dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Aprovado em _____ de _____ de 2009.

Comissão Examinadora:

_____

Eduardo Pagani Julio orientador.
Mestre em Computação/UFF.

_____

Eduardo Barrére.
Doutor em Engenharia de Sistemas e Computação.

_____

Marcelo Lobosco.
Doutor em Engenharia de Sistemas e Computação COPPE/UFRJ.

JUIZ DE FORA

JULHO, 2009

Resumo

Em grande parte dos ataques digitais, principalmente ataques de negação de serviço, pacotes IP com endereços de origem forjados são usados para ocultar a verdadeira origem do atacante, esta técnica é conhecida como IP *Spoofing*. Uma possível estratégia de defesa é rastrear a origem do ataque, de forma a penalizar o atacante ou isolá-lo da rede. Diversos sistemas para rastreamento de pacotes IP foram propostos, a maioria baseado em marcação de pacotes ou transmissão de mensagens de rastreamento usando esquemas probabilísticos para minimizar o impacto no tráfego total da rede. Por ser uma conseqüência direta do modelo de desenvolvimento dos protocolos TCP/IP, não há ainda nenhuma solução nativa contra o uso da técnica de IP *Spoofing*. Este trabalho apresenta as principais técnicas presentes na literatura que se propõem a mitigar este problema. Através de simulações com a implementação dos Filtros de Bloom, Filtro de Bloom Generalizado e Filtro de Bloom Concatenado, o trabalho avalia o uso de cada filtro como alternativa para redução do *overhead* no datagrama IP na técnica de marcação de pacotes e determina valores ideais para que o filtro seja eficiente.

Palavras-chave: IP *Spoofing*. Rastreamento de pacotes. Ataques DoS.

Abstract

In most digital attacks, especially denial of service attacks, IP packets with forged source addresses, are used to hide the true origin of the attacker, this technique is known as IP Spoofing. A possible strategy of defense is to trace the origin of the attack in order to punish the attacker or isolate it from the network. Several systems for tracking the IP packets have been proposed, most based on marking of packages or transmission of tracking messages using probabilistic schemes to minimize the impact in the total traffic on the network. As a direct consequence of the TCP/IP development model, there is no native solution against the use of the IP Spoofing technique. This paper presents the main techniques in the literature that proposes to mitigate this problem. Through simulations evaluates the use of Bloom filter as an alternative to reduce the overhead in the IP datagram in the technique of marking the packet and determines optimal values for the filter is efficient.

Key-words: IP Spoofing. Trace packets. DoS Attacks.

Agradecimentos

A todos que colaboraram na realização deste trabalho, a minha família pelo apoio e compreensão incondicionais, e a meu orientador que dedicou seu tempo e empenho na execução deste trabalho.

SUMÁRIO

Lista de Figuras

Lista de Reduções

ACK	*Acknowledgment*
BGP	*Border Gateway Protocol*
DDoS	*Distributed Denial of Service*
DoS	*Denial of Service*
FB	*Filtro de Bloom*
FBC	*Filtro de Bloom Concatenado*
FBG	*Filtro de Bloom Generalizado*
ICMP	*Internet Control Message Protocol*
IPSEC	*IP Security Protocol*
IPv4	*Internet Protocol version 4*
IPv6	*Internet Protocol version 6*
L2TP	*Layer 2 Tunneling Protocol*
RFC	*Request for Comments*
SA	Sistema Autônomo
SYN	*Synchronized*
TCP	*Transmission Control Protocol*
VPN	*Virtual Private Network*

1. Introdução

A Internet é cada vez mais difundida e essencial no dia-a-dia das pessoas. Junto com o uso maciço da rede, surgem também problemas relacionados à segurança dos dados trafegados e confiabilidade dos serviços prestados. No entanto, os protocolos que controlam a rede e a maioria dos serviços da Internet não possuem segurança adequada [11]. Essa fragilidade facilita a ação de pessoas mal intencionadas tornando a rede vulnerável [9].

Pesquisa realizada pela *Arbor Networks* com especialistas em segurança de provedores de Internet demonstrou uma crescente preocupação com ataques de negação de serviço, que em 2007 atingiram volume recorde[4]. Em ataques de negação de serviço (*Denial of Service* - DoS),e ataques de negação de serviço distribuídos (*Distributed* DoS - DDoS), servidores são sobrecarregados com requisições de dados e, assim, têm seu funcionamento interrompido até que o ataque pare e as conexões pendentes expirem esvaziando os *buffers*, permitindo novas conexões [10].

A preocupação com ciberataques também foi abordada em um relatório da *Secure Computing*, conduzido nos Estados Unidos, Europa e Canadá durante agosto e setembro de 2008 com 199 especialistas de segurança ligados à indústria de infra-estrutura. Mesmo com o aumento da regulamentação e novas leis, mais da metade dos entrevistados declarou preocupação com a falta de segurança do setor [4].

Esse trabalho apresenta técnicas diversas para o rastreamento de pacotes em redes IP com uma análise crítica sobre as vantagens e desvantagens de cada método. Técnicas de Rastreamento de Pacotes IP não tem o intuito de parar um ataque e são usadas para identificar a origem dos pacotes ofensivos, durante ou depois dos ataques.

Este trabalho está organizado em 5 capítulos. No Capítulo 2 são apresentadas definições de redes de computadores e problemas relacionados às redes, apresentando ao leitor conceitos necessários para entendimento do objetivo e técnicas apresentadas no trabalho.

No Capítulo 3 são apresentadas as técnicas propostas na literatura, criando assim a base para se avançar na discussão sobre o Rastreamento de Pacotes em Redes IP.

No Capítulo 4 é tecida uma comparação entre algumas das técnicas apresentadas no Capítulo 3. Também apresenta uma ferramenta de simulação utilizada para obter uma comparação entre as técnicas simuladas.

No capítulo 5 são relatadas as conclusões obtidas no trabalho e sugestões para trabalhos futuros.

2 Referencial Teórico

Para um melhor entendimento do estudo apresentado neste trabalho faz-se necessária uma revisão de alguns conceitos e a apresentação de problemas relacionados às redes de computadores e às técnicas expostas nos capítulos seguintes a este.

2.1 Protocolos

2.1.1 Protocolo IP

IP (*Internet Protocol*) é um protocolo usado entre duas ou mais máquinas em rede para encaminhamento de dados. Dados em redes IP são enviados como pacotes ou datagramas. Particularmente, no IP nenhuma validação é necessária antes de uma máquina tentar enviar pacotes à outra máquina nunca contactada anteriormente [21,22].

O protocolo IP oferece um serviço não confiável de datagramas, ou seja, perdas ou duplicações de pacotes podem acontecer. Pacotes IP podem ser roteados (encaminhados através de redes interconectadas). A RFC 791 de setembro de 1981 descreve o protocolo IP versão 4 com 32 bits para endereçamento. O IP versão 6 descrito posteriormente em dezembro de 1995, porém ainda não implementado, tem 128 bits para endereçamento, evitando um possível esgotamento dos endereços [21,22].

2.1.2 Datagrama IP

O datagrama ou pacote IP é a unidade básica de dados no nível IP, e é composto por duas áreas, uma área de cabeçalho e outra de dados [21,22].

O cabeçalho contém informações que identificam o conteúdo do datagrama. Na área de dados está encapsulado o pacote do nível superior, ou seja, um pacote TCP ou UDP. A Figura 2.1 exibe o formato do datagrama IPv4 e a Figura 2.2 o formato do datagrama IPv6 [21,22].

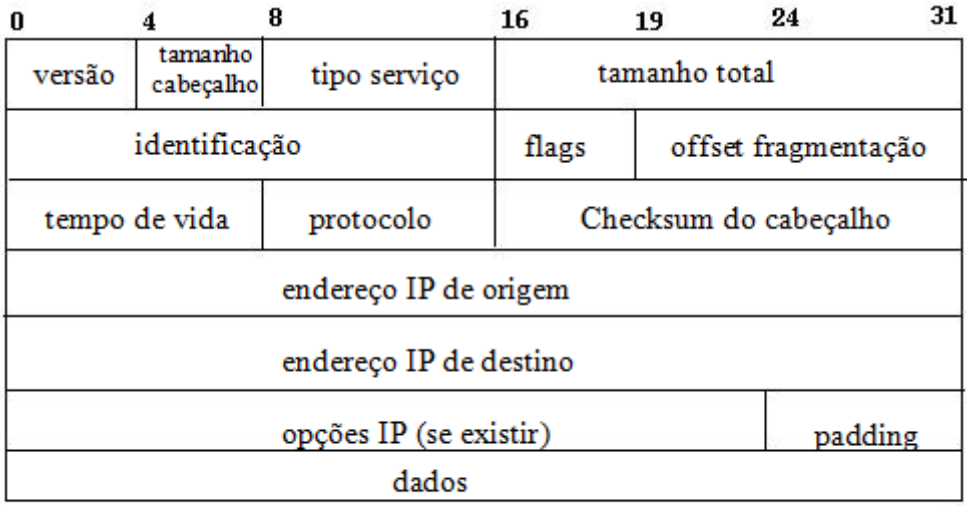

Figura 2.1 – Datagrama IPv4[14]

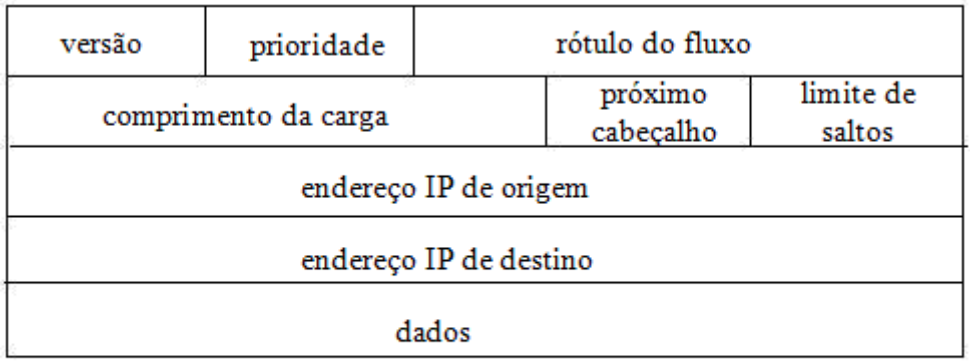

Figura 2.2 – Datagrama IPv6[15]

2.1.3 TCP - *Transmission Control Protocol*

O protocolo tem como objetivo principal comunicar aplicações de dois *hosts* diferentes. O TCP é um protocolo de nível de transporte que trabalha com mensagens de reconhecimento, especificação do formato da informação e mecanismos de segurança. Ele garante a entrega de todos os PDU's (*Protocol data Unit*), pois realiza transmissões orientadas à conexão [2].

O protocolo TCP utiliza o protocolo IP, que é um protocolo não orientado à conexão. O TCP então fica responsável pelo controle dos procedimentos da transferência segura de dados. Cabe salientar que o IP não é o único protocolo não orientado à conexão que pode ser utilizado pelo TCP [1,5].

2.2 Ataques a Redes

2.2.1 IP *Spoofing*

IP *spoofing* é a técnica utilizada para falsificar o endereço real do remetente de um pacote na rede. O Protocolo IP tem características que permitem o reencaminhamento de pacotes sem validação do endereço IP nem relação destes pacotes com os roteadores por onde ele passou até chegar ao seu destino. Estas características permitem que um indivíduo mal intencionado na rede falsifique o IP com uma simples manipulação do cabeçalho do datagrama IP [9,10].

2.2.2 DoS e DDoS

Os ataques DoS (*Denial of Service*), também denominados Ataques de Negação de Serviço, consistem em tentativas de impedir usuários legítimos de utilizarem um determinado serviço de um computador. Para isso, são usadas técnicas que podem sobrecarregar uma rede a tal ponto em que os verdadeiros usuários desta não consigam usá-la [8,9,10,11].

Ao estabelecer uma conexão com um sistema servidor o sistema cliente troca com este uma sucessão de mensagens.

O sistema cliente inicia enviando uma mensagem SYN para o servidor. O servidor confirma a mensagem SYN enviando SYN-ACK para o cliente. O cliente então finaliza o estabelecimento da conexão respondendo com uma mensagem ACK. A conexão entre cliente e servidor está aberta, e os dados podem ser trocados. A Figura 2.3 ilustra este fluxo de mensagens, chamado de *Three-Way Handshake.*

O ponto onde um potencial ataque pode acontecer está entre o momento que o sistema servidor envia um SYN-ACK de volta para o cliente e o momento que o cliente envia o ACK para o servidor para estabelecer a conexão. Esta fase é dita conexão *half-open* (semi-aberta). O servidor já alocou um *buffer* para cada conexão pendente. Esta estrutura de dados é de tamanho finito, e esta pode ser levada a um *overflow* pela criação intencional de muitas conexões parciais [9,10].

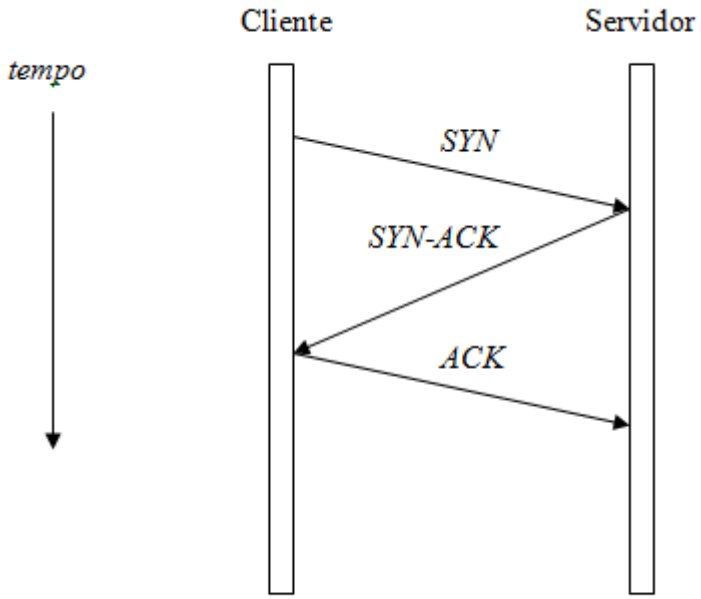

Figura 2.3 – *Three-Way Handshake* [9]

A criação de conexões *half-open* é facilmente completada com IP *spoofing*. O sistema atacante envia mensagens SYN para o sistema servidor vítima. Estas conexões parecem ser legítimas, mas de fato referem-se a um sistema cliente que não é capaz a responder à mensagem SYN-ACK. Isto significa que a mensagem final ACK nunca será enviada para o servidor vítima [9].

A estrutura de dados para conexões *half-open* eventualmente esgotará seu tamanho, então o sistema estará impossibilitado de receber novas conexões de entrada até que a tabela seja esvaziada. Normalmente existe um *timeout* associado com a conexão pendente, então as conexões *half-open* irão eventualmente expirar e o sistema servidor da vítima irá se recuperar. Porém, o atacante pode simplesmente continuar enviando pacotes *IP-spoofed* requisitando novas conexões mais rápido que a vítima pode fazer as conexões pendentes expirarem [2].

Na maioria dos casos, a vítima que sofre um ataque terá dificuldade em aceitar qualquer nova conexão. Nestes casos, o ataque não afeta as conexões já existentes nem a habilidade de criar conexões de saída.

Porém, em alguns casos, o sistema talvez esgote a memória, se tornando inoperável. A localização do sistema atacante é escondida porque o endereço de origem no pacote SYN são frequentemente falsificados [10].

Um variante de poder ampliado do ataque DoS é o ataque DDoS. Está técnica utiliza até milhares de computadores para atacar uma determinada máquina. Esse é um dos tipos mais eficazes de ataques e já prejudicou sites conhecidos, tais como CNN, Amazon, Yahoo, Microsoft e eBay [3].

Para garantir a eficiência dos ataques DDoS é necessário um grande número de máquinas chamadas “escravas” ou “zumbis”. Uma das maneiras encontradas para se escravizar estas máquinas é inserindo programas DoS em vírus que, ao infectarem as máquinas, as deixam em estado de espera para que em algum momento as máquinas “mestre” recebam o sinal da máquina “atacante” e o repassem para os “zumbis” para o disparo do ataque distribuído [3].Desta forma uma máquina atacante tem o poder de escravizar muitas máquinas “mestras”, que por sua vez podem ter milhares de máquinas “zumbis” sob seu controle, por essa razão o nome *Distributed Denial of Service*. A Figura 2.4 apresenta a hierarquia usada nos ataques DDoS.

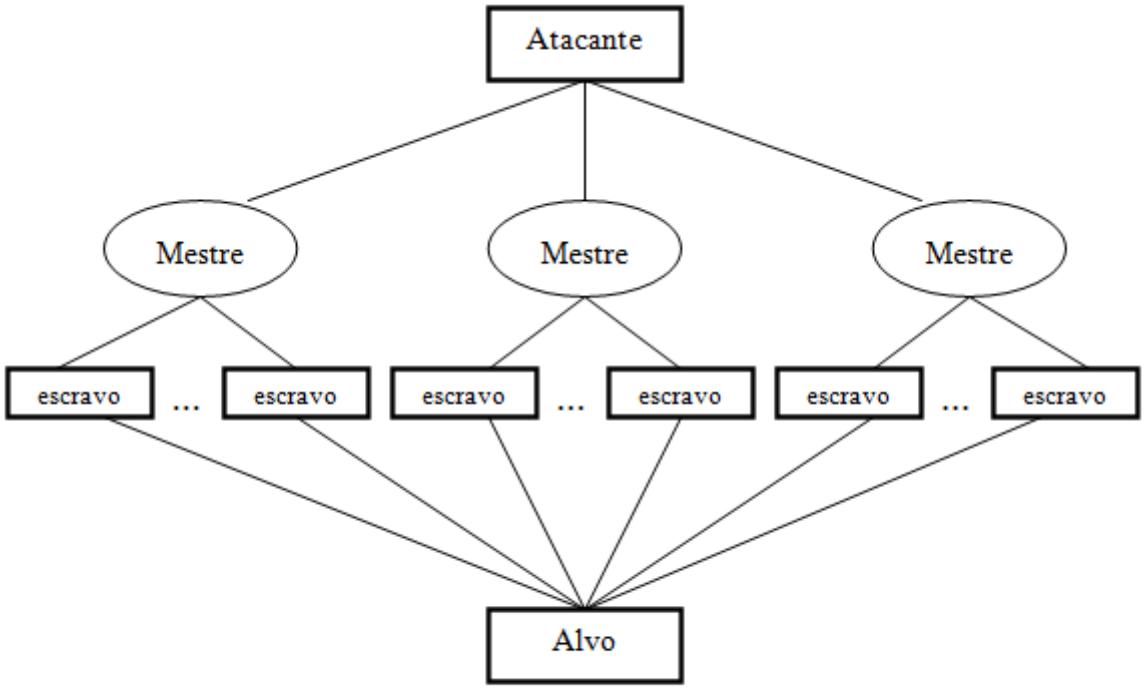

Figura 2.4 – Ataques DDoS

2.3 Conclusão

Este capítulo apresentou os conceitos básicos necessários para a compreensão do problema tratado.

Após a compreensão dos conceitos apresentados, torna-se possível o entendimento das técnicas descritas no Capítulo 3 e da relevância do estudo destas para segurança nas redes de computadores e principalmente da Internet.

3 Técnicas de Rastreamento de Pacotes

A dificuldade de se rastrear ataques em uma rede, em especial, ataques DoS e DDoS tem estimulado muitas pesquisas na tentativa de desenvolver soluções para o combate destes ataques. É praticamente impossível responder a um ataque sem que se descubra a real origem do mesmo. Portanto as técnicas de rastreamento de pacotes são essenciais para o combate de ataques. Diversas técnicas para implementar soluções para o rastreamento de pacotes em redes IP foram propostas. A seguir são apresentadas as principais técnicas encontradas na literatura.

3.1 *Tracing Anonymous Packets to their Approximate Source*

Burch e Cheswick [6] sugerem uma técnica baseada na inundação de roteadores para se descobrir quais pertencem à rota de ataque. No momento em que o ataque é identificado, a vítima inunda seus roteadores vizinhos com curtas rajadas de tráfego e, se o ataque é interrompido quando um roteador é inundado, esse roteador é selecionado como parte da rota de ataque. Em seguida inundam-se os roteadores vizinhos ao identificado e esse processo é repetido até que se encontre a origem do ataque. Essa técnica exige um conhecimento apurado da topologia da rede e necessita que roteadores no caminho do ataque estejam dispostos a serem inundados. O ataque deve durar tempo suficiente para que todos os nós sejam encontrados. Uma interrupção súbita do ataque faria com que a reconstrução da rota fosse interrompida e consequentemente a rota não fosse encontrada.

3.2 Marcação de Pacotes

A marcação de cada pacote com endereço IP dos roteadores que ele atravessar é uma maneira simples de identificar a rota real do pacote. Esse método conseguiria reconstruir a rota de ataque a partir de um único pacote. Porém segundo Laufer *et al.* [19], esse método não é ideal por ter desvantagens importantes como a inserção de dados (endereço IP dos roteadores) no cabeçalho dos pacotes aumentando o processamento no roteador e levando a um aumento progressivo do tamanho do mesmo, gerando uma consequente fragmentação do pacote que acarretaria em um processamento desnecessário ainda maior. Além disso, nada

impede que o atacante insira endereços IP inválidos no cabeçalho do pacote levando à inconsistência na reconstrução da rota.

3.3 *Network Support for IP Traceback*

Savage *et al.* [24] introduzem uma técnica baseada em auditoria, onde a vítima vai acumulando as informações necessárias para a reconstrução da rota. Os pacotes são marcados de maneira aleatória com dados que identificam o roteador. Assim a vítima, munida dos pacotes de ataque, pode coletar as informações presentes neles e usá-las para reconstruir a rota até o atacante. As desvantagens desta técnica estão na necessidade de alto poder de processamento da vítima para reconstrução da rota e a necessidade de um grande número de pacotes para que a reconstrução seja possível. Uma vantagem está no fato de o rastreamento poder ocorrer após o fim do ataque.

3.4 ICMP *Traceback*

Esta técnica propõe a escolha probabilística de pacotes que passam pelos roteadores. Quando um pacote é escolhido, o roteador envia uma mensagem ICMP ao endereço de destino do pacote (possível vítima). Este pacote ICMP contém informações a respeito do pacote escolhido e dos roteadores vizinhos ao responsável pela mensagem. O autor sugere que 1 a cada 20.000 pacotes sejam escolhidos. A vítima recebe as mensagens ICMP e a partir das informações contidas nas mensagens, pode reconstruir a rota do ataque.

Essa técnica tem muitas desvantagens, entre elas: pacotes ICMP geralmente são tratados de maneira diferente dos pacotes TCP/IP pelos roteadores, podendo ser bloqueados ou sofrerem restrições. Se alguns roteadores na rota de ataque não participarem da reconstrução, fica difícil encontrar os outros roteadores que estão do lado oposto e são necessários milhares de pacotes para que a rota possa ser reconstruída.

3.5 *Tracing Network Attacks to Their Sources*

Na abordagem feita por Tatsuya Baba e Shigeyuki Matsuda em seu trabalho *Tracing Network Attacks* [28] as informações de auditoria são armazenadas na própria rede em dispositivos acoplados ao roteador denominados “*tracers*”. Os roteadores coletam informações a respeito dos pacotes roteados e armazenam

nesses dispositivos. Durante a reconstrução da rota os "*tracers*" são consultados de maneira recursiva até identificarem a origem dos pacotes.

Esse modelo exige grande capacidade de armazenamento nos roteadores o que o torna inviável em grande escala, além do problema de privacidade acarretado pela invasão de um desses roteadores que contém informações de todos os pacotes roteados.

3.6 *Deciduous: Decentralized source identification for network-based intrusions*

Chang *et. al* [13] em um de seus trabalhos, descrevem o sistema *Deciduous*, esse sistema é baseado no protocolo IPSec. O IPSec é um conjunto de padrões utilizados para garantir uma comunicação segura entre dois computadores, mesmo que as informações estejam sendo enviadas através de um meio não seguro, como por exemplo a Internet. Observe que esta definição é parecida com a definição de VPN – *Virtual Private Network*. Por isso que a combinação do protocolo de tunelamento L2TP e IPSec é uma das opções mais indicadas para a criação de conexões do tipo VPN [23].

Para utilização do sistema proposto por Chang *et al.* [13], é necessário que a topologia da rede seja conhecida. Se existe um canal IPSec entre um roteador e a vítima, e os pacotes de ataque que chegam à vítima estiverem autenticados pelo IPSec, o ataque tem origem em um roteador alem deste que forma o túnel com a vítima, se o pacote não estiver autenticado, o ataque partiu de um roteador dentro do canal IPSec.

Para implementação do sistema, nenhuma nova funcionalidade precisa ser acrescida nos roteadores, porém é necessário que os clientes conheçam a topologia da rede do provedor. A vítima deve construir diversos túneis IPSec com os roteadores do provedor. A grande vulnerabilidade deste sistema é que os roteadores do provedor estão ao alcance do atacante, uma vez que a topologia da rede será conhecida.

3.7 Rastreamento de Pacotes IP Contra Ataques de Negação de Serviço

Snoeren em seu trabalho *Hash-Based IP Traceback* [26] sugere o uso do Filtro de Bloom [20] para reduzir o armazenamento de um grande conjunto de

informações por parte do roteador. O Filtro de Bloom é utilizado para armazenar de maneira compacta os dados dos pacotes, ainda assim o sistema exige grande capacidade de armazenamento.

O Filtro de Bloom é uma estrutura de dados usada para representar de forma compacta um conjunto de elementos. O filtro é composto por um vetor de bits e um grupo de funções *hash** diferentes. O vetor de bits é inicialmente preenchido com bits 0 e em seguida as posições do vetor correspondentes ao resultado de cada função *hash* aplicada ao valor que se quer compactar são preenchidas com bits 1. Dessa maneira, quando se quer descobrir se um determinado elemento está no filtro, basta aplicar as funções *hash* no elemento e verificar se as posições resultantes estão preenchidas com o bit 1.

Laufer *et al.* [19] propõem o uso do Filtro de Bloom de maneira diferente da proposta por Snoeren [26]. A técnica se baseia em armazenar informações sobre os roteadores no filtro que ficará no próprio pacote. Com essa técnica é possível armazenar a rota percorrida pelo pacote com apenas alguns bits extras no pacote.

Para melhor eficiência do filtro, seu estado inicial, ou seja, quando o pacote é lançado na rede deve ter todos os bits com valor 0. O uso do Filtro de Bloom para guardar o endereço IP dos roteadores por onde o pacote passou tem como vulnerabilidade o fato de o atacante poder inicializar o filtro contido em seus pacotes com o bit 1 ao invés do bit 0. Com isso apareceriam problemas de falsos positivos no rastreamento. Os falsos positivos [19] ocorrem quando, ao se reconstruir a rota, IPs que não fazem parte da mesma são incluídos nela devido ao fato das posições no vetor referentes ao resultado da função *hash* aplicada a esses IPs terem sido previamente preenchidas com o bit 1.

Laufer *et al.* [19] propuseram como solução para esse problema a criação de um Filtro de Bloom Generalizado (FBG). O Filtro de Bloom Generalizado também faz uso de um vetor de bits, mas ao invés de um conjunto de funções *hash*, são usados dois conjuntos de funções, o primeiro preenche uma parte dos bits do vetor com bits 1 e o segundo preenche a outra parte com bits 0, quando o resultado de funções que preenchem com bit 0 e funções que preenchem com bit 1 colidem na mesma posição é determinado o preenchimento com o bit 0. Essa nova abordagem com dois grupos de funções *hash* elimina a necessidade de o filtro estar inicialmente preenchido com bits 0. Essa nova implementação do Filtro de Bloom, ao contrário da versão original, pode gerar falsos negativos.

* Uma função *hash* é uma equação matemática que utiliza texto para criar um código chamado *message digest* (resumo da mensagem)[13].

Os falsos negativos [19] ocorrem quando alguma posição que deveria ser preenchida com o bit 0 é preenchida com o bit 1 ou quando uma posição que deveria ser preenchida com o bit 1 é preenchida com o bit 0. Isso pode ocorrer caso um outro elemento sobrescreva algum bit setado por um elemento inserido anteriormente. Por outro lado, se ocorrer das posições que deveriam ser preenchidas com o bit 0 terminarem preenchidas com bits 1 e as que deveriam ser preenchidas com o bit 1 serem preenchidas com o bit 0 são gerados os falsos positivos [19], que podem ocorrer por uma configuração inicial do vetor ou por resultado da aplicação das funções *hash* a um conjunto específico de elementos.

Na aplicação proposta por Laufer *et al.* [19] o número de elementos no Filtro de Bloom quando o pacote chega à vítima é o número de roteadores por onde o pacote passou. O tamanho do filtro é exatamente o número de bits alocados no cabeçalho do pacote para o FBG. As funções que zeram e preenchem bits em cada roteador devem ser as mesmas para que a reconstrução da rota possa ocorrer.

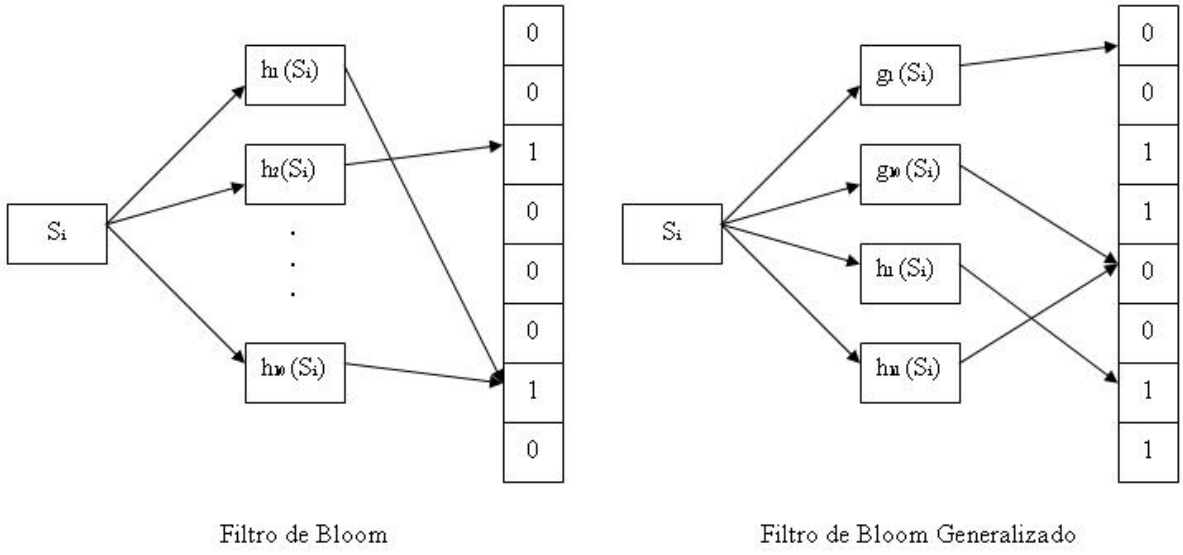

Figura 3.1 – Filtro de Bloom e Filtro de Bloom Generalizado [19]

Em um trabalho mais recente, Moreira *et al.* [31] propõem um novo filtro, chamado Filtro de Bloom Concatenado(FBC). O FBC [31] assim como as outras implementações do Filtro de Bloom descritas anteriormente, é uma estrutura capaz de armazenar dados de maneira compacta. Sua proposta é ser um filtro menos vulnerável ao seu estado inicial. O FBC é uma concatenação de pequenos filtros, denominados subfiltros. Somente um elemento é inserido em cada subfiltro. Além da

vantagem de se poder alterar o estado inicial do filtro, existe uma segunda vantagem que é a probabilidade nula de falsos negativos, devido ao fato de somente um elemento ser inserido em cada subfiltro.

A inserção no Filtro de Bloom Concatenado se dá de maneira "sequencial", ou seja, o i-ésimo elemento é inserido no i-ésimo subfiltro. Neste caso o filtro deve ter subfiltros suficientes para armazenar todos os elementos.

A verificação de presença de um determinado elemento no filtro usa uma espécie de contador para identificar o subfiltro onde o elemento deveria estar, desta maneira não é necessária a verificação de todos os subfiltros, somente aquele que o contador apontar. Uma discussão sobre o uso dos 3 filtros apresentados é abordada no capítulo 4.

3.8 Uma rede sobreposta no nível de sistemas autônomos para rastreamento de tráfego IP

Castelucio *et al.* [7] propõem uma abordagem baseada numa rede sobreposta com o protocolo BGP [12] e Sistemas Autônomos (SAs). Os roteadores com protocolo BGP usam a mensagem *Update* para trocarem informações de rota entre si. Roteadores vizinhos são conhecidos como *peers* BGP. A mensagem *Update* contém o *Path Attribute*, uma coleção de atributos associados às rotas que podem influenciar na escolha das mesmas. Entre os atributos do *Path Attribute* está o *Community Attribute* definido na RFC 1997 [12]. Um novo *Community Attribute* é criado, o chamado *IP Traceback Community* que guarda informações sobre os SAs, indicando que estes estão aptos a formar a rede sobreposta e realizar o rastreamento de trafego no nível de SAs.

Características presentes no *Community Attribute* e herdadas pelo *IP Traceback Attribute* permitem que ele seja repassado de maneira transparente por roteadores que não fazem parte da rede sobreposta, de modo que ao final de uma sequência de mensagens *Update*, todos os membros da rede sobreposta que contem o sistema proposto instalado sejam encontrados. Neste ponto todos os roteadores com o sistema instalado têm uma tabela contendo a rota para os demais SAs; esta tabela é chamada tabela de *overlay*.

A utilização da rede sobreposta com BGP elimina a necessidade de que todos os roteadores da rede tenham o sistema instalado, como ocorre na maioria das implementações expostas até o momento, e a reconstrução de rota se dá entre os roteadores que contém o sistema proposto instalado. Um exemplo de rede com

esta técnica é ilustrado na Figura 3.2, onde os roteadores marcados têm o sistema proposto instalado.

A marcação de pacotes neste sistema é uma modificação do esquema de marcação proposto por Laufer *et al.* [19]. Segundo Castelucio [7] a modificação tem como objetivo evitar que dois roteadores vizinhos apareçam no filtro de Bloom no momento de reconstrução da rota. Para que isso não ocorra, a modificação faz com que o pacote guarde informação de todos os roteadores com o sistema de rastreamento instalado. Assim o pacote chega ao seu destino com as informação de todos os SAs com o sistema instalado, por onde passou.

Figura 3.2 - Rede Sobreposta BGP [7]

A reconstrução da rota é feita baseada nas informações dos SAs por onde o pacote passou. O administrador do SA que deseja fazer o rastreamento inicia o mesmo procurando o SA mais próximo por onde o pacote passou. Quando descobre, manda o pacote para ele. Assim o processo se repete até que o SA mais próximo do atacante possa ser encontrado. Quando o SA é encontrado, este pode filtrar os pacotes vindos da fonte de ataque, bloqueando o ataque e evitando gasto de recursos desnecessários na rede.

3.9 Considerações

Este capítulo apresenta as principais técnicas para o Rastreamento de Pacotes em Redes IP. O Capítulo 4 se propõe a ilustrar melhor a comparação entre os Filtros de Bloom, Filtro de Bloom Generalizado e Filtro de Bloom Concatenado,

onde são apresentados resultados de simulação gerados com uma ferramenta desenvolvida para comparar a *performance* dos filtros.

4 Estudo de Caso

O estudo de caso faz uso de uma ferramenta de simulação para avaliar o Filtro de Bloom convencional, o Filtro de Bloom Generalizado e o Filtro de Bloom Concatenado, descritos na seção 3.7 do Capítulo 3, e apresenta resultados de simulação que apontam pontos fracos e fortes no uso de cada filtro na técnica proposta por Laufer *et al.* [19]. A análise das simulações aponta valores que otimizam o uso dos filtros combinando eficiência no rastreamento e menor *overhead* no datagrama IP.

As modificações feitas no Filtro de Bloom convencional se propõem a evitar que o atacante interfira na reconstrução da rota de ataque através da manipulação do estado inicial do filtro. As simulações apresentadas ilustram o impacto do uso de cada um dos três filtros na técnica proposta em [19].

4.1 Ferramenta de Simulação

Para simular o uso da técnica prosposta no trabalho Rastreamento de Pacotes IP Contra Ataques de Negação de Serviço [19], Laufer *et al.* desenvolveram uma aplicação [16], em linguagem C++, que simula o uso do Filtro de Bloom Generalizado para reconstrução da rota de ataque. Para o objetivo deste trabalho, que é avaliar não só o FBG, mas também o Filtro de Bloom chamado de convencional e o Filtro de Bloom Concatenado, algumas modificações na ferramenta foram necessárias e são descritas na seção 4.2.

A ferramenta recebe como parâmetros um arquivo texto **`<file>`** gerado com o gerador de topologia ***nem*** [30], baseado em amostragem do mapa da Internet. O gerador ***nem*** extrai aleatoriamente um sub-grafo de um mapa de rede, mantendo suas propriedades originais, como grau de vizinhos, distância média e diâmetro da topologia. Os “nós da rede” descritos no arquivo tem numeração sequêncial e nas simulações deste trabalho vão de 0 a 9999, ou seja, a topologia utilizada tem 10000 nós. A Figura 4.1 apresenta o formato do arquivo.

```
2 30
2 31
2 35
```

Figura 4.1 – Formato do Arquivo de Topologia

Os outros parâmetros são **<k0>** número de funções *hash* que setam bits como 0, **<k1>** número de funções *hash* que setam bits como 1, **<m>** tamanho em bits do Filtro de Bloom, **<n>** número de nós na rota de ataque e **<p1(0)>** fração de bits iniciados com 1 no filtro. O número de parâmetros varia de acordo com a implementação do Filtro utilizado.

Filtro de Bloom convencional: **<file> <k1> <m> <n> <p1(0)>**.
Filtro de Bloom Generalizado: **<file> <k0> <k1> <m> <n> <p1(0)>**.
Filtro de Bloom Concatenado: **<file> <k1> <m> <n>**.

A rota de ataque é gerada aleatoriamente de acordo com a topologia encontrada no parâmetro **<file>** e com tamanho igual ao parâmetro **<n>**. Após a criação da rota, é simulado um ataque, onde o pacote caminha pela rota gerada e são inseridos em seu filtro dados sobre cada nó por onde ele passou. Quando o pacote chega à vítima, inicia-se a reconstrução da rota, a reconstrução é feita da maneira descrita para cada filtro na seção 3.7 do capítulo 3. Ao final da reconstrução, a ferramenta retorna se a reconstrução chegou ao atacante, o número de possíveis atacantes encontrados e o número de falsos positivos e falsos negativos encontrados no filtro. A Figura 4.2 exibe a execução do simulador.

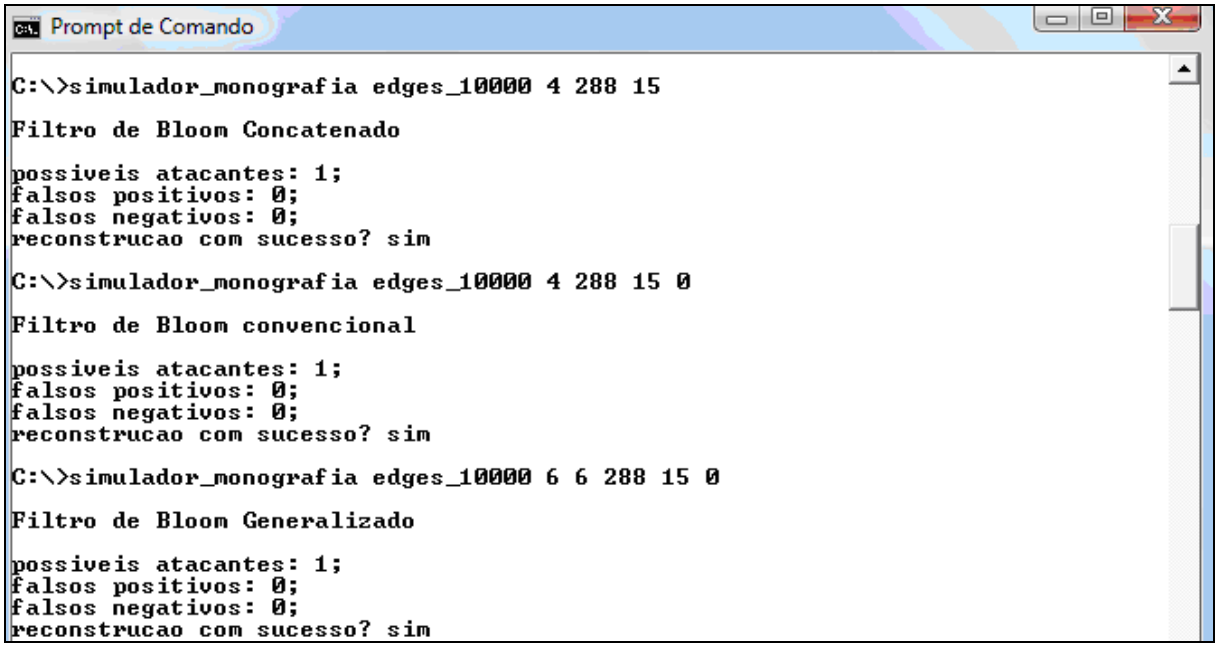

Figura 4.2 – Execução do Simulador

4.2 Modificações na Ferramenta

Para que a ferramenta descrita na seção 4.1 atendesse ao estudo e aos objetivos deste capítulo, algumas modificações foram necessárias e são descritas a seguir.

As principais modificações foram a inclusão dos arquivos ***cbf.h, cbf.cpp*** que contém a implementação do Filtro de Bloom Concatenado seguindo o algoritmo proposto por Moreira *et al.* [31] e que tem seu funcionamento descrito na seção 3.6. Também foram incluídos os arquivos ***bf.h*** e ***bf.cpp*** contendo a implementação do Filtro de Bloom convencional descrito em [20].

Os arquivos *network.cpp* e *network.h* já existentes e responsáveis pela geração da rota de ataque, marcação do filtro, e reconstrução da rota para o FGB foram modificados para incluir estas funcionalidades com o uso dos demais filtros e o cálculo dos Falsos Positivos e Falsos Negativos presentes nos filtros em cada simulação. Para essas modificações foram feitas sobrecargas nos métodos *marking, reconstruction e build_graph_alternative* além da inclusão dos métodos *false_positive* e *false_negative*.

O arquivo *main.cpp* também foi modificado, apenas para que seja possível escolher com qual dos três filtros se deseja fazer a simulação.

O Anexo A apresenta a listagem do código das modificações realizadas na ferramenta.

Após estas modificações, a ferramenta pode servir de maneira satisfatória aos propósitos deste trabalho e nas seções seguintes são apresentados os resultados das simulações, análises dos mesmos e inferências sobre estes resultados.

4.3 Cenários de Simulação

Para as simulações foram criados diversos cenários onde cada um dos parâmetros de entrada no simulador variou entre 4 valores escolhidos inicialmente. A Tabela 4.1 apresenta os valores escolhidos para cada parâmetro.

A combinação dos parâmetros de acordo com a Tabela 4.1 gerou 1.344 cenários diferentes de simulação e cada um desses cenários foi executado 20 vezes para evitar que a rota gerada aleatoriamente pelo simulador influencie no resultado.

Tabela 4.1- Valores para cada parâmetro do simulador

<k0>	3	4	5	8
<k1>	3	4	5	8
<m>	100	128	200	256
<n>	10	12	15	25
<p1(0)>	0	25	50	75

Os resultados retornados para cada uma das simulações foram gravados em um arquivo com extensão csv para facilitar a leitura e análise dos dados.

4.4 Análise dos Resultados de Simulação

Partindo das simulações apresentadas na seção 4.3, esta seção analisa os resultados obtidos e extrai destes alguns dados importantes na busca de um filtro ideal para o sistema proposto por Laufer *et al.* [19].

As melhorias criadas a partir do Filtro de Bloom convencional buscaram dar ao sistema de rastreamento uma maior confiabilidade, eliminando principalmente a possibilidade de que a manipulação do estado inicial do filtro por parte do atacante permita ao mesmo escapar do rastreamento.

A Figura 4.3 apresenta a comparação entre o Filtro de Bloom convencional, Filtro de Bloom Generalizado e Filtro de Bloom Concatenado em relação à média de atacantes encontrados em cada filtro quando se varia o estado inicial do mesmo. Entende-se como estado inicial do filtro a configuração do mesmo no momento em que o pacote é lançado na rede, ou seja, a quantidade de bits inicialmente preenchidos com valor 1 e a quantidade de bits inicialmente preenchidos com valor 0. O número de atacantes rastreados indica a eficiência do filtro, o valor ideal para este número é 1, o que indica que apenas o verdadeiro atacante foi encontrado.

Com a análise dos resultados apresentados na Figura 4.3 podemos comprovar a evolução do FBG em relação ao FB convencional e verificar que o FBC não tem sua eficiência comprometida pela variação do estado inicial do filtro. Quanto à eficiência do filtro, o FBC elimina ainda a ocorrência de falsos negativos que estão presentes no FBG, isto garante ao FBC a certeza de sempre encontrar o atacante, mesmo que haja falsos positivos no filtro.

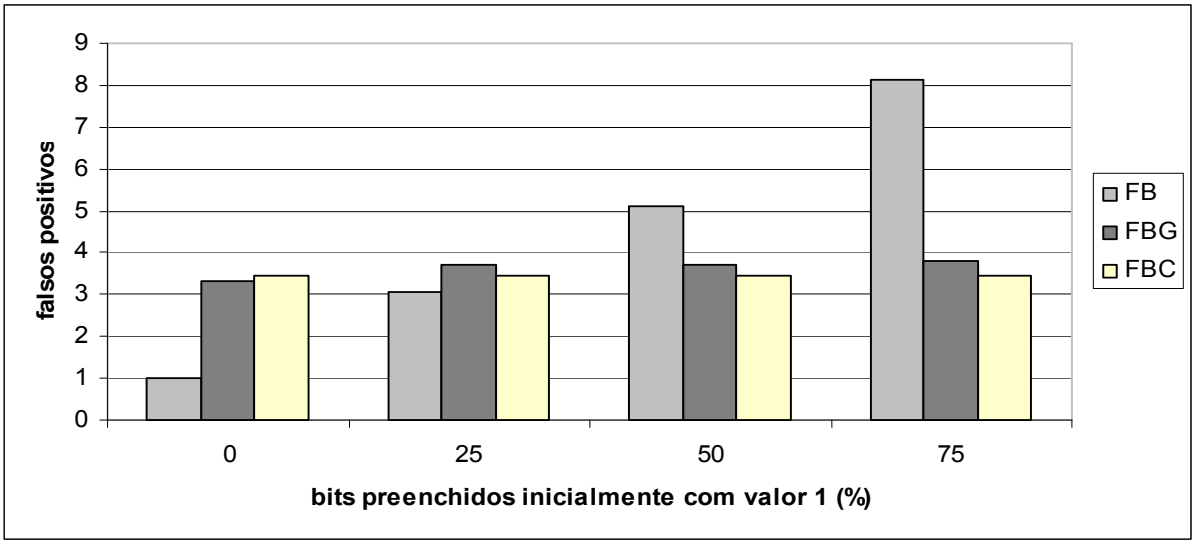

Figura 4.3 – nº de falsos positivos X estado inicial do filtro

A maior relevância desta seção é tentar encontrar valores próximos do ideal para os parâmetros de cada filtro, a fim de compara-los quanto à eficiência e *overhead* adicionado ao datagrama IP.

A primeira análise busca a melhor proporção entre o número de funções *hash* e o tamanho do filtro, para esta análise é apresentada a Figura 4.4 que exibe o gráfico de comparação entre o número de bits alterados pelas funções *hash* aplicadas a todos os elementos inseridos no filtro e o valor médio de atacantes rastreados. Esta análise é importante e valida a análise de Laufer *et al.* [19] comprovando que o número de funções *hash* interfere diretamente na ocorrência de falsos positivos e falsos negativos nos filtros. Vale lembrar que pela definição os falsos negativos só ocorrem no Filtro de Bloom Generalizado.

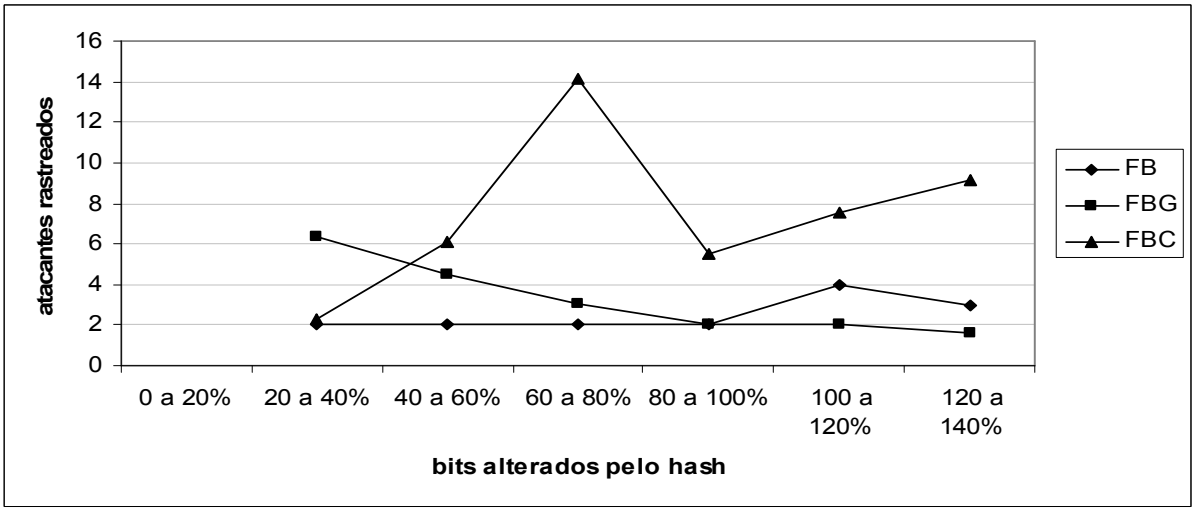

Figura 4.4 – bits alterados pelos *hashs* X atacantes rastreados

A Figura 4.4 mostra que o FB e FBC têm um melhor comportamento quando o número de *hashs* aplicadas aos elementos inseridos no filtro altera o valor de poucos bits no mesmo, ou seja, o número de *hashs* **k** aplicados ao número de elementos no filtro **n** deve estar entre 20 e 40% do total de bits do filtro **m**.

```
(k * n ~ 30% * m)
```

No caso do FBG este valor se mostra maior, principalmente pelo fato de dois grupos diferentes de funções *hash*, um grupo de *hashs* **k0** que setam valor 0 nos bits e um grupo de *hashs* **k1** que setam valor 1 nos bits, a figura 4.4 mostra a relação.

```
((k1 + k0) * n ~ 90% * m)
```

Estas relações são utilizadas para definir o melhor número de funções *hash* em relação ao tamanho do filtro e ao número de elementos a serem inseridos no mesmo.

Outra análise importante é a verificação da proporção entre o número de bits para cada elemento no filtro e o número médio de atacantes rastreados. Com esta análise pode-se encontrar um tamanho de filtro que não acrescente um *overhead* exagerado ao datagrama IP e ao mesmo tempo, não comprometa a eficiência do sistema de rastreamento. A Figura 4.5 apresenta os valores obtidos a partir desta análise.

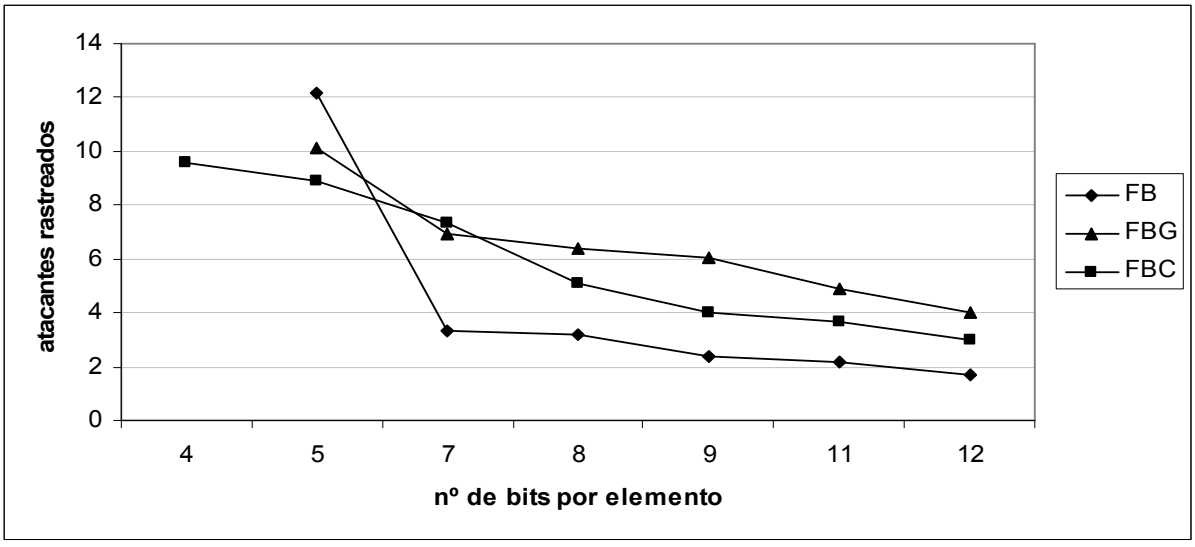

Figura 4.5 – nº de bits por elemento X atacantes rastreados

Os três filtros demonstram melhor eficiência com o aumento do número de bits por elemento, mas é importante que seja respeitado um limite satisfatório para o tamanho do filtro e a eficiência do rastreamento, pois o principal objetivo do filtro está na diminuição do *overhead* no datagrama IP. Considerando o tamanho de 32 bits de um endereço IPv4 e de 128 bits de um endereço IPv6, um filtro com 12 bits por elemento, gera uma economia significativa para o armazenamento de cada nó por onde o pacote passou. A economia está em torno de 62,5% em relação ao IPv4 e de 90,6% em relação ao IPv6.

Tomando 12 bits por elemento como um valor satisfatório entre *overhead* e eficiência, podemos calcular valores que sejam satisfatórios para o número de funções *hash* e tamanho real dos filtros.

Segundo estatísticas da Internet [29], o número médio de saltos de um pacote na rede é de aproximadamente 15 saltos e mais de 95 % dos pacotes trafegados precisam de menos de 25 saltos para chegar ao seu destino final. Considerando estes valores e o valor 12 bits por elemento no filtro, sabendo que devido ao fato da vítima não marcar o pacote, o número de elementos no filtro corresponde ao número de saltos do pacote menos 1. Temos o tamanho do filtro **m** igual a 288 bits no pior caso e no caso médio 168 bits.

$$(m = (25 - 1) * 12 = 288)$$

ou

$$(m = (15 - 1) * 12 = 168)$$

Laufer *et al.*[19] faz uma análise parecida, e aponta valores entre 256 e 320 bits para o tamanho do filtro, com intuito de diminuir os falsos positivos. Diminuir os falsos positivos diminui também o número de falsos atacantes, o que nos leva a uma relação entre os resultados apresentados pelo autor e os apresentados por este trabalho.

O número de funções *hash* seguindo os resultados de simulação apresentados no gráfico da Figura 4.4 é.

Para o FB e FBC:

$$(k * n \sim 30\% * m),\ m = 288,\ n = (25 - 1)$$

$$k \sim 4$$

Para o FBG:

$$((k1 + k0) * n \sim 90\% * m),\ m = 288,\ n = (25 - 1)$$

$$(k1 + k0) \sim 11$$

tomando como regra a descrição do Filtro de Bloom Generalizado [5] onde o número de funções hash **k0 = k1** e com o devido arredondamento

$$k1 = k0 = 6$$

Assim temos os melhores valores para o FB: **k** = 4, **m/n** = 12 que devem tornar o sistema de rastreamento com o Filtro de Bloom convencional capaz de encontrar a rota com eficácia satisfatória, quando o filtro está inicialmente preenchido com bits 0 ou seja, o melhor caso. Para o FBG temos **k0** = 6, **m/n** = 12 e para o FBC **k** = 4 e **m/n** = 12 que para o FBC corresponde ao tamanho de cada subfiltro.

4.5 Simulações com Valores Otimizados

A seção 4.4 apresentou valores que otimizam a relação eficiência VS *overhead* para cada filtro, esses valores foram novamente submetidos ao simulador com uma bateria de 40 simulações para cada filtro, para verificarmos a eficácia dos filtros com a configuração descrita. A Figura 4.6 apresenta a *performance* de cada filtro em relação ao seu estado inicial.

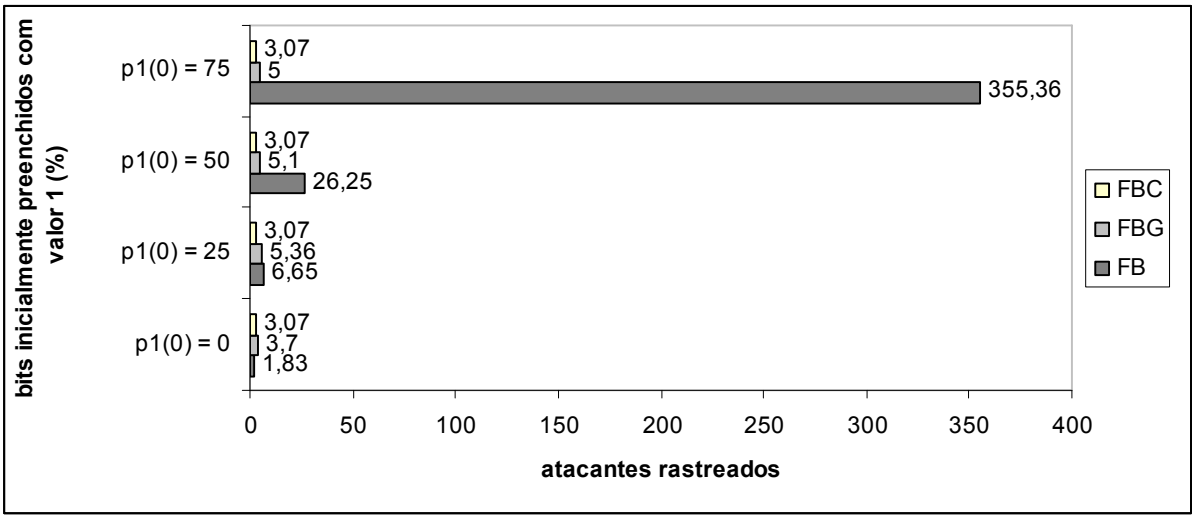

Figura 4.6 - atacantes rastreados X estado inicial do filtro

O Filtro de Bloom convencional com tamanho **m** = 288 e com número de funções *hash* **k** = 4 é capaz de rastrear a origem do ataque em uma rota de tamanho

n = 25 com uma eficácia média de 1,83 atacantes quando seu estado inicial, ou seja, o estado antes do pacote ser lançado na rede tem todos os bits com valor 0. Essa foi a melhor eficiência entre todos os filtros, porém como demonstrado por Laufer *et al.* [19] e descrito na seção 3.7 do capítulo 3 o Filtro de Bloom convencional é muito vulnerável ao seu estado inicial. Quando o estado inicial tem 25% dos bits preenchidos com valor 1 a eficácia cai para uma média de 6,25 atacantes encontrados. Essa média aumenta ainda mais quando se tem 50% dos bits preenchidos inicialmente com valor 1, nesse caso a média é de 26,5 atacantes encontrados um valor inaceitável por ser maior que a própria rota, que tem 25 nós apenas. Em uma terceira situação, onde 75% dos bits do filtro estão inicialmente preenchidos com 1 a média de atacantes rastreados chega a 355. Essa queda da eficiência quando se manipula o estado inicial, comprova a importância de se encontrar um filtro mais seguro e menos vulnerável ao estado inicial.

O Filtro de Bloom Generalizado proposto para minimizar essa vulnerabilidade foi testado com tamanho **m** = 288, número de funções *hash* **k1** = 6 e **k0** = 6 em uma rota também de tamanho **n** = 25, e demonstrou uma melhora considerável. Quando todos os bits tem valor 0 no estado inicial a eficácia média é de 3,7 atacantes rastreados. Quando se tem 25% dos bits com valor 1 no estado inicial, o FBG teve eficiência de 5,36 atacantes encontrados. No caso onde 50% dos bits estão inicialmente preenchidos com valor 1 a eficácia está em 5 atacantes rastreados, demonstrando uma melhora considerável em relação aos 26,5 rastreados pelo FB. Essa evolução do FBG em relação à segurança fica ainda mais clara quando analisamos o caso onde 75% dos bits estão preenchidos com valor 1. Para o FB como mostrado anteriormente o valor é de 355 atacantes rastreados, enquanto que para o FBG o valor é 5 atacantes rastreados.

O Filtro de Bloom Concatenado se propõe a ser ainda mais seguro, e eliminar qualquer vulnerabilidade ao estado inicial. O FBC foi submetido ao simulador com a configuração: tamanho **m** = 288 e número de funções *hash* **k** = 4 em uma rota de tamanho **n** = 25, e demonstrou eficácia média 3,07 atacantes encontrados indiferente do seu estado inicial.

4.6 Conclusão

A análise das simulações mostra que o Filtro de Bloom Concatenado cumpre de fato com seu objetivo de eliminar a interferência do estado inicial do filtro

na reconstrução da rota, porém limita o número de nós rastreados à quantidade de subfiltros presentes no filtro.

Com o uso de 12 bits por elemento nos filtros é possível atingir uma eficiência satisfatória no rastreamento e ainda conseguir uma economia significativa no *overhead* do datagrama IP. Essa economia está em torno de 62,5% em relação ao IPv4 e de 90,6% em relação ao IPv6.

O Filtro de Bloom convencional em seu melhor caso, onde o filtro está inicialmente preenchido apenas com bits 0, teve maior eficácia que os demais filtros, constatando uma perda de eficiência em troca de maior rubustez e segurança por parte do Filtro de Bloom Generalizado e do Filtro de Bloom Concatenado.

5 Considerações Finais

A segurança das redes e Internet é um tema cada vez mais pesquisado e debatido, e um dos problemas que atingem as redes IP é a dificuldade de se conhecer a origem real dos pacotes trafegados por ela. Esse problema é inerente do modelo e dos protocolos usados nesse tipo de redes, principalmente a Internet. A dificuldade de se encontrar a real origem dos pacotes facilita a ação de usuários mal intencionados na rede. As técnicas de Rastreamento de Pacotes em Redes IP buscam identificar a origem de possíveis ataques e facilitar para que os responsáveis sejam punidos. Este trabalho apresenta as principais técnicas para o Rastreamento de Pacotes em Redes IP avaliando de maneira crítica cada uma delas.

O estudo de caso apresentado determina valores ótimos para o uso do Filtro de Bloom convencional, Filtro de Bloom Generalizado e Filtro de Bloom Concatenado como alternativa para minimizar o *overhead* no datagrama IP na técnica de marcação dos pacotes com informação dos roteadores por onde estes passaram. O trabalho mostra que com um filtro de tamanho 12 bits por elemento é possível atingir uma eficiência satisfatória no rastreamento e manter uma economia considerável no *overhead*, no caso de endereços IPv4 a economia é de aproximadamente 62,5% e com endereços IPv6 a economia é ainda maior, aproximadamente 90,6%.

Considerando o caso ótimo de rastreamento, onde apenas 1 atacante é encontrado, a análise dos filtros mostra que o Filtro de Bloom convencional em seu caso ótimo, onde em seu estado inicial todos os bits estão com valor 0, teve sua eficiência mais próxima do ideal, média 1,83 atacantes encontrados, em relação ao Filtro de Bloom Generalizado com média de 3,7 atacantes encontrados, e Filtro de Bloom Concatenado com média de 3,07 atacantes encontrados. Porém a eficiência do FB é muito vulnerável ao seu estado inicial e isso compromete seu uso em no sistema de rastreamento.

O tamanho ideal e o número de funções *hash* foram propostos para que se padronize o filtro tendo um rastreamento eficiente e que ao mesmo tempo não adicione um *overhead* desnecessário ao datagrama IP.

O estudo de caso mostra que o FBG e FBC perderam em eficiência enquanto aumentaram sua segurança e robustez.

Propõe-se como trabalho futuro, a modificação da ferramenta de simulação para que os nós da rede sejam identificados por endereços IP ao invés de números seqüenciais. Essa modificação tornará a ferramenta mais próxima da realidade das redes. Propõe-se ainda a busca de um filtro com a mesma robustez e segurança do Filtro de Bloom Concatenado e com eficiência de rastreamento mais próxima do ideal de 1 único atacante encontrado no rastreamento.

Referências Bibliográficas

[1] ARNETT, M. Desvendando o TCP/IP. ed. Campus, 1996

[2] T.C.M.B. Carvalho. Arquiteturas de redes de computadores OSI e TCP/IP - 2. ed.- c1997

[3] Ataques DoS (*Denial of Service*) e DDoS (*Distributed DoS*). Disponível em: < http://www.infowester.com/col091004.php > Acesso em 10 nov 2008.

[4] Ataques DoS – Arbor Networks. Disponível em: < http://www.arbornetworks.com/ index.php?option=com_esearch&searchphrase=exact&ordering=oldest&search=&Itemid=553&id=16&search=research > Acesso em 12 nov 2008.

[5] BENNETT, G. *Internetworking* com TCP/IP - tecnologia e infra-estrutura. ed. IBPI Press, 1998.

[6] BURCH, H.; CHESWICK, B. "*Tracing anonymous packets to their approximate source," in Proc. USENIX LISA '00, Dec. 2000.*

[7] CASTELUCIO, André O.; RONALDO, M. Salles; ARTUR Z. Uma Rede Sobreposta no Nível de Sistemas Autônomos para Rastreamento de Tráfego IP. Instituto Militar de Engenharia – IME - Rio de Janeiro - RJ – Brasil, Laboratório Nacional de Computação Científica – LNCC - Petrópolis - RJ – Brasil.

[8] CERT. *CERT Advisory CA-1996-01 UDP Port Denial-of-Service Attack*, fevereiro de 1996. http://www.cert.org/advisories/CA-1996-01.html.

[9] CERT. *CERT Advisory CA-1996-21 TCP SYN Flooding and IP Spoofing Attacks*, setembro de 1996. http://www.cert.org/advisories/CA-1996-21.html.

[10] CERT. *CERT Advisory CA-1998-01 Smurf IP Denial-of-Service Attacks*, janeiro de 1998. http://www.cert.org/advisories/CA-1998-01.html.

[11] CERT. *CERT Advisory CA-2000-21 Denial-of-Service Vulnerabilities in TCP/IP Stacks,* novembro de 2000. http://www.cert.org/advisories/CA-2000-21.html.

[12] CHANDRA, R.; TRAINA P., and T. Li. *BGP Communities Attribute*. *Internet Engineering Task Force*. RFC 1997, 1996.

[13] CHANG, H.Y.; NARAYAN, R.; WU, S.F.; VETTER, B.M.; WANG, X.; BROWN, M.; YUILL, J.J.; SARGON, C.; JOU, F.; GONG, DECIDUOUS: *decentralized source identification for network-basedintrusions F. Integrated Network Management, 1999. Distributed Management for the Networked Millennium. Proceedings of the Sixth IFIP/IEEE International Symposium on Volume , Issue , 1999 Page(s):701 – 714*

[14] Datagrama IPv4: http://penta.ufrgs.br/Esmilda/datagram.html

[15] Datagrama IPv6: http://www.cert.org/advisories/CA-1996-21.html

[16] *GBF_TRACEBACK:* http://www.cs.ucla.edu/~rlaufer/research/gbf_traceback-1.0.tar.gz

[17] GONT, F. *ICMP Attacks Against TCP. Internet Draft: draft-gont-tcpm-icmpattacks-03.txt* (dezembro de 2004).

[18] Funções Hash Disponível em: < https://wwws.prodemge.gov.br/index.php ?option=com_content&task=view&id=167&Itemid=180 > Acesso em 25 jun 2009.

[19] LAUFER, R. P. ; VELLOSO, P. B. ; DUARTE, O. C. M. B. . Um Novo Sistema de Rastreamento de Pacotes IP contra Ataques de Negação de Serviço. In: XXIII Simpósio Brasileiro de Redes de Computadores (SBRC 2005), 2005, Fortaleza, CE, 2005.

[20] MITZENMACHER, M. *Compressed Bloom Filters. IEEE/ACM Transactions on Networking 10, 5* (outubro de 2002), 604–612.

[21] RFC 791- *Internet Protocol*(inglês). Disponível em: <www.ietf.org/rfc/rfc0791.txt> Acesso em 18 maio 2009.

[22] RFC 1883 - *Internet Protocol*, Version 6 (inglês).
Disponível em < http://www.rfc-archive.org/getrfc.php?rfc=1883 > Acesso em 18 maio 2009.

[23] ROSSI, Marco Antonio G.; FRANZIN O., VPN - *Virtual Private Network*(Rede Privada Virtual) - GPr Sistemas/ASP Systems - Agosto/2000 www.gpr.com.br/download/vpn.pdf

[24] SAVAGE, S., WETHERALL, D., KARLIN, A., E ANDERSON, T. *Network Support for IP Traceback. IEEE/ACM Transactions on Networking 9, 3* (junho de 2001), 226–237.

[25] SNOEREN, A. C., PARTRIDGE, C., SANCHEZ, L. A., JONES, C. E., TCHAKOUNTIO, F., SCHWARTZ, B., KENT, S. T., E STRAYER, W. T. *Single-Packet IP Traceback. IEEE/ACM Transactions on Networking 10, 6* (dezembro de 2002), 721–734.

[26] SNOEREN, Alex C.; PARTRIDGE, C.; SANCHEZ, Luis A.; JONES, Christine E.; TCHAKOUNTIO F.; KENT, Stephen T.; STRAYER W. Timothy, *Hash-Based IP Traceback - BBN Technologies 10 Moulton Street, Cambridge, MA 02138* www.ccert.edu.cn/upload/4/30.pdf

[27] STONE, R. *CenterTrack: An IP Overlay Network for Tracking DoS Floods*. Em 9th USENIX Security Symposium (Denver, CO, EUA, agosto de 2000), pág. 199–212.

[28] TATSUYA B.; MATSUDA S., *IEEE Internet Computing archive Volume 6 , Issue 2 (March 2002)* Paginas: 20 – 26 Ano de Publicação: 2002 ISSN:1089-7801

[29] T. 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.

[30] U. Magoni e J.-J. Pansiot, "Internet Topology Modeler Based on Map Sampling", em *IEEE International Symposium on Computer Communications*, julho de 2002.

[31] V. 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.

Anexo A – Listagem do Código dos Arquivos Modificados no Simulador