FILTRO DE PACOTES PARA O NÍVEL DE APLICAÇÃO

Bruno Cuencas Donath (bolsista CNPq)[1], Fabrício Sérgio de Paula[2]

[1]Estudante do Curso de Ciência da Computação da UEMS, Unidade Universitária de Dourados; Email: bruno.internet@gmail.com

[2]Professor do Curso de Ciência da Computação da UEMS, Unidade Universitária de Dourados; Email: fabricio@uems.br

Segurança Computacional.

Resumo

Na Internet, os pacotes trafegam utilizando regras chamadas de protocolo. Protocolos de aplicação utilizam um número de porta para se comunicarem e, com base nesse número, a maioria dos *firewalls* bloqueia tráfego indesejável. Entretanto, esse tipo de bloqueio pode ser burlado facilmente. Este trabalho apresenta um filtro que bloqueia tráfego baseando-se na forma do conteúdo dos pacotes recebidos e não simplesmente no número de porta. O mecanismo que utilizamos, *l7-filter*, permite especificar padrões ao *iptables* para que este filtre o conteúdo da conexão e, então, realize o bloqueio, se necessário. Durante este trabalho foi possível bloquear tráfego de aplicações padrão, mas houve falha com aplicações P2P. Dessa forma, este trabalho serve de complemento a pesquisas que já existem hoje, mas ainda há espaço para aperfeiçoá-lo em busca de uma maior eficácia.

Palavras-Chave: *Firewall*. Extração de assinaturas. Protocolos de aplicação.

Introdução

Diversas tecnologias têm sido empregadas visando garantir os quesitos de segurança computacional. *Firewalls* são aparatos muito utilizados e consistem na análise de protocolos até o nível de transporte, decidindo que tipo de tráfego é permitido ou não em uma rede (Nakamura & de Geus, 2003).

O nível de transporte, implementado pelos protocolos TCP e UDP da arquitetura TCP/IP, contém a identificação das portas envolvidas na comunicação (Stevens, 2000) (Peterson & Davie, 2003). Os *firewalls* utilizam essa identificação para bloquear certos tipos de aplicação indesejáveis em uma organização (*chat*, transferência de arquivos, P2P, etc.).

Dois mecanismos são usados para ludibriar *firewalls*: 1) aplicações que usam portas distintas das usuais; 2) tunelamento dentro de outras aplicações. Uma forma de contornar esse problema é identificar a aplicação envolvida em uma comunicação através de padrões contidos no conteúdo do tráfego na rede, ao invés de verificar apenas a identificação de porta.

O objetivo deste trabalho é estudar mecanismos de filtragem de pacotes e implementar uma aplicação capaz de identificar e bloquear tráfego de rede no nível de aplicação da pilha de protocolos TCP/IP. O filtro de pacotes proposto deve ser capaz de realizar a identificação de aplicações através da busca, no nível de aplicação, por padrões previamente conhecidos e fornecidos. Os padrões de busca podem ser extraídos automaticamente, o que é assunto de pesquisas correlatas que vêm sendo desenvolvidas.

Materiais e Métodos

Como parte das atividades propostas neste projeto, foram realizados estudos e experiências práticas envolvendo a pilha de protocolos TCP/IP, bem como a configuração de *firewalls* realizando a identificação por conteúdo. As fontes de referências bibliográficas incluem artigos disponíveis na Internet, e o acesso às bibliotecas da UEMS e UFGD.

Foi utilizado um laboratório com 02 computadores dedicados. Nesse laboratório, foram simuladas e implementadas diversas arquiteturas de *firewalls*, através do uso de máquinas virtuais gratuitas tais como *User-mode Linux*[1]e *VirtualBox*[2]

Foi utilizado o sistema operacional Linux e ferramentas de código aberto e gratuitas, de acordo com os requisitos da pesquisa. Como base para o estudo e desenvolvimento do filtro de pacotes foi utilizado o *framework netfilter*/*iptables*[3].

Resultados e Discussão

Programas de navegação na *web*, correio eletrônico e P2P (*peer-to-peer*) são considerados aplicações de rede. Tais aplicações usam regras para se comunicarem, chamadas de protocolos. Os protocolos são responsáveis pelo envio e recebimento de informações dentro da rede. O TCP e o IP (*Transmission Control Protocol* e *Internet Protocol* respectivamente) são os protocolos mais importantes na Internet (Cantú, 2003).

A sigla TCP/IP não indica um ou dois protocolos, mas uma pilha de protocolos. Existem diversos protocolos que compõem essa pilha, além do IP (nível de rede) e do TCP (nível de transporte), tais como o UDP (nível transporte) e muitos outros mais do nível de aplicação da pilha: SMPT (e-mail), FTP (transferência de arquivos), TELNET (*shell* remoto), HTTP (páginas *web*), EDONKEY (transferência de arquivos P2P), etc.

---

[1] Disponível em http://user-mode-linux.sourceforge.net.
[2] Disponível em http://www.virtualbox.org.
[3] Disponível em http://www.netfilter.org.

Os protocolos de aplicação se comunicam utilizando um número de porta que, em geral, é bem conhecido. As aplicações mais comuns usam portas com o número fixo padrão, como o protocolo HTTP utiliza porta 80, e o protocolo SMTP utiliza porta 25.

Para manter a segurança na rede, companhias protegem e isolam seus sistemas internos da Internet com um *firewall* de rede. *Firewall* é um subsistema de *software*/*hardware* que intercepta pacotes antes de permitir a eles a entrada na rede interna. Para cada pacote, o *firewall* compara componentes conhecidos com as regras de segurança aplicadas e determina se o pacote deveria ser permitido passar (*Firewall Architecture*, 2001).

A primeira linha de defesa de um *firewall*, e a utilizada neste trabalho, é o filtro de pacotes (*packet filtering*). Sistemas *packet filtering* observam pacotes entre que estão e saindo de maneira seletiva. Eles permitem ou bloqueiam certos tipos de pacotes, refletindo a política de segurança adotada (Ciferri, Ciferri & França, 1997).

Os *kernels* Linux têm filtro de pacotes desde a versão 1.1. O *iptables*, atual *software* de filtro de pacotes, é a quarta geração no Linux e surgiu em 1999 para o *kernel* versão 2.4 (Eycheme, 2002). Por exemplo, o *iptables* pode bloquear uma conexão HTTP bloqueando conexões de porta 80.

Como citado, nosso objetivo é verificar o conteúdo dos pacotes para bloquear a aplicação. O *l7-filter* é um classificador para *o iptables/netfilter* que identifica pacotes baseado na camada de aplicação (Levandoski, Sommer, & Strait,2009). Ele examina os dados da aplicação para determinar qual protocolo está sendo usado para marcá-los adequadamente.

O *l7-filter* aceita todas as conexões enviadas para ele, cabendo ao *iptables* decidir quais enviar/recusar. O *iptables* trabalha com três pilhas para verificação de pacotes: a pilha INPUT que trabalha com os pacotes para o computador; a pilha OUTPUT que fiscaliza os pacotes que saem do computador; e a pilha FORWARD, que fiscaliza pacotes de encaminhamento, caso o computador seja roteador.

No nosso caso foi utilizada a pilha INPUT. Os testes foram realizados em um microcomputador com processador Intel Core i3 2,13GHz, 1,5 GB de RAM, 500GB HDD a 5400rpm, e utilizando a distribuição Ubuntu Linux versão 10.04 e *kernel* versão 2.6.32.

Para que os pacotes da máquina se redirecionem ao *l7-filter* utilizou-se o seguinte comando: `iptables –A INPUT -j NFQUEUE --num-queue 0`. A marcação de pacotes no *l7-filter* foi realizada da seguinte forma: `l7-filter -f [config-file]`. O arquivo de configuração apresenta os protocolos a serem marcados, utilizando os padrões (*strings*) da pasta `/etc/l7-protocols/protocols` que podem ser editados ou adicionados pelo usuário.

Com o l7-filter configurado em modo *kernel*, foi utilizado o seguinte comando para bloqueio: `iptables -t filter -A INPUT -m layer7 --l7proto` [nome do `protocolo] -j DROP.`

Os protocolos utilizados para o teste foram: HTTP, FTP, TELNET e EDONKEY. Com exceção do EDONKEY, as assinaturas utilizadas foram uma evolução das extraídas em trabalho anterior, retiradas com o auxílio do programa *wireshark*[4], no qual tiveram êxito de detecção de 31% em conjunto: HTTP com 29% de 25.965 conexões, FTP com 44% de 800 e TELNET com 52% de 1.974 (Donath & de Paula, 2009).

A eficiência do filtro para cada um deles é mostrado na Tabela 1. A tabela está organizada da seguinte forma: a coluna Protocolo apresenta o nome do protocolo a ser identificado e a coluna Porta indica o número da porta usada comumente; a coluna Padrão mostra a expressão regular; a coluna Exemplo apresenta uma *string* que pode ser gerada a partir da expressão regular dada; e coluna Eficiência apresenta a eficiência do filtro para bloqueio dos protocolos mencionados.

Tabela 1: Eficiência l7-filter no bloqueio de protocolos

Protocolo	Porta	Padrão (Expressão Regular)	Exemplo	Eficiência
HTTP	80	`http/(0\.9|1\.0|1\.1)[1-5][0-9][0-9]`	`http/1.1 200`	OK[1]
HTTP (Alternativo)	80	`post [\x09-\x0d -~]* http/[01]\.[019]`	`post` **CNI**[4] `http/1.1`	OK[1]
FTP	21	`220[\x09-\x0d -~]*FTP`	`220` **CNI**[4] `ftp`	OK[1]
TELNET	23	`\xff[\xfb-\xfe].\xff[\xfb-\xfe].\xff[\xfb-\xfe]`	ÿû.ÿû.ÿû	OK[1] OM[2]
EDONKEY	?	`(\xe3|\xc5|\xd4)`	(Å)	UM[3]

[1] Conseguiu Identificar o protocolo
[2](*overmatch*) - Reconheceu outros protocolos com esse padrão.
[3](*undermatch*) – Casualmente reconheceu o protocolo.
[4]Caracteres não imprimíveis

Como pode ser visto na tabela, os protocolos HTTP, FTP e TELNET conseguem ser facilmente reconhecidos, pois além de terem portas de número fixo, possuem uma estrutura organizada. Com o padrão TELNET foi possível identificar algumas instâncias do protocolo HTTP falsamente, e seria possível identificar outros protocolos que utilizam \xfb, \xfc, \xfd, \xfe, \xff, em hexadecimal, freqüentemente. Nesse caso, para melhor efeito, é necessário melhorar o sistema de geração de assinaturas e de reconhecimento de pacotes.

O EDONKEY se comportou como o esperado. Ele não possui padrão nenhum em seus pacotes e também utiliza portas aleatórias que podem ser definidas pelo usuário. O

[4] Analisador de protocolos. Disponível em: http://www.wireshark.org/

padrão apresentado, além de não identificar corretamente o protocolo, identifica outros protocolos falsamente. Protocolos P2P carregam essa característica.

Conclusões

Esse trabalho fez uma introdução ao problema do filtro de pacotes por conteúdo de conexão e apresentou a eficiência deste método. Através dos testes realizados na sessão anterior, pode-se constatar que o *l7-filter* é uma excelente opção para se utilizar na forma de filtro de pacotes, desde que os padrões sejam bem recolhidos. Isso ocorre porque o *l7-filter* depende unicamente das assinaturas e tenta acertar toda a expressão regular em qualquer lugar do pacote, o que poderia causar lentidão se a expressão for demasiada complexa.

Pretende-se estender o estudo sobre esse assunto de forma a criar uma aplicação viável que filtre protocolos com assinaturas geradas automaticamente. Em posse dessas assinaturas, deseja-se implementar um filtro que possa bloquear ou limitar o uso de alguns protocolos em tempo competitivo. Deseja-se também, verificar a possibilidade de gerar padrões para aplicações P2P que sejam viáveis, o que deixaria o trabalho bastante completo.

Agradecimentos

Os autores agradecem pelo apoio financeiro (bolsa) concedida pela CNPq e também por todos os acadêmicos, coordenadores e professores da UEMS que, diretamente ou indiretamente, contribuíram para a realização desse trabalho.

Referências

Cantú, E. 2003. **Redes de computadores e Internet.** CEFET , p. 16-17,33-52.

Ciferri, D.; Ciferri, R. & França, S. (Dezembro de 16-17 de 1997). **Firewalls**. IV *Workshop* de Administração de Sistemas Heterogêneos – UFPE, p. 01.

Donath, B. C & de Paula, F. S. 2009. **Estudo de um algoritmo para extração de substrings mais freqüentes.** 7º ENIC, UEMS, p. 15-16.

Eycheme, H. 2002. **iptables - Linux man page.** Disponível em die.net: http://linux.die.net/man/8/iptables (último acesso em março de 2010).

Firewall Architecture. 2001. **NEXTEP Broadband White Paper** , p. 1-4.

Levandoski, J.; Sommer, E. & Strait, M. 2009. **Application layer packet classifier for Linux.** Disponível em http://l7-filter.sourceforge.net (útimo acesso em Junho de 2009).

Nakamura, E. & de Geus; P. 2003. **Segurança de redes em ambientes cooperativos.** São Paulo, Ed Futura, 488p.

Peterson, L. & Davie, B. 2003. **Redes de Computadores uma abordagem de sistemas.** Rio de Janeiro, Ed. Elsiever, 616p.

Stevens, R. 2000. **TCP/IP Illustrated.** 1ª Edição, Vol. I. Addison Wesley, 600p.