BIBLIOTECA PARA CAPTURA DE PACOTES VISANDO ANÁLISE DO NÍVEL DE APLICAÇÃO

Jhonatan Ríchard Raphael[1]; Fabrício Sérgio de Paula[2]

[1]Estudante do Curso de Ciência da Computação da UEMS; e-mail: 017362@comp.uems.br
[2]Professor do Curso de Ciência da Computação da UEMS; e-mail: fabricio.paula@gmail.com

Redes de Computadores e Segurança Computacional.

Resumo

Este projeto buscou estudar e implementar uma biblioteca para capturar pacotes que possa ser facilmente utilizada para acessar o conteúdo de alguns protocolos da pilha TCP/IP. Com essa biblioteca o tráfego de aplicações é acessado de uma forma mais simples, visando a extração de assinaturas e a identificação de protocolos de aplicação. Assim, este projeto complementa uma pesquisa que visa extrair automaticamente padrões de protocolos de aplicação e a posterior e identificação desses protocolos baseados nos padrões encontrados. Como resultados foram desenvolvidos a biblioteca proposta e sua respectiva documentação.

Palavras-chave: Redes de Computadores. Arquitetura TCP/IP. Obtenção de Tráfego.

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, 2004). Os *firewalls* utilizam essa identificação para bloquear certos tipos de aplicação indesejáveis em uma organização (chat, ftp, p2p, etc.).

Para ludibriar *firewalls*, dois mecanismos têm sido empregados: 1) aplicações que usam portas distintas das usuais; 2) tunelamento dentro de outras aplicações. Assim, um *firewall* comum não consegue identificar a aplicação real, não funcionando de maneira adequada.

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. Baseando-se nessa idéia de identificação por conteúdo, alguns trabalhos correlatos foram realizados (HAFFNER, 2005; IPP2P, 2003).

O objetivo deste trabalho foi estudar e implementar uma biblioteca para capturar pacotes que possa ser utilizada para acessar facilmente e de forma organizada o conteúdo dos protocolos de aplicação da pilha TCP/IP. Entretanto, antes de acessar o nível de aplicação, foi necessário interpretar os protocolos no nível de transporte e rede. Dessa forma, a biblioteca também provém a interpretação dos protocolos Ethernet, TCP e UDP no nível de transporte, e IP, no nível de rede (STEVENS, 2000).

Material e Métodos

Foram realizados estudos e experiências práticas sobre a pilha de protocolos TCP/IP. As fontes de referências bibliográficas incluem artigos disponíveis na Internet, e o acesso às bibliotecas da UEMS e UFGD. As atividades práticas foram realizadas através do uso de ferramentas das ferramentas de captura e análise tcpdump[1] e wireshark[2].

Este projeto contou com o acesso aos recursos computacionais da UEMS, que inclui um laboratório com 02 computadores dedicados. Foi utilizado o sistema operacional Linux.

A biblioteca referência para captura de pacotes e decodificação dos protocolos utilizada foi a libpcap[3]: uma biblioteca de código aberto escrita em C, que fornece uma interface de alto nível para sistemas de rede de captura de pacotes. Foram criadas também outras bibliotecas para definir os cabeçalhos dos protocolos envolvidos: ip.h, ether.h, udp.h, tcp.h.

Resultados e Discussões

A Ethernet (também conhecida sob o nome de norma IEEE 802.3) especifica um padrão de enlace de dados em uma rede, onde cada máquina nessa rede possui um endereço distinto de 48 bits (TANENBAUM, 2003).

[1] Disponível em http://www.tcpdump.org.
[2] Disponível em http://www.wireshark.org.
[3] Disponível em http://www.tcpdump.org.

Acima do nível de enlace, situam-se os protocolos de rede, transporte e aplicação, de acordo com a pilha de protocolos TCP/IP (STEVENS, 2000; TANEMBAUM, 2003).

Segundo Carvalho (1997), a grande característica da Arquitetura TCP/IP é a simplicidade de implementação dos seus protocolos, que, mesmo assim, atendem aos requisitos de interconexão exigidos pela maioria dos sistemas.

O protocolo IP (*Internet Protocol*) implementa o nível de rede da arquitetura TCP/IP. É responsável pelo mecanismo de transmissão de datagramas entre quaisquer duas máquinas na Internet (TANENBAUM, 2003). O IP oferece três definições importantes. Primeira, o protocolo IP define a unidade básica de transferência de dados utilizada através de uma interligação em redes TCP/IP. Segunda, o *software* IP desempenha a função de *roteamento*, escolhendo um caminho por onde os dados serão enviados. Terceira, o IP inclui um conjunto de regras que concentram a idéia da entrega não confiável de pacotes (COMER, 1998).

A Camada de Transporte, representada pelos protocolos UDP e TCP, possui protocolos fim-a-fim, que consideram apenas a origem e o destino da comunicação. O UDP oferece um meio para as aplicações enviarem datagramas IP encapsulados sem que seja necessário estabelecer uma conexão (TANENBAUM, 2003). O UDP não é confiável no que se diz respeito à entrega dos pacotes ao destino final, podendo ocorrer perda de dados. O TCP é um protocolo de transporte orientado a conexão, utilizado para a maioria das aplicações da Internet. Sua principal característica é possuir uma entrega confiável e em sequência (TANENBAUM, 2003).

Visando facilitar o manuseio de pacotes Ethernet, IP, UDP e TCP, foi desenvolvida, neste trabalho, uma biblioteca de captura de pacotes que pode ser facilmente utilizada para acessar o conteúdo dos protocolos de aplicação da pilha TCP/IP, chamada de Libcaptura. Com os datagramas capturados, foi possível fornecer procedimentos para realizar a decodificação dos protocolos brutos.

Depois de realizado um estudo sobre a biblioteca base libpcap e gerado um projeto de como seria a biblioteca, iniciou-se a implementação. Durante o desenvolvimento, foram alteradas partes do projeto buscando melhorar a sua qualidade. Além disso, fatores como otimização (tempo de execução), facilidade ao usuário e tratamentos especiais foram levados em conta. Uma das funcionalidades implementadas é dar ao usuário a disponibilidade de criar uma função que manipule pacotes IP, TCP e UDP durante determinada captura, o que foi fundamental para ter sido alcançado o objetivo proposto.

Durante a implementação e testes, gerou-se uma documentação contendo todas as informações necessárias ao usuário, como ajuda e utilização da biblioteca corretamente.

Foram incluídos: descrição da biblioteca, especificações úteis, informações de cada módulo implementado (descrição, retorno, parâmetros e exemplo) e instalação no modo estático e dinâmico. Foram anexadas em seu final, informações que facilitam o uso ao usuário.

Depois de finalizada a implementação, foram realizados testes buscando verificar a corretude e eficiência da biblioteca. Cada módulo da biblioteca foi testado sob diversas situações (ex.: captura proveniente de arquivo, captura de dispositivo de rede, escrita em disco e tela). Posteriormente, foi realizada uma análise de desempenho (média de tempo de execução), comparando os resultados obtidos em relação ao tcpdump (utiliza a mesma biblioteca base para captura de pacotes: libpcap), executando uma mesma tarefa. Um diferencial da biblioteca desenvolvida surge do fato de ser propícia para uso em outros programas em C, ao contrário da ferramenta tcpdump. Alguns dados da análise a seguir:

Tarefa 1: ler do disco um arquivo de conexões chamado inside.tcpdump (539.946 kbytes e 1.947.815 pacotes) e escrever informações (em média 99 caracteres) de todos os tipos de pacotes (sem restrição quanto a protocolo) na saída padrão (monitor).

Tabela 1 - Comparação em tempo de execução para a Tarefa 1.

Número de pacotes	tcpdump	Libcaptura
10	0,15s	0,17s
100	0,38s	0,45s
1000	0,62s	0,98s

Analisando a Tabela 1, pode-se constatar que a diferença de desempenho é pequena, onde tcpdump é mais eficiente. Para haver justiça na comparação, foi desativada no tcpdump a opção de converter endereços para nomes, o acarreta perda de desempenho em uma impressão na tela.

Tarefa 2: ler do disco um arquivo de conexões chamado inside.tcpdump (539.946 kbytes e 1.947.815 pacotes) e despejar (escrever) esses pacotes em outro arquivo criado com extensão .tcpdump sem restrição de protocolos.

Tabela 2 – Comparação em tempo de execução para a Tarefa 2.

Número de pacotes	tcpdump	Libcaptura
1.947.815	9,5s (205.033 pacotes/s)	10,95s (177.882 pacotes/s)

Observando a Tabela 2, verifica-se que há um pequeno ganho de eficiência de tcpdump em relação a Libcaptura. Essa diferença pode ser justificada pelo modo de acesso ao disco de cada uma das aplicações, onde o restante do tempo é o processamento em si da tarefa.

Conclusões

Conclui-se que através do desenvolvimento deste projeto, além de adquirir conhecimentos teóricos necessários para realização do mesmo, foi possível também obter conhecimentos práticos, como técnicas de programação (na medida em que os problemas apareciam) e específicos do sistema operacional Linux. Dessa forma, desenvolvendo-se uma biblioteca que possa analisar o conteúdo de aplicação de um datagrama TCP/IP.

Em relação à funcionalidade e opções, Libcaptura oferece ao usuário diversas formas de personalização a captura em C, eliminando a utilização de outras ferramentas e *scripts* nessas situações. Quanto ao desempenho, pode-se afirmar que Libcaptura é bem eficiente, tendo em vista que a diferença de desempenho ao tcpdump não é significativa.

Além de alcançar o objetivo proposto, o desenvolvimento da biblioteca também proporcionou idéias para projetos futuros, como a implementação de um mecanismo para capturar dados de uma "conversa" UDP entre duas aplicações.

Agradecimentos

Agradeço à UEMS/PIBIC pela oportunidade de realizar minha primeira pesquisa na graduação, oferecendo o apoio financeiro e a infraestrutura requerida para a mesma; e às pessoas que contribuíram para a realização desse trabalho de alguma maneira.

Referências

Carvalho, Tereza C. M. de Brito. 1997. **Arquiteturas de redes de computadores OSI e TCP/IP**. São Paulo: Ed. Makron Books, 669p.

Haffner, P., Sen, .S, Spatscheck, O., Wang, D. 2005. **Acas: automated construction of application signatures**. Em MineNet '05: Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data, New York, páginas 197–202.

IPP2P. **IPP2P project is to identify peer-to-peer (p2p) data in ip traffic**. Disponível em: http://www.ipp2p.org. (último acesso em 01/06/2009).

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

Peterson, L., Davie, B. 2004. **Redes de Computadores: uma abordagem de sistemas**. Rio de Janeiro: Ed. Elsevier, 588p.

Stevens, R. 2000. **TCP/IP Illustrated, volume 1.** Ed. Addison-Wesley, 600p.

Tanenbaum, Andrew S. 2003. **Redes de Computadores**. Rio de Janeiro, Ed. Elsevier, 955p.