REMONTAGEM DE SESSÕES EM FLUXOS DE DADOS TCP/IP

Marcelo H. P. C. Chaves[*, 1], Antonio Montes[**, 1]

(1) Área de Sistemas de Informação

Laboratório Associado de Computação e Matemática Aplicada

Instituto Nacional de Pesquisas Espaciais (INPE)

(*) Mestrado, e-mail: mhp@lac.inpe.br (**) Orientador

RESUMO

Com o advento da Internet e rápida disseminação de vulnerabilidades relacionadas a sistemas computacionais, surge a necessidade do desenvolvimento de ferramentas que auxiliem administradores de redes e analistas de segurança no processo de detecção de intrusões. Este trabalho propõe a modelagem de informações extraídas de fluxos de dados em redes TCP/IP, que permita remontar e inferir o estado das sessões, utilizando apenas o cabeçalho dos pacotes. Esta modelagem irá viabilizar a implementação de um sistema de detecção de intrusão baseado nestas características.

Palavras-Chave: Detecção de Intrusão; Reconstrução de Sessões; Modelo; Redes TCP/IP

1 - INTRODUÇÃO

Este trabalho apresenta um modelo, gerado a partir do fluxo de dados em redes de computadores, baseado na pilha de protocolos TCP/IP, que visa a reconstrução de sessões. Foi tomado como referência para a modelagem o sistema de detecção de intrusões SHADOW [9]. O desenvolvimento também baseou-se em algumas premissas, tais como a análise *offline* dos dados coletados e a utilização apenas do cabeçalho dos pacotes, para a reconstrução e inferência do estado das sessões.

Nas abordagens comumente adotadas no desenvolvimento de sistemas de detecção de intrusões baseados em assinaturas de ataques, cada pacote é verificado individualmente, podendo disparar alertas sem que informações acerca de eventos anteriores e posteriores tenham sido levadas em consideração. Este fato possibilita que pacotes sejam especialmente montados, de forma que inúmeros alarmes falsos sejam gerados, sobrecarregando assim o sistema de detecção de intrusões e o processo de avaliação dos alarmes pelos analistas de segurança [3]. A grande vantagem da abordagem proposta neste trabalho, em contrapartida às abordagens normalmente adotadas no desenvolvimento de SDIs, é a análise de todo o conjunto de pacotes que compõem cada sessão. Decisões não são tomadas apenas sobre as informações contidas em um único pacote, mas sobre todo o histórico que o envolve. Isto minimiza a possibilidade de ocorrências de ataques como o descrito anteriormente.

Como o objetivo deste trabalho é criar um modelo que permita reconstruir e inferir sobre o estado das sessões, faz-se necessário estabelecer uma definição concisa para sessão. Uma sessão TCP/IP é definida por qualquer sequência de pacotes, que caracterize a troca de informações entre dois endereços IP, e que tenha início, meio e fim (mesmo que toda a comunicação esteja contida em um único pacote). Esta definição possibilita que, mesmo comunicações utilizando protocolos não orientados à conexão, como o UDP e o ICMP, sejam tratadas como sessões.

2 - A MODELAGEM

Nesta seção é apresentada a modelagem do fluxo de dados, em redes de computadores baseadas na pilha de protocols TCP/IP, visando a reconstrução de sessões. É apresentado o método escolhido para a representação do modelo, o processo utilizado em sua geração, bem como as metodologias abordadas para a resolução de alguns problemas relacionados à sua representação. Também é apresentada a modelagem das sessões TCP/IP e uma proposta para o problema relacionado à fragmentação de pacotes.

2.1 - Processo de Geração do Modelo

Segundo Gibbons [2], um grafo consiste em um conjunto de vértices interconectados por um conjunto de arestas. Para um grafo G, denota-se o conjunto de vértices por V e o conjunto de arestas por E, sendo $G=(V,E)$. Uma aresta no grafo G pode ser especificada por dois vértices que esta conecta. Se a aresta e conecta os vértices i e j, então denota-se $e=(i,j)$ ou $e=(j,i)$. Além disso, se $(i,j) \in E$, então i é adjacente a j e j é adjacente a i.

O fluxo de dados TCP/IP, base para a modelagem apresentada neste trabalho, nada mais é do que um conjunto de endereços IP e a comunicação (ou transferência de dados) entre cada par de endereços deste conjunto. Desta forma, o modelo é representado pelo grafo $G=(V,E)$, onde o conjunto de vértices V é constituído por todos os endereços IP do tráfego de rede, e cada aresta do conjunto de arestas E representa a comunicação entre dois endereços IP (vértices) do conjunto V.

Foi escolhida a análise *offline* para o fluxo de dados TCP/IP, ou seja, os pacotes já foram previamente coletados e armazenados. Cada pacote coletado é, então, lido e retornado para o processo de geração do grafo G.

Inicialmente, os conjuntos *V* (vértices) e *E* (arestas) do grafo *G=(V,E)* são vazios. À medida que cada pacote é lido e retornado para o processo de geração do grafo, três casos podem ocorrer. Para apresentar os casos, seja *i* um dos IPs do pacote e *j* o outro IP; e *e=(i,j)* a comunicação entre estes IPs, representada pelos dados contidos no pacote.

No primeiro caso, os vértices *i* e *j* não existem no conjunto *V* do grafo *G=(V,E)*. Ocorre na leitura do primeiro pacote entre dois IPs, onde ambos ainda não apareceram nos dados coletados. São adicionados ao conjunto *V* os vértices *i* e *j* e é adicionada ao conjunto *E* a aresta *e=(i,j)*. No segundo caso, o vértice *i* não existe no conjunto *V* do grafo *G=(V,E)*. Este ocorre na leitura do primeiro pacote entre dois IPs, onde um dos IPs ainda não apareceu nos dados coletados. É adicionado ao conjunto *V* o vértice *i* e é adicionada ao conjunto *E* a aresta *e=(i,j)*. No terceiro caso, os vértices *i* e *j* existem no conjunto *V*, mas a aresta *e=(i,j)* não existe no conjunto *E* do grafo *G=(V,E)*. Ocorre na leitura do primeiro pacote entre dois IPs, onde ambos já apareceram nos dados coletados, mas ainda não houve comunicação entre eles. É adicionada ao conjunto *E* a aresta *e=(i,j)*. A geração do grafo *G* consiste na criação de uma lista de adjacências [2], onde cada vértice tem uma lista associada de vértices adjacentes.

2.2 - Problema de Localização de Vértices (IPs)

A maioria das implementações de grafos necessita de um método sistemático para localização de vértices. Foram vistos três casos na seção 2.1, que envolvem a identificação de vértices, de forma que ações são tomadas de acordo com a existência ou não de um determinado vértice no conjunto *V*. Normalmente, métodos de identificação (localização) de vértices em um grafo, como *depth-first search* [2], têm complexidade linear - *O(max(n,|E|))* - onde n é o número de vértices e *|E|* é o número de arestas. Neste caso a complexidade está associada ao tempo de execução do método ou ao número de comparações realizadas para encontrar o vértice. Para minimizar o número de comparações feitas na busca de um determinado vértice, utilizou-se um método baseado na implementação de uma árvore binária balanceada [4]. São utilizados os vértices do grafo *G* como nós da árvore. A localização (ou identificação) de um nó, em uma árvore binária balanceada, tem complexidade logarítmica - $O(\log_2 n)$ - onde n é o número de nós (equivalente ao número de vértices no grafo *G*). Este método melhorou o desempenho na localização em pelo menos uma ordem de magnitude.

2.3 - Modelagem das Sessões TCP/IP

Na modelagem das sessões TCP/IP, cada aresta do grafo *G* contém três atributos, que são apontadores para as sessões referentes aos protocolos TCP [8], UDP [5] e ICMP [7], associadas aos dois IPs. Esta associação se dá através das listas de adjacências dos IPs, onde cada elemento da lista mantém uma referência para uma estrutura de apontadores das sessões TCP/IP. As sessões são organizadas em listas encadeadas de estruturas, que contém informações diferenciadas por protocolo. Cada sessão mantém um apontador para uma outra lista de estruturas, de modo que cada elemento desta lista contém dados referentes a cada pacote da sessão. Os apontadores para as sessões são inicialmente nulos. À medida que cada pacote é lido e retornado para o processo de geração do grafo *G*, este é analisado para verificar se pertence a alguma sessão existente. Caso pertença, a lista de estruturas contendo informações dos pacotes da sessão é atualizada. Caso contrário, uma nova sessão é criada, a lista de sessões e a lista de estruturas, contendo informações dos pacotes da nova sessão, são atualizadas. Além disso, cada elemento da lista encadeada de sessões mantém a informação sobre o datagrama IP [6] e o estado da sessão. Esta é atualizada, à medida que pacotes são lidos e elementos são inseridos na lista de informações de pacotes da sessão.

Cada elemento da lista encadeada de dados de pacotes deve manter a informação referente a direção da comunicação. Neste trabalho, a representação do tráfego de rede utiliza um grafo não direcionado. Portanto, é necessário criar um mecanismo (ou codificação) capaz de identificar quem é o endereço de origem e quem é o de destino para os dados de um pacote. Para apresentar a codificação proposta, tomam-se A e B como os enredeços IP de um dado pacote. A "seta" partindo de A para B indica que A é o IP de origem e B é o IP de destino. A recíproca é verdadeira.

- Se $A < B$ e $A \rightarrow B$, ou $A > B$ e $B \rightarrow A$, então código recebe 0 (zero).
- Se $A < B$ e $B \rightarrow A$, ou $A > B$ e $A \rightarrow B$, então código recebe 1 (um).

Com isto, para determinar a direção da comunicação entre dois IPs de um pacote em uma sessão reconstruída, basta comparar os IPs e analisar o código (0 ou 1) relativo à direção, identificando quem é o IP de origem e quem é o de destino.

2.4 - Problema da Fragmentação

O problema da fragmentação de pacotes é uma constante nas discussões de sistemas de detecção de intrusão. Este trabalho propõe uma metodologia para o tratamento de pacotes fragmentados, baseada na utilização de uma tabela de fragmentos. Cada entrada na tabela de fragmentos contém: endereço IP de origem, endereço IP de destino, e os campos *identification* e *protocol* do datagrama IP. Estas informações são utilizadas para distinguir fragmentos de diferentes pacotes. Além disso, mantém uma lista encadeada com informações de cada fragmento

e um campo que diz qual é o estado do pacote. À medida que fragmentos são lidos, a lista encadeada de fragmentos é atualizada, procurando manter os fragmentos ordenados. O estado de cada entrada da tabela de fragmentos pode assumir diferentes valores: *reassembling* (permanece neste estado até que todos os fragmentos sejam recebidos, ou algum problema seja detectado), a*larm* (depois que todos os fragmentos foram recebidos, indica se há algum problema na remontagem do pacote), *time-exceeded* (utilizado quando a mensagem ICMP *Fragment Reassembly Time Exceeded* (*type* 11) é recebida) e *reassembled* (indica que todos os fragmentos foram recebidos e que o pacote pôde ser remontado corretamente). Uma entrada na tabela de fragmentos é tratada como um pacote e repassada para o processo de geração do modelo (visto na seção 2.1), somente se seu respectivo estado tiver o valor *reassembled*. Os estado *alarm* sinaliza um comportamento anômalo.

CONCLUSÕES

Neste trabalho foi apresentado um modelo, gerado a partir de dados extraídos do tráfego de redes TCP/IP, visando a reconstrução de sessões. A modelagem proposta, representada por um grafo, viabiliza, de uma forma relativamente simples, a geração de estatísticas relacionadas ao tráfego de rede sendo analisado em um dado período, que podem ser utilizadas na construção de perfis de comportamento para a rede monitorada.

Uma característica desta modelagem diz respeito a identificação de varreduras (*scans*) [1], assim como a identificação da utilização dos endereços IP da rede interna (rede sendo monitorada) para *spoofing*. A contagem do número de arestas associadas a um determinado vértice (endereço IP) e a avaliação dos estados de suas sessões podem indicar o uso de técnicas de varredura, como *stealthy scans* [1], e também podem indicar que o endereço IP está sendo utilizado para *spoofing*.

Uma outra aplicação do modelo proposto consiste na correlação de eventos caracterizados por um conjunto de sessões distintas. Atualmente, um grande número de ferramentas automatizadas para ataques a sistemas têm sido disponibilizadas, de modo que os atacantes precisam apenas de alguns "cliques" para comprometer todo um sistema de informação. Exemplos de ações tomadas por este tipo de ferramenta são: a varredura da rede alvo; identificação do sistema operacional e de um possível serviço vulnerável; conexão com a porta disponibilizada pelo serviço, caracterizando o ataque propriamente dito; conexão com servidores de FTP fora da rede atacada; abertura de uma ou mais portas-dos-fundos (*backdoors*) nas máquinas comprometidas; e conexão com as portas recentemente abertas. Através da reconstrução das sessões estabelecidas entre atacante e rede (ou máquina) alvo e da correlação entre estas, seria possível identificar e sinalizar tais ataques.

Algumas extensões propostas para a continuação deste trabalho são: implementação do modelo proposto, de forma que este possa ser testado e avaliado; avaliação da metodologia proposta para o tratamento dos pacotes fragmentados, e o desenvolvimento de um sistema, que utilize esta modelagem como base para a análise de dados e para a detecção de intrusões.

REFERÊNCIAS

[1] Fyodor. Nmap - Network Mapper. [online]. http://www.insecure.org/nmap. Jul., 2001.

[2] Gibbons, A. Algorithmic Graph Theory., Cambridge University Press, 1985.

[3] Giovanni, C. Fun with Packets: Designing a Stick. http://www.eurocompton.net/stick/papers/Peopledos.pdf. Set, 2001.

[4] Gonnet, G. H.; Baeza-Yates, R. Handbook of Algorithms and Data Structures: in Pascal and C. 2.ed. Chatham: Addison-Wesley, 1991. 424 p.

[5] Postel, J. User Datagram Protocol. Requests for Comments RFC 768, 1980.

[6] Postel, J. Internet Protocol: DARPA Internet Program, Protocol Specification. Requests for Comments RFC 791, 1981.

[7] Postel, J. Internet Control Message Protocol: DARPA Internet Program, Protocol Specification. Requests for Comments RFC 792, 1981.

[8] Postel, J. Transmission Control Protocol: DARPA Internet Program, Protocol Specification. Requests for Comments RFC 793, 1981.

[9] Stocksdale, G. NSWC Shadow Index. http://www.nswc.navy.mil/ISSEC/CID. Jul., 2001.