André Proto

Detecção de eventos de segurança de redes por intermédio de técnicas estatísticas e associativas aplicadas a fluxos de dados

São José do Rio Preto
2011

André Proto

Detecção de eventos de segurança de redes por intermédio de técnicas estatísticas e associativas aplicadas a fluxos de dados

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, Área de Concentração em Sistemas de Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Campus de São José do Rio Preto.

Orientador: Prof. Dr. Adriano Mauro Cansian

São José do Rio Preto
2011

Proto, André.
Detecção de eventos de segurança de redes por intermédio de técnicas estatísticas e associativas aplicadas a fluxos de dados / André Proto. - São José do Rio Preto : [s.n.], 2011.
73 f. : il. ; 30 cm.

Orientador: Adriano Mauro Cansian
Dissertação (mestrado) - Universidade Estadual Paulista, Instituto de Biociências, Letras e Ciências Exatas

1. Computação. 2. Redes de computadores – Medidas de segurança. 3. Redes de computadores – Protocolos. 4. Detecção de intrusão. 5. Análise de fluxos. I. Cansian, Adriano Mauro. II. Universidade Estadual Paulista, Instituto de Biociências, Letras e Ciências Exatas. III. Título.

CDU – 004.7

Ficha catalográfica elaborada pela Biblioteca do IBILCE
Campus de São José do Rio Preto - UNESP

André Proto

Detecção de eventos de segurança de redes por intermédio de técnicas estatísticas e associativas aplicadas a fluxos de dados

Dissertação apresentada para obtenção do título de Mestre em Ciência da Computação, área de concentração em Sistemas de Computação, junto ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Campus de São José do Rio Preto.

BANCA EXAMINADORA

Prof. Dr. Adriano Mauro Cansian
UNESP – São José do Rio Preto
Orientador

Prof. Dr. Paulo Licio de Geus
UNICAMP – Campinas

Prof. Dr. Marcos Antônio Cavenaghi
UNESP – São José do Rio Preto

São José do Rio Preto, 01 de Agosto de 2011.

Dedico este trabalho

A minha mãe Áurea Carriti Proto e em memória de meu pai Wilson Rosa Proto, por me darem a oportunidade da vida, pelo amor incondicional e pela educação que sempre me proporcionaram.

AGRADECIMENTOS

A Deus, que permitiu minha existência e oportunidades para minha evolução.

Ao Prof. Dr. Adriano Mauro Cansian pela orientação, amizade, respeito, confiança e oportunidade a mim oferecida. Obrigado por todo conhecimento que proporcionou a mim e aos meus colegas de laboratório.

A minha maravilhosa esposa Verônica Palácio de Pádua Melo Proto, por todo amor dedicado, carinho e companheirismo ao longo destes anos juntos.

Aos meus pais Áurea e Wilson (em memória) e meus irmãos Anderson e Andressa, por todo respeito, carinho e convivência durante toda a vida.

A minha amiga e companheira de laboratório Maira Lambort Batista, pela amizade, confiança e pelas contribuições importantes que ajudaram no desenvolvimento deste projeto.

Aos meus amigos Leandro Arabi Alexandre, Jorge Luiz Corrêa e Luiz Heitor Waiteman, minha amiga Isabela Liane de Oliveira e todos os companheiros do laboratório ACME!, companheiros durante todos estes anos de convivência na Universidade.

Aos professores e funcionários do DCCE – IBILCE/UNESP, em especial ao Prof. Dr. Aleardo Manacero Junior, pelas importantes contribuições e aprendizados a mim proporcionados.

A CAPES pela confiança no trabalho e pela bolsa de estudos a mim ofertada durante o primeiro ano de mestrado.

Aos companheiros de trabalho do IBGE por todo apoio e paciência oferecidos na última fase de elaboração deste projeto.

A todos que direta e indiretamente participam de minha vida.

“Embora ninguém possa voltar atrás e fazer um novo começo,
qualquer um pode começar agora e fazer um novo fim.”

Francisco Cândido Xavier

RESUMO

Este trabalho desenvolve e consolida um sistema de identificação e correlação de comportamentos de usuários e serviços em redes de computadores. A definição destes perfis auxiliará a identificação de comportamentos anômalos ao perfil de um grupo de usuários e serviços e a detecção de ataques em redes de computadores. Este sistema possui como estrutura base a utilização do padrão IPFIX – IP *Flow Information Export* – como exportador de informações sumarizadas de uma rede de computadores. O projeto prevê duas etapas principais: o desenvolvimento de um coletor de fluxos baseado no protocolo *NetFlow*, formalizado pela *Internet Engineering Task Force* (IETF) como padrão IPFIX, que acrescente melhorias na sumarização das informações oferecidas, consumindo menor espaço de armazenamento; a utilização de técnicas de mineração de dados estatísticas e associativas para detecção, correlação e classificação de comportamentos e eventos em redes de computadores. Este modelo de sistema mostra-se inovador na análise de fluxos de rede por meio da mineração de dados, empreendendo características importantes aos sistemas de monitoramento e segurança computacional, como escalabilidade de redes de alta velocidade e a detecção rápida de atividades ilícitas, varreduras de rede, intrusão, ataques de negação de serviço e de força bruta, sendo tais eventos considerados como grandes ameaças na Internet. Além disso, possibilita aos administradores de redes um melhor conhecimento do perfil da rede administrada.

Palavras-chave: Computação. Segurança de computadores e redes. Protocolos de redes. Detecção de intrusão. Análise de fluxos.

ABSTRACT

This work develops and consolidates an identification and correlation system of users and services behaviors on computer networks. The definition about these profiles assists in identifying anomalous behavior for the users and services profile and detecting attacks on computer networks. This system is based on the use of standard IPFIX – IP Flow Information Export – as a summarizing information exporter of a computer network. The project provides two main steps: the development of a flow collector based on the NetFlow protocol, formalized by Internet Engineering Task Force (IETF) as IPFIX standard, which improves the summarization of provided information, resulting in less storage space; the use of data mining association techniques for the detection, correlation and classification of behaviors and events in computer networks. This system model innovates in the analysis through IPFIX flow mining, adding important features to the monitoring of systems and computer security, such as scalability for high speed networks and fast detection of illicit activities, network scan, intrusion, DoS and brute force attacks, which are events considered big threats on the Internet. Also, it enables network administrators to have a better profile of the managed network.

Keywords: Computation. Computer and network security. Network protocols. Intrusion detection. Flow analysis.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DDL	*Data Definition Language*
DER	Diagrama Entidade Relacionamento
DHCP	*Dynamic Host Configuration Protocol*
DML	*Data Manipulation Language*
DNS	*Domain Name System*
DoS	*Denial of Service*
FTP	*File Transfer Protocol*
GPL	*General Public License*
HTTP	*Hypertext Transfer Protocol*
HTTPS	*Hypertext Transfer Protocol Secure*
ICMP	*Internet Control Message Protocol*
IEEE	*Institute of Electrical and Electronics Engineer*
IETF	*Internet Engineering Task Force*
IP	*Internet Protocol*
IPFIX	*IP Flow Information Export*
NIDS	*Network Intrusion Detection System*
NTP	*Network Time Protocol*
POP3	*Post Office Protocol*
QoS	*Quality of service*
RTSP	*Real Time Streaming Protocol*
SGBD	Sistema Gerenciador de Banco de Dados
SQL	*Structured Query Language*
SSH	*Secure Shell*
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*

SUMÁRIO

Capítulo 1 – Introdução

O avanço significativo do número de aplicações que transmitem dados na rede representa um desafio ao monitoramento e segurança dos sistemas e usuários que compõem uma rede de computadores. Os recursos de redes são cada vez mais exigidos pelas aplicações implicando na necessidade de maior desempenho, seja pelo aumento da capacidade da largura de banda, seja pelo melhor aproveitamento do tráfego com a priorização de certos serviços mais utilizados em uma rede específica.

No contexto da intrusão de sistemas, o desenvolvimento de aplicações atrelado a baixa preocupação com questões de monitoramento e segurança da informação leva ao crescimento do número de vulnerabilidades em software e consequentemente o número de ataques que comprometem tais recursos computacionais. Isso porque a Internet possibilita uma fonte riquíssima de informações, entre elas documentações e programas que ensinam um usuário comum a realizar uma intrusão em sistemas computacionais. Com isso, em um cenário atual, vê-se a facilidade de recursos que os criminosos cibernéticos dispõem para comprometer um sistema.

Os principais tipos de ataques reportados aos órgãos de resposta de incidentes estão relacionados a fraudes, disseminação de vírus e *worms*, exploração de vulnerabilidades em programas, ataques de negação de serviço e varreduras em redes de computadores para identificação de serviços ativos (CERT, 2009). Com o intuito de proteger sistemas computacionais, diversos mecanismos de segurança são desenvolvidos. Neste contexto surgem os antivírus, programas especializados em identificar e remover programas maliciosos em máquinas pessoais e estações de trabalho, e os *firewalls*, dispositivos dedicados para análise e bloqueio de tráfegos não desejados. Os *firewalls* são utilizados também em ambientes de médio porte, como empresas e departamentos, na busca de proteger e isolar o

ambiente de trabalho interno do acesso indesejados de usuários externos. Por fim, os Sistemas de Detecção de Intrusão (*Intrusion Detection System - IDS*) são sistemas especializados que analisam determinado tráfego de rede em busca de padrões de comportamento que evidenciem uma intrusão. Estes sistemas são instalados em pontos estratégicos de uma rede de computadores como *firewalls* ou dispositivos dedicados, analisando todo o tráfego pertencente a esta.

1.1 Identificação do problema e objetivos

No contexto de intrusão, dentre os IDSs existentes uma classe especial trouxe uma nova abordagem para análise de tráfego e detecção de intrusão: o *Network Intrusion Detection System* (NIDS). Esta nova classe possibilita a análise em âmbito de rede, e não apenas a análise individual por máquina conectada. Esse tipo de IDS tornou-se tendência na comunidade de segurança, visto seu maior poder de controle de um ambiente pelo administrador.

O grande desafio de um NIDS é realizar a análise do tráfego de uma rede de computadores de grande porte consumindo o mínimo possível dos recursos do dispositivo no qual está instalado. Sistemas como o *Snort* (ROESCH, 1998), que são implementados em dispositivos como *firewalls*, se adaptam bem em ambientes de rede de pequeno e médio porte, mas devido à sua abordagem em análise de conteúdo de cada pacote trafegado, seu uso em redes de grande porte torna-se computacionalmente custoso. Ou seja, a busca por informações valiosas em grandes bases de dados, com desempenho desejável, ainda é uma grande dificuldade nesses sistemas. Neste contexto surge uma alternativa viável em termos computacionais: o padrão IPFIX (QUITTEK *et al.*, 2004). Criado por um grupo de trabalho do *Internet Engineering Task Force* (IETF), o padrão IPFIX propõe uma série de especificações para exportação de informações de rede por meio de dispositivos localizados estrategicamente nesses ambientes, como roteadores e comutadores. Mais tarde tais especificações foram implementadas por diversos protocolos, sendo um destes o *NetFlow* (CLAISE, 2004), criado pela Cisco Systems, que tornou-se o protocolo oficial do padrão IPFIX.

O protocolo *NetFlow* exporta uma série de informações sobre o tráfego da rede em análise, possibilitando a manipulação de tais dados em busca de correlações com eventos em redes. Isso incentivou novas pesquisas em NIDS que se utilizam das informações provindas

do padrão IPFIX como base para a análise e detecção de intrusão em redes de computadores (MYUNG-SUP *et al.*, 2004; ZHENQI; XINYU, 2008; CORRÊA *et al.*, 2009).

Por outro lado, diversas pesquisas relacionadas à área de segurança da informação estão utilizando técnicas de outras áreas de conhecimento para associação e correlação de dados de ataques (PENG; ZUO, 2006; THURAISINGHAM *et al.*, 2008; SONG; MA, 2009). Entre essas estão às conhecidas técnicas de Mineração de Dados: possibilitam que informações relevantes (comerciais, por exemplo) sejam extraídas de determinado banco de dados com volume gigantesco de informações. Como os ambientes de rede geram uma grande quantidade de dados, essas técnicas se tornaram uma alternativa viável para a análise de tais informações.

Neste contexto, este trabalho tem como objetivo aliar as tecnologias provindas do padrão IPFIX, métodos estatísticos e associativos presentes em técnicas de mineração de dados e a abordagem de sistemas NIDS para identificar e correlacionar perfis de usuários e serviços em uma rede de computadores. Por meio desta análise, analistas e administradores de redes poderão identificar comportamentos anômalos de usuários ou serviços na rede, além de detectar atividades ilícitas e ataques a redes e sistemas. A identificação e correlação destas informações serão feitas em tempo hábil para contramedidas, além da possibilidade de aplicar tal sistema em redes de grande porte.

Este trabalho pode ser dividido em duas áreas de detecção principais: a detecção baseada em anomalias, na qual é utilizada uma metodologia estatística para a modelagem de padrões de comportamento de serviços e usuários; a detecção baseada em abuso, no qual é utilizada uma metodologia associativa para a geração de regras – similares a assinaturas – que representam o comportamento dos usuários da rede. Na primeira área a análise é feita sobre o tráfego de entrada de uma rede e cada padrão representa de modo geral o uso de determinado serviço. Já na segunda área a análise é feita sobre o tráfego de saída de uma rede e cada regra representa de modo individual o comportamento de um usuário acessando um determinado serviço.

1.2 Organização da monografia

Este documento é dividido nos seguintes capítulos: no Capítulo 2 é descrita a fundamentação teórica relativa ao tema, incluindo conceitos de ataque em redes, fluxos *NetFlow*, Mineração de Dados e conceitos de estatística. No Capítulo 3 são descritas as pesquisas mais recentes da área, relacionadas ao protocolo *NetFlow* e Mineração de Dados. No Capítulo 4 são descritas as metodologias desenvolvidas, a implementação do coletor capaz

de modificar o *NetFlow* gerando um padrão bidirecional dos dados exportados e as metodologias estatísticas e associativas de mineração de dados utilizadas para a identificação de perfis e ataques em redes. No Capítulo 5 são apresentados os resultados das metodologias desenvolvidas neste projeto e uma análise de desempenho das mesmas. Por fim, no Capítulo 6 são feitas as considerações finais sobre o tema.

Capítulo 2 – Fundamentação teórica

Este capítulo tem como objetivo descrever toda a fundamentação teórica das tecnologias utilizadas para o desenvolvimento deste projeto. Na Seção 2.1 são descritos conceitos de redes de computadores e a sua segurança, sendo detalhados alguns tipos de ataques relacionados a este trabalho. Em seguida na Seção 2.2 são descritos todos os conceitos relativos ao padrão IPFIX e o protocolo *NetFlow*. Na Seção 2.3 são detalhados conceitos e técnicas de mineração de dados em bancos de dados relacionais. Por fim na Seção 2.4 são relacionados conceitos importantes sobre estatística descritiva.

2.1 Redes de computadores e segurança

Todas as metodologias utilizadas foram aplicadas em redes que utilizam a pilha de protocolos TCP/IP. Maiores informações sobre a pilha de protocolos TCP/IP podem ser encontradas em Tanenbaum (2003). Alguns conceitos importantes tratados neste trabalho são descritos a seguir:

- **Serviços de rede:** o conceito de serviços de rede refere-se aos serviços da camada de aplicação. Estes serviços são utilizados pelas aplicações desenvolvidas e pelos usuários finais de uma rede;
- **Portas:** o conceito de portas refere-se às portas da camada de transporte. Estas definem a multiplexação e demultiplexação de serviços da camada de aplicação, em que cada serviço está atrelado a uma porta em específico;
- **Endereço IP:** Os endereços IPs tratados neste trabalho referem-se ao IP versão 4 da camada de rede.

Entre as várias definições de segurança de computadores e redes, uma das mais conceituadas está baseada na autenticidade, na confidencialidade, na integridade e na disponibilidade das informações (GOLLMANN, 1999). A autenticidade garante que um usuário é realmente quem este diz ser. A confidencialidade garante que apenas pessoas autorizadas tenham acesso a uma informação específica. A integridade garante que dados não sejam alterados de forma acidental ou intencional prejudicando a veracidade das informações. A disponibilidade garante que um sistema computacional funcione conforme esperado, não sofra degradações de acesso e esteja sempre disponível a requisições dos usuários.

A adequação de sistemas computacionais a essa definição é extremamente importante para a garantia de que o usuário se sinta seguro. No ambiente de redes de computadores aplicações bancárias, sites de compras, acesso remoto, bancos de dados, entre outros, são exemplos de aplicações que necessitam destas adequações.

Um ataque a um sistema computacional caracteriza-se pela tentativa ou a execução da violação de sua autenticidade, confidencialidade, integridade ou disponibilidade. No cenário de redes de computadores, existem diversos tipos de ataques para a violação desses quatro requisitos, seja por meio da exploração de vulnerabilidades ou pela consumação de recursos. Um resumo das principais técnicas está descrito a seguir:

- **Ataque de Negação de Serviço:** No ataque de negação de serviço um atacante utiliza um computador para tirar de operação um serviço ou outro computador conectado a Internet (CERT, 2006b). O objetivo do atacante é indisponibilizar serviços como páginas na internet, servidores de e-mail, comunicadores instantâneos, sistema de nomes de domínios (DNS) entre outros, de modo que usuários comuns não consigam utilizá-los;
- **Códigos Maliciosos:** Conhecido como *malware* são tipos de programas especificamente desenvolvidos para executar ações maliciosas em um computador (CERT, 2006b). Os tipos mais comuns são: vírus, *Worms***,** *keyloggers*, Cavalos de tróia, *Bot*, *Botnets* e *Rootkits.* A descrição de cada tipo pode ser encontrada em Cert (2006a);
- **Varredura de rede:** Esta é normalmente a primeira fase de um ataque. A varredura ou prospecção de rede é utilizada para reconhecer serviços disponíveis em uma ou mais redes, identificando serviços e *hosts* vulneráveis a ataques. Existem ferramentas conhecidas como *scanners* que foram desenvolvidas especialmente para dificultar e muitas vezes tornar invisível a varredura de uma rede para o administrador. Dentre as

ferramentas mais conhecidas estão *nmap*[1] e *Nessus*[2]. Para os analistas de segurança a varredura é um indicativo de pré-ataque, principalmente devido à rapidez de alguns tipos de ataques, que podem ter a duração de apenas alguns segundos (COOPER *et al.*, 2001). As varreduras são realizadas em duas etapas: na primeira é feito o reconhecimento da rede e dos *hosts* ativos; na segunda é feita a varredura das 65535 portas de cada *host* ativo, buscando por serviços disponíveis e informações destes;

- **Ataques de dicionário:** Neste tipo de ataque o invasor possui um banco de dados com sugestões de usuários e senhas comuns. Assim o atacante executa *scripts* que utilizam essa base para forçar uma tentativa de acesso a serviços remotos como SSH, Telnet, FTP, entre outros, realizando conexões de forma rápida e sequencial. Este tipo de ataque também é conhecido como ataque de força bruta e possui como forte característica o grande número de tentativas de conexões em uma mesma porta, de um ou mais *hosts*, em um curto espaço de tempo.

2.2 Fluxos de dados

Fluxo de dados ou fluxo de redes são termos utilizados para referenciar informações sobre o tráfego de uma rede provindas do protocolo *NetFlow*. Esta seção descreve os conceitos do *NetFlow* e a proposta do padrão de fluxos bidirecionais conhecida como *BiFlow*.

2.2.1 Protocolo *NetFlow*

A *Cisco* define um fluxo *NetFlow* como uma sequência unidirecional de pacotes entre *hosts* de origem e destino (CLAISE, 2004). Pode-se dizer em resumo que o *NetFlow* sumariza informações sobre o tráfego de um roteador ou *switch*. Pode-se entender um fluxo como um conjunto de dados na qual as seguintes informações aparecem com o mesmo valor: endereço IP de origem e de destino; porta de origem e destino referente ao protocolo da camada de transporte; valor do campo *Protocol* do datagrama IP; byte *Type of Service* do datagrama IP; e interface lógica de entrada do datagrama no roteador ou *switch*. Estes campos permitem que um fluxo represente concisamente o tráfego por meio de um ponto de observação como, por exemplo, em um roteador. Todos os datagramas com o mesmo valor em tais campos são contados, agrupados e empacotados em um registro de fluxo.

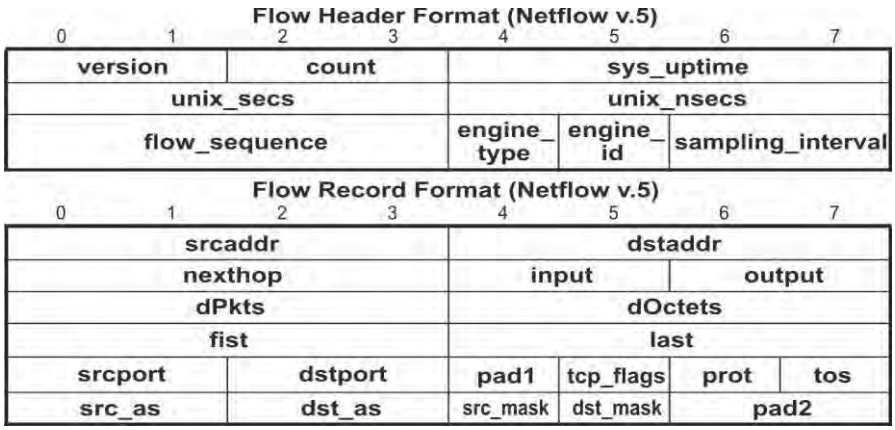
---

[1] *Nmap*. Desenvolvido por Fyodor e disponível em: http://www.insecure.org/nmap.

[2] *Nessus*. Disponível em: http://www.nessus.org.

A versão mais recente do *NetFlow* desenvolvida pela Cisco é a versão 9. Porém, devido às características dos equipamentos do ambiente utilizado neste projeto, a versão 5 do *NetFlow* foi utilizada. Na Figura 1 são ilustrados os campos do protocolo *NetFlow v.5*, bem como o seu cabeçalho.

Figura 1 – Formato de um datagrama *NetFlow*.



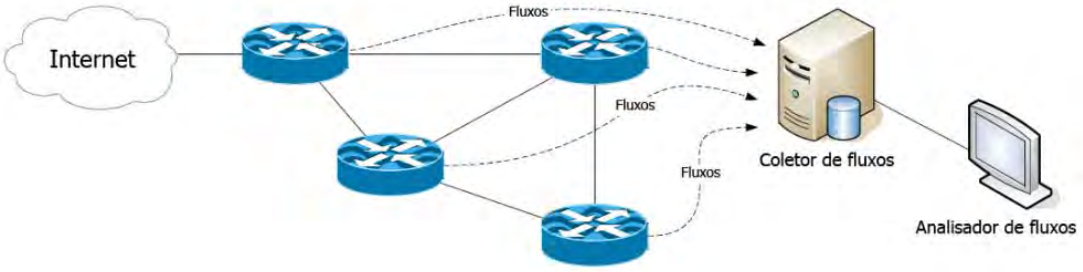
Fonte: próprio autor.

Os campos que realmente interessam neste trabalho estão descritos em *"Flow Record Format"*. Estes são responsáveis por representar as informações sumarizadas de uma conexão/sessão entre duas máquinas, descrevendo endereços de origem e destino, portas de origem e destino, interfaces de entrada e saída do roteador, número de pacotes e octetos envolvidos, *timestamp* de criação do fluxo e *timestamp* de sua última atualização (campos *first* e *last*), *flags* do TCP, entre outros. O campo *tcp_flags* registra as *flags* do segmento TCP no fluxo de forma acumulativa, ou seja, caso um cliente envie um segmento TCP com a *flag* SYN ligada e após algum tempo envie a *flag* FIN ligada, então esse campo guardará tanto a indicação de SYN quanto de FIN ligadas. A ordem de precedência dos *bits* é importante para entender os números inteiros resultantes das *flags*.

Os fluxos mantidos no *cache* de um dispositivo de rede são exportados nas seguintes situações: permanece ocioso por determinado período de tempo – padrão de 15 segundos; sua duração excede determinado período de tempo – padrão de 30 minutos; uma conexão TCP é encerrada com a *flag* FIN ou RST; a tabela de fluxos está cheia ou o usuário redefine as configurações de fluxo.

Os fluxos gerados nos equipamentos são exportados e armazenados conforme esquema da Figura 2, possibilitando uma fonte valiosa de informações. É possível obter detalhes de cada conexão estabelecida por qualquer máquina pertencente ao ambiente monitorado, fator este extremamente relevante nas análises de tráfego e de segurança.

Figura 2 – Arquitetura do ambiente de coleta de fluxos de dados.

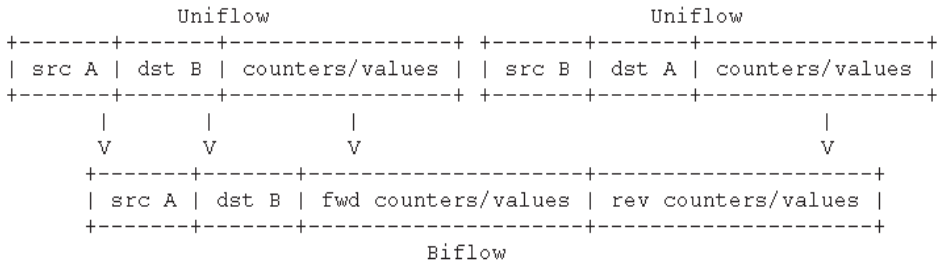


Fonte: próprio autor.

2.2.2 Conceitos do padrão *BiFlow*

O padrão *BiFlow* é uma proposta de protocolo de exportação de fluxos bidirecionais descrita no RFC 5103 (TRAMMELL; BOSCHI, 2008). O documento descreve um novo modelo de protocolo baseado no padrão IPFIX. Dentre as diferenças do *BiFlow* para os protocolos de exportação baseados no IPFIX, a principal delas é unir em um único segmento de fluxo a informação bidirecional de uma conexão/sessão. Isto porque a maioria dos protocolos e aplicações na Internet se baseia na comunicação entre dois *hosts*, ou seja, bidirecional. Com isso, a criação deste novo modelo vem eliminar informações duplicadas que existiam nos fluxos unidirecionais como, por exemplo, os campos *tos* e *protocol* de um fluxo *NetFlow*. Na Figura 3 é ilustrada a união de dois fluxos unidirecionais em um único fluxo bidirecional.

Figura 3 – Fluxos unidirecionais unificados em um fluxo bidirecional



Fonte: Figura retirada de Trammel e Boschi (2008).

Os campos do *BiFlow* podem ser subdivididos em três classes: direcionais, não-direcionais e campos de valores. Os campos direcionais referem-se aos campos com informações associadas a uma única direção do fluxo, como exemplo os campos *srcaddr* e *dstport* (IP de origem e porta de destino). Já os campos não-direcionais não estão associados necessariamente a uma única direção do fluxo, como, por exemplo, o campo *protocol*,

referente ao protocolo da camada de transporte. Estes últimos devem ser idênticos nos fluxos unidirecionais que compõem um fluxo bidirecional; já os campos direcionais devem ser idênticos aos campos de contrapartida entre os fluxos unidirecionais, por exemplo, o IP de origem de um fluxo unidirecional deve ser o IP de destino do outro fluxo unidirecional. Por fim os campos de valores representam as contagens e somas de pacotes e bytes pertencentes à comunicação.

Para determinar o iniciador de uma comunicação o RFC do *BiFlow* define três regras que deverão ser implementadas no exportador, que são: determinar a origem pelo iniciador da comunicação, assumindo que o IP de origem do primeiro pacote passante de uma comunicação é o iniciador, por meio da análise das *flags* do TCP ou analisando características dos protocolos de aplicação; determinar a direção baseado no perímetro de rede, em que um conjunto de endereços locais deve estar bem definido, sendo neste caso a origem sempre o comunicante externo; determinar a direção arbitrariamente. A prioridade na utilização das regras segue a ordem em que foram citadas.

Com base no *BiFlow*, pode-se obter uma redução significativa no armazenamento das informações disponibilizadas pelo protocolo. Apesar de em alguns casos não ser possível determinar os dois lados de uma conexão (origem e destino), ainda assim o *BiFlow* consegue por meio de sua proposta reduzir consideravelmente o número de tuplas[3] a serem armazenadas em um banco de dados. Isso auxilia computacionalmente no processamento dos dados fornecidos pelo protocolo, o que implica em um maior desempenho na consulta aos dados. Neste trabalho é desenvolvido um coletor de dados capaz de modificar as informações provindas do *NetFlow*, adaptando tais dados ao modelo de informações proposto pelo *BiFlow*.

2.3 Banco de dados relacional e Mineração de Dados

O banco de dados relacional é definido como uma coleção de dados relacionados (ELMASRI; NAVATHE, 2005); os “dados” são fatos que podem ser armazenados e que possuem um significado implícito. Um Sistema Gerenciador de Banco de Dados (SGBD) é uma coleção de programas que permite aos usuários ter um banco de dados. Assim, o SGBD é um sistema que visa facilitar a definição, construção, manipulação e compartilhamento de um banco de dados entre usuários e aplicações. O SGBD deve oferecer linguagens apropriadas para cada categoria de usuários. As duas principais linguagens são:

[3] Cada linha formada por uma lista ordenada de colunas representa um registro ou tupla.

- **Linguagem de Definição de Dados – *Data Definition Language* (DDL):** é usado por administradores de Banco de Dados (DBA) e projetistas para especificar os esquemas conceitual e interno;
- **Linguagem de Manipulação de Dados – *Data Manipulation Language* (DML):** é usado por usuários para manipulação dos dados como recuperação, inserção, remoção e modificação.

Com a utilização do SGBD, surgem vantagens como: controle de redundância; restrição de acesso não autorizado; garantia de armazenamento persistente para objetos programas; garantia de armazenamento de estruturas para o processamento eficiente de consultas; garantia de *backup* e restauração; fornecimento de múltiplas interfaces para os usuários; representação de relacionamentos complexos entre os dados; restrições de integridade e; permissão para inferências e ações utilizando regras.

Um SGBD possibilita o armazenamento de grande quantidade de dados relacionados, mas deixa a cargo do usuário a busca por informações relevantes nos mesmos. Neste contexto surge a mineração de dados (HAN; KAMBER, 2006) que é o processo de descobrir informações relevantes como padrões, associações, mudanças, anomalias e estruturas, em grandes quantidades de dados armazenados em banco de dados, depósitos de dados ou outros repositórios de informação. Devido à disponibilidade de enormes quantidades de dados em formas eletrônicas e à necessidade iminente de extrair delas informações e conhecimentos úteis a diversas aplicações, por exemplo, na análise de mercado, pesquisas científicas, administração empresarial, apoio à decisão, etc., a mineração de dados foi popularmente tratada como sinônimo de descoberta de conhecimento em bases de dados, apesar de, na visão de alguns pesquisadores, este seja considerado como um passo essencial da descoberta de conhecimento. Em geral, o processo de descoberta do conhecimento consiste de uma iteração entre três etapas principais:

- **Preparação:** os dados são preparados para serem apresentados às técnicas de mineração de dados. Os dados são selecionados de acordo com sua importância, purificados (remoção de inconsistências) e pré-processados para uma representação adequada à mineração de dados. A fase de pré-processamento possibilita a aplicação de uma série de metodologias estatísticas, que podem resultar desde uma modificação na representação dos dados até a obtenção de resultados satisfatórios para uma determinada área de pesquisa. Uma dessas metodologias foi utilizada neste trabalho para a identificação de

anomalias no tráfego das redes de computadores. Por isso, é feita na Seção 2.4 uma breve introdução ao conceito de estatística para a melhor compreensão desta etapa do projeto.

- **Mineração de dados:** Os dados são processados com as técnicas de mineração implementadas, na busca de informações importantes e relevantes à pesquisa.
- **Análise dos resultados:** o resultado da mineração de dados é avaliado determinando se algum conhecimento adicional foi descoberto. Neste passo vários métodos podem ser utilizados, como a representação dos resultados em um gráfico para análise comportamental do mesmo.

Um sistema de mineração de dados pode realizar pelo menos uma das seguintes tarefas: descrição de classes, associação, classificação, previsão, agrupamento e análise de série temporal. A descrição completa de cada tarefa pode ser encontrada em Han e Kamber (2006). As tarefas mais conhecidas são brevemente descritas a seguir:

- **Associação:** é a descoberta de relações de associação ou correlações entre um conjunto de itens. Uma regra de associação da forma X —+ Y é interpretada como "tuplas de base de dados que satisfazem X são prováveis que satisfaçam Y*";*
- **Classificação:** analisa um conjunto de dados de treinamento – por exemplo, um conjunto de objetos cuja classificação já é conhecida – e constrói um modelo para cada classe baseado nas características dos dados. Uma árvore de decisão ou um conjunto de regras de classificação é gerado por tal processo de classificação, que pode ser usado para entender melhor cada classe no banco de dados e para classificação de futuros dados;
- **Previsão:** esta função de mineração prediz os possíveis valores de alguns dados perdidos ou a distribuição de valores de certos atributos em um conjunto de objetos. Isso envolve a descoberta de um conjunto de atributos relevantes para o atributo de interesse;
- **Agrupamento:** análise de "*clusters*" ou de agrupamento consiste em identificar possíveis agrupamentos nos dados, onde um agrupamento é uma coleção de objetos que são semelhantes um ao outro. Diferentes medidas de similaridade baseadas em funções de distância podem ser especificadas para diferentes contextos de aplicação.

2.4 Conceitos de estatística

A fase de pré-processamento na mineração de dados oferece uma série de metodologias estatísticas a serem aplicadas aos dados em busca de uma melhor representação. Com isso, esta subseção faz uma breve introdução sobre a estatística descritiva. Estes conceitos são de suma importância para a compreensão da metodologia estatística utilizada neste projeto para a identificação de anomalias no tráfego em uma rede de computadores.

A estatística descritiva é usada para descrever e sumarizar um conjunto de dados. Os principais elementos a serem utilizados neste trabalho são:

- **Mediana:** Em um conjunto de dados ordenados, o elemento que separa ao meio tal conjunto, ou seja, existe a mesma quantidade de elementos tanto abaixo quanto acima dele, é considerado mediana. Em alguns casos em que o conjunto possui o número de elementos pares, não havendo um único elemento central, dois elementos centrais são somados e divididos por dois para resultarem na devida mediana;
- **Quartil:** É um dos três valores que divide o conjunto ordenado de dados em quatro partes iguais. Em Dekking et al. (2005) é definido como o primeiro quartil a 25ª porcentagem empírica dos dados de uma amostra e o terceiro quartil como a 75ª porcentagem empírica dos dados. Já o segundo quartil representa a mediana de uma amostra;
- **Pontos discrepantes:** Também conhecidos como *outliers*, são elementos de uma amostra que estão distantes do restante dos elementos da mesma.

2.5 Considerações finais

Neste capítulo foram apresentados os principais conceitos e tecnologias envolvidos neste trabalho. Os conceitos de fluxo *NetFlow*, estatística descritiva e mineração de dados foram os principais pontos abordados. No capítulo seguinte é descrito o estado da arte do assunto, relacionando diversos trabalhos que atuam na área de detecção de intrusão e mineração de dados.

Capítulo 3 – Trabalhos relacionados

Neste capítulo é realizado um apanhado geral do estado da arte dos trabalhos relacionados com fluxos *NetFlow* e a aplicação da mineração de dados em informações sobre tráfego de redes de computadores. A grande maioria dos trabalhos deste tipo foca o contexto de detecção de intrusão. Os artigos aqui relacionados servem de base para a proposta deste trabalho.

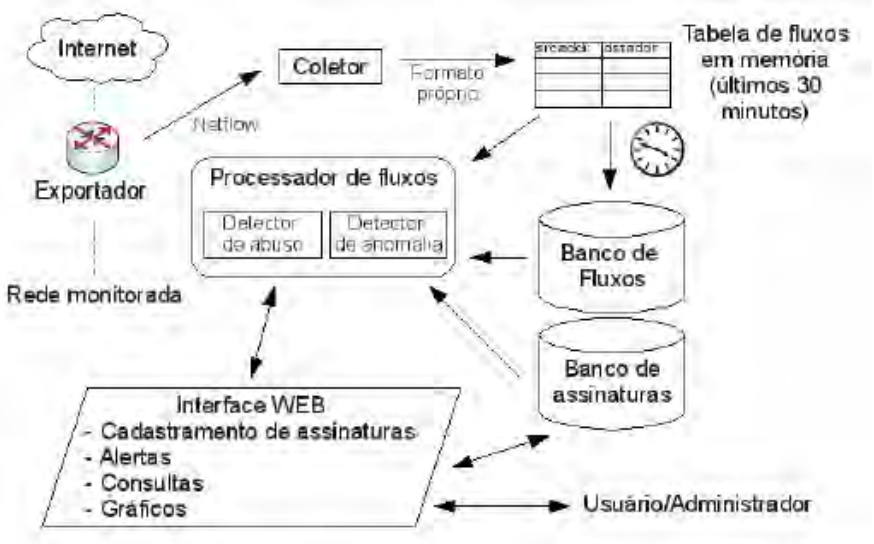
3.1 Fluxo de redes e detecção de intrusão

Diversos trabalhos com fluxos *NetFlow* aplicados à detecção de eventos em redes de computadores são encontrados na comunidade acadêmica. O trabalho de Proto e Cansian (2008) aborda uma nova metodologia para detecção de eventos em redes de computadores utilizando o protocolo *NetFlow* e o armazenamento de informações em Banco de Dados. Por meio de consultas SQL aos dados armazenados é possível identificar alguns eventos na rede, incluindo ataques às redes de computadores. O trabalho apenas propõe uma arquitetura para o armazenamento de informações, deixando a cargo de novos trabalhos a busca por metodologias mais robustas de detecção de intrusão aliadas a tal arquitetura.

Já o trabalho de Zhenqi e Xinyu (2008) propõe um IDS que utiliza *NetFlow* para detectar ataques como DDoS e disseminação de *worms*. Tal trabalho compara o formato de fluxos *NetFlow*, buscando por semelhanças entre os fluxos que representam o ambiente em determinado momento e fluxos qualificados como ataques, além de propor técnicas para a reação a esses ataques (regras de bloqueios em roteadores ou firewalls). Tal trabalho enfrenta dificuldades com falso-positivos na sua detecção.

Já o trabalho de Corrêa et al. (2009) apresenta um modelo de rastreamento de fluxos baseado em assinaturas, sendo o processamento das informações totalmente voltado a fluxos de rede *NetFlow*. A metodologia utiliza um modelo que cria e reconhece assinaturas classificadas em duas vertentes: abuso e anomalia. As assinaturas são baseadas em passos, em que cada evento pode conter um ou mais passos que representem seu comportamento. Utilizando a arquitetura de armazenamento proposta em Proto e Cansian (2008), as assinaturas são cadastradas por um administrador por meio da interface web, sendo utilizadas por um Processador de fluxos que faz o monitoramento do ambiente em tempo real. A arquitetura do sistema proposto está descrita na Figura 4.

Figura 4 – Arquitetura do sistema proposto por Corrêa et al. (2009).



Fonte: Figura retirada de Corrêa et al. (2009).

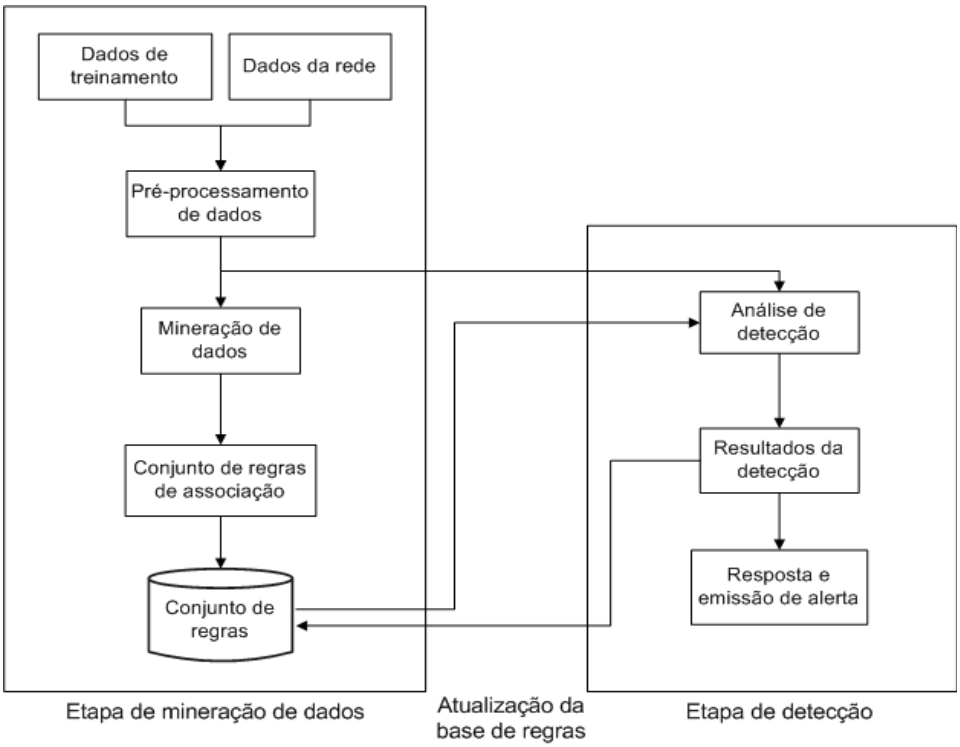
3.2 Mineração de dados e detecção de intrusão

Os trabalhos relacionados à aplicação de mineração de dados para detecção de intrusão são bastante recentes. O trabalho de Thuraisingham et al. (2008) discute várias técnicas de mineração de dados aplicadas à segurança cibernética com sucesso. Tal trabalho aplica essas técnicas em informações relacionadas à detecção de códigos maliciosos pela mineração de executáveis binários, detecção de intrusão em redes e detecção de anomalias. Um dos principais argumentos para a realização do trabalho é o número crescente de casos de ciberterrorismo causados por intrusos de fora e dentro de uma organização atacada, além da frequente ocorrência de fraudes relacionadas a cartões de créditos e ataques a infra-estruturas de um Estado.

Em seu trabalho, o autor comenta que técnicas de mineração de dados para detecção de anomalias auxiliam na identificação de eventos não usuais. Já as técnicas de classificação podem ser usadas para agrupar diversos ataques cibernéticos, utilizando os perfis determinados para detectar um ataque quando este ocorrer. As técnicas de predição podem ser usadas para determinar futuros ataques em potencial, baseadas no aprendizado de conversas de terroristas feitas por meio de e-mails e telefones. Outra aplicação de mineração de dados pode ser feita em dados de auditoria. Apesar de promissor, o trabalho de Thuraisingham et al. (2008) não apresenta resultados concretos sobre a aplicação das ferramentas desenvolvidas.

O trabalho de Song e Ma (2009) apresenta uma arquitetura para detecção de intrusão utilizando mineração de dados. Tal arquitetura possui um conjunto de três módulos principais: módulo de pré-processamento, módulo de análise de associação de dados e um módulo de agrupamento de dados. O modelo do sistema está descrito na Figura 5.

Figura 5 – Modelo de IDS baseado em mineração de dados.

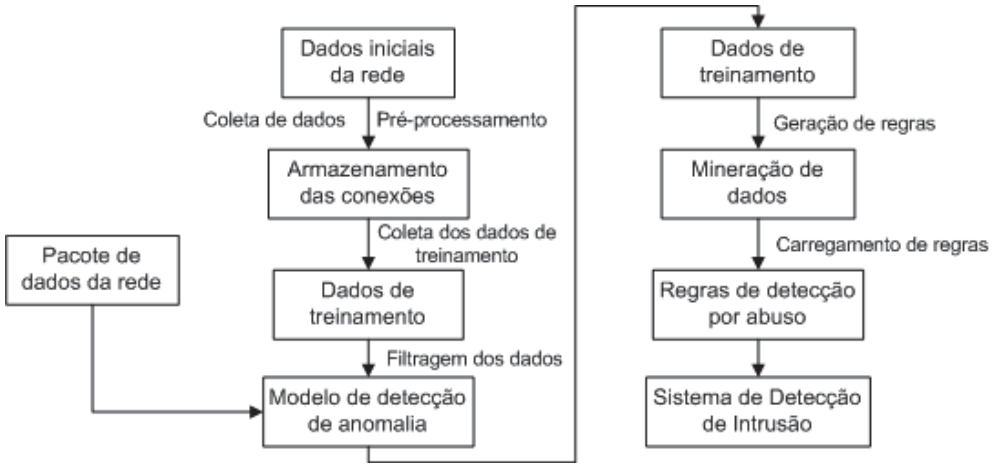


Fonte: Figura adaptada de Song e Ma (2009).

Segundo o autor, o esforço da mineração de dados deve se concentrar em como distinguir efetivamente comportamentos normais e anormais de um conjunto inicial de atributos, além de gerar automaticamente e efetivamente regras de intrusão após coletar dados iniciais de uma rede de computadores. A aplicação de algoritmos de análise de associações e algoritmos de agrupamento em mineração de dados pode auxiliar nesta tarefa. O modelo de

detecção proposto por Song e Ma (2009) executa os seguintes passos: coletar o conjunto de dados que representam o comportamento da rede e usar os algoritmos de associação e agrupamento para determinar o conjunto normal de atividades, obtendo padrões de comportamentos normais para detecção por anomalia; usar os padrões de comportamentos normais para filtrar ainda mais os dados de intrusão; obter dados de intrusão para servir como conjunto de treinamento e; usar o algoritmo de classificação para realizar mineração de regras, auxiliando a distinguir comportamentos normais e de intrusão. Um esquema destes passos é apresentado na Figura 6.

Figura 6 – Processo de mineração do IDS proposto por Song e Ma (2009).



Fonte: Figura adaptada de Song e Ma (2009).

No trabalho de Song e Ma (2009), os dados coletados para análise envolvem basicamente informações de redes sobre conexões TCP. Esses dados formam uma tupla de informações, que inclui:

- *StartTime*: Tempo inicial da conexão;
- *Duration*: Duração da conexão;
- *SrcIP*: Endereço IP de origem;
- *DstIP*: Endereço IP de destino;
- *Service*: Serviço da camada de aplicação;
- *DstBytes*: Número de bytes enviados para a porta de destino;
- *Flag*: Características da conexão, podendo assumir valores como SF, quando ambas as partes enviaram *flags* FIN; REJ, originador da conexão gerou a *flag* SYN e o destino respondeu com a *flag* RST; SO, quando a origem envia um SYN e não recebe respostas; entre outros.

É importante notar que todas as informações citadas pelo autor (SONG; MA, 2009) podem ser obtidas pelo protocolo *NetFlow*. Com isso, a adaptação de tal trabalho para utilizar o padrão IPFIX é bastante promissora.

Os dados coletados por um IDS deverão ser armazenados para análise dos algoritmos de mineração de dados. Para isso, dois módulos principais são utilizados:

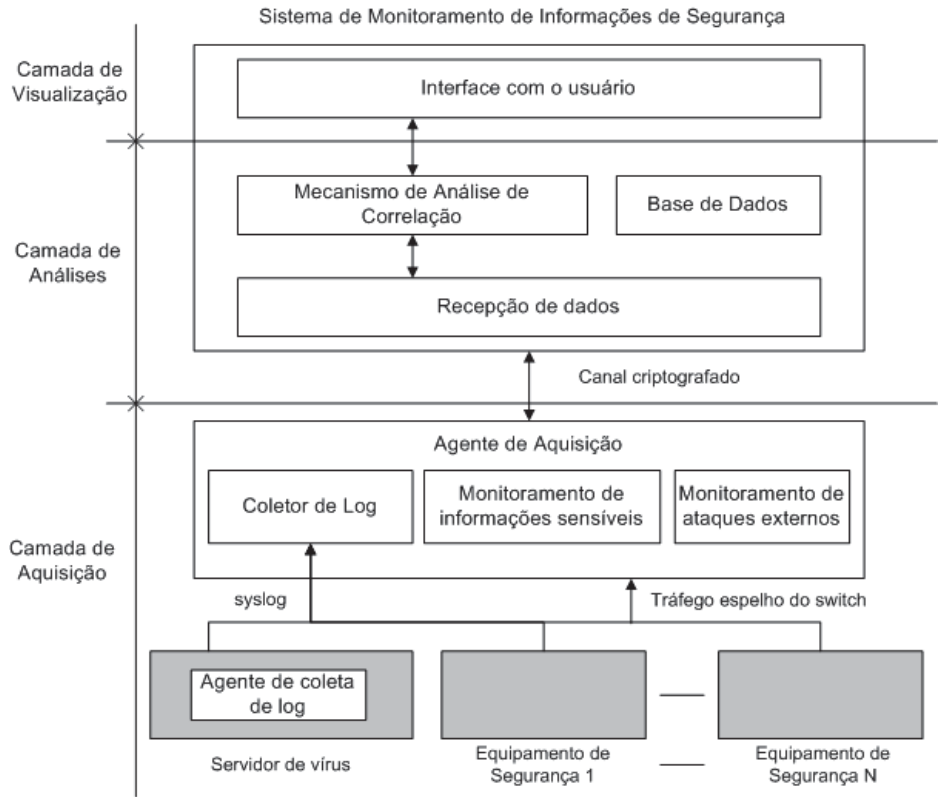
- **Módulo de análise associativa:** utilizado para detecção de intrusão por assinaturas, realiza a associação dos dados com o objetivo de gerar automaticamente novas regras para detecção de tráfego abusivo. Utiliza um algoritmo de associação para desempenhar tal tarefa, e deve ser aplicado em uma base com maior número de tráfego intruso (SONG; MA, 2009);
- **Módulo de análise de agrupamento:** utilizado para detecção de intrusão por anomalia, realiza a identificação de eventos anômalos por meio do treinamento baseado nos dados de tráfegos lícitos. Utiliza o algoritmo *k-means* para agrupamento, sendo aplicado sobre uma base cujos dados de tráfegos normais são predominantes (SONG; MA, 2009).

O trabalho de Guangjuan et al. (2009) propõe uma arquitetura de um sistema de monitoramento de informações de segurança. A motivação baseia-se na grande quantidade de dispositivos e ferramentas existentes para detecção de eventos, que acabam gerando informações descentralizadas dificultando o gerenciamento. O sistema propõe uma centralização dessas informações, possibilitando análises estatísticas e a aplicação de algoritmos de mineração para correlação de eventos. Suas funções básicas são: coletar eventos de vários equipamentos de segurança, como *firewalls*, IDS, IPS, proteção de páginas web e anti-vírus; monitorar a borda da rede em tempo real e fazer análises dos dados; integrar os vários eventos de segurança e correlacioná-los; mostrar relatórios dos eventos detectados. Dadas as funções do sistema, este foi dividido em três camadas, ilustradas na Figura 7.

A camada de aquisição é responsável por obter todos os registros de eventos fornecidos pelos equipamentos de segurança. Os dados coletados são filtrados e pré-processados, devido aos diferentes formatos de informações fornecidas pelos diferentes equipamentos. Os dados são então enviados para a camada de análise, para a centralização e análise dos mesmos. A camada de análise realiza as funções de recebimento de dados, análise em tempo real e análise profunda. A análise em tempo real é utilizada para detectar o posicionamento do evento, realizar a correlação de tempo e montar uma base de conhecimento. Já a análise profunda

desempenha o armazenamento de dados estatísticos, além de formação de relatórios. Por fim, a camada de exibição faz a interface com o usuário dos dados analisados.

Figura 7 – Arquitetura do sistema de monitoramento de segurança.



Fonte: Figura adaptada de Guangjuan et al. (2009).

O foco principal do trabalho de Guangjuan et al. (2009) é o mecanismo de análise de correlação (*Correlation Analysis Engine*). Este mecanismo utiliza técnicas de mineração de dados sobre as informações. Os métodos de mineração de registros utilizados são:

- **Associação:** tem por objetivo buscar relações entre um conjunto de dados, usado para descrever pontos discrepantes de eventos normais ou invasões. Entre os algoritmos usados, são citados os algoritmos *Apriori*, STEM, AIS e DHP;
- **Análise de sequência:** tem por objetivo encontrar relações dentro de uma sequência de tempo para identificar comportamentos anormais de programas ou usuários. A maior parte dos algoritmos desta análise se baseia no *Apriori*;
- **Classificação:** Analisando dados de um conjunto de treinamento, este método faz uma descrição exata para cada categoria, e então utiliza essas regras para classificar novos dados. O método de classificação é um método de aprendizado supervisionado e um

modelo típico de análise utiliza modelos de regressão linear, árvore de decisão ou redes neurais. Este método é principalmente utilizado para detecção por abuso;

- **Agrupamento:** Diferente da classificação, o agrupamento não depende de um conjunto de dados prévios. Os dados de entrada não estão marcados e são classificados como grupos naturais. O método de agrupamento pode ser considerado como um método de treinamento não-supervisionado, constituído de uma variedade de técnicas incluindo agrupamento de sistema, agrupamento dinâmico, agrupamento de *fuzzy*, etc. É mais comumente utilizado na detecção por anomalia.

O trabalho de Peng e Zuo (2006) é o que mais se aproxima da proposta deste projeto. Este propõe a criação de um NIDS que utiliza um algoritmo de associação (mineração de dados) para identificar ataques em redes de computadores. O algoritmo utilizado no trabalho é o *FP-growth* (HAN *et al.*, 2004), um algoritmo otimizado baseado no *Apriori* e que se utiliza de uma estrutura *FP-Tree* (HAN *et al.*, 2004). Os dados de entrada, apesar de obtidos por meio de um capturador de pacotes (*sniffer*) são os mesmos encontrados no protocolo *NetFlow*, o que se identifica ainda mais com este projeto. Um exemplo das associações encontradas na aplicação do algoritmo *FP-growth* está descrito na Figura 8.

Figura 8 – Conjunto de itens frequentes identificados em Peng e Zuo (2006).

TID	Items
10.60.46.58-dIP, 202.198.16.220-sIP	2
10.60.46.58-dIP, 80-sPt	2
TCP-po, 1717-dPt	3
⋮	⋮
10.60.46.58-dIP, 80-sPt, 202.198.16.220-sIP	2
TCP-po, 80-sPt, 1717-dPt	2
TCP-po, 80-sPt, 202.198.16.220-sIP	2
⋮	⋮
TCP-po, 192.168.0.1-sIP, 192.168.0.2-dIP, 1717-dPt	2
TCP-po, 10.60.46.58-dIP, 202.198.16.220-sIP, 80-sPt	2

Fonte: Figura retirada de Peng e Zuo (2006).

Para o treinamento do sistema proposto o autor de Peng e Zuo (2006) utilizou dados de ataques coletados em MIT (2000). Os ataques a serem detectados foram divididos em quatro categorias: DoS, R2L (acesso não autorizado de uma máquina remota), U2R (acesso não autorizado de um usuário local com privilégios administrativos) e varredura de redes. O NIDS proposto analisa em tempo real os dados coletados, comparando com os itens frequentes encontrados no treinamento. Nesta etapa os dados são selecionados em uma janela de tempo

sobreposta que contemple não apenas um pacote, mas um conjunto de pacotes, semelhante ao modelo de janelas deslizantes. É muito importante para o sistema a forma de definir o tamanho desta janela, de modo que ele contemple todos os pacotes de um determinado ataque.

Quanto aos resultados do trabalho de Peng e Zuo (2006), o autor apresentou apenas algumas percentagens de acerto na detecção das classes de eventos proposta. Os acertos ficaram entre 88% e 97%, o que indica um resultado bastante promissor. Também foram indicadas percentagens de falso-positivos, que ficaram em torno de 0.75% e 12.9%. Apesar de o autor não apresentar resultados mais claros sobre o NIDS desenvolvido, tal trabalho foi fundamental para a formação da metodologia proposta neste projeto. Os detalhes do algoritmo *FP-growth* utilizado em Peng e Zuo (2006) são descritos no Capítulo 4.

Um trabalho importante a ser considerado por aplicar algoritmos de mineração de dados sobre *NetFlow* é o trabalho de Caracas et al. (2008). O trabalho em si não possui enfoque em detecção de intrusão, mas os autores utilizam técnicas de mineração sobre *NetFlow* para obter dependências entre entidades e serviços em um contexto comercial e o impacto de tais dependências. Um exemplo mais prático refere-se à manutenção de um servidor: quais serviços serão afetados enquanto tal tarefa for executada? Quais os riscos de mercado quando operações afetam tais serviços?

A principal contribuição de Caracas et al. (2008) é a proposta de um algoritmo que emprega técnicas de monitoramento passivo de rede baseadas no *NetFlow*, cuja vantagem é não necessitar de acesso privilegiado na rede para analisar tais dados. O algoritmo facilita a criação de um modelo de dependência de serviços, com relações abrangendo múltiplos componentes. A principal ideia é identificar eventos correlacionados e criar regras de associação entre pares de fluxos em um modelo de dependência, na forma $f_1 \rightarrow f_2$.

3.3 Considerações finais

Este capítulo descreveu os trabalhos relacionados a fluxos *NetFlow*, mineração de dados e detecção de intrusão. Com base nestes trabalhos, foi desenvolvida uma proposta de uma arquitetura avançada de detecção de eventos utilizando métodos estatísticos e associativos, além do desenvolvimento de um protocolo baseado no *NetFlow*. Esta proposta é discutida no próximo capítulo.

Capítulo 4 – Metodologia e desenvolvimento

Neste capítulo é descrita a proposta e o desenvolvimento das metodologias deste trabalho. A proposta abrange a aplicação de técnicas de mineração de dados em informações provindas pelo padrão IPFIX para a detecção de eventos de rede. O Capítulo é dividido em três seções: na primeira é descrito o novo coletor de fluxos que adapta o *NetFlow* ao padrão bidirecional; na segunda seção é descrita a metodologia estatística desenvolvida para detecção baseada em anomalias no tráfego de uma rede; na terceira seção é descrita a aplicação de uma metodologia de mineração de dados associativa utilizada para a detecção baseada em abuso com a identificação de comportamentos ilícitos dos usuários de uma rede.

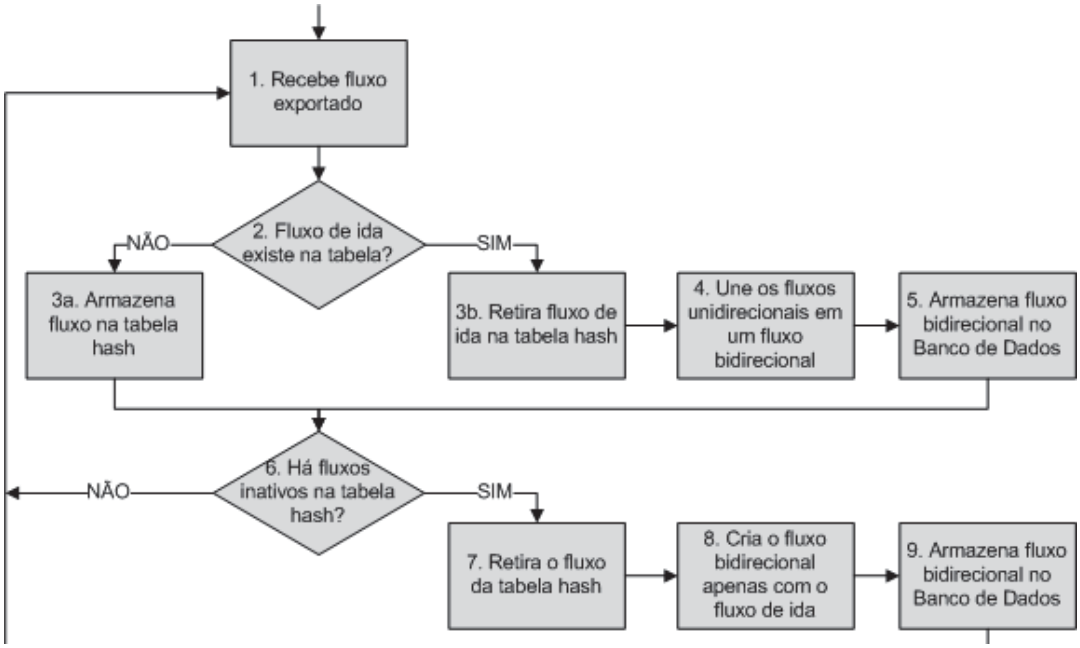
4.1 Coletor de fluxos bidirecionais

A experiência do trabalho de Proto e Cansian (2008) demonstrou que um ambiente de rede de grande porte gera uma quantidade considerável de informações de redes, mesmo quando a tecnologia *NetFlow* é utilizada. O ambiente citado por Proto e Cansian (2008) armazena aproximadamente onze milhões de tuplas por dia na base de dados. O processo de realizar buscas neste montante de dados pode ser algo computacionalmente custoso em termos de tempo e processamento. Um detalhe importante é que na linguagem SQL e nos SGBDs, a busca baseada em restrições acontece de forma vertical, em que cada tupla é percorrida para atender determinadas restrições. Por isso, a utilização do *BiFlow* como alternativa para reduzir a quantidade de tuplas e consequentemente o custo de processamento, se torna útil e viável para este trabalho.

A grande questão de utilizar o *BiFlow* é que ainda não existem aplicações ou dispositivos de rede que exportem fluxos neste padrão. Com isso, a opção mais viável

encontrada neste projeto foi implementar um coletor de fluxos que fosse capaz de receber como dados de entrada fluxos de rede unidirecionais no padrão *NetFlow*, convertendo-os e armazenando-os em um banco de dados no formato do padrão *BiFlow*. No fluxograma da Figura 9 é ilustrado de forma geral o funcionamento do coletor de fluxos.

Figura 9 – Fluxograma de processamento do coletor.



Fonte: próprio autor.

O coletor implementa uma tabela *hash* para armazenar os fluxos unidirecionais que ainda não tiveram um fluxo correspondente identificado. Esta tabela possui como chave de índice o somatório dos campos *srcaddr, dstaddr, srcport, dstport* – IPs de origem e destino, portas de origem e destino – do *NetFlow*, considerados no contexto da Internet como identificadores únicos de uma comunicação. O objetivo é fazer com que o índice de um fluxo de retorno de uma comunicação colida com o índice de seu fluxo de ida correspondente, localizando-os rapidamente.

Quando um fluxo em análise encontra o seu correspondente, é iniciado o processo de criação do fluxo bidirecional. Nesta fase é necessária a determinação de qual fluxo iniciou a comunicação. Como as regras indicadas no RFC do *BiFlow* são aplicáveis apenas no dispositivo exportador, pois exigem a análise dos pacotes de dados de uma comunicação, foram necessárias a criação de novas regras para essa tarefa. As regras são descritas a seguir, sendo que a prioridade segue a ordem em que elas são relatadas:

- **1ª regra:** O fluxo que possuir a porta de destino menor que 1024 e a porta de origem maior que o mesmo valor é considerado o fluxo de origem. Conforme regras definidas pelo IANA, serviços rodando em portas menores que 1024 devem ser executados apenas por usuários com acesso administrativo, sendo então mais comum a execução de aplicações que ofereçam serviços de rede como, por exemplo, o HTTP, SSH, FTP e outros. Estes serviços tendem sempre a receber uma requisição de comunicação;
- **2ª regra:** O fluxo com o campo *first* com valor menos recente é considerado o fluxo de origem;
- **3ª regra:** O fluxo que está na tabela *hash* é escolhido como fluxo de origem.

Pode-se questionar o porquê da 2ª regra não ter maior prioridade do que a 1ª regra. Em um banco de fluxos, é comum encontrar a ocorrência de fluxos que representam a continuidade de uma comunicação antiga, ou seja, a existência de dois ou mais fluxos representando a mesma comunicação. Isso acontece devido às regras de exportação definidas no protocolo *NetFlow*, conforme descritas na Seção 2.2.1. Com isso, se fosse utilizada a 2ª regra como a de maior prioridade, isso induziria à escolha errônea da origem na maioria dos fluxos com essa característica. Todo este processo de escolha da origem e a criação do fluxo bidirecional ocorre na etapa 4 do fluxograma da Figura 9.

Para cada posição na tabela *hash* existe uma lista encadeada de fluxos não identificados, visto que devido à simplicidade da função de índices, existe a possibilidade de colisões entre fluxos distintos ou até mesmo semelhantes mas de momentos diferentes na linha do tempo. Para tratar este problema, cada fluxo na lista é comparado com o fluxo em análise, seguindo os critérios dos quatro campos identificadores de uma comunicação – IPs e portas – além da diferença entre o tempo de criação do fluxo de destino e o tempo do último pacote do fluxo de origem – campos *first* e *last* – ser menor que 15 segundos. Este tempo é determinado de acordo com os padrões de exportação que o protocolo *NetFlow* estabelece.

Uma segunda tarefa a ser feita pelo coletor é a retirada de fluxos inativos na tabela *hash* e posteriormente seu armazenamento como fluxo bidirecional, tendo os campos do fluxo reverso como zerados. Isso é necessário, pois há casos de fluxos de ida que não possuem fluxos de retorno, como no caso de tentativas de conexão mal-sucedidas. Além disso, a retirada destes fluxos na tabela evita que a mesma cresça indefinidamente. O tempo necessário para que um fluxo seja considerado inativo é de 60 segundos, sendo este valor determinado por testes realizados ao longo do desenvolvimento do coletor.

Por fim, após a criação dos fluxos bidirecionais, estes são armazenados em um banco de dados no formato SQL. O formato de um fluxo bidirecional definido neste projeto, bem como a descrição de cada campo está descrito na Tabela 1. O formato das tabelas criadas no banco de dados segue o mesmo utilizado em Proto e Cansian (2008).

Tabela 1 – Campos do fluxo bidirecional armazenado.

Campos	Tipo de campo no BiFlow	Descrição
Srcaddr	Direcional	IP de origem
Dstaddr	Direcional	IP de destino
Input	Direcional	Interface de entrada no exportador
Output	Direcional	Interface de saída no exportador
Src_pkts	Valor	Pacotes enviados pela origem
Dst_pkts	Valor	Pacotes enviados pelo destino
Src_doctets	Valor	Bytes enviados pela origem
Dst_doctets	Valor	Bytes enviados pelo destino
Src_first	Direcional	*Timestamp* de início da origem
Dst_first	Direcional	*Timestamp* de início do destino
Src_last	Direcional	*Timestamp* do último pacote da origem
Dst_last	Direcional	*Timestamp* do último pacote do destino
Srcport	Direcional	Porta de origem
Dstport	Direcional	Porta de destino
Src_Tcpflags	Valor	*Flags* TCP da origem
Dst_Tcpflags	Valor	*Flags* TCP do destino
Prot	Não-direcional	Protocolo da camada de transporte
ToS	Não-direcional	Tipo de serviço

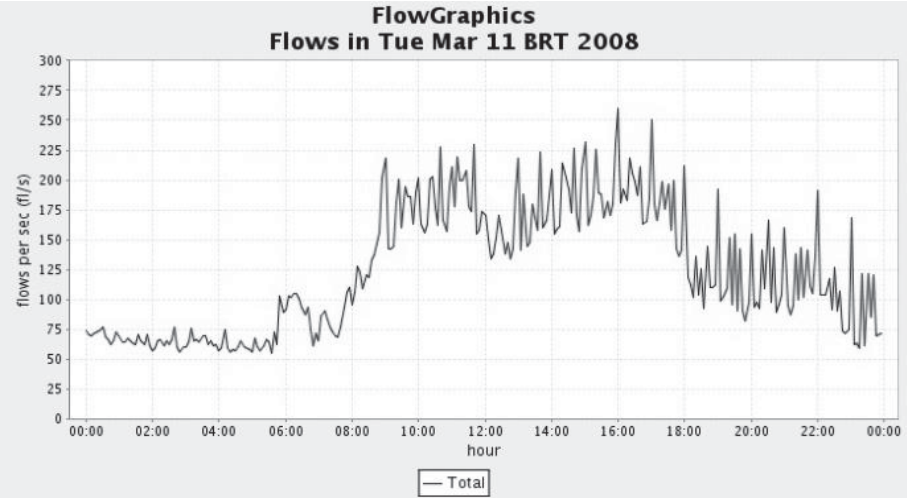
4.2 Modelo estatístico para detecção de anomalias

Neste trabalho é desenvolvida uma metodologia estatística para a determinação do padrão de tráfego dos usuários de uma rede e a posterior detecção de anomalias na mesma. A definição deste padrão é focalizada na utilização de serviços e refere-se a um contexto geral da rede, ou seja, uma análise sumarizada de todo o tráfego de rede gerado pelos seus usuários em determinados serviços. A modelagem do padrão será diferente para cada serviço de rede, como será visto a seguir. Para realização dos objetivos, foi definido um modelo para a definição do padrão de tráfego na rede e um modelo de identificação de pontos discrepantes, sendo descritos a seguir.

4.2.1 Modelo para definição do padrão de tráfego

O modelo para a definição do padrão de tráfego deve se basear no comportamento de cada ambiente em determinados horários do dia, isto porque o tráfego de uma rede no período diurno não possui o mesmo comportamento do período noturno. Na Figura 10 é ilustrado o comportamento do tráfego de uma rede ao longo do dia.

Figura 10 – Exemplo de comportamento do tráfego de uma rede.



Fonte: próprio autor.

Neste trabalho em especial, é necessário não apenas definir o padrão de tráfego da rede como um todo, mas também de cada serviço utilizado. Além disso, o processo de treinamento visa coletar fluxos de um intervalo de dias, buscando um padrão de tráfego. Em cada dia deste intervalo foram selecionadas janelas de tempo de cinco minutos de fluxos; isso significa que a cada cinco minutos de dados, são computadas as médias de fluxos por segundo relacionadas a cada serviço de rede no ambiente. Para isso foi utilizada uma consulta em linguagem SQL (ELMASRI; NAVATHE, 2005) executada sobre os dados no banco de dados relacional. A consulta está descrita na Figura 11.

Figura 11 – Consulta para obtenção do comportamento de serviços.

```
SELECT      timestamp 'epoch' + ((EXTRACT(epoch FROM
            date_trunc('minute',src_first))) - ((EXTRACT(epoch FROM
            date_trunc('minute',src_first)))::int%300)) * INTERVAL '1
            second' AS Time, dstport AS Service, count(*) AS Flows,
            sum(src_dpkts) AS Pkts, sum(src_doctets) AS Bytes
FROM        TableDay
WHERE       dstport <= 1024 && dstaddr << 192.168/16
GROUP BY    (EXTRACT(epoch FROM date_trunc('minute',src_first))) -
            ((EXTRACT(epoch FROM date_trunc('minute',src_first)))
            ::int%300), dstport
ORDER BY    Time, Service;
```

Fonte: próprio autor.

A consulta leva em consideração apenas os dados da origem no fluxo, ou seja, dados do iniciador da comunicação. Na Figura 12 é possível observar os resultados obtidos pela consulta, incluindo as médias de fluxos para serviços que rodam em portas TCP/UDP menores ou iguais a 1024. Este intervalo de portas foi escolhido porque engloba a maioria dos serviços de rede utilizados em um ambiente computacional. Esses dados foram armazenados em uma nova tabela, a fim de serem novamente trabalhados em uma segunda etapa.

Figura 12 – Resultado da consulta SQL.

Time	Service	Flows	Pkts	Bytes
2009-01-01 00:00:00	0	24	484	188311
2009-01-01 00:00:00	21	2	3	144
2009-01-01 00:00:00	22	2	6	304
2009-01-01 00:00:00	25	35	285	103863
2009-01-01 00:00:00	53	633	665	47678
2009-01-01 00:00:00	80	48	688	50636
2009-01-01 00:00:00	123	65	170	12920
2009-01-01 00:00:00	137	75	75	16575
2009-01-01 00:00:00	143	4	22	1364
...	...	...	...	...
2009-01-01 23:55:00	445	10	14	560
2009-01-01 23:55:00	496	95	100	4800
2009-01-01 23:55:00	500	2	2	1352
2009-01-01 23:55:00	769	117	128	9928
2009-01-01 23:55:00	771	72	141	12288

5282 rows in set (3.62 sec)

Fonte: próprio autor.

Com os dados coletados, é possível realizar o cálculo do padrão de tráfego da rede. Este padrão é representado por uma série de amostras contendo as médias de fluxos de cada conjunto de cinco minutos de tráfego, em determinado serviço. Cada amostra representa determinado horário do dia, conforme intervalos de cinco minutos. Na Figura 13 é apresentado um exemplo de amostras de dados coletados referentes a um serviço de rede.

Figura 13 – Exemplo da amostra de dados de um serviço de rede.

Fonte: próprio autor.

Uma questão a ser levantada refere-se a como remover possíveis anomalias no tráfego dentro dos dias coletados para criação do padrão. Isso é resolvido utilizando-se a mesma fórmula de identificação de pontos discrepantes, que é discutida na próxima subseção.

4.2.2 Modelo para a identificação de pontos discrepantes

Segundo Dekking et al. (2005), uma amostra de dados pode ser dividida em cinco pontos de sumarização: o mínimo, o máximo, a amostra de mediana, primeiro e terceiro quartis. A distância entre o terceiro e primeiro quartis é chamada de *interquartile range* (IQR) e pode ser vista em (1), no qual Q_3 e Q_1 são o terceiro e primeiro quartis respectivamente. O ponto mínimo de uma amostra é aquele que está a 1.5*IQR de distância de Q_1 (2); o ponto máximo está a 1.5*IQR de distância de Q_3 (3). Qualquer elemento da amostra que estiver fora dos limites de mínimo e máximo é considerado um ponto discrepante (DEKKING *et al.*, 2005; HAN; KAMBER, 2006).

$$IQR = Q_3 - Q_1 \quad (1)$$

$$Min = Q_1 - 1.5\ IQR \quad (2)$$

$$Max = Q_3 + 1.5\ IQR \quad (3)$$

Para este trabalho apenas o ponto máximo de uma amostra é utilizado. Assim, os elementos de uma determinada amostra cuja quantidade de fluxos ultrapasse o valor máximo da mesma são considerados anômalos.

A metodologia citada no parágrafo anterior é usada tanto na identificação de anomalias no tráfego quanto na remoção de elementos discrepantes das amostras coletadas para definição dos padrões. Para tal, um algoritmo de remoção de pontos discrepantes foi desenvolvido e está descrito na Figura 14.

Figura 14 – Algoritmo para remoção de pontos discrepantes.

```
Faça{
      existe_outlier = 0;
Calcule Q1 e Q3 da amostra X;
      Faça MAX = Q3 + 1.5*(Q3-Q1);
      Percorra todos os elementos da amostra X {
            Se elemento > MAX
                  Remova elemento;
                  Faça  existe_outlier = 1;
      }
} Enquanto existe_outlier = 1;
```

Fonte: próprio autor.

O algoritmo apresentado calcula o ponto máximo e remove os pontos discrepantes até que esses não mais existam. Ele é executado para cada amostra coletada de cada serviço da rede. A cada iteração é feita a remoção de pontos discrepantes da amostra; com a nova amostra obtida calcula-se novamente Q_1 e Q_3. Assim, um novo valor de MAX é calculado, sendo a amostra percorrida novamente a fim de remover outros pontos discrepantes. Quando em uma iteração não for encontrado nenhum ponto discrepante o algoritmo é encerrado.

Terminado o algoritmo, o valor de MAX será utilizado como limiar para definir a anomalia de um tráfego. Assim um sistema monitora em tempo real a média de fluxos dos últimos cinco minutos da data atual. Para cada valor coletado, o sistema compara com o valor MAX de determinada amostra relacionada ao serviço e hora/minuto correspondente. Como exemplo, supondo que dados do serviço SSH (porta 22) sejam coletados no intervalo entre 10h00min e 10h05min, a média de fluxos por segundo deste intervalo será comparada com o valor MAX da amostra correspondente ao mesmo intervalo. Caso á média seja superior ao valor MAX, então concluir-se-á que há uma anomalia no tráfego SSH neste intervalo de tempo.

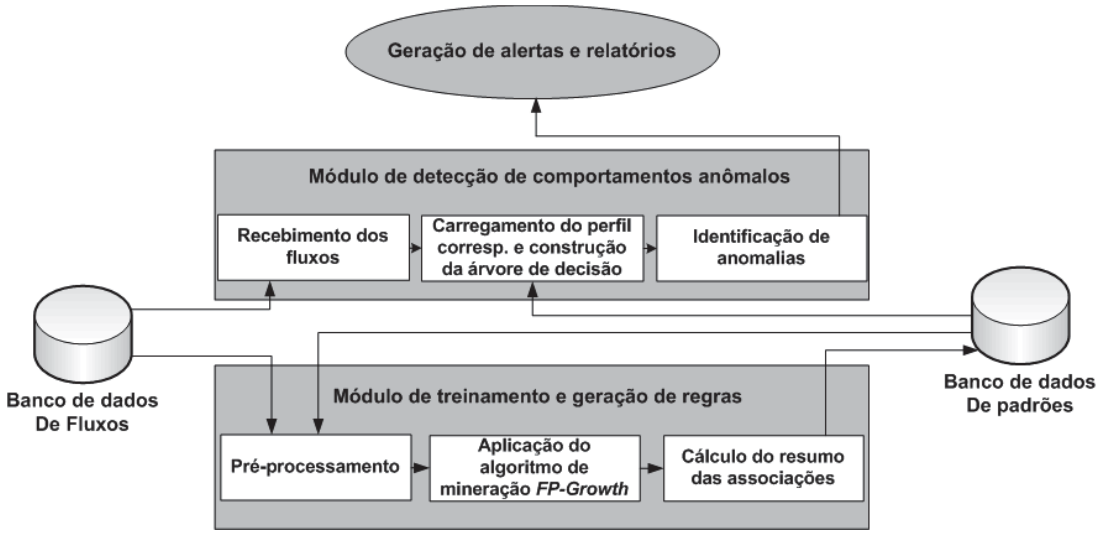
4.3 Metodologia associativa para identificação de perfil de usuários

Os algoritmos de mineração de dados associativos vêm sendo utilizados de forma promissora em pesquisas com dados de redes de computadores. Muito utilizados em banco de dados comerciais, estes algoritmos têm como finalidade identificar associações entre as atividades realizadas por clientes, como a compra de produtos ou a utilização de determinados serviços; com isso, esses algoritmos auxiliam na caracterização de perfis de usuários. Para este trabalho o mesmo conceito é utilizado, ou seja, o objetivo é identificar o perfil dos usuários da rede local e, com os perfis traçados, é possível identificar comportamentos que não coincidem com aqueles definidos.

Todo o processo para a identificação dos perfis é realizado a partir do modelo de treinamento não-supervisionado e utiliza o algoritmo *FP-growth* descrito no trabalho de Peng e Zuo (2006). A aplicação do algoritmo é apenas uma parte do processo, que também inclui o pré-processamento dos dados, a geração de árvores de decisão que representam o perfil do usuário e o acesso aos serviços de rede e o armazenamento dos perfis identificados em um banco de dados relacional. Uma descrição resumida do sistema desenvolvido é ilustrada na Figura 15. Dois módulos foram desenvolvidos, um para a realização do treinamento não-supervisionado e outro para a detecção de comportamentos divergentes aos perfis.

Para entender melhor toda a metodologia desenvolvida, é necessário conceituar o algoritmo *FP-growth* utilizado, além de descrever a estrutura da base de dados desenvolvida para armazenar os perfis dos usuários. Para isso, três subseções são utilizadas: a primeira descreve o funcionamento do algoritmo associativo *FP-growth*; a segunda descreve a metodologia de treinamento desenvolvida; por fim a terceira descreve o módulo para a detecção de comportamentos anômalos aos perfis de usuários.

Figura 15 – Diagrama com os módulos do sistema desenvolvido.



Fonte: próprio autor.

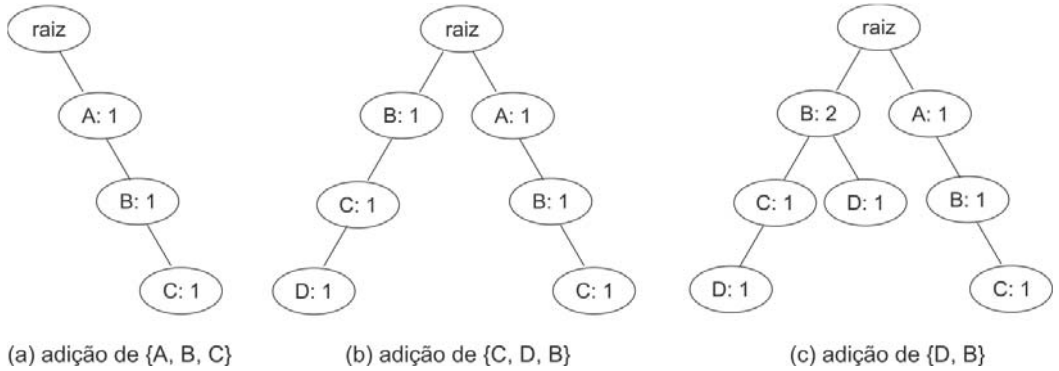
4.3.1 Algoritmo de mineração *FP-growth*

Nesta seção é feita uma breve descrição do algoritmo *FP-growth* (HAN *et al.*, 2004), utilizado neste trabalho como o algoritmo de mineração de dados associativo para a identificação de relações na base de dados de fluxos.

O *FP-growth* é um algoritmo otimizado baseado no algoritmo *Apriori* (HAN; KAMBER, 2006) e utiliza como estrutura fundamental uma árvore de padrões frequentes denominada *FP-Tree*. Esta por sua vez é definida como uma árvore de prefixos estendida que armazena de forma compacta informações de um conjunto de transações, suficientes para a determinação de conjuntos frequentes. Cada nó da árvore armazena um item da base de dados, além de sua frequência de ocorrência nas transações representadas pelo nó. Uma propriedade importante de uma árvore de prefixos é que todos os caminhos a partir do nó raiz obedecem a uma ordenação estabelecida entre os itens. Por exemplo, para um conjunto de quatro itens distintos {A, B, C, D} ordenados alfabeticamente, a árvore de prefixos poderá conter caminhos como ABC ou ACD, mas nunca caminhos como DC ou BDA.

Cada transação de uma base de dados é um caminho da *FP-Tree* desde a raiz. Como exemplo, considerando-se uma base de dados com três transações: {A, B, C}, {D, C, B} e {D, B}. Cada transação deve ser ordenada conforme acordo pré-estabelecido como, por exemplo, a ordem alfabética. Para cada item tem-se um contador global que representa a frequência de cada item na base. Com isso, a árvore é preenchida a partir do nó raiz. Procura-se a existência do primeiro item e, se existente, incrementa-se seu contador; caso contrário inicia-se um novo nó na árvore. Neste exemplo, a primeira transação {A, B, C} é adicionada na árvore, iniciando o contador dos itens (A=1, B=1, C=1) e seus nós, conforme visto na Figura 16(a). Para a segunda transação, a frequência dos itens é atualizada (A=1, B=2, C=2, D=1) e a transação é ordenada. Como o primeiro item da transação (B) não é encontrado imediatamente, um novo caminho é criado (B,C,D), conforme visto na Figura 16(b). Por fim, para a terceira transação as frequências são atualizadas (A=1, B=3, C=2, D=2), a transação é ordenada e os itens são adicionados na árvore, incrementando-se o contador do nó B e adicionando um novo nó D, conforme visto na Figura 16(c).

Figura 16 – Exemplo de construção da árvore de padrões frequentes.



Fonte: próprio autor.

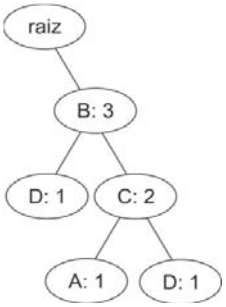
Duas propriedades importantes da árvore de padrões frequentes são:

- É possível reconstruir todas as transações da base de dados a partir de buscas por profundidade na árvore, sempre decrementando o contador dos nós a cada busca;
- A frequência dos nós em um caminho é monotonicamente decrescente, ou seja, a frequência de um nó filho é sempre menor ou igual ao seu ancestral imediato.

Para o algoritmo *FP-growth* o critério de ordenação é diferente do visto pelo exemplo anterior. O algoritmo utiliza como critério de ordenação a frequência dos itens na base de dados. Por isso, o algoritmo possui duas fases: na primeira, a frequência de cada item na base de dados é determinada; na segunda a árvore *FP-Tree* é construída conforme ilustrado

anteriormente. Se o algoritmo fosse utilizado no exemplo anterior, a ordem nas transações utilizadas seria { B, C, A }, { B, C, D } e { B, D }, seguindo a frequência de cada item na base (B=3, C=2, D=2, A=1), o que resultaria na árvore representada pela Figura 17.

Figura 17 – *FP-Tree* gerado pelo algoritmo *FP-growth* do exemplo anterior.



Fonte: próprio autor.

O exemplo a seguir aplica o algoritmo em dados de redes. As transações disponíveis na base de dados estão descritas na Tabela 2. Cada item representa uma informação da comunicação, por exemplo, *dstport-80*. A frequência de cada item é calculada (ver Tabela 3), sendo possível a construção da árvore *FP-Tree* conforme apresentada na **Erro! Fonte de referência não encontrada.**.

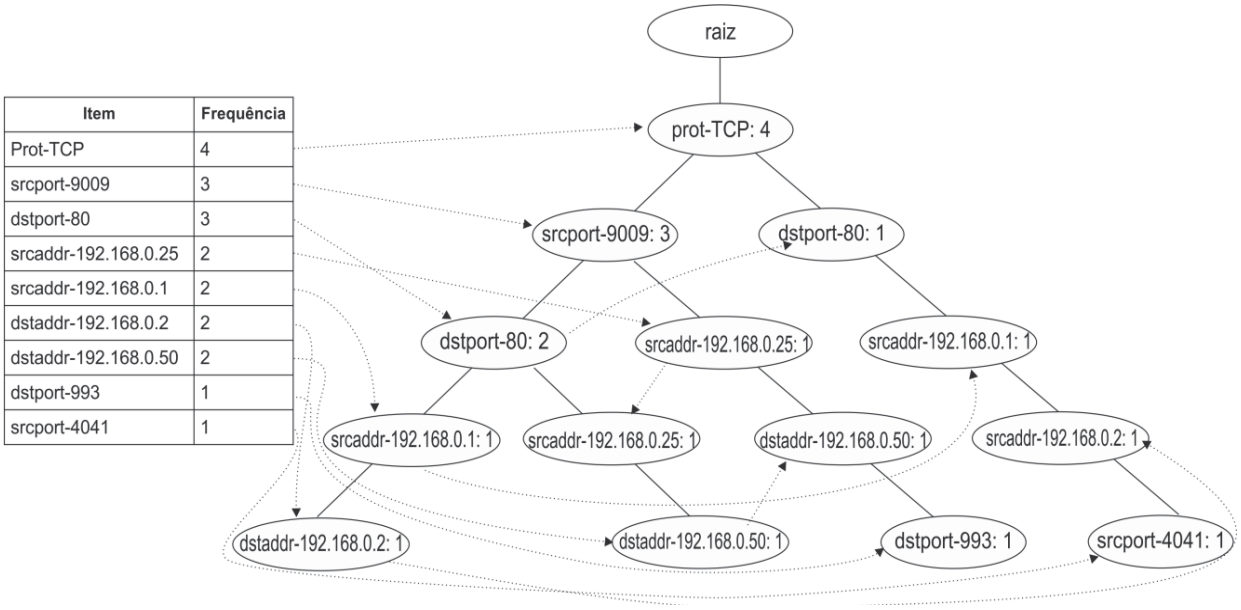
Tabela 2 – Transações na base de dados.

ID	Itens
1	Prot-TCP, srcaddr-192.168.0.1, srcport-9009, dstaddr-192.168.0.2, dstport-80
2	Prot-TCP, srcaddr-192.168.0.25, srcport-9009, dstaddr-192.168.0.50, dstport-993
3	Prot-TCP, srcaddr-192.168.0.1, srcport-4041, dstaddr-192.168.0.2, dstport-80
4	Prot-TCP, srcaddr-192.168.0.25, srcport-9009, dstaddr-192.168.0.50, dstport-80

Tabela 3 – Frequência dos itens (*1-itemset*) nas transações anteriores.

Item	Frequência
Prot-TCP	4
Srcport-9009	3
Dstport-80	3
Srcaddr-192.168.0.25	2
Srcaddr-192.168.0.1	2
Dstaddr-192.168.0.2	2
Dstaddr-192.168.0.50	2
Dstport-993	1
Srcport-4041	1

Figura 18 – *FP-Tree* gerado pelo algoritmo *FP-growth* sobre os dados da Tabela 2.



Fonte: próprio autor.

4.3.2 Metodologia de treinamento utilizando *FP-growth*

A metodologia aqui descrita tem como objetivo treinar os dados e formar regras que definam o perfil dos usuários da rede. Estas regras são transformadas no formato de árvore de decisão, sendo esta usada posteriormente para a detecção de comportamentos atípicos ao perfil identificado. Tal tarefa é realizada pelo módulo de treinamento desenvolvido, conforme apresentado na Figura 15. O treinamento pode ser dividido em três tarefas principais: Pré-processamento dos dados; aplicação do algoritmo *FP-growth*; sumarização das relações obtidas. Cada uma delas é discutida a seguir.

- **Pré-processamento:** Nesta fase, os dados de entrada são transformados de forma que facilitem a identificação de associações. O formato de um dado de entrada é semelhante ao apresentado na Tabela 1. Nesta etapa, alguns campos são sumarizados, entre eles os campos *dstaddr*, número de pacotes da origem e do destino (*dpkts*) e número de bytes da origem e do destino (*doctets*). No primeiro, para cada fluxo utilizado é consultado o *Whois* (DAIGLE, 2004) do IP de destino, obtendo-se o dono do domínio no qual este IP pertence. Como exemplo, o IP 200.145.1.254 é convertido para a *string* “UNESP – Universidade Estadual Paulista”; já o IP 74.125.229.113 é convertido para a *string* “Google Inc”. Com isso é possível correlacionar os fluxos cujo IP de destino sejam de um

mesmo domínio, ou seja, fluxos cujo IP de destino são 200.145.1.1 e 200.145.201.1 têm seus campos de destino alterados para um mesmo nome: “UNESP – Universidade Estadual Paulista”. Já os campos *dpkts* e *doctets* tem seus valores transformados em números de base 10. Como exemplo, um fluxo com os campos *src_dpkts=12* e *dst_doctets=23.890* terá os valores dos mesmos alterados para *src_dpkts=10* e *dst_doctets=10.000*. Estes valores são obtidos com a fórmula (4), sendo X o número de dígitos do campo calculado. Com isso, fluxos que possuam, nestes campos, valores com a mesma base 10 terão seus valores igualados, por exemplo, sejam os fluxos A e B cujo campo *src_dpkts* tenha valor 120 e 245 respectivamente. Neste caso, tal campo de ambos os fluxos terá seu valor alterado para 100.

$$valor = 10^{(X-1)} \quad (4)$$

- **Aplicação do algoritmo *FP-growth***: Uma grande quantidade de fluxos é selecionada, pré-processada conforme descrito anteriormente e aplicada como dados de entrada no algoritmo *FP-growth*. Os dados de entrada são selecionados pela restrição IP de origem/porta de destino, sendo os IPs de origem apenas aqueles pertencentes à rede em análise. Foi utilizada uma aplicação desenvolvida por Borgelt (2010) para a execução desse algoritmo e o resultado é uma série de associações entre os campos dos fluxos e a frequência de ocorrência para cada associação.

- **Sumarização das regras:** O algoritmo resulta em uma série de regras de associações entre os campos dos fluxos, sendo que muitas regras são semelhantes umas às outras, podendo então ser ignoradas. Assim, as associações são filtradas com base nos seguintes critérios:
  - Uma associação só é selecionada quando contém ao menos um campo do tipo direcional/não-direcional e dois campos do tipo valor;
  - Quando existem associações com campos e frequência semelhantes entre elas, a associação com maior número de campos é selecionada.

Em relação à sumarização de regras, uma série de operações lógicas é aplicada para se obter uma maior sumarização das mesmas, a fim de simplificar a criação da árvore de decisão final que será utilizada na fase de detecção. Para isso, os campos dos fluxos são identificados como as sentenças e as operações E e OU são identificadas entre os campos de uma mesma

associação (no caso da E) e entre as associações (no caso da OU). Como exemplo, as associações *(prot=6, dpkts=10, doctets=100)* e *(prot=6, dpkts=100)* são transformadas na operação lógica:

(prot=6 && dpkts=10 && doctets=100) || (prot=6 && dpkts=100)

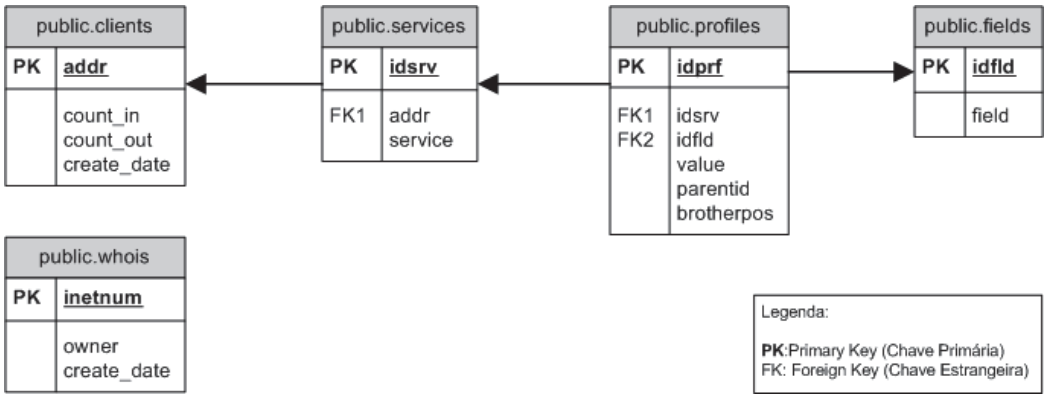
Utilizando propriedades da álgebra booleana, estas operações são simplificadas resultando em:

(prot=6 && ((dpkts=10 && doctets=100) || dpkts=100)

Após a obtenção da regra sumarizada, esta é transformada em uma árvore de decisão que servirá como base para o módulo de detecção identificar comportamentos anômalos. Cada campo do fluxo pertencente à regra obtida será um nó da árvore, sendo a ordem de construção da árvore a mesma utilizada pela estrutura *FP-Tree*, ou seja, os nós ancestrais possuem frequências de ocorrência maiores ou iguais aos de seus filhos. Além disso, a operação OU representa uma nova ramificação na árvore de decisão e a operação E uma relação entre nó pai e filho. A estrutura da árvore é baseada na árvore de múltiplos filhos.

Os dados selecionados para treinamento nesta fase são divididos de acordo com o IP de origem e o serviço acessado, ou seja, a porta de destino. Assim, um conjunto de fluxos cujos elementos possuem a tupla *srcaddr,dstport* de valor idêntico entre eles é selecionado, pré-processado e aplicado ao algoritmo de mineração. As regras obtidas após todas as etapas serem concluídas são armazenadas em uma base de dados de padrões. A estrutura de tabelas da base de dados está representada na Figura 19 e possui um conjunto de relações, facilitando o recarregamento dos perfis identificados na fase de detecção. Além disso, os campos *parentid* e *brotherpos* da tabela *profiles* são fundamentais para a reconstrução da árvore de decisão usada pelo módulo de detecção, identificando o nó superior imediato e sua posição em relação aos outros nós de mesmo nível. Para um maior desempenho na fase de pré-processamento, os *whois* consultados são armazenados em uma tabela no banco de padrões no formato *[endereço de rede/máscara, dono do domínio]*. Assim, quando for necessário obter um novo *whois*, uma consulta é feita primeiramente no Banco de Dados. Se em outro momento houve uma consulta *whois* para o mesmo dono de domínio, este é encontrado rapidamente.

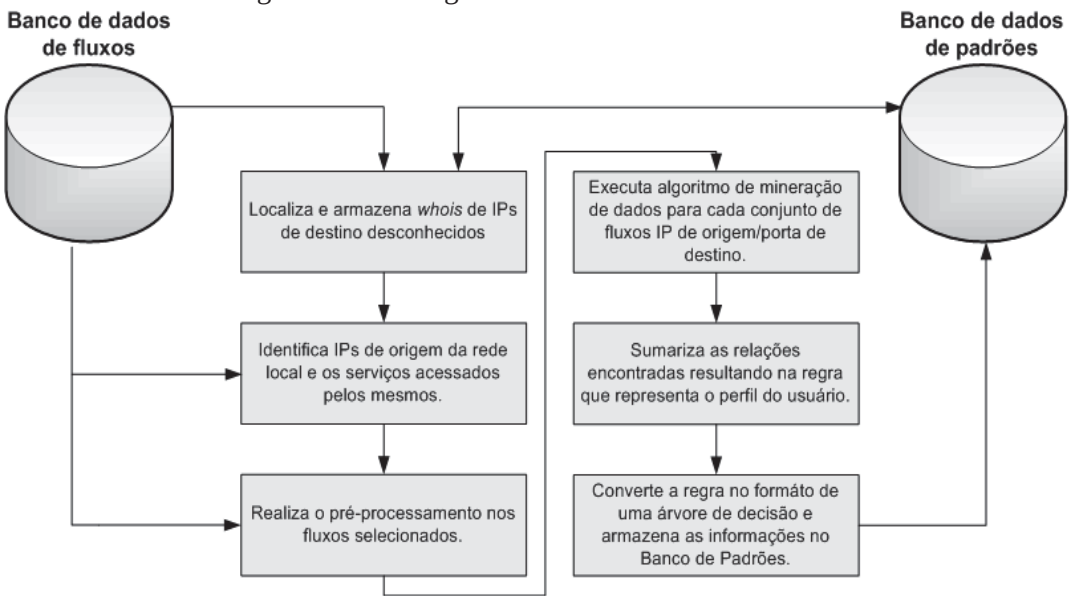
Figura 19 – Diagrama de tabelas do banco de dados de padrões.



Fonte: próprio autor.

O módulo de treinamento para a execução de tal tarefa está descrito na Figura 20, foi desenvolvido para plataforma *Unix* e utiliza alguns aplicativos externos para tarefas específicas, como a consulta ao *whois* e a aplicação do algoritmo de mineração. Em uma primeira etapa o módulo seleciona, por meio de uma tabela que contenha fluxos de treinamento, os IPs locais pertencentes à rede em análise. Em uma segunda etapa, para cada IP são identificados os serviços acessados, ou seja, porta de destino. Assim, será construída uma regra para cada conjunto *[IP de origem / porta de destino]* que defina o perfil do usuário daquele IP naquele serviço em específico. Os dados de entrada são então selecionados e pré-processados, resultando nos dados conforme exemplo apresentado na Tabela 4. Antes disso o módulo de treinamento certifica-se de obter todos os donos de domínio dos IPs de destino por meio do *whois*, utilizando um comando externo presente em sistemas *Unix*. Caso nenhum dono seja encontrado para determinado IP, tal campo não é alterado.

Figura 20 – Fluxograma do módulo de treinamento.



Fonte: próprio autor.

O algoritmo de mineração é então executado sobre os dados com suporte mínimo de 0.1, também por meio de um comando externo desenvolvido por Borgelt (2010). O resultado da execução do algoritmo pode ser visto na Tabela 5. Obtidas as associações, estas são filtradas e transformadas em operações lógicas que por sua vez são sumarizadas, obtendo-se uma operação mais simplificada. Por fim, a árvore de decisão é construída com base nas operações lógicas obtidas no passo anterior, conforme exemplo na

Figura 21.

Tabela 4 – Exemplo de dados pré-processados[4].

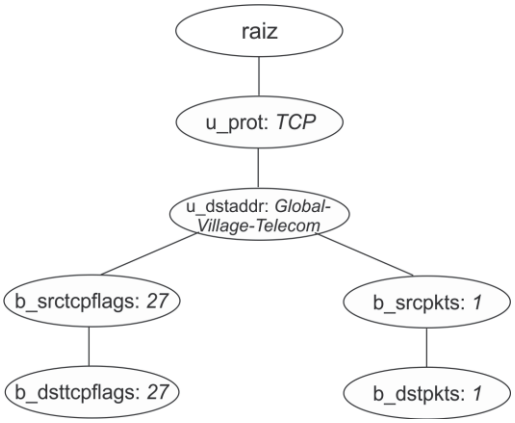
Dados de entrada pré-processados
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1348 b_srcpkts=1 b_dstpkts=1 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=100 b_dstbytes=100
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1347 b_srcpkts=1 b_dstpkts=1 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=100 b_dstbytes=100
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1346 b_srcpkts=1 b_dstpkts=1 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=100 b_dstbytes=100
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1338 b_srcpkts=10 b_dstpkts=10 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=1000 b_dstbytes=10000
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1333 b_srcpkts=10 b_dstpkts=100 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=1000 b_dstbytes=100000
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1356 b_srcpkts=1 b_dstpkts=1 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=1000 b_dstbytes=100
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1355 b_srcpkts=1 b_dstpkts=1 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=100 b_dstbytes=100
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1354 b_srcpkts=1 b_dstpkts=1 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=100 b_dstbytes=100
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1361 b_srcpkts=10 b_dstpkts=10 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=1000 b_dstbytes=1000
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1336 b_srcpkts=10 b_dstpkts=10 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=1000 b_dstbytes=10000
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1335 b_srcpkts=10 b_dstpkts=10 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=1000 b_dstbytes=100000
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1334 b_srcpkts=10 b_dstpkts=10 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=1000 b_dstbytes=10000
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1359 b_srcpkts=1 b_dstpkts=1 b_srctcpflags=26 b_dsttcpflags=26 b_srcbytes=100 b_dstbytes=100
u_prot=6 u_dstaddr=Global-Village-Telecom-[180174] u_srcport=1337 b_srcpkts=10 b_dstpkts=10 b_srctcpflags=27 b_dsttcpflags=27 b_srcbytes=1000 b_dstbytes=10000

[4] As iniciais *u* e *b* indicam campos direcionais/não-direcionais ou de valor, respectivamente.

Tabela 5 – Associações encontradas pelo algoritmo de mineração já filtradas.

Associações encontradas	Suporte
b_dstbytes=100000, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_srctcpflags=27, b_dsttcpflags=27, b_srcbytes=1000, b_srcpkts=10	14,3%
b_dstbytes=10000, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_dstpkts=10, b_srcpkts=10, b_srcbytes=1000, b_dsttcpflags=27, b_srctcpflags=27	28,6%
b_dstpkts=10, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_srcpkts=10, b_srcbytes=1000, b_dsttcpflags=27, b_srctcpflags=27	42,9%
b_srcbytes=100, b_srctcpflags=27, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_srcpkts=1, b_dstpkts=1, b_dstbytes=100, b_dsttcpflags=27	35,7%
b_srcbytes=100, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_srcpkts=1, b_dstpkts=1, b_dstbytes=100	42,9%
b_srcpkts=10, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_srcbytes=1000, b_dsttcpflags=27, b_srctcpflags=27	50,0%
b_dstbytes=100, b_srctcpflags=27, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_srcpkts=1, b_dstpkts=1, b_dsttcpflags=27	42,9%
b_dstbytes=100, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_srcpkts=1, b_dstpkts=1	50,0%
b_dstpkts=1, b_srctcpflags=27, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_srcpkts=1, b_dsttcpflags=27	42,9%
b_dstpkts=1, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_srcpkts=1	50,0%
b_srcpkts=1, b_srctcpflags=27, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_dsttcpflags=27	42,9%
b_srcbytes=1000, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_dsttcpflags=27, b_srctcpflags=27	57,1%
b_dsttcpflags=27, u_prot=6, u_dstaddr=Global-Village-Telecom-[180174], b_srctcpflags=27	92,9%

Figura 21 – Árvore de decisão gerada no exemplo.



Fonte: próprio autor.

4.3.3 Detecção de comportamentos atípicos

Na fase de treinamento foram definidas diversas regras que representam o comportamento dos usuários de uma rede de computadores. Estas regras são armazenadas em

um banco de dados de padrões, para serem utilizadas nesta fase de detecção. Esta fase consiste em três passos básicos: receber o fluxo a ser analisado; carregar o perfil correspondente ao fluxo, analisando o par *[IP de origem / porta de destino]*; determinar se o comportamento é normal ou atípico. No momento de carregar o perfil, o módulo de detecção realiza uma consulta no banco de dados de padrões e carrega a árvore de decisão correspondente na memória. Com a árvore de decisão disponível, cada nó da árvore é percorrido, iniciando-se no nó raiz. Para cada nó, o campo do fluxo e seu valor registrado são comparados ao fluxo em análise: caso sejam idênticos, o percurso prossegue seu caminho para o primeiro filho do nó atual; caso contrário, o irmão do nó atual é selecionado para a próxima comparação. Se o nó atual não possuir irmãos ou se todos já foram analisados, o percurso retorna ao ancestral imediato, que deverá procurar por um próximo irmão, seguindo recursivamente os passos anteriores. O algoritmo recursivo que realiza esta tarefa de percurso na árvore está descrito na Figura 22.

Figura 22 – Algoritmo de percurso da árvore de decisão.

```
Booleano Verifica_perfil(fluxo, node){
      Se node->campo = fluxo->campo & node->valor=fluxo->valor
            Resultado = VERDADEIRO;
      Senão
            Resultado = FALSO;
      Se Resultado = VERDADEIRO {
            Se existe filhos
                  Resultado = Verifica_perfil(fluxo, node->filho);
            Senão
                  Resultado = VERDADEIRO;
      }
      Se Resultado = FALSO {
            Se existe irmãos
                  Resultado = Verifica_Perfil(fluxo, node->irmão);
            Senao
                  Resultado = FALSO;
      }
      Retorna Resultado;
}
```

Fonte: próprio autor.

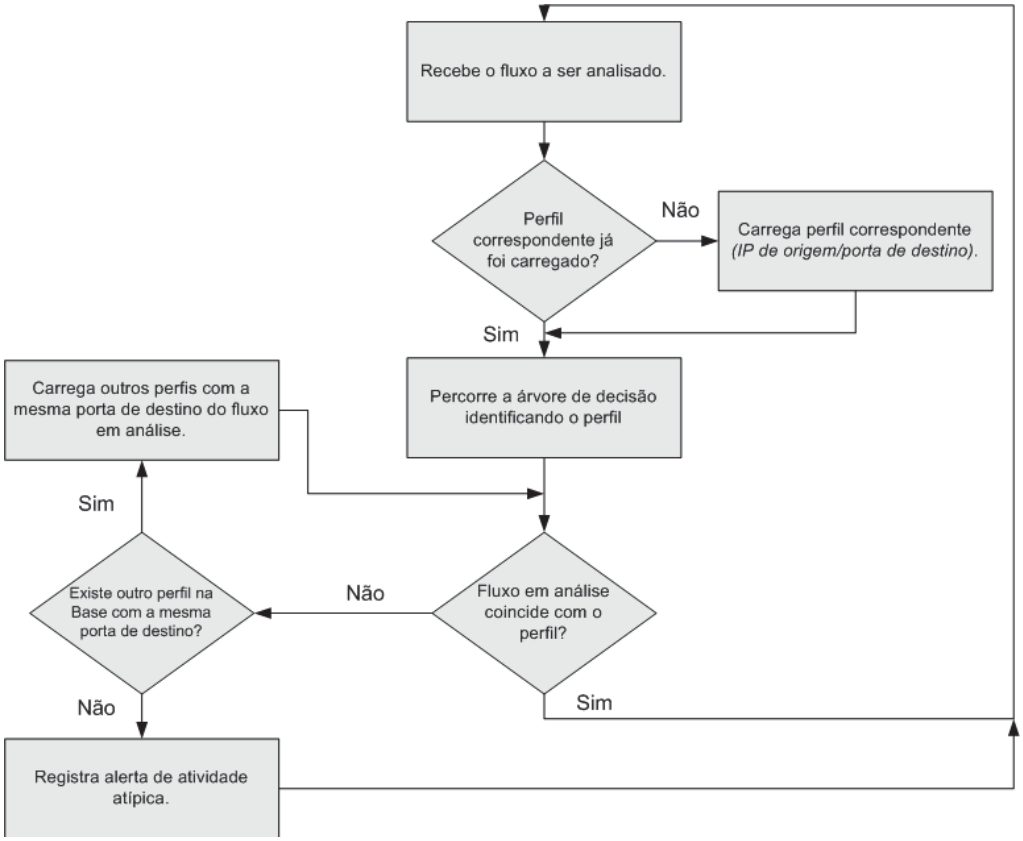
Observa-se no algoritmo que um nó consegue acessar o seu nó irmão diretamente. Isso é possível devido à estrutura da árvore de decisão ser baseada na estrutura de múltiplos filhos, no qual o ancestral de um nó possui um ponteiro apenas para o primeiro filho; já este filho possui um ponteiro para o próximo irmão e assim por diante. Com base no algoritmo, pode-se concluir que:

- **Um fluxo é considerado normal** quando os nós da árvore de decisão são percorridos até um nó folha e os valores comparados forem idênticos em todos os casos;

- **Um fluxo é considerado atípico** quando não foram encontrados percursos até o nó folha de uma árvore de decisão que satisfaçam a identidade dos valores de cada nó do mesmo.

Como descrito em toda a Subseção 4.3.2, a metodologia identifica uma regra para cada conjunto *[IP de origem/porta de destino]* de uma rede. Isso poderia ser entendido como: para cada IP de origem tem-se o comportamento de um único usuário da rede. Porém é sabido que, devido às tecnologias existentes como a utilização de IP dinâmico (DHCP) e o uso de ambientes de trabalho multiusuários, nem sempre o mesmo usuário estará presente no mesmo IP. Para resolver este problema, o módulo de detecção, ao identificar um fluxo atípico por meio do algoritmo apresentado nesta seção, carrega todos os outros perfis existentes na base de dados de padrões que possuam a mesma porta de destino do fluxo em análise, fazendo a mesma análise para cada perfil carregado. Este processo visa identificar comportamentos semelhantes entre usuários com o objetivo de evitar falso-positivos. Assim, se nenhum dos perfis na base de dados satisfizer os requisitos de fluxo normal, o módulo considerará o fluxo em análise como atípico, registrando um alerta de detecção. Na Figura 23 é ilustrado o fluxograma do módulo de detecção.

Figura 23 – Fluxograma do módulo de detecção.



Fonte: próprio autor.

4.4 Considerações finais

Neste capítulo foram descritas todas as metodologias desenvolvidas neste trabalho, identificando os pontos principais, algoritmos e tecnologias utilizados. As metodologias descritas tendem a se complementar em dois fatores: primeiro pelas abordagens distintas de detecção por anomalia (metodologia estatística) e as semelhanças com a detecção por abuso (metodologia associativa); um segundo fator é que a metodologia estatística realiza a análise do tráfego da rede local no sentido externo para interno e a metodologia associativa analisa o tráfego no sentido interno para externo. Neste último fator, nada impede que algumas pequenas adaptações façam as metodologias analisarem o tráfego da rede em sentido inverso; porém para este trabalho a abordagem foi focalizada nos sentidos tal como descritos, no intuito de abordá-los de forma complementar. O próximo capítulo apresentará os resultados obtidos com a aplicação das metodologias desenvolvidas.

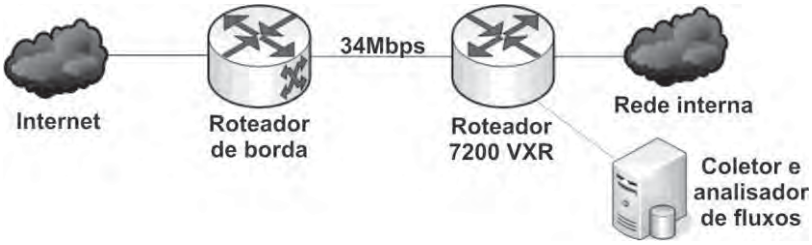
Capítulo 5 – Testes e resultados

Neste capítulo são descritos os testes realizados e os resultados obtidos com as metodologias desenvolvidas neste projeto. Na Seção 5.1 é descrito o ambiente de testes utilizado neste trabalho. Na Seção 5.2 são discutidos os resultados obtidos com o coletor de fluxos bidirecionais desenvolvido. Na Seção 5.3 são descritos os testes e resultados obtidos com a aplicação da metodologia estatística para a detecção de anomalias no tráfego da rede do ambiente. Já na Seção 5.4 são descritos os testes e resultados da aplicação do algoritmo *FP-growth* em fluxos do mesmo ambiente. Por fim, na Seção 5.5 são descritas as considerações finais sobre os resultados encontrados.

5.1 Ambiente de testes

O ambiente no qual as metodologias foram testadas está presente no Instituto de Biociências, Letras e Ciências Exatas da UNESP – Universidade Estadual Paulista. Trata-se de um ambiente real que possui milhares de dispositivos de rede entre roteadores, *switches*, computadores e dispositivos móveis, além de abranger centenas de usuários de serviços da rede espalhados por diversos departamentos. O ambiente possui um roteador CISCO 7200 VXR que exporta fluxos *NetFlow* versão 5. A máquina coletora é um PC x86 Pentium(R) Dual-core E5300 2.60GHz, 2GB RAM e HD IDE 250 GB SATA, dedicado a coleta, armazenamento e análise dos fluxos no banco de dados (ver Figura 24).

Figura 24 – Representação da topologia do ambiente.



Fonte: próprio autor.

O dispositivo exportador de fluxos gera em média onze milhões de fluxos *NetFlow* diariamente. Isso representa a chegada de mais ou menos 128 fluxos por segundo no coletor.

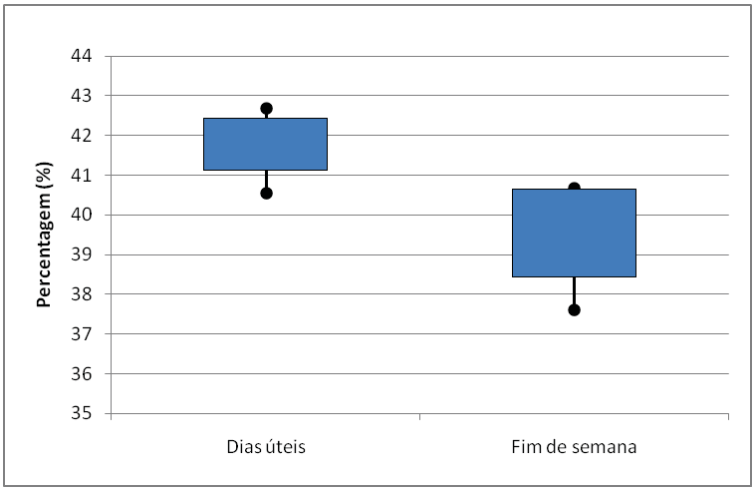
5.2 Resultados com o coletor de fluxos bidirecionais

O coletor de fluxos bidirecionais desenvolvido foi executado na máquina coletora e armazenou os fluxos convertidos em um banco de dados relacional. O SGBD escolhido foi o PostgreSQL (POSTGRESQL, 2011) devido à sua aceitação pela comunidade, além de ser uma aplicação de código aberto com licença GPL e apresentar constante evolução no seu desenvolvimento e no desempenho de suas tarefas. Apesar de outros trabalhos realizados pelo autor terem utilizado o MySQL (MYSQL, 2008) como SGBD, a mudança ocorreu principalmente devido à descontinuidade do desenvolvimento deste último. Além disso, o PostgreSQL possibilita o tipo de dados *inet* – uma representação real do IP – além de diversas funções de manipulação desse tipo, ao contrário do MySQL.

O coletor de fluxos foi executado ininterruptamente por mais de sessenta dias. Selecionando por amostra a quantidade de fluxos de cada dia, constatou-se a redução média de 41% do número de tuplas armazenadas no Banco de Dados se comparado ao armazenamento de fluxos unidirecionais. Além disso, nota-se que essa redução varia de acordo com o dia da semana: para os dias úteis, a redução do número de tuplas armazenadas no banco é de 41,8% em média em relação ao total de fluxos unidirecionais; para os finais de semana, este número médio passa a ser de 39,3%. Esses valores se justificam devido ao número de fluxos de ida sem fluxos de retorno identificados pelo coletor. A média diária de fluxos que não tiveram dados de retorno identificados foi de 32,6% para os dias úteis; já nos finais de semana este número foi de 40,9%. Essa discrepância entre os dias úteis e o fim de semana indica que a proporção de ataques em relação a comunicações lícitas é maior no fim de semana, já que neste período o uso da rede por usuários é consideravelmente menor. Na Figura 25 é possível ver o gráfico de extremidade que representa a percentagem de redução de

tuplas na base de dados, no qual é comparado o armazenamento dos fluxos bidirecionais em relação ao total de fluxos unidirecionais gerados. O gráfico está dividido entre os períodos de dias úteis e finais de semana, sendo que o eixo Y representa a percentagem de redução do número de tuplas no banco. As extremidades dos itens no gráfico representam os valores mínimo e máximo; já o ínicio e o fim dos retângulos representam o 1º e o 3º quartil da amostra analisada. Neste gráfico é possível observar visivelmente a diferença da redução do número de tuplas no banco nos períodos citados.

Figura 25 – Redução de tuplas na base com o armazenamento de fluxos bidirecionais.



Fonte: próprio autor.

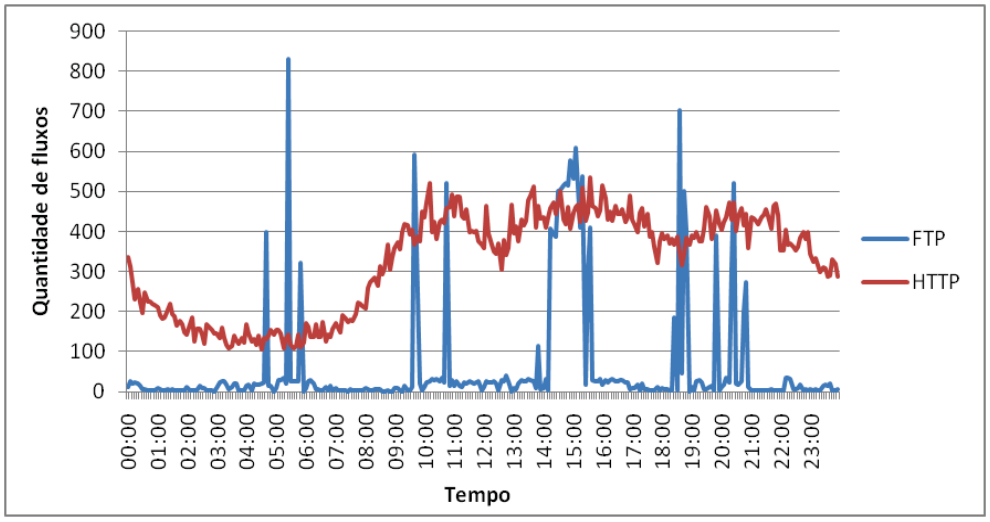
5.3 Resultados da metodologia estatística

O primeiro passo para os testes com a metodologia estatística foi o treinamento para a definição do padrão de tráfego do ambiente. Neste caso, todas as sub-redes do ambiente de teste foram utilizadas. Foram coletadas amostras em um período de três meses de fluxos, totalizando 92 dias.

5.3.1 Serviços monitorados

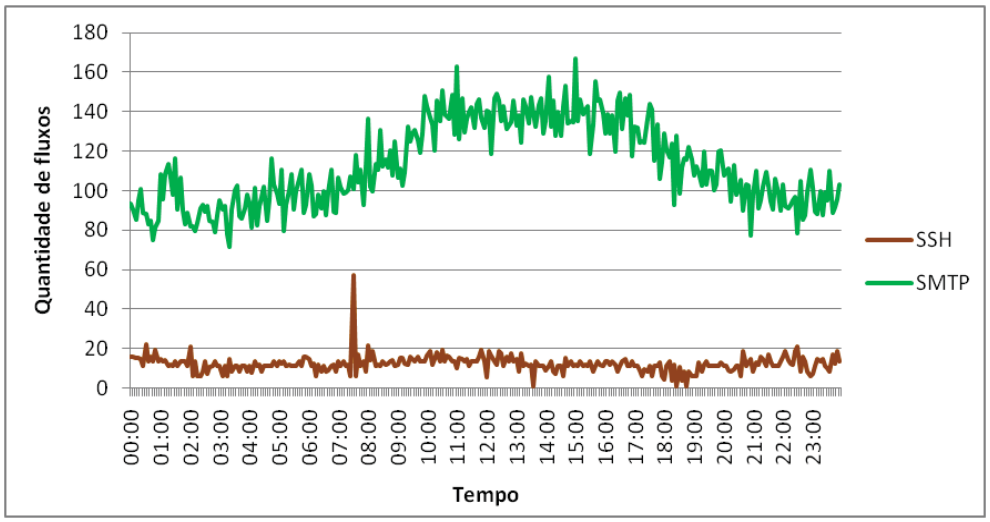
Para este trabalho, quatro serviços em particular foram monitorados pelo sistema proposto: FTP, um protocolo de transferência de arquivos; SSH, um protocolo de acesso remoto; SMTP, um protocolo para serviço de e-mails e; HTTP, um protocolo de serviços web. Os pontos máximos calculados na fase de definição do padrão de tráfego são mostrados na Figura 26 (FTP e HTTP) e Figura 27 (SMTP e SSH). Estes gráficos já mostram a representação dos limitantes de tráfego dos serviços, excluídos os pontos discrepantes das amostras.

Figura 26 – Limitantes superiores dos serviços FTP e HTTP por tempo.



Fonte: próprio autor.

Figura 27 – Limitantes superiores dos serviços SSH e SMTP por tempo.



Fonte: próprio autor.

Na Figura 26 nota-se no serviço HTTP uma maior quantidade de tráfego em horários diurnos (6h às 18h) bem diferente do apresentado em horários noturnos (18h às 6h), diferença essa explicada pela quantidade de usuários em atividade na rede nos períodos citados. Porém o que mais chama a atenção é a representação do tráfego FTP, que possui pontos no tempo cujo valor é excessivamente alto se comparado a outros valores. Isso ocorre, pois em determinados pontos da amostra coletada a quantidade de ataques de dicionário ao serviço FTP foi tão frequente quanto comportamentos normais de usuários, influenciando no processo de remoção de pontos discrepantes. Já na Figura 27 é possível notar uma maior normalidade nos limitantes, sendo que o SMTP apresenta característica parecida ao comportamento do

serviço HTTP; já o SSH possui apenas um dos limitantes com valor excessivo se comparado aos outros, devido aos mesmos motivos apresentados pelo FTP.

5.3.2 Resultados de detecção

O sistema de detecção monitorou o ambiente proposto por um período de seis dias, o que equivale a analisar 1728 períodos de cinco minutos cada. Na Tabela 6 é apresentado o número de eventos detectados nos serviços propostos e a quantidade de falso-positivos referentes ao mesmo. Para o cálculo de falso-positivos foi realizada a análise de *logs* nos servidores sob ataque ou a utilização de técnicas descritas em (CORRÊA; PROTO; CANSIAN, 2008) para a comprovação ou não de cada evento detectado.

Tabela 6 – Eventos anômalos detectados.

Serviço	Eventos detectados	Falso-positivos	Percentagem de acertos
FTP	10	0	100,00%
SSH	615	2	99,67%
SMTP	153	4	97,38%
HTTP	33	14	57,57%

Para o serviço FTP dez eventos foram detectados e, dentre estes, nenhum falso-positivo foi encontrado. Todos estes ataques se caracterizam como varreduras do serviço, no qual o atacante deseja saber se em uma determinada rede possui um servidor FTP sendo executado.

Já na detecção de eventos no serviço SSH observa-se um grande número de eventos detectados, sendo apenas dois falso-positivos encontrados. Os ataques em geral se caracterizam por “ataques de dicionário” ou varreduras no serviço. O grande número de eventos pode ser explicado pelo número de atacantes utilizando tal técnica de intrusão, além do fato de que a modalidade “ataque de dicionário” costuma ser executada por várias horas seguidas, às vezes por dias. Curiosamente um dos falso-positivos encontrados refere-se a um dia em específico no qual pesquisadores do instituto utilizaram por diversas vezes o serviço em um horário não convencional – entre 1h e 3h da manhã – para a manutenção de servidores.

Quanto à detecção de eventos no serviço SMTP, 153 eventos foram detectados, sendo apenas quatro eventos falso-positivos. A modalidade de ataques neste serviço pode ser dividida entre varreduras e envio de SPAMs, sendo que para identificar SPAMs foram utilizadas técnicas que verificam se o computador que enviou tal mensagem tem permissão para enviar e-mails pelo domínio DNS (MOCKAPETRIS, 1987) ao qual pertence e se o

mesmo possui um serviço SMTP sendo executado a todo momento, características de um provedor de e-mails lícito. Dentre os 149 eventos detectados e comprovados, 5 são varreduras, 139 são SPAMs e 5 são varreduras e SPAMs detectados no mesmo período.

Por fim a detecção de eventos no serviço HTTP foi a que apresentou maior número de falso-positivos. Dos 33 eventos detectados, 14 foram falso-positivos, totalizando aproximadamente 57% de acertos. A modalidade de ataques detectada neste serviço ficou restrita a varreduras apenas. Um dos fatores principais que explica este resultado diz respeito ao uso do serviço *web* pelos usuários. A variação no uso de tal serviço é bastante frequente, tanto nos períodos do dia quanto em determinadas épocas do ano. De fato, a quantidade de tráfego do serviço *web* medido há alguns meses é bem diferente do mês atual, devido à grande popularidade de tal serviço e o crescimento de usuários utilizando o mesmo.

5.3.3 Desempenho do sistema

A fim de aferir o desempenho do sistema, os tempos de treinamento – coleta de amostras para definição do padrão de tráfego – e de monitoramento do ambiente foram computados. O tempo para a coleta das amostras e cálculo do padrão do tráfego é em média de aproximadamente 74 minutos contando-se 92 dias de dados, num total de mais de 274 milhões de fluxos, ou seja, em média aproximadamente 48 segundos para o cálculo de um dia. Vale ressaltar que esta coleta ocorre apenas uma vez, sendo os resultados armazenados no banco de dados. Outro detalhe importante é que para cada dia de fluxos gera-se em média 7.800 tuplas a serem processadas pelo sistema, visto que a consulta SQL (ver Seção 4.2.1) utiliza agrupamentos baseados na porta de destino e em intervalos de cinco minutos. Assim, em 92 dias foram geradas mais de 717.600 tuplas a serem processadas pelo sistema.

Já a consulta responsável por monitorar a quantidade de fluxos em tempo real é executada em aproximadamente 2.57 segundos para analisar aproximadamente 1024 serviços, uma média de 4.150 fluxos lidos por ciclo. Somando-se o tempo necessário para realizar a consulta com o tempo de processamento para identificação de anomalias, a média é de, aproximadamente, 3.72 segundos. Em determinados momentos no qual houve um grande número de requisições no banco de dados devido a outras aplicações utilizando o mesmo, a média do tempo de processamento para identificação de anomalias foi de aproximadamente 25.13 segundos. Esses dados estão descritos na Tabela 7.

Tabela 7 – Tempos de processamento.

Descrição	Tempo médio (segundos)
Coleta de amostras para treinamento (92 dias).	4 680
Consulta SQL para monitoramento do ambiente em tempo real (dados dos últimos 5 minutos).	2,57
Consulta SQL mais processamento necessário para identificação de eventos em tempo real (banco de dados sem sobrecarga).	3,72
Consulta SQL mais processamento necessário para identificação de eventos em tempo real (banco de dados sobrecarregado).	25,13

5.4 Resultados da metodologia de mineração de dados associativa

Ao contrário dos testes na metodologia anterior, o treinamento desta metodologia foi realizado para uma sub-rede do ambiente apenas. Esta sub-rede conta com cerca de cento e cinquenta *hosts*, sendo em média a utilização simultânea da rede feita por setenta *hosts*. A escolha por apenas uma sub-rede foi feita devido ao tempo necessário para o treinamento e o monitoramento do sistema, problema descrito na Subseção 5.4.3.

5.4.1 Fase de treinamento

Nesta fase, mais de 1,379 milhões de fluxos bidirecionais relativos à sub-rede escolhida foram selecionados ao longo de uma semana para a identificação do perfil dos usuários. Como a amostra de dados pode conter tanto fluxos que representem comunicações lícitas quanto tentativas de ataque, os fluxos foram selecionados em dias úteis da semana, em que o número de fluxos lícitos tende a ser maior que o de ataques. Depois de realizado o treinamento, cerca de noventa IPs tiveram seu perfil definido, sendo encontrados 87.436 perfis somando-se todos os serviços utilizados pelos IPs na amostra de dados selecionada, uma média de 971 serviços acessados por IP. Apesar desta média alta, a mediana da amostra é de 227, o que indica que alguns IPs da amostra elevaram o valor médio excessivamente. Usuários destes IPs por sua vez provavelmente usaram aplicações *peer-to-peer* ou varredura de rede, o que justifica as diversas comunicações em portas distintas. Quanto à fase de consulta ao *whois* dos IPs de destino, 152.723 IPs distintos foram encontrados na amostra. Estes geraram na base de dados 100.953 entradas de *whois* distintas, o que representa uma melhoria considerável na correlação entre os fluxos. Todos esses dados estão sumarizados na Tabela 8.

Tabela 8 – Informações sobre os fluxos utilizados no treinamento.

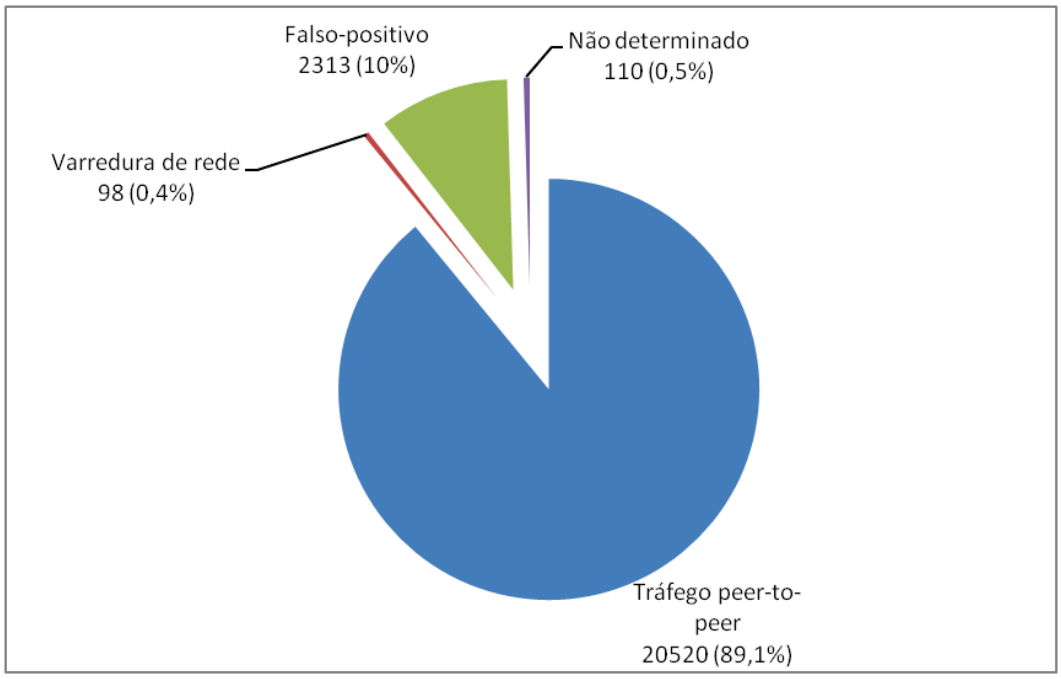
Descrição	Valor
Número de fluxos selecionados para treinamento	1 379 354
Números de IPs distintos com perfis identificados	90
Número regras obtidas (IP de origem/porta de destino)	87 436
Número de IPs de destino distintos dentro da amostra	152 723
Número de whois obtidos nas consultas	100 953

5.4.2 Resultados da detecção

Para a fase de resultados, o módulo de detecção monitorou em tempo real por oito dias os fluxos gerados no ambiente. Neste período, 1.450.159 fluxos foram analisados, sendo 50.259 fluxos identificados como anômalos ao perfil dos usuários da rede. Isso representa aproximadamente 3,5% de fluxos considerados anômalos, indicando que os perfis encontrados na fase de treinamento representaram de forma satisfatória o comportamento dos usuários no ambiente.

Devido à grande quantidade de fluxos anômalos, uma amostra de aproximadamente vinte mil fluxos foi selecionada para classificação. O resultado pode ser visto na Figura 28.

Figura 28 – Classificação dos fluxos identificados como anômalos.



Fonte: próprio autor.

Quatro classes distintas foram encontradas nos fluxos analisados:

- Tráfego *peer-to-peer*;
- Varredura de rede;
- Falso-positivos;
- Não determinado.

A grande maioria dos fluxos da amostra analisada refere-se a tráfego *peer-to-peer* (incluindo *torrent*), com cerca de 89,1%. A característica predominante nesta classe é a grande quantidade de conexões a diversos destinos distintos com portas de destino distintas. As portas utilizadas neste tipo de tráfego na maioria das vezes possuem valores altos, ou seja, acima de 1024, tanto para porta de origem quanto destino, pois são aplicações executadas por um usuário padrão do sistema operacional. O fato de se gerar grande quantidade de conexões explica a grande quantidade de fluxos gerados nesta classe.

Para a classe de varredura de rede, apenas 0,4% dos fluxos analisados foram classificados nesta categoria. Tal classe tem como característica predominante o grande número de conexões em portas de destino distintas realizadas em um curto intervalo de tempo; além disso, o número de bytes transmitidos é inexpressivo e geralmente tem o mesmo valor na maioria dos fluxos desta classe. O fluxo de ida possui apenas a *flag TCP SYN* e a resposta geralmente contém a *flag TCP RST*. Uma questão que surge neste contexto é como explicar a pequena quantidade de fluxos se tal classe possui as características semelhantes ao tráfego *peer-to-peer*. Em verdade os fluxos aqui analisados são respostas a varreduras de rede executadas por um atacante de fora da rede, pois, por algum dos motivos citados na Seção 4.1 o coletor não encontrou os fluxos de ida correspondentes dos mesmos, considerando-os como fluxos unidirecionais. Como a análise desta metodologia baseia-se apenas no comportamento de usuários internos acessando serviços externos, os fluxos originais de ida da varredura executada pelo atacante – externo para interno – não foram analisados pelo módulo de detecção.

Já para a classe de falso-positivos, aproximadamente 10% dos fluxos foram classificados nesta categoria. Dentre os falso-positivos, a grande maioria dos fluxos tem como porta de destino o serviço HTTP. Esses têm como característica predominante o grande número de bytes trafegados na comunicação, o que indica o *download* de grandes arquivos feito pelos usuários. Esta característica não é muito comum nos campos *src_bytes* e *dst_bytes* dos perfis identificados na fase de treinamento, devido à baixa frequência desses tipos de fluxos em tal fase. Outros fluxos considerados falso-positivos referem-se a serviços como

DNS, FTP, POP3 e ICMP, RTSP (*stream* de vídeo) e NTP (sincronização de tempo). Para estes últimos os falso-positivos são explicados na sua maioria também pela baixa quantidade de fluxos de tais protocolos na amostra de treinamento, o que possibilita a não abrangência de todos os comportamentos dos usuários acessando os mesmos. A Tabela 9 apresenta o quantitativo de cada serviço identificado como falso-positivo.

Tabela 9 – Fluxos de serviços identificados como falso-positivos.

Serviço	Qtde. de fluxos	Percentagem
Protocolo HTTP	1 858	80,54%
Protocolo DNS	286	12,40%
Protocolo RTSP	109	4,72%
Protocolo FTP	24	1,04%
Protocolo NTP	13	0,56%
Protocolo HTTPS	11	0,48%
Protocolo POP3	3	0,13%
Protocolo ICMP	3	0,13%

Por fim a classe de não determinados refere-se a fluxos no qual não foi possível identificar um padrão que indique se o fluxo é originado de uma comunicação lícita ou ilícita. Esta classe tem como característica o acesso a serviços de portas não catalogadas pelo IANA, com a quantidade de bytes variada e às vezes sem resposta do destino. As portas dos serviços acessados nesta categoria estão abaixo de 1024, o que indica acesso a serviços executados por administradores do sistema operacional. Para esta categoria seria necessária uma análise mais profunda da comunicação, com a busca de informações adicionais em arquivos de *log* ou aplicações em firewalls que possibilitem a análise do conteúdo dos pacotes originados por tais comunicações.

5.4.3 Desempenho do sistema

Nesta etapa os dois módulos do sistema tiveram o tempo de execução calculado. O módulo de treinamento teve um tempo total de execução de aproximadamente 17h43min. Para um melhor entendimento do tempo de processamento deste módulo, três etapas distintas do treinamento tiveram seus tempos calculados:

- 1ª etapa: Cálculo do *whois* dos IPs de destino dos fluxos de treinamento;

- 2ª etapa: Seleção de IPs dos usuários e os serviços utilizados;
- 3ª etapa: Geração dos perfis dos usuários.

O tempo de processamento de cada etapa pode ser visto na Tabela 10. Observa-se que a etapa que consome maior tempo é a de consulta ao *whois* dos IPs de destino, isso porque o processo depende de chamadas externas ao sistema, além da alta taxa de comunicação de rede com os servidores de *whois*. Em média é possível consultar dois *whois* de IPs por segundo, isso devido às limitações definidas pelos servidores de *whois* para evitar varreduras em suas bases de dados. Por outro lado o uso de um banco de dados de *whois* armazenando as consultas já realizadas reduz o tempo de treinamento drasticamente; assim se os mesmos fluxos fossem utilizados em uma segunda rodada de treinamento, o tempo para consulta ao *whois* seria expressivamente menor, visto que o banco de *whois* já estaria construído.

Tabela 10 – Tempo de processamento das etapas do treinamento.

Etapa	**Segundos**	**Horas (aprox.)**
Obter *whois* dos IPs de destino dos fluxos de treinamento	55095	15h18min
Seleção de IPs dos usuários e os serviços utilizados	15	-
Geração dos perfis dos usuários	8699	2h25min
Total	63809	17h43min

A segunda etapa foi executada em poucos segundos, pois é constituída apenas de algumas consultas SQL no banco. Já para a terceira etapa foram necessárias mais duas horas para concluir a tarefa. Como visto na Tabela 8, mais de 87 mil regras de perfil foram obtidas com o treinamento. Isso representa uma geração média de dez regras de perfil a cada segundo de execução.

Já para o cálculo de desempenho do módulo de detecção, foram determinados três tipos de ocorrências possíveis em sua execução. São eles:

- **1º tipo:** Fluxo classificado pelo perfil principal carregado;
- **2º tipo:** Fluxo classificado por perfis de outros IPs;
- **3º tipo:** Fluxo identificado como anômalo.

No módulo de treinamento, a parte mais custosa em termos de processamento refere-se ao carregamento do perfil correspondente ao fluxo em análise. Por isso, quando é necessário carregar outros perfis para comparação, o custo de processamento é maior, diminuindo o

desempenho do módulo. A média de fluxos por segundo analisados para cada tipo determinado anteriormente pode ser vista na Tabela 11.

Tabela 11 – Tempo de processamento do módulo de detecção.

Tipo	Fluxos analisados por segundo (média)
Fluxo classificado pelo perfil principal carregado	95,17
Fluxo classificado por perfis de outros IPs	14,02
Fluxo identificado como anômalo	11,61

A média de fluxos para o 1º tipo é bastante superior aos outros tipos, mostrando que o desempenho do módulo diminui consideravelmente quando necessita carregar outros perfis para analisar um mesmo fluxo. Com uma média de aproximadamente um fluxo do 2º tipo para cada oito fluxos analisados, a média ponderada geral é de 86 fluxos analisados a cada segundo. Este desempenho é mais do que suficiente para analisar a sub-rede selecionada nos testes deste Capítulo, visto que tal ambiente gera em torno de 23 fluxos por segundo. Porém para uma análise em redes maiores o desempenho do sistema pode não ser satisfatório.

É natural que o desempenho deste módulo seja inferior a outras metodologias existentes. Isso porque a metodologia analisa singularmente cada fluxo gerado no ambiente, diferentemente de outras ferramentas que realizam um resumo dos dados antes de analisá-los, técnica utilizada inclusive pela metodologia estatística descrita na Subseção 4.2. Uma forma de resolver este problema é a implementação do módulo de detecção em uma arquitetura distribuída, ou então a seleção por amostra de fluxos a serem analisados, técnica utilizada por protocolos como o *sFlow* (PHAAL; PANCHEN; MCKEE, 2001).

5.5 Identificação de falso-negativos na detecção

Uma medição importante para qualquer metodologia de detecção de eventos é a identificação de possíveis falso-negativos. Para avaliar esta questão, foi desenvolvido um ambiente de testes que simula oito *hosts* em uma rede local acessando diversos serviços da rede em períodos aleatórios de tempo. Cada *host* foi emulado com a ferramenta *Netkit (BATTISTA et al., 2011)*, sendo que estes acessavam serviços como HTTP, FTP, SSH, NetBIOS e MSN. Para simular da melhor forma possível um ambiente real, o acesso a tais serviços era realizado com maior ou menor frequência de acordo com o período do dia.

Foram geradas duas semanas de tráfego normal para serem utilizadas no treinamento das duas metodologias desenvolvidas, estatística e associativa. Em uma segunda etapa, foram realizados os seguintes ataques a essas redes: força bruta em SSH, varredura de rede, DoS em

HTTP, vulnerabilidade no serviço NetBIOS e vulnerabilidade no serviço HTTP (servidor Apache). Para efeito de comparação, a ferramenta SNORT (ROESCH, 1998) também foi executada e seus resultados foram comparados com as metodologias deste projeto. O resultado de cada detecção está apresentado na Tabela 12, no qual a letra S indica que o evento foi detectado e a letra N que não foi detectado.

Tabela 12 – Detecção de eventos no ambiente de teste.

Evento	Metodologia estatística	Metodologia associativa	SNORT
Força bruta em SSH	S	S	S
Varredura de rede	S	S	S
DoS em HTTP	S	**N**	**N**
Vulnerabilidade em NetBIOS	**N**	S	S
Vulnerabilidade em HTTP	**N**	**N**	**N**

Todas as metodologias identificaram com sucesso os ataques de força bruta em SSH e varredura de rede. Isso porque estes ataques possuem características particulares que facilitam sua detecção: a geração de uma quantidade discrepante de fluxos em relação ao padrão do tráfego de uma rede e o acesso a diversas portas distintas de um *host*.

Já o ataque de DoS em HTTP apenas a metodologia estatística detectou o evento, visto que tal ataque gera um número discrepante de fluxos do ambiente. Na metodologia associativa o serviço HTTP, por ser largamente utilizado, gera uma ampla gama de comportamentos de usuários, no qual o ataque aplicado se assemelhou a um desses comportamentos. Para a ferramenta do SNORT, a falta de uma assinatura que represente tal ataque foi o principal fator para que o mesmo não fosse identificado.

Já no ataque de vulnerabilidade do NetBIOS, apenas a metodologia estatística apresentou falso-negativo, pois este tipo de ataque não gera uma anomalia de fluxos no ambiente. A metodologia associativa, ao contrário, identificou este evento devido às características do ataque, que não se assemelha aos comportamentos dos usuários acessando tal serviço. Já o SNORT possui um conjunto de assinaturas que identificou tal ataque.

Por fim no ataque de vulnerabilidade em HTTP todas as metodologias apresentaram falso-negativos. As razões são as mesmas para os serviços anteriores: na metodologia estatística, este tipo de ataque não gera uma anomalia de fluxos; para a metodologia associativa, a ampla gama de comportamentos de usuários no serviço HTTP possibilita que o ataque se assemelhe a um comportamento normal de um usuário; para a ferramenta SNORT, a falta de uma assinatura que represente o ataque em específico.

5.6 Considerações finais

Neste capítulo foram analisados os testes e resultados obtidos com as metodologias propostas neste projeto. Isso inclui as anomalias identificadas por cada metodologia, além do desempenho de cada sistema implementado. Além disso, um ambiente de testes foi montado para aferir possíveis falso-negativos na detecção realizada pelas metodologias. No próximo capítulo serão descritas as conclusões finais do projeto e as considerações sobre as possíveis evoluções das metodologias propostas em trabalhos futuros.

Capítulo 6 – Conclusões

Este trabalho propôs a aplicação de duas metodologias, estatística e associativa, em fluxos de dados padrão IPFIX para a identificação e correlação de comportamentos e eventos em redes de computadores. As informações disponibilizadas pelo protocolo *NetFlow* formam a base dos dados que representam uma rede em análise, sendo que, a partir deste protocolo foi desenvolvido um coletor capaz de transformar o formato de fluxos unidirecionais em um padrão bidirecional proposto em Trammell e Boschi (2008). A partir de então duas metodologias foram aplicadas: a primeira estatística, na qual foi desenvolvida uma aplicação que define o padrão do tráfego de serviços e usuários de uma rede e que monitora o tráfego de entrada da mesma; a segunda associativa, que modela o perfil dos usuários internos da rede em análise e monitora o tráfego de saída por meio da análise singular dos fluxos gerados pelo ambiente. A primeira pode ser definida como uma metodologia de detecção por anomalia, por definir um padrão comum de tráfego por meio de limitantes; a segunda se assemelha a uma metodologia de detecção por abuso, por criar uma série de regras que define o comportamento de um perfil e por meio delas identificar comportamentos não usuais. Apesar de diferentes as metodologias tendem a se complementar pelas abordagens distintas no qual pertencem.

6.1 Contribuições e restrições

O coletor de fluxos bidirecionais apresentou resultados satisfatórios para o escopo deste trabalho. O maior ganho com a sua implementação foi à diminuição de tuplas armazenadas no banco de dados, aumentando o desempenho das consultas ao mesmo. Por outro lado, as dificuldades em se determinar as correspondências entre os fluxos de ida e de retorno no coletor são muito maiores do que se fossem determinadas no próprio dispositivo exportador,

seguindo as orientações propostas pelo protocolo *BiFlow*. Assim, quando este novo protocolo estiver disponível em novos dispositivos de rede, os resultados serão ainda melhores.

Para a metodologia estatística, o objetivo principal foi modelar o padrão do tráfego de um ou mais serviços de rede por meio de amostras do tráfego baseadas em intervalos de tempo. Dada as amostras, são removidos os pontos discrepantes e, pela mesma técnica, são definidos os valores máximos para separação entre tráfego normal e anômalo. Testes foram realizados com o monitoramento de quatro serviços bastante utilizados por usuários da Internet: FTP, SSH, SMTP e HTTP. Dentre eles, os serviços FTP, SSH e SMTP apresentaram resultados bastante positivos no que diz respeito à detecção de eventos e o pouco número de falso-positivos. Porém o serviço HTTP apresentou o maior número de falso-positivos, explicado pelas características do serviço e pela sua escalabilidade no ambiente, ou seja, o padrão de utilização deste serviço se modifica constantemente conforme o crescimento do número de usuários do ambiente e de seu uso devido à sua popularidade. Apesar dos testes apresentados referirem-se apenas a esses quatro tipos de serviços, este modelo pode ser aplicado a qualquer outro serviço que o usuário desejar. O desempenho do sistema foi satisfatório, principalmente no que se refere ao tempo de resposta no monitoramento do ambiente. Tal medida reflete o baixo custo computacional para a análise do tráfego de uma rede de grande porte. Pode-se notar também que a maior parte do tempo de execução é realizada com tarefas de consulta à base de dados, o que envolve operações de entrada e saída de dados no disco, entre outras.

Já para a metodologia associativa, a aplicação do algoritmo de mineração de dados associativo *FP-growth* em um conjunto de fluxos de uma sub-rede específica possibilitou a obtenção de regras que representem o comportamento de seus usuários. Por meio da utilização do modelo de árvore de decisão, os fluxos do ambiente são analisados e comparados com as regras obtidas e, em caso de não correspondidas, um alerta é emitido. A principal característica deste modelo é a análise individual de cada fluxo gerado pelo ambiente, característica essa que diferencia a metodologia de outros sistemas de detecção baseado em fluxos e propostos em outros trabalhos. Apesar de o monitoramento abranger o tráfego com sentido interno para externo, algumas adaptações na metodologia podem ser realizadas a fim de que o tráfego de sentido externo para interno possa ser analisado. O objetivo passaria a ser então de modelar os perfis dos serviços disponibilizados pela rede em análise e assim detectar qualquer atividade ilícita sobre estes serviços. A detecção de comportamentos atípicos aos perfis foi satisfatória ao apresentar 90% de acertos, considerando o tráfego *peer-to-peer* uma atividade ilícita de acordo com as políticas da rede.

Os testes demonstraram que os usuários têm consumido boa parte dos recursos da rede com aplicações deste porte. O grande problema desta metodologia refere-se ao seu desempenho, principalmente no que diz respeito à fase de detecção. Com um desempenho médio de 86 fluxos analisados por segundo, este módulo necessitaria de algumas adaptações em sua implementação para ser utilizado em redes de grande porte, como a amostragem dos fluxos a serem analisados ou a sua execução em sistemas distribuídos.

6.2 Trabalhos futuros e publicações

Como trabalhos futuros para a metodologia estatística, a coleta de amostras do tráfego para a definição dos padrões deverá ser aperfeiçoada de modo que se atualize ao longo do tempo, tratando os problemas citados no serviço HTTP. Um dos objetivos será atualizar dinamicamente o padrão de comportamento dos serviços, de forma que estes acompanhem a evolução do seu uso pelos usuários da rede em análise. Já para a metodologia associativa, algumas adaptações poderão ser realizadas para que o modelo defina comportamentos de ataques e não mais dos usuários. Assim, os fluxos serão comparados com perfis de ataques definidos, na busca de identificá-los.

Por fim, este trabalho já produziu como resultado o artigo publicado no periódico *LNCS - Transactions on Computational Science XI - Special Issue on Security in Computing* (PROTO *et al.*, 2011). O artigo expõe a metodologia estatística apresentada neste projeto e os resultados obtidos com sua detecção.

Referências Bibliográficas

BATTISTA, G. D. et al. Netkit - The poor man's system to experiment computer networking. 2011. Disponível em: < http://wiki.netkit.org >. Acesso em: 10 Jul. 2011.

BORGELT, C. FPgrowth - Frequent Item Set Mining. 2010. Disponível em: <http://www.borgelt.net/fpgrowth.html>. Acesso em: 01 Ago. 2010.

CARACAS, A. et al. **Mining Semantic Relations using NetFlow**. 3rd IEEE/IFIP International Workshop on Business-driven IT Management. Salvador, Brazil**:** 110 - 111 p. 2008.

CERT. Estatísticas dos incidentes Reportados ao Cert.br. 2009. Disponível em: <http://www.cert.br/stats/incidentes/>. Acesso em: 20 Dez. 2009.

CERT, B. Cartilha de Segurança para a Internet - Parte VIII: Códigos Maliciosos (Malware). 2006a. Disponível em: < http://cartilha.cert.br/download/cartilha-08-malware.pdf >. Acesso em: 09 Out. 2008.

_____. Cartilha de Segurança para Internet - Part I: Conceitos de Segurança. 2006b. Disponível em: < http://cartilha.cert.br/download/cartilha-01-conceitos.pdf >. Acesso em: 08 Out. 2008.

CLAISE, B. RFC 3954: Cisco Systems NetFlow Services Export Version 9. 2004. Disponível em: < http://www.ietf.org/rfc/rfc3954.txt >. Acesso em: 27 nov. 2009.

COOPER, M. et al. **Intrusion Signatures and Analysis**. Editora Pearson, 2001. 448 ISBN 9780735710634.

CORRÊA, J. L. et al. **Detectando eventos em redes utilizando um modelo de rastreamento de fluxos baseado em assinaturas**. IX Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais. Campinas - SP**:** 189 a 202 p. 2009.

CORRÊA, J. L.; PROTO, A.; CANSIAN, A. M. **Modelo de armazenamento de fluxos de rede para análises de tráfego e de segurança**. VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg). Gramado - RS 2008.

DAIGLE, L. RFC 3912: WHOIS Protocol Specification. 2004. Disponível em: <http://tools.ietf.org/html/rfc3912>. Acesso em: 03 Mar. 2011.

DEKKING, F. M. et al. **A modern introduction to probability and statistics**. Springer, 2005. ISBN 1852338962.

ELMASRI, R. E.; NAVATHE, S. **Sistemas de Banco de Dados**. Addison-Wesley, 2005. 744 ISBN 8588639173.

GOLLMANN, D. **Computer Security**. John Wiley & sons ltda., 1999. ISBN 0-471-97844-2.

GUANGJUAN, L. et al. **Information Security Monitoring System based on Data Mining**. Fifth International Conference on Information Assurance and Security. Xi´an, China. 1**:** 472 - 475 p. 2009.

HAN, J.; KAMBER, M. **Data Mining: Concepts and Techniques**. 2ª Edição. Morgan Kaufmann Publishers, 2006. ISBN 9781558609013.

HAN, J. et al. Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. **Data Mining and Knowledge Discovery,** v. 8, n. 1, 2004. Disponível em: <http://dx.doi.org/10.1023/B:DAMI.0000005258.31418.83>.

MIT. DARPA Intrusion Detection Scenario Specific Data Sets. 2000. Disponível em: <http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/index.html>. Acesso em: 20 Out. 2010.

MOCKAPETRIS, P. RFC 1034: Domain Names - Concepts and Facilities. 1987. Disponível em: < ftp://ftp.rfc-editor.org/in-notes/rfc1034.txt >.

MYSQL. MySQL 5.1 Reference Manual. 2008. Disponível em: <http://dev.mysql.com/doc/refman/5.1/en/index.html>. Acesso em: 10 mai. 2008.

MYUNG-SUP, K. et al. A flow-based method for abnormal network traffic detection. **Network Operations and Management Symposium, 2004. NOMS 2004. IEEE/IFIP,** v. 1, p. 599-612, 2004. ISSN 1542-1201.

PENG, T.; ZUO, W. Data Mining for Network Intrusion Detection System in Real Time. **International Journal of Computer Science and Network Security,** v. 6, p. 173-177, 2006. Disponível em: < http://paper.ijcsns.org/07_book/200602/200602C11.pdf >.

PHAAL, P.; PANCHEN, S.; MCKEE, N. RFC 3176: InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks. 2001. Disponível em: <http://tools.ietf.org/html/rfc3176>. Acesso em: 03 Mai. 2011.

POSTGRESQL. The PostgreSQL Global Development Group. 2011. Disponível em: <http://www.postgresql.org/>. Acesso em: 10 Jan. 2011.

PROTO, A. et al. Statistical Model Applied to NetFlow for Network Intrusion Detection. **Transactions on Computational Science XI,** v. 6480, n. Special Issue on Security in Computing, p. 179-191, 2011. Disponível em: < http://dx.doi.org/10.1007/978-3-642-17697-5 >.

PROTO, A.; CANSIAN, A. M. Banco de dados de fluxos para análises de tráfego e de segurança. **UNESP**, São José do Rio Preto, p. 61, 2008.

QUITTEK, J. et al. RFC 3917: Requirements for IP Flow Information Export: IPFIX. 2004. Disponível em: < http://www.ietf.org/rfc/rfc3917.txt >. Acesso em: 27 dez. 2010.

ROESCH, M. Lightweight intrusion detection technology. 1998. Disponível em: <http://www.snort.org/>. Acesso em: 26 dez. 2010.

SONG, C.; MA, K. **Design of Intrusion Detection System Based on Data Mining Algorithm**. International Conference on Signal Processing Systems**:** 370 - 373 p. 2009.

TANENBAUM, A. S. **Redes de Computadores**. 4. 2003. 955 ISBN 8535211853.

THURAISINGHAM, B. et al. **Data Mining for Security Applications**. IEEE/IFIP International Conference on Embedded and Ubiquitous Computing. Shanghai, China. 2**:** 585 - 589 p. 2008.

TRAMMELL, B.; BOSCHI, E. RFC 5103: Bidirectional Flow Export Using IP Flow Information Export (IPFIX). 2008. Disponível em: < http://tools.ietf.org/html/rfc5103 >. Acesso em: 05/04/2011.

ZHENQI, W.; XINYU, W. **NetFlow Based Intrusion Detection System**. International Conference on Multimedia and Information Technology. Phuket, Thailand 2008.