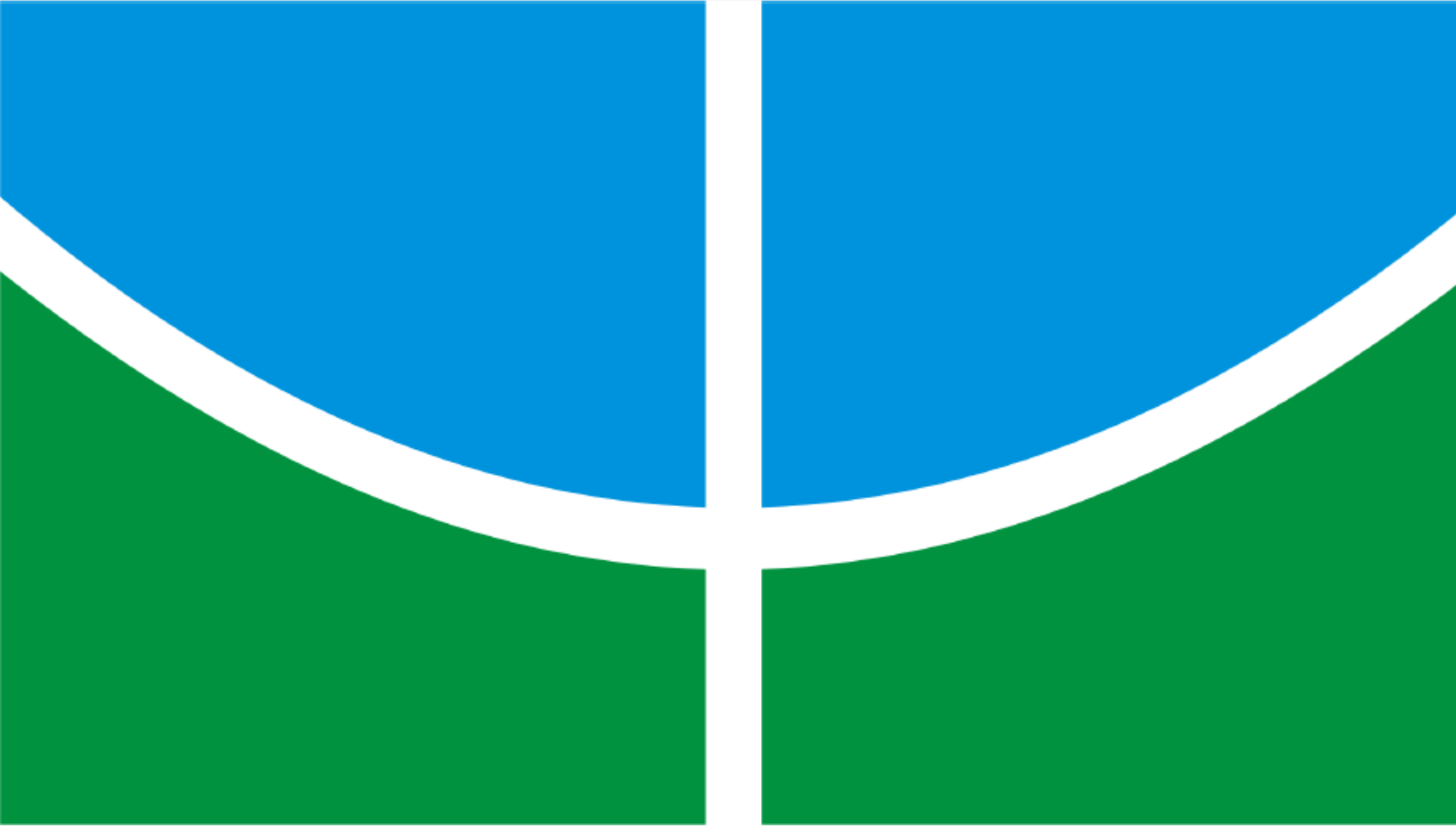


DETECÇÃO CEGA DE TRÁFEGO MALICIOSO ATRAVÉS DA VARIAÇÃO TEMPORAL DO MAIOR AUTOVALOR

DANILO FERNANDES TENÓRIO

DISSERTAÇÃO DE MESTRADO EM ENGENHARIA ELÉTRICA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FACULDADE DE TECNOLOGIA
UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

DETECÇÃO CEGA DE TRÁFEGO MALICIOSO ATRAVÉS DA VARIAÇÃO TEMPORAL DO MAIOR AUTOVALOR

DANILO FERNANDES TENÓRIO

ORIENTADOR: RAFAEL TIMÓTEO DE SOUSA JÚNIOR

COORIENTADOR: JOÃO PAULO C. LUSTOSA DA COSTA

DISSERTAÇÃO DE MESTRADO EM ENGENHARIA ELÉTRICA

PUBLICAÇÃO: PPGEE.DM - 548/2013

BRASÍLIA/DF: DEZEMBRO - 2013

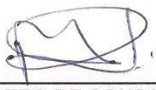
UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

DETECÇÃO CEGA DE TRÁFEGO MALICIOSO ATRAVÉS DA VARIAÇÃO TEMPORAL DO MAIOR AUTOVALOR

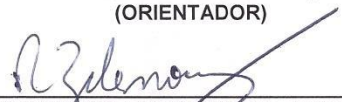
DANILO FERNANDES TENÓRIO

DISSERTAÇÃO DE MESTRADO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA, COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE.

APROVADA POR:



RAFAEL TIMÓTEO DE SOUSA JÚNIOR, Dr., ENE/UNB
(ORIENTADOR)



RICARDO ZELENOVSKY, Dr., ENE/UNB
(EXAMINADOR INTERNO)

EDISON PIGNATON DE FREITAS, Dr., UFSM
(EXAMINADOR EXTERNO)

Brasília, 10 de dezembro de 2013.

FICHA CATALOGRÁFICA

TENÓRIO, DANILO FERNANDES

Detecção Cega de Tráfego Malicioso Através da Variação Temporal do Maior Autovalor [Distrito Federal] 2013.

xviii, 116p., 210 x 297 mm (ENE/FT/UnB, Mestre, Engenharia Elétrica, 2013).
Dissertação de Mestrado - Universidade de Brasília. Faculdade de Tecnologia.

Departamento de Engenharia Elétrica.

1. Detecção Cega 2. Tráfego Malicioso

3. Variação Temporal 4. Maior Autovalor

I. ENE/FT/UnB II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

TENÓRIO, D. F. (2013) Detecção Cega de Tráfego Malicioso Através da Variação Temporal do Maior Autovalor, Publicação PPGEE.DM - 548/2013, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 116p.

CESSÃO DE DIREITOS

AUTOR: Danilo Fernandes Tenório

TÍTULO: Detecção Cega de Tráfego Malicioso Através da Variação Temporal do Maior Autovalor

GRAU: Mestre ANO: 2013

É concedida à Universidade de Brasília permissão para reproduzir cópias desta dissertação de mestrado e para emprestar ou vender tais cópias somente para propósitos acadêmicos e científicos. O autor reserva outros direitos de publicação e nenhuma parte dessa dissertação de mestrado pode ser reproduzida sem autorização por escrito do autor.

Danilo Fernandes Tenório
Universidade de Brasília - Faculdade de Tecnologia
Departamento de Engenharia Elétrica
70910-900 Brasília-DF Brasil

Dedico este trabalho aos meus pais, à minha esposa Brunna e aos leitores desta obra, pelo interesse e confiança.

AGRADECIMENTOS

Agradeço inicialmente a Deus, por ter me acompanhado fielmente durante os estudos e guiado meus passos, fazendo com que eu pudesse tomar as decisões certas nas horas certas. Além disso, posso dizer que por diversas vezes senti sua presença me iluminando, fazendo com que eu tivesse força e sabedoria para não desistir deste grande desafio.

Aos meus pais Paulo (*in memoriam*) e Marluce, pelos exemplos de vida, amor e por terem me dado a educação que é o alicerce para tudo o que eu faço.

À minha esposa Brunna pela motivação, ajuda, amor, paciência e compreensão que precisei para poder desenvolver este trabalho. Sem ela nada disso teria sido feito.

Gostaria de agradecer ao meu ex-chefe, Ten Cel QEM R/1 Luis Gustavo Varges Resende, pois foi ele quem me deu a oportunidade, me incentivou e deu todo apoio necessário para que eu pudesse ingressar e dar início ao mestrado acadêmico na Universidade de Brasília.

Ao Ten Cel QEM Marcio Nascimento Bispo e ao Maj Art Eduardo da Cruz Perez, meus chefes no 7º Centro de Telemática de Área (7º CTA), por terem me permitido continuar esse sonho, pelo apoio, confiança e, sobretudo, amizade que tiveram comigo em todos os momentos que precisei estudar e chegar à conclusão desta obra.

Ao 7º CTA, Organização Militar do Exército Brasileiro, local onde eu trabalho e que me proporciona o desenvolvimento prático, onde posso aplicar os conhecimentos acadêmicos e aprender cada vez mais sobre o ambiente profissional de Tecnologia da Informação.

Ao orientador professor Dr. Rafael Timóteo de Sousa Júnior, pelas aulas, oportunidade, confiança e orientação necessária para o desenvolvimento e conclusão deste trabalho.

Ao coorientador, professor e amigo de longas datas Dr. João Paulo Carvalho Lustosa da Costa, por acreditar em mim, no meu trabalho e ter mostrado os caminhos e direcionamentos necessários para que eu pudesse evoluir como aluno e pesquisador. Gostaria de dizer que sua coorientação, participação e ensinamentos foram decisivos no sucesso alcançado.

Ao professor Dr. Edison Pignaton de Freitas gostaria de agradecer o interesse por este trabalho e todo esforço empregado na revisão desta dissertação. Suas dicas foram fundamentais para a melhoria desta obra.

Aos professores da UnB que colaboraram com a minha formação, em especial ao Prof. Dr. Anderson Clayton A. Nascimento pela atenção e pelas aulas em criptografia e segurança de dados, e ao Prof. Dr. Ricardo Zelenovsky pela disponibilidade e interesse nesta dissertação.

Meus sinceros agradecimentos e muito obrigado a todos vocês.

Disse-lhe Jesus: “Se podes alguma coisa!... Tudo é possível ao que crê”.

Marcos 9:23

RESUMO

DETECÇÃO CEGA DE TRÁFEGO MALICIOSO ATRAVÉS DA VARIAÇÃO TEMPORAL DO MAIOR AUTOVALOR

Autor: Danilo Fernandes Tenório

Orientador: Rafael Timóteo de Sousa Júnior

Coorientador: João Paulo C. Lustosa da Costa

Programa de Pós-graduação em Engenharia Elétrica

Brasília, dezembro de 2013

Atualmente, a vida das pessoas e das empresas está cada vez mais dependente dos meios de comunicação empregados, por exemplo, em *smartphones*, em computadores, em *tablets*. Dessa forma, nada mais claro que as pessoas e as empresas armazenem informações em tais dispositivos, requerendo obviamente segurança das mesmas, seja ela para que esteja sempre disponível quando solicitada, não seja alterada por quem não tenha autorização para isso, ou não seja divulgada publicamente.

Entretanto, esses mesmos meios de comunicação são usados para efetuar ataques contra a segurança da informação. Desse modo, nas redes de comunicação, o tráfego de interesse e utilidade para os usuários mistura-se a um tráfego malicioso voltado a causar problemas de segurança. Assim, uma das mais importantes medidas de proteção para tais redes consiste em detectar tal tráfego malicioso da maneira mais rápida e precisa, de modo a permitir que sejam tomadas decisões quanto à aplicação de contramedidas de segurança.

Esta dissertação propõe uma técnica inovadora de detecção automática de tráfego malicioso por detecção de anomalias na composição do tráfego monitorado em uma rede. Para descrever tal técnica, esta dissertação apresenta todo o embasamento matemático necessário para o melhor entendimento da algorítmica desenvolvida nessa técnica, assim como das ações preliminares, incluindo coleta do tráfego e filtragem de dados, necessárias para se chegar aos resultados desejados.

A técnica de detecção proposta utiliza conceitos matemáticos bem conhecidos e emprega processamento digital de sinais, com o objetivo de detectar ataques de negação de serviço

(*synflood* e *fraggle*) e de escaneamento de portas de comunicação (*portscan*) em redes de computadores.

Para validar a técnica proposta, foram desenvolvidos os correspondentes módulos de *software*, o que permitiu a experimentação com testes de efetividade utilizando amostras de tráfego de redes. Os resultados obtidos são apresentados e discutidos, incluindo os resultados de detecção dos ataques supracitados.

ABSTRACT

GREATEST EIGENVALUE TIME VECTOR APPROACH FOR BLIND DETECTION OF MALICIOUS TRAFFIC

Author: Danilo Fernandes Tenório

Supervisor: Rafael Timóteo de Sousa Júnior

Co-Supervisor: João Paulo C. Lustosa da Costa

Programa de Pós-graduação em Engenharia Elétrica

Brasília, December of 2013

Currently, the life of people and organizations is increasingly dependent on the media applied, for instance, on smartphones, on computers, on tablets. Thus, nothing more clear that people and organizations store information on such devices, obviously requiring its security, that is: it is always available when requested, it is not changed by anyone not authorized to do so, and it is not publicly disclosed.

However, these same media are used to perform attacks against information security. Thus, in communication networks, the traffic of interest and usefulness to users mixture to a malicious traffic aimed to cause security problems. Therefore, one of the most important measures to protect such networks is to detect such malicious traffic more quickly and accurately to allow decisions to be made regarding the implementation of safety countermeasures.

This work proposes an innovative technique for automatic detection of malicious traffic by detecting anomalies in the composition of the monitored traffic on a network. To describe such technique, this dissertation presents all the necessary mathematical foundation for better understanding the algorithmic developed in this technique as well as the preliminary actions, including collecting and filtering data traffic needed to get the desired results.

The proposed detection technique uses well known mathematical concepts and employs digital signal processing in order to detect denial of service (synflood and fraggle) and scan communication ports (portscan) attacks in computer networks.

To validate the proposed technique it was developed the corresponding software modules allowing experimentation with effectiveness tests using samples of network traffic. The

results obtained are presented and discussed, including the detection results of the abovementioned attacks.

SUMÁRIO

LISTA DE TABELAS

LISTA DE FIGURAS

LISTA DE SÍMBOLOS, NOMENCLATURA E ABREVIAÇÕES

SYN	Flag SYN ativa do protocolo TCP
ACK	Flag ACK ativa do protocolo TCP
SYN/ACK	Flags SYN e ACK ativas do protocolo TCP
RST/ACK	Flags RST e ACK ativas do protocolo TCP
log	Informações de registros diversos
OpenVAS	*Open Vulnerability Assessment System*
DoS	*Denial of Service*
DDoS	*Distributed Denial of Service*
PCA	*Principal Component Analysis*
PC	*Principal Component*
MOS	*Model Order Selection*
RAM	*Random Access Memory*
DNS	*Domain Name System*
OSI	*Open Systems Interconnection*
ACL	*Access Control List*
IP	*Internet Protocol*
VPN	*Virtual Private Network*
VoIP	*Voice Over IP*
DMZ	*DeMilitarized Zone*
MZ	*Militarized Zone*
IDS	*Intrusion Detection System*
IPS	*Intrusion Prevention System*
IDPS	*Intrusion Detection and Prevention System*
HIDS	*Host-based Intrusion Detection System*
HIDPS	*Host-based Intrusion Detection and Prevention System*
NIC	*Network Interface Card*
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*
NIDS	*Networkt-based Intrusion Detection System*
LMS	*Log Management System*
SLM	*Security Log Management*

SEM	*Security Event Management*
SIM	*Security Information Management*
SIEM	*Security Information and Event Management*
OSSIM	*Open Systems Security Information Management*
OSSEC	*Open Source Host-based Intrusion Detection System*
OSVDB	*Open Sourced Vulnerability Database*
NFSen	*Netflow Sensor*
NFDump	*Netflow Dump*
SVD	*Single Value Decomposition*
EEF	*Exponentially Embedded Families*
BIC	*Bayesian Information Criterion*
MAP	*Maximum a Posteriori*
KIC	*Kullback Information Criterion*
CME	*Conditional Model Order Estimator*
AIC	*Akaike's Information Theoretic Criterion*
MDL	*Minimum Description Length*
EDC	*Efficient Detection Criterion*
SURE	*Stein's Unbiased Risk Estimator*
EFT	*Exponential Fitting Test*
EVD	*Eigenvalue Decomposition*
LAN	*Local Area Network*
DHCP	*Dynamic Host Configuration Protocol*
OS	*Operating System*
ICMP	*Internet Control Message Protocol*
MAC	*Media Access Control*
VTMA	Variação Temporal do Maior Autovalor

1 – INTRODUÇÃO

1.1 – CONTEXTUALIZAÇÃO

A necessidade de segurança é um fato que vem transcendendo o limite da produtividade e da funcionalidade em sistemas computacionais. Enquanto a velocidade e a eficiência em todos os processos de negócios significam uma vantagem competitiva, a falta de segurança nos meios que habilitam a velocidade e a eficiência pode resultar em grandes prejuízos e falta de novas oportunidades de negócios. A segurança deve ser contínua e evolutiva, pois o arsenal de defesa usado por uma organização pode funcionar contra determinados tipos de ataques; porém, talvez seja falha contra novas técnicas desenvolvidas para driblar esse arsenal de defesa (Geus; Nakamura, 2010).

Nesse contexto, um dos maiores desafios em uma rede de comunicação é a garantia da segurança, relacionada à integridade, disponibilidade e confidencialidade dos dados. Existem várias maneiras de prover segurança, levando em conta tanto aspectos técnicos, através da utilização de equipamentos ou sistemas de segurança, quanto aspectos administrativos e pessoais, relacionados respectivamente ao estabelecimento de políticas de segurança e às campanhas de conscientização, por exemplo. Em relação a equipamentos ou sistemas de segurança pode-se empregar, por exemplo: *firewalls*, sistemas de detecção de intrusão e sistemas de prevenção de intrusão (CERT.br[1], 2010).

Os *firewalls* atuam como a primeira linha de defesa na proteção de servidores e de recursos de rede contra acessos não autorizados e tráfego malicioso. *Firewalls* são tipicamente implantados na borda da rede ou no ponto de entrada de uma rede privada. O tráfego de entrada e saída da *Internet* é inspecionado por *firewalls* de rede. Baseado em um conjunto de regras, eles podem permitir ou bloquear o tráfego de entrada ou saída. Para isso, os *firewalls* de rede trabalham com base em regras que interrogam os pacotes sequencialmente, regra por regra, até que seja encontrada uma correspondência e o mesmo seja descartado ou liberado para prosseguir até o destino (Salah; Elbadawi; Boutaba, 2012).

Sistemas de detecção de intrusão e sistemas de prevenção de intrusão são sistemas de segurança que são utilizados respectivamente para detectar (passivo) e prevenir (pró-ativo) ameaças a sistemas de computadores e redes de computadores. Tais sistemas utilizam diversas formas de funcionamento, tais como: baseado em assinaturas; baseados em ano-

[1]Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil. Disponível em: http://www.cert.br/

malias; ou híbrido, combinando duas ou mais formas (Mudzingwa; Agrawal, 2012).

Em 24 de junho de 2011, a Revista Época edição 684 apresentou a seguinte reportagem de capa: “Os hackers invadem o Brasil”, noticiando sobre ataques a sítios oficiais brasileiros:

> [...] um ramo brasileiro do grupo realizou o maior ataque à *Internet* do país. Os hackers tiraram do ar o Portal Brasil e vários *sites* oficiais: da Presidência, do Senado e dos ministérios do Esporte e da Cultura. Também tentaram derrubar o *site* da Receita Federal e de empresas privadas, mas apenas causaram lentidão. Nos ataques, sobrecarregaram os computadores com vários acessos simultâneos, tirando o serviço do ar.

Tal notícia denota a dimensão que a preocupação com segurança e a busca por soluções em segurança da informação tem tomado, o que motiva o interesse pelo tema alvo desta dissertação: detecção de ataques de negação de serviço, DoS, utilizados por atacantes com o objetivo de tornar serviços de rede indisponíveis; e escaneamento de portas, utilizado para descobrir serviços de rede que estejam em funcionamento e posteriormente encontrar as vulnerabilidades associadas a cada serviço. Algumas das técnicas utilizadas por atacantes para executar ataques de DoS são o *synflood* e o *fraggle*. Ataques de escaneamento de portas são conhecidos como *portscan*. Esses tipos de ataques são objetos de estudo desta dissertação.

1.2 – DEFINIÇÃO DO PROBLEMA

Detectar automaticamente e cegamente tráfego malicioso em qualquer computador conectado a uma rede de dados, utilizando a variação temporal do maior autovalor.

O termo “automaticamente” significa que não se precisa de intervenção humana para avaliar se houve ou não o ataque. O termo “cegamente” refere-se ao fato se não se precisar de informações prévias, tais como assinaturas de ataques ou períodos de aprendizagem, para detectar o ataque.

1.3 – OBJETIVOS

Modelar o tráfego de rede de dados em três componentes: sinal legítimo, sinal malicioso e ruído. Tal modelagem leva em conta o tráfego entrante e sainte em determinadas portas de comunição (TCP ou UDP). Dessa forma, a modelagem preocupa-se apenas com a camada de transporte, fazendo com que o escopo de detecção de ataques restrinja-se a esta camada.

Apresentar conceitos utilizados neste trabalho para detecção de ataques: Análise de Componentes Principais (PCA), Esquemas de Seleção de Ordem do Modelo (MOS), correlação e covariância de dados, autovalores e autovetores.

Validar a solução proposta considerando três tipos de tráfego malicioso: *portscan*, *synflood* e *fraggle*.

1.4 – CONTRIBUIÇÕES

Desenvolvimento de uma nova técnica (VTMA) totalmente inovadora para detecção de ataques (*portscan*, *synflood* e *fraggle*) através do uso de conceitos matemáticos bem conhecidos, como por exemplo PCA e MOS. Legado para diversos trabalhos futuros tanto na área acadêmica quanto na área profissional. Possibilidade de utilização do método VTMA em outras áreas do conhecimento, não se restringindo apenas à engenharia.

1.5 – NOTAÇÃO UTILIZADA

Nesta dissertação, os escalares são denotados por letras em itálico (a, b, A, B, α, β), vetores por letras minúsculas em negrito (**a**, **b**), matrizes por letras maiúsculas em negrito (**A**, **B**), $a_{i,j}$ denota o elemento (i, j) da matriz **A**. Os sobrescritos ^{T} e $^{-1}$ são usados para designar matriz transposta e matriz inversa, respectivamente.

1.6 – TRABALHOS RELACIONADOS

Vários métodos têm sido propostos para a identificação e caracterização de atividades maliciosas. Métodos clássicos normalmente empregam mineração de dados (He; Hu; Yao; Kan; Wang; Xiang, 2008) (Ghourabi; Abbes; Bouhoula, 2010) e análise regular de arquivos (Raynal; Berthier; Biondi; Kaminsky, 2004) para detectar padrões que indicam a presença de ataques específicos no tráfego analisado.

Séries múltiplas de mineração de dados são utilizadas em (He; Hu; Yao; Kan; Wang; Xiang, 2008) para analisar fluxo de dados em uma rede de comunicação com o objetivo de identificar características de tráfego malicioso em ambientes de larga escala. A mineração de dados é frequentemente usada para designar o processo de extrair informações úteis a partir de grandes bases de dados. Com o objetivo de melhorar o desempenho da tarefa de detecção de intrusão, pesquisadores têm aplicado técnicas de mineração de dados em análise de *logs* (Ghourabi; Abbes; Bouhoula, 2010). No entanto, a exigência da coleta prévia de grandes volumes de dados torna-se um ponto fraco do processo.

A utilização de análise regular de arquivos (Raynal; Berthier; Biondi; Kaminsky, 2004) consiste em detectar padrões que indicam a presença de ataques específicos no tráfego analisado, além do estudo estatístico dos dados sobre o tráfego coletado. Uma característica essencial desse método é o fato de que ele depende do conhecimento prévio dos ataques que se destinam a serem identificados, bem como também da coleta de quantidades significativas de *logs* para que o método funcione de forma adequada, reduzindo os falsos positivos.

O uso de PCA pode ser visto empregado na detecção de ataques em (Almotairi; Clark; Mohay; Zimmermann, 2009), porém, utilizando PCA apenas, sem ser combinando com nenhuma outra técnica, como por exemplo o MOS, necessita do caráter subjetivo da intervenção humana, tornando-o impraticável para análises automáticas e propenso a erros, tais como os falsos positivos.

Técnicas cegas de detecção automática de tráfego malicioso têm sido desenvolvidas para *honeypots* (David; da Costa; Nascimento; Holtz; Amaral; Sousa Júnior, 2011) (da Costa; de Freitas; David; Serrano; Amaral; Sousa Júnior, 2012). No entanto, o tráfego em *honeypot* é mais simples, pois não existem aplicações legítimas em execução. Ele emula comportamentos de *host* de produção dentro de uma rede, a fim de ludibriar e atrair os atacantes (Zakaria; Kiah, 2012).

Os dados coletados em sistemas *honeypot*, como captura de tráfego e *logs* do sistema operacional, são analisados a fim de se obter informações sobre técnicas de ataque, tendências gerais de ameaça e *exploits*. Devido ao fato de *honeypots* não gerarem tráfego legítimo, a quantidade de dados capturados é significativamente reduzida em comparação com um IDS de rede que captura e analisa a maior quantidade de tráfego de rede possível (David; da Costa; Nascimento; Holtz; Amaral; Sousa Júnior, 2011).

O uso de esquemas de Seleção de Ordem do Modelo para detecção cega de componentes altamente correlacionados, como, por exemplo, atividades significantes de rede e identificação de atividades maliciosas em *honeypot* – somente através da aplicação de esquemas de seleção ordem do modelo – foi proposto em (David; da Costa; Nascimento; Holtz; Amaral; Sousa Júnior, 2011). Critérios de Seleção de Ordem do Modelo são geralmente avaliados em simulações, comparando a ordem do modelo resultante com a verdadeira ordem do modelo (Dan; Ming; Tao; Rong, 2009).

Esta dissertação relaciona-se com os trabalhos apresentados pelo fato de utilizar uma técnica para detecção de ataques, como todos os citados, porém a proposta apresentada na Seção 1.3 diferencia-se de: (da Costa; de Freitas; David; Serrano; Amaral; Sousa Júnior, 2012), pelo fato de tratar um tráfego mais complexo, composto de sinal legítimo, ruído e ataque; (He; Hu; Yao; Kan; Wang; Xiang, 2008) e (Ghourabi; Abbes; Bouhoula, 2010) pelo fato de não necessitar de uma quantidade significativa de *logs*, a fim de detectar ataques; (Raynal; Berthier; Biondi; Kaminsky, 2004) por não precisar de coleta prévia de dados para se fazer comparações e decidir pela existência ou não de tráfego malicioso; e (Almotairi; Clark; Mohay; Zimmermann, 2009) pois a detecção de ataques é automática, não necessitando de intervenção humana.

Em (R. Puttini; et al, 2006) é possível observar uma técnica alternativa à apresentada nessa dissertação, através de uma explicação geral sobre módulos de um sistema de detecção de intrusão com base em detecção de anomalias de tráfego, e ainda uma discussão sobre falsos positivos e falsos negativos na detecção de intrusão.

Tabela 1.1 - Comparativo entre os trabalhos relacionados e esta dissertação.

Trabalhos relacionados	**Técnicas empregadas**				
	Mineração de dados	**Análise regular de arquivos**	**PCA**	**MOS**	**VTMA**
(He; et al, 2008)	x	-	-	-	-
(Raynal; et al, 2004)	-	x	-	-	-
(Almotairi; et al, 2009)	-	-	x	-	-
(da Costa; et al, 2012)	-	-	x	x	-
(Tenório, 2013)	-	-	x	x	x

A Tabela 1.1 faz um comparativo entre os trabalhos relacionados e esta dissertação (Tenório, 2013), mostrando que o presente trabalho faz o uso de uma nova técnica (VTMA), que é apresentada ao longo desta obra.

1.7 – ESTRUTURA DA DISSERTAÇÃO

No Capítulo 2 é feita uma revisão sobre conceitos e definições referentes à segurança da informação. São apresentados também alguns tipos de ataques, bem como sistemas de segurança: *firewalls* e detectores de intrusão. Além disso, é feita uma breve introdução sobre gerenciamento de evento e informação de segurança. No Capítulo 3 são apresentados

conceitos relacionados a autovalores e autovetores. O Capítulo 4 referencia-se ao tema Análise de Componentes Principais (PCA), onde são citadas suas principais características. O Capítulo 5 descreve o assunto Seleção de Ordem do Modelo (MOS), mostrando os principais esquemas de MOS e suas diferenças. No Capítulo 6 é apresentada a solução proposta, são estudados os ataques de interesse, mostrados detalhes da modelagem, de como os dados foram coletados e por fim como os ataques foram detectados. O Capítulo 7 traz o cenário analisado, os resultados experimentais obtidos, e a comprovação da detecção dos ataques. No Capítulo 8 são feitas as considerações finais e sugeridos trabalhos futuros associados ao tema desta dissertação.

2 – SEGURANÇA DA INFORMAÇÃO

2.1 – INTRODUÇÃO

O motivo principal que faz com que a tecnologia da informação seja tratada com atenção é sua dependência cada vez maior por parte de organizações de toda a natureza. De fato, a tecnologia efetivamente faz parte dos negócios das organizações, principalmente quando se fala em informática e telecomunicações, incluindo a *Internet*.

O mundo moderno e globalizado faz com que as organizações busquem o mais alto nível de competitividade, no qual novos mercados são disputados vorazmente. O concorrente agora pode estar em qualquer parte do mundo e para superá-lo é necessário mais do que nunca fabricar produtos de qualidade, prestar bons serviços e manter um bom relacionamento com os clientes, sejam eles internos ou externos. Nesse cenário, a competitividade global é ditada principalmente pela velocidade, qualidade e eficácia – seja das decisões, das implementações ou das comunicações. Dessa maneira, a infraestrutura de tecnologia da informação e de telecomunicações, que permite a comunicação entre pessoas e recursos deve ser bem projetada e dimensionada de modo a otimizar os investimentos. Mais do que isso, o uso eficiente da tecnologia como meio de evolução dos negócios e de desenvolvimento de novas oportunidades é vital para a sobrevivência de qualquer organização (Geus; Nakamura, 2010).

A sobrevivência corporativa é um tema que combina segurança computacional com o gerenciamento de risco de negócios a fim de proteger serviços de informações distribuídas e ativos. A premissa fundamental é a de que nenhum sistema é totalmente imune a ataques, acidentes ou falhas. Portanto, o foco deste assunto não é apenas de conter atacantes, mas também de garantir que funções críticas sejam sustentadas e serviços essenciais sejam entregues, apesar da presença de ciberataques, falhas e acidentes (Lipson; Fisher, 1999).

Em um cenário em que organizações de qualquer natureza dependem cada vez mais da tecnologia da informação, qualquer falha na segurança da informação pode afetar negativamente os negócios da organização causando impactos significativos. No caso do comércio eletrônico, por exemplo, a indisponibilidade ou problemas de acesso em um sítio faz com que o usuário realize compras no concorrente, pois bastam apenas alguns cliques no *mouse* para a mudança entre diferentes lojas virtuais. Roubos de informações, como cadastro de clientes ou de dados de cartões de crédito, também resultam em grandes impactos (Geus; Nakamura, 2010).

2.2 – CONCEITOS E DEFINIÇÕES

Segurança de sistemas de informação é a proteção contra acessos não autorizados, podendo a informação ser modificada ou não, estando armazenada, em processamento ou em trânsito; e contra a negação de serviço a usuários autorizados, incluindo medidas necessárias para detectar, documentar e conter tais ameaças (ANST[1]).

Garantia da informação é o conjunto de operações que protegem e defendem a informação e os sistemas de informação garantindo sua disponibilidade, integridade, autenticidade, confidencialidade e não-repúdio. Inclui também o provimento de restauração de sistemas de informação, incorporando proteção, detecção e capacidades de reação (ANST[1]).

A segurança da informação pode ser caracterizada de várias formas. Um dos modelos adotados é o do NSTISSI[2], que modela a segurança da informação em três dimensões, como mostrado nas Figuras 2.1 e 2.2: propriedades de segurança da informação, estados da informação e medidas de segurança.

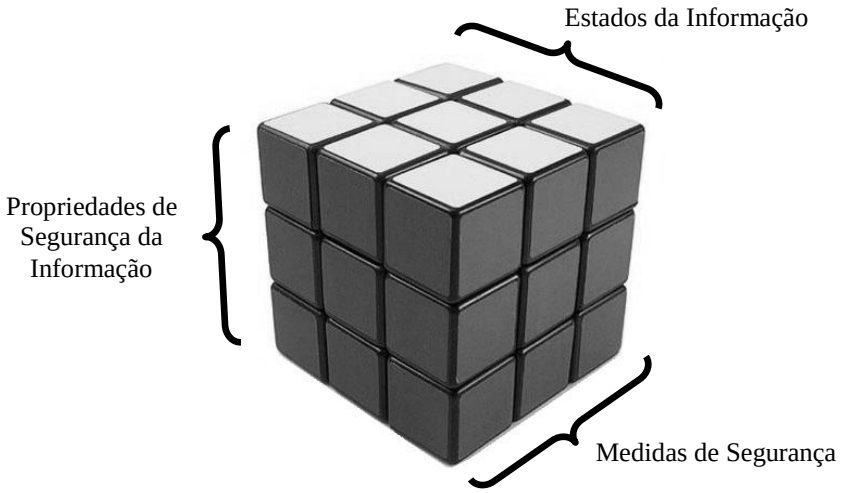


Figura 2.1 - Modelo de segurança da informação (NSTISSI 4011).

São aceitas três propriedades de segurança da informação: confidencialidade, integridade e disponibilidade. Essas propriedades podem ter diferentes prioridades de acordo com a missão da organização. Por exemplo, uma instituição financeira que permite que seus clientes façam transações bancárias pela *Internet*, provavelmente estará mais preocupada com a confidencialidade da informação, ou seja, que a informação não caia na mão de in-

[1]*American National Standard for Telecommunications – Telecom Glossary 2013.* Disponível em: *http://www.atis.org/glossary/definitionsList.aspx?find=I&kw=0*
[2]*National Training Standard for Information Systems Security Professionals*

trusos, mas apenas no destinatário correto que neste caso é a própria instituição bancária. Por outro lado, um sítio de buscas, como por exemplo o *Google*, estará mais preocupado com a disponibilidade, ou seja, que o sítio não fique sem acesso pelos seus usuários. A propriedade de integridade está associada a não alteração da informação por pessoas não autorizadas para isso (modificado - Stallings, 2005).

Em relação ao estado da informação, a informação não é apenas uma entidade estática, ela existe em processamento, por exemplo, quando está na memória RAM de um computador; armazenada, por exemplo, quando está em um disco rígido; ou em transmissão, por exemplo, em um cabo de rede de dados (CERT.br, 2010).

As medidas de segurança estão relacionadas às preocupações das organizações em implementar meios que efetivamente mitiguem os riscos em seus ativos de informação, como por exemplo: treinamentos, campanhas de conscientização, aquisição de equipamentos de segurança, etc (CERT.br, 2010).

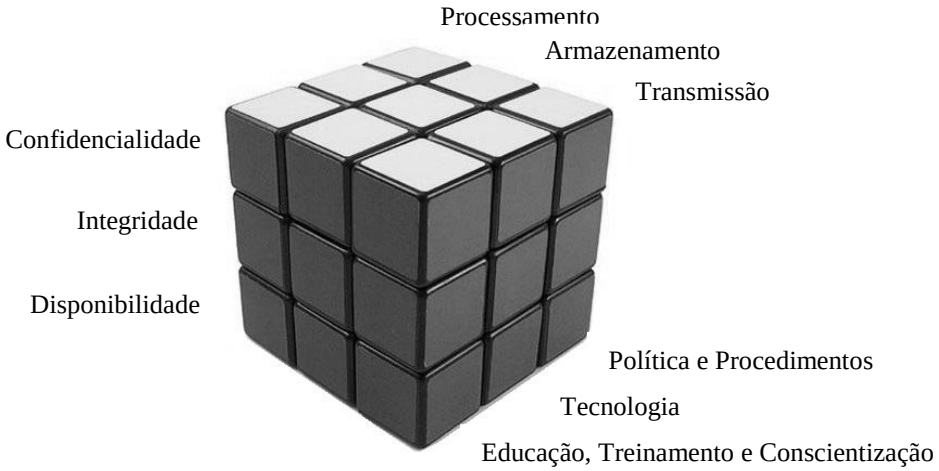


Figura 2.2 - Modelo detalhado de segurança da informação (NSTISSI 4011).

Este modelo de segurança enfatiza a necessidade de sustentar a confidencialidade, integridade e disponibilidade da informação em qualquer estado que a mesma possa estar. É importante notar que o conceito de armazenamento inclui qualquer meio no qual a informação possa ser gravada, incluindo até mesmo um papel impresso (CERT.br, 2010).

Organizações devem desenvolver políticas e procedimentos que regulam acessos, uso, modificação, transmissão e descarte da informação. Para defender e proteger os ativos e recursos de informação, tecnologias apropriadas devem ser seguramente configuradas e implantadas. Administradores de rede e de sistemas, bem como usuários, devem ser bem

informados em relação a suas responsabilidades no tocante à segurança da informação e também serem capazes de aplicar procedimentos apropriados a fim de garantir a salvaguarda da informação[3] (CERT.br, 2010).

2.3 – CONSCIENTAÇÃO DE AMEAÇAS

2.3.1 – Sequência geral de um ataque

Antes que um ladrão entre dentro de uma residência, é natural que ele primeiro leve um tempo coletando informações, a fim de auxiliá-lo em sua invasão. Ele quer determinar o melhor meio de entrar e identificar alguns obstáculos, tais como alarmes, vizinhos curiosos ou um cachorro. Atacantes de sistemas realizam passos semelhantes antes de dar início a um ataque. Eles precisam, por exemplo, identificar um ponto de entrada na rede de dados que não esteja bloqueado por um *firewall* ou monitorado por um IDS.

Levantamento de informações ou reconhecimento é considerado a primeira fase do passo a passo até se chegar ao alvo, e é uma tentativa sistemática de localizar, recolher, identificar e registrar informações sobre o alvo. O atacante tenta descobrir o máximo de informações possíveis sobre a vítima. Este primeiro passo é considerado simplesmente uma aquisição passiva de informações (Gadge; Patil, 2008).

No *scanning* é feito o mapeamento da rede e são utilizadas ferramentas e técnicas para determinar quais sistemas estão funcionando e são alcançáveis através da *Internet*, identificar nomes de domínios, redes associadas ao alvo em questão, etc (Hamisi; Mvungi; Mfinanga; Mwinyiwiwa, 2009).

Com a enumeração é possível, por exemplo, identificar contas válidas de usuários ou recursos de compartilhamento fragilmente protegidos (Hamisi; Mvungi; Mfinanga; Mwinyiwiwa, 2009).

Vulnerabilidade é a fraqueza de um ativo ou grupo de ativos que pode ser explorada por uma ou mais ameaças (ISO/IEC[4] 17799:2005).

Um sistema de informação de uma empresa pode ser considerado seguro se não contém vulnerabilidades. Ainda que existam ameaças, a ausência de vulnerabilidades torna tais

[3]Medidas objetivando a mitigação de riscos, como por exemplo: gerenciamento de senhas fortes, guardas de segurança, mecanismos de controle de acesso e treinamentos de conscientização sobre ameaças (Harris, 2010).
[4]*International Organization for Standardization/International Electrotechnical Commission*

ameaças sem efeito, já que não há nenhuma fragilidade a ser explorada. As vulnerabilidades podem ser de dois tipos: técnicas ou de gestão. Vulnerabilidades técnicas são fraquezas associadas aos *softwares* e *hardware* dos ativos. A má configuração e conexão dos ativos de uma organização e a especificação precária de políticas podem dar origem a vulnerabilidades gerenciais. Políticas inadequadas podem levar a problemas de segurança, da mesma forma usuários podem potencialmente fazer o mau uso de seus direitos e violar a segurança dos ativos da empresa (Sengupta; Mazumdar; Bagchi, 2009).

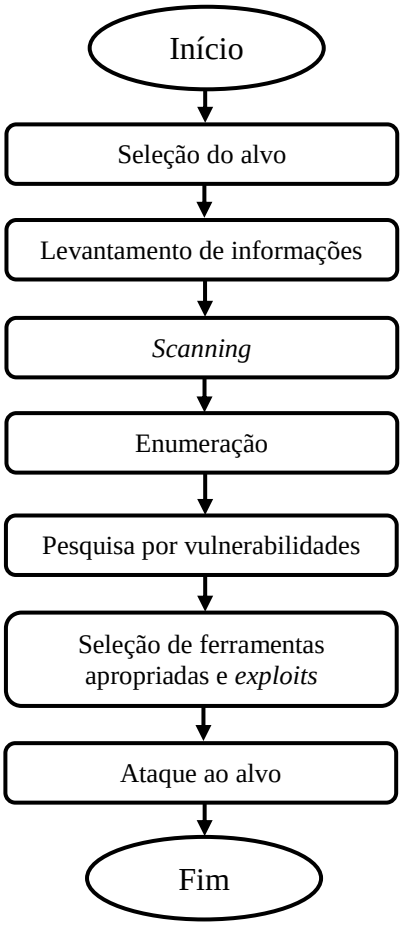


Figura 2.3 - Sequência de um ataque (modificado – Gadge; Patil, 2008).

Uma vez as vulnerabilidades sendo descobertas pelos atacantes, o próximo passo é selecionar ferramentas apropriadas e explorá-las. Uma das formas de se efetivar um ataque é utilizando *exploits*.

Exploit é um termo genérico para descrever pequenas partes de códigos executáveis que são projetados para explorar uma vulnerabilidade específica de um sistema. Estes pequenos códigos executáveis podem tanto ser executados isoladamente ou serem inseridos no

código de programas. Esses *exploits*, que podem ser de utilização local ou remota, variam muito quanto à sua forma, à linguagem de programação e ao poder de ataque. Geralmente há um diferente *exploit* para cada tipo de serviço, para cada tipo de vulnerabilidade ou para cada tipo de sistema operacional (Mourão, 2012).

A Figura 2.3 sintetiza em forma de fluxograma a sequência de um ataque, fazendo referência aos tópicos apresentados nessa seção.

2.3.2 – Ataques para obtenção de informações

Na Seção 2.3.1 foi discutida de forma qualitativa a sequência geral de um ataque. Não foi feita nenhuma menção de quais técnicas computacionais podem ser utilizadas para efetivar cada passo abordado.

Algumas técnicas e ferramentas podem ser utilizadas para obtenção de informações importantes para o ataque, como por exemplo: *dumpster diving*, informações públicas, engenharia social, ataques físicos, *packet sniffing*, *scanning* de vulnerabilidades e *firewalking*.

Tais técnicas podem ser aplicadas tanto sobre a óptica do atacante quanto sobre a óptica da defesa. Neste útlimo caso as técnicas são usadas para análise de segurança, visando identificar vulnerabilidades para posteriores correções e melhorias necessárias.

2.3.2.1 – Dumpster diving

Também conhecido como *trashing*, *dumpster diving* refere-se ao conceito de vasculhar uma empresa ou lixo individual de documentos descartados, informações e outros itens preciosos que poderiam ser usados em um ataque contra essa empresa ou pessoa. O intruso teria de ter acesso físico ao local, mas a área onde o lixo é mantido normalmente não é altamente protegida. O *dumpster diving* não é ético, mas não é ilegal (Harris, 2010).

2.3.2.2 – Informações públicas

As informações públicas podem ser obtidas livremente, principalmente na própria *Internet*, através, por exemplo, de pesquisas no *Google*. O *Google* tem provado ser um dos melhores meios de se realizar buscas, fornecendo ao atacante informações preciosas. Ele exporá inadvertidamente informações sensíveis de *websites* mal configurados, resultando em uma enorme quantidade de dados disponíveis. Através do uso adequado de operadores de

procura é possível filtrar os resultados de pesquisa chegando a arquivos cada vez mais específicos. Outro tipo de informação pública que pode ajudar pessoas mal intencionadas são consultas a servidores de DNS, fornecendo detalhes sobre sistemas, topologia e usuário.

2.3.2.3 – Engenharia social

Engenharia Social é a arte de explorar as fraquezas dos seres humanos, a fim de obter informação. Uma vez que a primeira fase de penetração no perímetro de segurança de uma organização é geralmente feita através da interação do atacante com o pessoal da própria organização, testes de penetração que envolvam métodos de engenharia social tornam-se extremamente importantes, quando se quer avaliar a segurança dos sistemas de informação da organização (Pavkovic; Perkov, 2011).

2.3.2.4 – Ataques físicos

O ataque físico à organização, em que são roubados equipamentos, *softwares* ou fitas magnéticas, constitui o método menos comum. Esse tipo de ataque permite que haja contato físico direto com os sistemas de informação, o que facilita as ações, pois não é necessário que técnicas de ataques remotos sejam utilizadas. Com o acesso direto ao sistema, além do roubo do próprio equipamento, é possível executar uma série de ações maliciosas ou destrutivas, tais como copiar documentos confidenciais, ler *e-mails* de terceiros, obter informações privilegiadas (como os salários de todos os funcionários ou estratégia de novos produtos), modificar arquivos importantes e alterar configurações ou privilégios de alguns usuários. A imaginação e a intenção do atacante é que vai limitar as ações no sistema a que ele obtém acesso físico, de modo que ele pode simplesmente destruir todas as informações, se assim desejar. Outros problemas relacionados a ataques físicos são o uso de *sniffers* ou analisadores de protocolos para capturar informações e senhas, além de implantação de sistema para capturar tudo o que o funcionário digita (*keystroke logger*). A perda de sigilo decorrente do uso dessas técnicas é uma das mais encontradas nas organizações (Geus; Nakamura, 2010).

2.3.2.5 – Packet sniffing

Packet sniffing é um método de verificar cada pacote à medida que o fluxo de pacotes flui através da rede, isto é, uma técnica na qual um usuário “fareja” dados pertencentes a outros

usuários da rede. *Sniffers* podem funcionar como ferramentas administrativas de rede ou para fins maliciosos, dependendo da intenção do usuário. Administradores de rede utilizam-nos para monitorar e validar o tráfego de rede. Este ataque também é conhecido como *eavesdropping* (Ansari; Rajeev; Chandrashekar, 2003).

2.3.2.6 – Scanning de vulnerabilidades

A função de um *scanner* de vulnerabilidades é a de verificar falhas em programas provenientes, por exemplo, de erros de desenvolvimento e falhas de configuração. Tais falhas constituem vulnerabilidades que podem ser exploradas por pessoas mal intencionadas.

Ferramentas de *scanner* de vulnerabilidades possuem um grande banco de dados de vulnerabilidades e a capacidade de explorar muitas das vulnerabilidades que eles identificam. Novas vulnerabilidades são encontradas a cada semana em sistemas operacionais, servidores *web*, *software* de banco de dados e aplicações (Harris, 2010).

Tais ferramentas podem ser utilizadas tanto por administradores de rede, no intuito de verificar as fraquezas existentes na rede de sua administração a fim de sanar tais problemas antes que um atacante as descubra e coloque os sistemas em risco; ou pode ser utilizado por pessoas mal intencionadas, no sentido de descobrir quais são as falhas presentes no alvo de ataque e explorá-las.

2.3.2.7 – Firewalking

Firewalking é um termo associado à ferramenta de segurança *firewalk*, que tem a finalidade de tentar determinar em um *firewall* informações referentes a protocolos da camada de transporte (modelo OSI). Com tais informações é possível descobrir quais portas estão liberadas, obtendo informações sobre suas regras de filtragem (Goldsmith; Schiffman, 1998).

Firewalking utiliza técnicas semelhantes ao *traceroute* para determinar se é possível ou não um determinado pacote passar da máquina do atacante até um *host* de destino através de um dispositivo de filtragem de pacotes. Esta técnica pode ser utilizada também para mapear roteadores encontrados antes de *firewalls* (Goldsmith; Schiffman, 1998).

2.3.3 – Ataques de negação de serviço

Ataque de negação de serviço é uma tentativa explícita de fazer com que uma rede ou sistema de informação fique indisponível para uso por seus usuários. Tal ataque pode ser agravado caso seja distribuído, dando origem ao DDoS, no qual são utilizados muitos computadores para lançar em grande escala ataques coordenados de DoS contra um ou mais alvos. Em um ataque de DDoS, o atacante usa mestres para controlar os escravos, que por sua vez sobrecarregam a vítima com inúmeras requisições, conforme ilustrado na Figura 2.4. O ataque de DDoS tem a capacidade de exaurir os recursos computacionais da vítima dentro de um curto período de tempo (Robinson; Thomas, 2011).

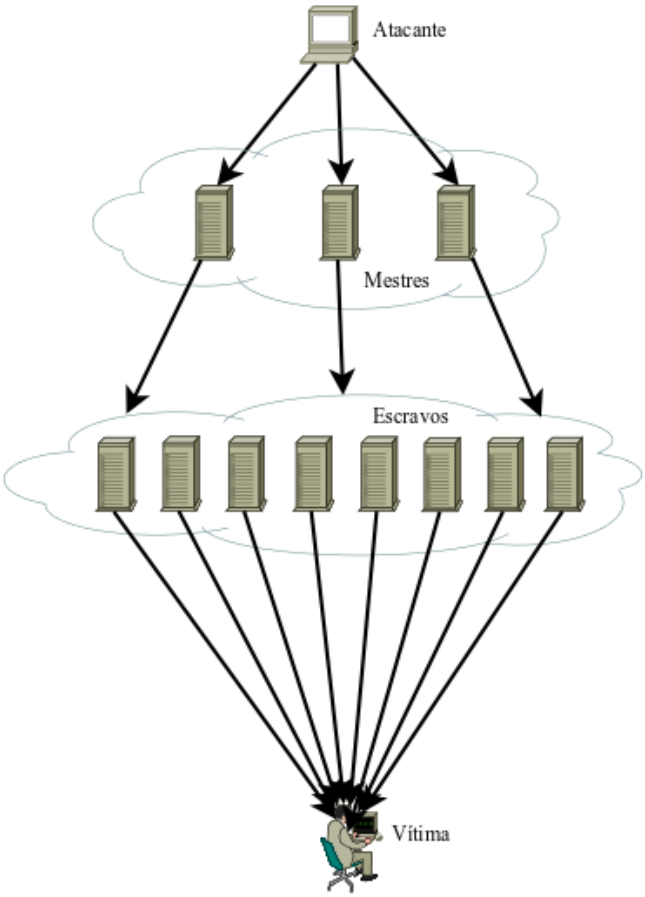


Figura 2.4 - Ataque de DDoS.

Pode-se citar duas técnicas utilizadas por atacantes para causar negação de serviço: *synflood* e *fraggle*. Tais técnicas serão detalhadas na Seção 6.1, por fazerem parte dos ataques modelados nesta dissertação.

2.4 – FIREWALL

Cada vez mais organizações estão aderindo à *Internet*, a fim de estabelecer um negócio de comércio eletrônico ou mesmo acessar informações com maior velocidade. Quando a rede de dados de uma organização está conectada à *Internet* sem fazer uso de medidas de segurança adequadas, ela se torna vulnerável a ataques de adversários externos. Sem *firewalls* uma organização torna-se incapaz de se prevenir contra muitas formas de acessos indesejáveis à sua rede de dados privativa, sistemas e ativos de informação (CERT.br, 2010).

Segundo Bill Cheswick e Steve Bellovin, em "*Firewalls and Internet Security: Repelling the Wily Hacker"* (Cheswick; Bellovin, 1994), *firewall* é um ponto entre duas ou mais redes, no qual circula todo o tráfego. A partir desse único ponto, é possível controlar e autenticar o tráfego, além de registrar, por meio de *logs*, todo tráfego da rede, facilitando sua auditoria (Avolio, 1999).

De acordo com Chapman, *firewall* é um componente ou conjunto de componentes que restringe o acesso entre uma rede protegida e a *Internet*, ou entre conjuntos de redes (Chapman; Zwicky; Cooper, 2000).

Partindo dessas duas definições clássicas, pode-se dizer que *firewall* é um ponto entre duas ou mais redes, que pode ser um componente ou um conjunto de componentes, por onde passa todo o tráfego, permitindo que o controle, a autenticação e os registros de todo o tráfego sejam realizados, como pode ser visto na Figura 2.5. O *firewall* deve ser visto como uma implementação da política de segurança. Ele é tão seguro quanto à política de segurança com que ele trabalha (Geus; Nakamura, 2010).

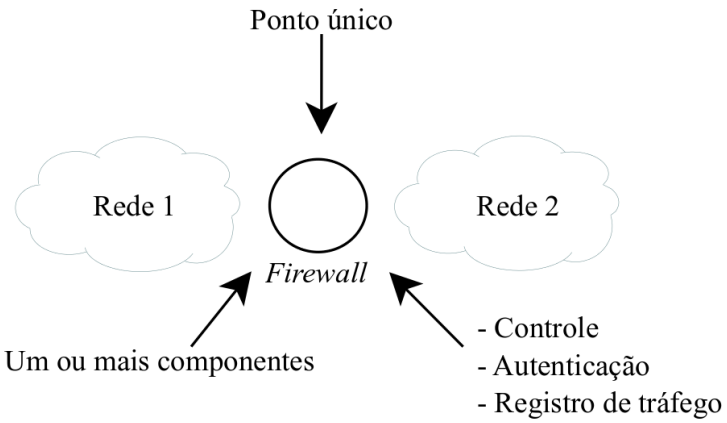


Figura 2.5 - Características de *firewall* (Geus; Nakamura, 2010).

Firewall pode ser um roteador, servidor ou dispositivo de *hardware* especializado. Ele monitora os pacotes que entram e saem da rede que protege e filtra os pacotes que não atendem aos requisitos da política de segurança, podendo descartar esses pacotes, acondicioná-los ou redirecioná-los, dependendo da regra submetida. Os pacotes são filtrados com base em endereços IP de origem e destino e portas de serviço (*socket*), tipo de protocolo, informações de cabeçalho, *bits* de sequência, dentre outros (Harris, 2010).

2.4.1 – Funcionalidades

De acordo com as definições apresentadas, o *firewall* é formado por diversos componentes, cada um com uma função específica, desempenhando diferentes papéis no sistema de segurança da organização. As principais funcionalidades associadas aos *firewalls* são: *packet filtering*, *proxy services*, *network address translation* e *virtual private network* (Chapman; Zwicky; Cooper, 2000).

Packet filtering é uma função do *firewall* que examina e toma decisões de permitir ou negar, baseadas nas informações do cabeçalho do pacote e, em alguns casos, no estado de seção do protocolo. *Proxies* fornecem um maior nível de segurança quando comparados à função de *packet filtering*, uma vez que podem operar tanto no cabeçalho do pacote como no seu conteúdo (CERT.br, 2010).

Network address translation é um processo pelo qual um roteador muda alguns dados nos cabeçalhos dos pacotes para modificar os endereços de rede. Isso permite a um roteador ocultar os endereços de *hosts* da rede em um dos lados. Esta técnica pode permitir que um grande número de *hosts* se conectem à *Internet* usando um único endereço público, um pequeno número de endereços alocados, ou pode permitir que *hosts* de uma rede com endereços privados se conectem à *Internet* usando endereços válidos (ou públicos). Não é realmente uma técnica de segurança, embora possa proporcionar uma pequena quantidade de segurança adicional. No entanto, ele geralmente é executado nos mesmos roteadores que fazem parte do *firewall* (Chapman; Zwicky; Cooper, 2000).

Virtual private network consiste de um conjunto de computadores que se interligam por meio de uma rede relativamente insegura e que fazem uso de encriptação e protocolos especiais para garantir a segurança (Stallings, 2005).

2.4.2 – Evolução tecnológica

Os primeiros *firewalls* foram implementados em roteadores, no final da década de 80, por serem os pontos de ligação natural entre duas redes. As regras de filtragem dos roteadores, conhecidas também como lista de controle de acesso (ACL), tinham como base decisões do tipo "permitir" ou "descartar" os pacotes, que eram tomadas de acordo com a origem, o destino e o tipo das conexões (Avolio, 1999).

A partir disso, tudo mudou rapidamente, de modo que a própria definição de que *firewalls* devem separar "nós" de "eles" foi modificada, fazendo com que eles passassem por uma evolução natural e crescimento em sofisticação e funcionalidade. O mundo tornou-se mais integrado e os serviços básicos, na atualidade, são o acesso *web*, acesso a bancos de dados via *Internet*, acesso a serviços internos da organização pela *Internet* (via VPN, por exemplo), serviços de áudio, vídeo, videoconferência, voz sobre IP (VoIP), entre tantos outros. Além disso, as organizações tem cada vez mais usuários utilizando uma maior variedade de serviços (modificado – Geus; Nakamura, 2010).

Dessa forma, muitos tipos diferentes de *firewalls* foram surgindo, a fim de atender a diferentes demandas, uma vez que cada ambiente pode ter requisitos específicos e objetivos de segurança diferentes, sendo necessário um tipo adequado de dispositivo. *Packet filtering* (filtro de pacotes), *stateful* (filtro de pacotes baseado em estados), *proxy*, e *firewall* híbrido são alguns dos tipos de *firewalls* existentes. A Tabela 2.1 traz um resumo de cada um desses tipos de *firewall*.

Tabela 2.1 - Diferenças entre *firewalls* (modificado – Harris, 2010).

Tipo de *firewall*	**Camada OSI**	**Caractecrísticas**
Packet filtering	Camada de rede	Verifica endereços de origem e destino, portas e serviços solicitados.
Stateful	Camada de rede	Verifica o estado e o contexto do pacote. Mantém o controle de cada fluxo usando uma tabela de estado.
Proxy	Camada de aplicação	Faz uma verificação profunda dentro de cada pacote e consegue tomar decisões granulares de controle de acesso.
Firewall híbrido	Camada de rede e camada de aplicação	Misturam os elementos das tecnologias do *packet filtering*, *stateful* e *proxy*.

2.4.3 – Arquiteturas

A arquitetura de um *firewall* também deve ser definida de acordo com as necessidades e política da organização, utilizando as funcionalidades e tecnologias descritas na Seção 2.4.1 e Seção 2.4.2, respectivamente.

Firewalls podem ser colocados em pontos distintos de uma rede, a fim de satisfazerem necessidades particulares. Além das caraterísticas já descritas, *firewalls* também podem ser usados para estabelecer uma DMZ entre uma rede interna e uma rede externa.

A zona desmilitarizada (DMZ) é uma rede que possui um papel essencial na arquitetura, pois permite que serviços sejam providos para os usuários externos (por meio de *bastion hosts* – computadores que devem ser projetados e configurados para resistirem a ataques, pois serão os alvos de ataques externos) ao mesmo tempo em que protege a rede interna dos acessos externos. Todo acesso externo fica restrito à zona desmilitarizada, não sendo possível um atacante que comprometa um servidor na DMZ chegar aos sistemas internos.

Há três arquiteturas clássicas do *firewall*: *dual-homed host*, *screened host* e *screened subnet*. Além dessas arquiteturas, Geus e Nakamura (Geus; Nakamura, 2010) citam a quarta arquitetura, que é a do *firewall* cooperativo, que estende as arquiteturas clássicas ao ambiente cooperativo, tratando de componentes, como redes privadas virtuais (VPN), sistemas de detecção de intrusão (IDS), banco de dados e infraestrutura de chaves públicas.

O ambiente cooperativo é caracterizado pelo relacionamento e pela integração dos mais diversos sistemas de diferentes organizações, nos quais as partes envolvidas cooperam entre si, na busca de um objetivo comum: velocidade e eficiência nos processos e nas realizações de negócios, que representam os elementos-chave para o sucesso de qualquer tipo de organização (Geus; Nakamura, 2010).

Dual-homed host refere-se a um dispositivo que possui duas interfaces de rede: uma voltada para a rede externa e a outra voltada para a rede interna. Em tal dispositivo é desabilitada a funcionalidade de encaminhamento e roteamento de pacotes, de modo que as duas redes fiquem realmente segregadas, por razões de segurança. A rede interna (a ser protegida) tem que se conectar necessariamente primeiro ao dispositivo, para que possa se comunicar com a rede externa (insegura) e vice-versa, mas nunca diretamente uma rede com a outra, de acordo com a Figura 2.6 (modificado – Harris, 2010).

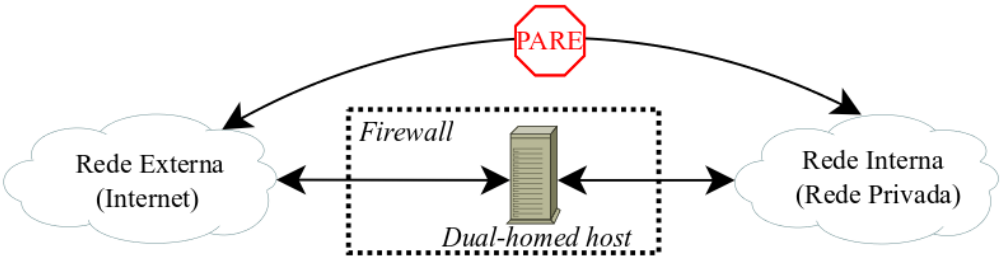


Figura 2.6 - Arquitetura *dual-homed host*.

A arquitetura *screened host* é formada por um filtro de pacotes e um *bastion host*. Da mesma forma que na aquitetura *dual-homed host*, a rede interna não se comunica diretamente com a rede externa, e vice-versa, de acordo com a Figura 2.7. O filtro deve ter regras que permitam o tráfego para a rede interna somete por meio do *bastion host*, de modo que os usuários externos que queiram acessar um sistema da rede interna devem, primeiramente, se conectar ao *bastion host*. O *bastion host* pode funcionar também como um *proxy*, exigindo que os usuários internos acessem a *Internet* por meio dele (Chapman; Zwicky; Cooper, 2000).

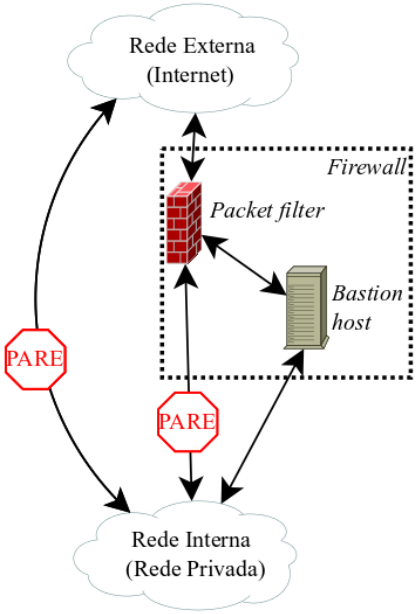


Figura 2.7 - Arquitetura *screened host*.

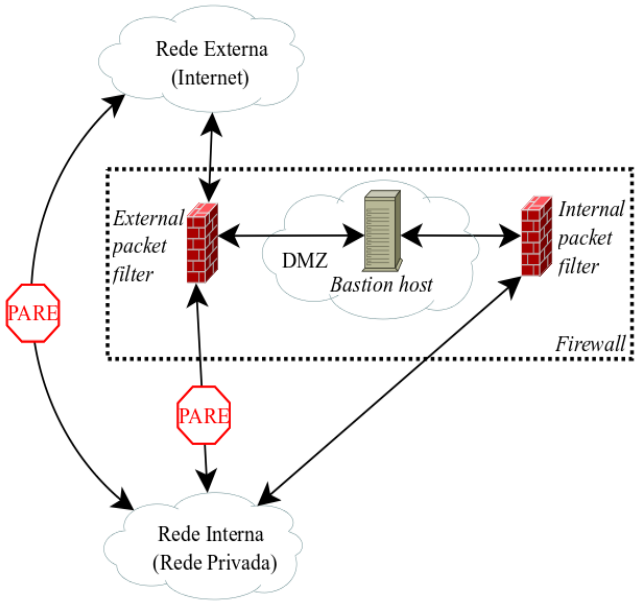


Figura 2.8 - Arquitetura *screened subnet*.

A arquitetura *screened subnet* difere da *screened host* por adicionar um segundo filtro de pacotes. Nessa arquitetura a rede externa conecta-se ao filtro de pacotes externo, a rede interna ao filtro de pacotes interno e a DMZ é localizada entre os dois filtros de pacotes. O tráfego entre a rede interna e a rede externa deverá, necessariamente, passar pelos dois filtros de pacotes e pela DMZ. Nesta arquitetura, o *bastion host* fica isolado na rede da

DMZ, o que propicia maior segurança para a rede interna da organização, pois, por exemplo, com um ataque ao *bastion host* não é possível se utilizar um *sniffer* para a captura de pacotes de usuários internos (modificado – CERT.br, 2010).

É possível também implementar a arquitetura *screened subnet* através de um único dispositivo com três interfaces, uma para a rede interna, outra para a rede externa e a terceira para a DMZ. Os filtros de pacotes funcionariam neste caso em cada interface. É interessante optar por esta configuração quando se pretende economizar recursos financeiros.

2.5 – SISTEMA DE DETECÇÃO DE INTRUSÃO

O *firewall* foi abordado na Seção 2.4, e visto que é parte de um sistema de segurança necessário para a proteção das organizações; porém, ele não é a solução definitiva para os problemas de segurança. Ele pode funcionar como a primeira linha de defesa, realizando controle de acesso no nível de rede. O que não é bloqueado pelo *firewall* obviamente é permitido, como é o caso do acesso a serviços legítimos. O tráfego relativo a esse acesso, uma vez permitido, não é mais inspecionado pelo *firewall*, devendo a segurança agora ser feita pelo próprio serviço legítimo ou por outros meios, como por exemplo, por intermédio de um IDS. Nesse contexto, o *firewall* não tem a capacidade de detectar intenções maliciosas dentro de um pacote.

IDS é um dispositivo ou aplicação de *software* que monitora a rede ou *hosts* à procura de atividades maliciosas ou violações de políticas (detecção), analisa o tráfego suspeito (análise), e produz informações de eventos (resposta) para a estação de gerenciamento. As detecções podem ser feitas com base em algum conhecimento prévio, através de duas técnicas: por meio de assinaturas de ataques ou por meio de desvios comportamentais (Thukral; Maqsood; Upadhyay, 2013).

A detecção por meio de assinatura combina dados observados com a descrição pré-definida de comportamento intrusivo. Uma desvantagem dessa abordagem é que ela só pode detectar intrusões que correspondam a regras pré-definidas, ou seja, ao conjunto de assinaturas, que precisa ser constantemente atualizado para conhecer uma nova ameaça. Por outro lado, este método pode ser altamente eficaz na identificação, com cada vez mais precisão, de ataques conhecidos e suas variações. Além disso, pode produzir um baixo número de falsos alarmes (Stiawan; Abdullah; Idris, 2011).

Detecção baseada em anomalia ou desvios comportamentais é feita comparando o tráfego da rede com padrões normais de uso aceitável pré-estabelecidos, obtidos através de períodos de testes realizados previamente (“período de treinamento”). Uma vantagem desta abordagem é a capacidade de detectar novos ataques que ainda não possuam assinaturas definidas. Infelizmente, esta abordagem produz muitos falsos alarmes e tende a consumir muito tempo com pesquisas, a fim de se obter perfis acurados e abrangentes do comportamento normal. Isso requer um grande conjunto de dados de análise com *logs* de sistemas do ambiente de rede (Wu; Banzhaf, 2010).

Um dos grandes desafios associados à adoção dos IDS é reduzir a quantidade de falsos alarmes: falso positivo e falso negativo. Há quatro tipos de alerta: o verdadeiro negativo, que é o tráfego do usuário normal e nenhum alarme é gerado; o verdadeiro positivo, que é gerado alarme devido ao tráfego suspeito; falso negativo, onde não é gerado alarme quando há um tráfego suspeito; e o falso positivo, que produz um alerta quando identifica um tráfego de atividade normal (Stiawan; Abdullah; Idris, 2011).

Há sistemas que possuem todas as habilidades de um sistema de detecção de intrusão e que também podem impedir possíveis incidentes. Tais sistemas são conhecidos como sistemas de prevenção de intrusão (IPS). A combinação de IDS e IPS gera um novo termo conhecido como sistema de detecção e prevenção de intrusão (IDPS), o qual é capaz de detectar e prevenir ataques (Mukhopadhyay; Chakraborty; Chakrabarti, 2011).

Devido à proximidade de significados, esta dissertação utiliza por vezes o termo “Sistema de Detecção e Prevenção de Intrusão” (IDPS) para se referir a ambos os conceitos: IDS e IPS.

2.5.1 – Tipos

Os tipos primários de IDPS são os seguintes: baseado em *host* ou baseado em rede. Com o processo natural de evolução tecnológica houve o surgimento do IDS híbrido.

2.5.1.1 – IDPS baseado em host

Monitora as características de um único *host* e os eventos suspeitos que ocorrem dentro dele. Exemplos de características que um IDPS baseado em *host* pode monitorar são: o tráfego de rede, *logs* do sistema, processos em execução, atividade de aplicativos, acesso e modificação de arquivos e alterações na configuração do sistema. HIDPS são implantados

em *hosts* críticos, tais como servidores de acesso público e servidores que contêm informações confidenciais (Mukhopadhyay; Chakraborty; Chakrabarti, 2011).

As principais vantagens dos HIDS são: relativamente fáceis de implantar e gerenciar, e geralmente não utilizam recursos excessivos do *host* no qual está instalado. Como principais desvantagens dos HIDS, pode-se citar: um pouco trabalhoso monitorar múltiplos computadores hospedando individualmente um HIDS; e se um host é comprometido, o IDS pode cessar o funcionamento, não gerando mais alertas (CERT.br, 2010).

2.5.1.2 – IDPS baseado em rede

IDPS baseado em rede utiliza sensores, que podem ser computadores com o *software* necessário instalado, ou equipamentos dedicados, cada um com suas placas de interface de rede em modo promíscuo. Quando uma placa de rede é colocada em modo promíscuo, o *driver* NIC captura todo o tráfego, faz uma cópia de todos os pacotes e, em seguida, passa uma cópia para a pilha TCP e uma cópia para um analisador, a fim de procurar por tipos específicos de padrões (Harris, 2010).

O NIDS pode operar em dois modos: passivo ou *inline*. O NIDS que opera em modo passivo captura o tráfego do segmento de rede – isso é possível de ser implementado através do espelhamento de portas de roteadores – enquanto o NIDS que opera em modo *inline* faz com que todo o tráfego da rede passe por ele. Dessa forma, o NIDS *inline* é capaz não apenas de detectar os ataques, mas também de prevení-los, pois os pacotes do ataque não chegam aos servidores (aos alvos). NIDS *inline* são chamados de sistemas de prevenção de intrusão (IPS). Uma vez que existe IPS baseado em *host*, NIDS que opera em modo *inline* pode ser caracterizado como um IPS baseado em rede.

Como principais vantagens dos NIDS pode-se citar: possibilidade de monitorar múltiplas máquinas através de apenas uma localização física; e, se um *host* ou rede monitorada cessar o funcionamento, mesmo assim o NIDS pode gerar alertas. As principais desvantagens dos NIDS são: uma vez que captura todos os pacotes de uma rede, pode produzir uma enorme quantidade de dados difícil de gerenciar; o *hardware* do NIDS deve ser bastante poderoso para poder processar todos os pacotes; e, caso o NIDS venha a falhar, várias máquinas deixam de ser monitoradas ao mesmo tempo (CERT.br, 2010).

2.5.1.3 – IDS híbrido

A utilização dos dois tipos de IDS ao mesmo tempo traz grandes benefícios. O IDS híbrido tem como objetivo combinar os pontos fortes do HIDS e do NIDS, a fim de oferecer uma melhor capacidade de detecção de intrusões (Ranum, 2001).

O IDS híbrido opera como o NIDS, coletando o tráfego da rede, processando os pacotes, detectando e respondendo a ataques. A diferença é que ele faz isso como um HIDS, ou seja, processa os pacotes endereçados ao próprio sistema. Com isso, desaparece o problema de desempenho, comum no NIDS. Por outro lado, ainda existe o problema da escalabilidade, pois um IDS híbrido deve ser instalado em cada equipamento (Ranum, 2001).

2.5.2 – Componentes de um IDPS

Segundo Scarfone e Mell (Scarfone; Mell, 2007), os componentes típicos de um IDPS são: sensor ou agente, gerente, servidor de banco de dados e interface de visualização.

O sensor ou agente monitora e analisa os eventos. O termo sensor é tipicamente utilizado no monitoramento de redes com *hardware* específico (*appliances*), enquanto o termo agente é tipicamente utilizado em sistemas HIDS, nos quais há código instalado nos equipamentos a serem monitorados.

Gerente é um equipamento central que recebe e trata as mensagens dos sensores e agentes. Os gerentes podem analisar as mensagens em conjunto e correlacioná-las, identificando intrusões que sensores ou agentes individuais não poderiam. Pode haver um ou múltiplos gerentes, dependendo da complexidade do ambiente no qual o IDPS está instalado.

O banco de dados é o repositório das mensagens e eventos gerados pelos sensores, agentes ou gerentes. Normalmente, esse banco de dados é intrínseco ao gerente. No entanto, isso não é uma regra.

Interface de visualização é um programa que provê os meios para que os usuários e administradores de IDPS possam executar suas tarefas no sistema. Podem ser instalados como clientes em computadores pessoais ou *laptops*, ou não requererem essa instalação, sendo apenas acessados através de um navegador *web*. Podem existir interfaces distintas para administração, monitoramento ou análises.

Uma solução de IDPS pode ser instalada conectando diretamente cada componente à rede de dados que se quer defender ou cria-se uma rede separada, com endereçamento próprio, desenhada especialmente para a instalação e conexão dos equipamentos de segurança. No entanto, os agentes e sensores devem ter uma interface de rede que os conectem também com a rede que se quer defender, não havendo essa necessidade para os gerentes, servidores de banco de dados e interfaces de visualização (Scarfone; Mell, 2007).

2.5.3 – Honeypot

Honeypot é um sistema concebido e configurado para ser acessível para um intruso e que possui vulnerabilidades conhecidas. Um *honeypot* fornece um ambiente e informações adicionais que podem ser utilizados para apoiar a análise de intrusão. Assim, os detalhes da técnica utilizada e do ataque em si podem ser capturados e estudados. Ele serve como um sensor para um IDS, esperando por ataques de intrusos, enganando-os, pois não sabem que seus passos estão sendo totalmente monitorados. Ter um *honeypot* servindo como um sensor fornece indicações e advertências de um ataque. *Honeypots* têm a capacidade de detectar intrusos num ambiente controlado e conservar um estado conhecido (Allen; Christie; Fithen; McHugh; Pickel; Stoner, 2000).

Honeypots são divididos em duas categorias, dependendo do seu nível de interação com potenciais atacantes: *honeypots* de alta interatividade e *honeypots* de baixa interatividade. Em um *honeypot* de baixa interatividade são instaladas ferramentas para emular sistemas operacionais e serviços com os quais os atacantes irão interagir. Esse tipo de *honeypot* limita a interação com outros *hosts* na rede, reduzindo os riscos de comprometer a segurança da rede como um todo, caso um atacante consiga com sucesso ignorar os mecanismos de isolamento implementados nos serviços emulados. *Honeypots* de alta interatividade são complexos de serem implementados, executando sistemas operacionais reais e implementações avançadas de serviços comuns com que um usuário mal-intencionado pode interagir totalmente dentro de áreas de segurança e mecanismos de isolamento em geral. Este tipo de *honeypot* captura mais detalhes sobre as atividades maliciosas realizadas por um atacante, permitindo que os sistemas de análise possam determinar exatamente as vulnerabilidades que foram exploradas, as técnicas de ataque utilizadas e o código malicioso executado (Juan, 2009).

Para operar um *honeypot* necessita-se de um bloco de endereçamento IP da instituição e que não esteja sendo utilizado. Este bloco deve ser visto como uma rede isolada ou um

novo segmento de rede da instituição. Não deve fazer parte de qualquer rede ou segmento de rede que esteja previamente sendo utilizado (Hoepers; Steding-Jessen; Chaves, 2007).

É fortemente recomendado que não haja poluição de dados, como por exemplo, o administrador do *honeypot* acessando-o via rede para realizar procedimentos de manutenção, ou realizando varreduras no bloco de endereçamento IP do *honeypot* para verificar se os serviços estão realmente funcionando, entre outros. Caso contrário, pode ser extremamente difícil distinguir entre os eventos que fazem parte dos procedimentos de administração e manutenção e os eventos gerados por atividades maliciosas (Hoepers; Steding-Jessen; Chaves, 2007).

Também é muito importante que não haja qualquer tipo de filtragem para o bloco de endereçamento IP alocado para o *honeypot*, pois assim existem mais chances de se observar técnicas utilizadas pelos atacantes antes desconhecidas, ataques novos, etc (Hoepers; Steding-Jessen; Chaves, 2007).

2.5.4 – Topologias de IDS

O IDS pode ser utilizado em diversas localidades da rede de uma organização, pois cada posição significa um tipo de proteção específica. Algumas das posições em que o NIDS pode ser utilizado são observadas na Figura 2.9. Para aumentar o nível de segurança, um HIDS ou IDS híbrido pode ser utilizado em cada um dos servidores.

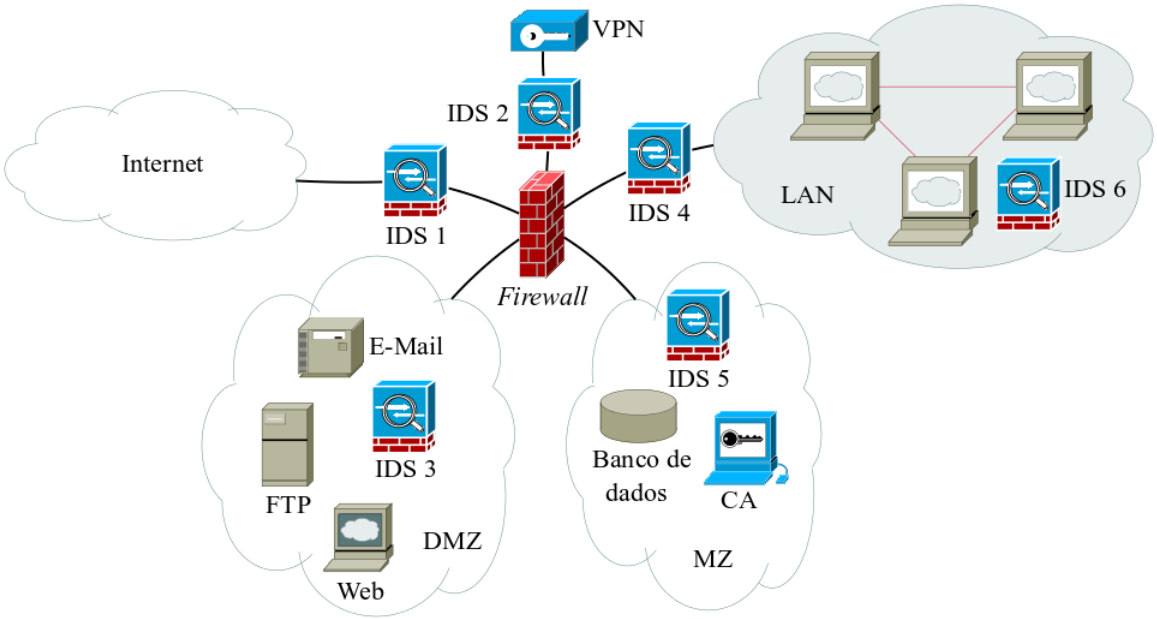


Figura 2.9 - Topologias de IDS (modificado - Geus; Nakamura, 2010).

Referenciando-se à Figura 2.9, serão explicados abaixo detalhes das localizações possíveis de um IDS em uma rede de dados, de acordo com o ponto de vista de Geus e Nakamura (Geus; Nakamura, 2010). Tais localizações são tratadas como: IDS 1, IDS 2, IDS 3, IDS 4, IDS 5 e IDS 6.

IDS 1 detecta todas as tentativas de ataque contra a rede da organização, até mesmo as tentativas que não teriam nenhum efeito, como os ataques a servidores *web* inexistentes. Essa localização oferece uma rica fonte de informação sobre os tipos de tentativas de ataques que a organização estaria sofrendo.

IDS 2 apresenta o IDS funcionando no próprio *firewall*. Nesta configuração o IDS pode detectar tentativas de ataque contra o *firewall*.

IDS 3 detecta tentativas de ataque contra servidores localizados na DMZ, que são capazes de passar pelo *firewall*. Assim, ataques contra serviços legítimos situados na DMZ, podem ser detectados por esse IDS.

IDS 4 detecta tentativas de ataque contra recursos internos que passaram pelo *firewall* e que podem acontecer via VPN.

IDS 5 detecta tentativas de ataque contra servidores localizados na MZ que passaram pelo *firewall*, pela VPN ou por algum outro serviço da DMZ, como o servidor *web*. Isso ocorre porque os recursos da MZ não podem ser acessados diretamente pelo usuário, a não ser via algum servidor da DMZ, ou via VPN.

IDS 6 detecta tentativas de ataques internos na organização. Esse posicionamento passa a ser importante em ambientes cooperativos, devido ao aumento de pontos de segurança característicos. O provimento de acesso cada vez maior a recursos internos faz com que a vigilância interna seja um fator de sucesso para o ambiente cooperativo.

Uma consideração importante, segundo Northcutt e Novak (Northcutt; Novak, 2002) é que, quando o IDPS fica antes do *firewall*, como o IDS 1, a detecção é considerada simultânea aos ataques (detecção de ataques). Já quando o IDS fica depois do *firewall*, como os IDS 3, 4, 5 e 6, a detecção passa a ser de intrusões, uma vez que o atacante já passou pelo *firewall* (detecção de intrusões).

2.6 – SIEM

Nas seções anteriores foram abordadas, dentre outros assuntos, questões referentes a tipos de ataques e como se defender utilizando, por exemplo, *firewalls* e detectores de intrusão. Não foi citado em nenhum momento como tratar os incidentes de rede de forma eficaz, nem como gerenciar os vários eventos gerados pelos diversos equipamentos de uma organização. A resposta para essas e outras questões torna-se fundamental ao tentar descobrir por vulnerabilidades, incidentes e até mesmo entender o passo a passo de um ataque. Nesse contexto é oportuno citar o *Security Information and Event Management* (SIEM), um conceito que envolve gerenciamento de eventos de informações de segurança.

O SIEM é atualmente um dos mais importantes caminhos de pesquisa na área de segurança de rede de computadores. A essência desta tecnologia é fornecer uma coleção ordenada de registros de *logs* de segurança (eventos de segurança) a partir de uma variedade de fontes, e sua conservação a longo e curto prazo em um repositório centralizado de dados, em um formato comum para a modelagem e análise, a fim de detectar e prever ataques e desenvolver contramedidas. A análise dos dados em sistemas SIEM baseia-se, como regra, nos métodos de correlação de eventos, mineração, raciocínio lógico e de visualização de dados (Kotenko; Polubelova; Saenko, 2012).

O SIEM surgiu a partir da união de diferentes tecnologias, porém complementares: *Log Management System*, *Security Event Correlation*, *Security Event Management* e *Security Information Management*.

LMS é um sistema que coleta e armazena, em um único local, arquivos de *log* (de sistemas operacionais, aplicações, etc) a partir de múltiplos *hosts* e sistemas, permitindo acesso centralizado aos *logs* em vez de acessá-los a partir de cada sistema individualmente (AlienVault[5], 2013).

Para um determinado código de *software*, três falhas a tentativas de acessar a mesma conta de usuário a partir de três clientes diferentes são apenas três linhas em seu arquivo de *log*. Para um analista, é uma seqüência peculiar de eventos dignos de investigação e correlação de *logs* (à procura de padrões em arquivos de *log*), gerando alertas quando essas coisas acontecem (AlienVault, 2013).

SEM é similar a um LMS, mas comercializado para os analistas de segurança em vez de

[5]Disponível em: http://www.alienvault.com/

administradores de sistema. SEM é capaz de destacar as entradas de *log* com diferentes níveis de segurança. São sistemas focados na agregação de dados com uma quantidade razoável de informações capazes de informar quais os incidentes de segurança podem ser tratados de imediato (Afzaal; Di Sarno; D'Antonio; Romano, 2012).

SIM é um sistema de gerenciamento de ativos, mas com recursos para incorporar informações de segurança também. *Hosts* podem ter relatórios de vulnerabilidades listadas, alertas de detecção de intrusão e antivirus podem ser mostrados mapeados para os sistemas envolvidos (AlienVault, 2013).

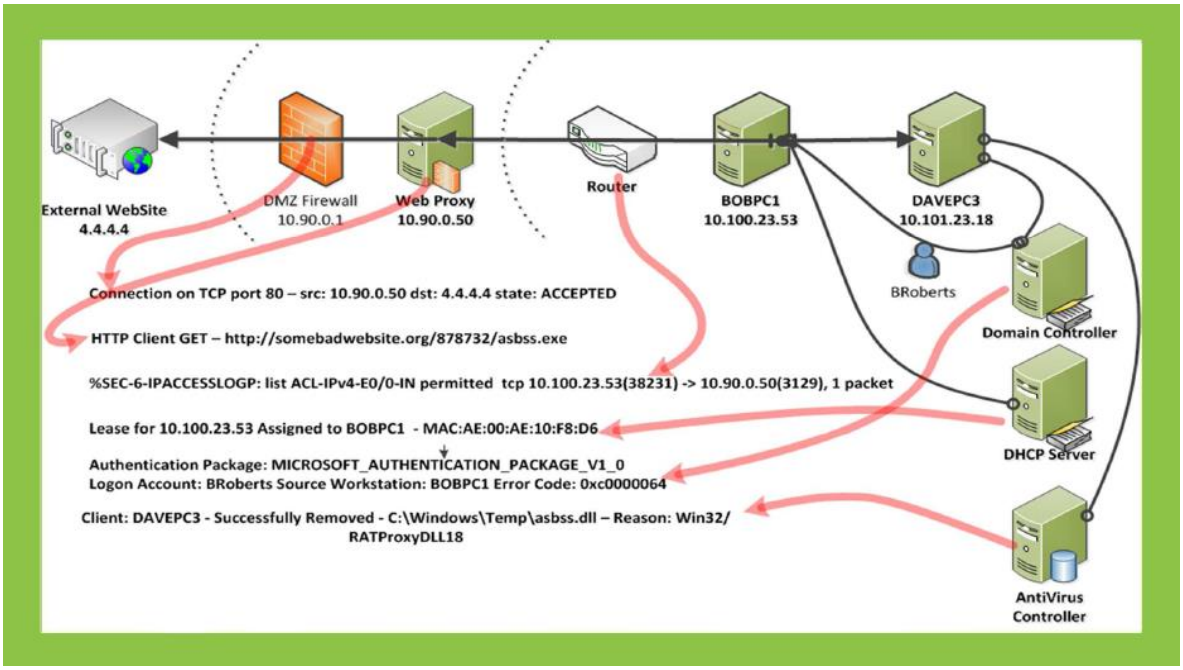


Figura 2.10 - Exemplo de um SIEM (AlienVault, 2013).

Existem diversos tipos de ferramentas que implementam SIEM, uma das mais conhecidas é o OSSIM, desenvolvido pela AlienVaut. O OSSIM possui uma infraestrutura robusta de coleta, mecanismo de correlação e avaliação de riscos, elaboração de relatórios e gerenciamento de ferramentas que são muito impressionantes. O resultado é uma plataforma coesa que oferece abstração de dados e permite que o analista de segurança possa monitorar milhões de eventos e focar em específicos, como procurar "agulhas no palheiro", que é o que realmente interessa. Customização é enfatizada aos usuários na medida em que podem escolher como implantar a tecnologia, quais ferramentas usar e como configurar e ajustar o dispositivo para satisfazer as suas necessidades individuais. Mais de quinze das mehores ferramentas de código aberto são compilados no OSSIM, tais como: Snort, OpenVas, Ntop, Nagios, (Miller; Harris; Harper; VanDyke; Blask, 2011).

Com o SIEM é possível olhar amplamente para o que está acontecendo em toda a rede, através de informações obtidas dos controles de segurança ou fontes de informação. O IDS só entende pacotes, protocolos e endereços IP; segurança de *endpoints* está associada a arquivos, nomes de usuários e *hosts*; seus *logs* de serviços mostram *logins* de usuários, atividades de serviços e alterações de configuração; o sistema de gestão de ativos vê aplicativos, processos de negócios e proprietários. Nenhuma delas, por si só, pode dizer o que está acontecendo com o negócio da organização em termos de segurança da continuidade dos processos de negócio, mas juntos, isso é possível (AlienVault, 2013).

A Figura 2.10 mostra o que um SIEM apresentaria a um profissional de segurança: que a máquina de Bob foi comprometida com o arquivo *asbss.exe*, obtido através de um acesso a um *website* infectado. Este *malware* usou a conta de Bob para tentar infectar *DAVEPC3*, mas o antivírus conseguiu atuar. A máquina de Bob, *BOBPC1*, no entanto, provavelmente está comprometida. Um tratamento para esse incidente seria bloquear o domínio malicioso (conta de Bob) e sanitizar *BOBPC* (AlienVault, 2013).

O SIEM é um forte aliado à forense digital, uma vez que possui um módulo de armazenamento forense (*forensic storage*) que ajuda a reter evidências digitais contra pessoas mal-intencionadas responsáveis por falhas de segurança (Afzaal; Di Sarno; D'Antonio; Romano, 2012).

Forense digital é um ramo da ciência forense referente à evidência de irregularidades encontradas em computadores e mídias de armazenamento digital para diversos fins. O objetivo da forense digital é explicar o estado atual de um sistema de computador, um dispositivo de armazenamento médio, um documento eletrônico, ou até mesmo pacotes movendo-se através de uma rede (Miller; Harris; Harper; VanDyke; Blask, 2011).

Uma organização pode exigir que um analista possua habilidades em forense digital por muitas razões: quando necessário para processos judiciais, para analisar sistemas de informática, mídia e comunicações pertencentes aos réus ou litigantes; para recuperar dados de uma organização ou usuário em caso de falha de *hardware*, falha de *software*, ou problema acidental; para analisar um sistema de computador depois de um evento de comprometimento, a fim de determinar como o atacante ganhou acesso ao sistema e as ações realizadas pelo atacante; e para reunir provas para usar contra um empregado que uma organização suspeita de transgressão (Miller; Harris; Harper; VanDyke; Blask, 2011).

3 – AUTOVALORES E AUTOVETORES

Autovalores e autovetores[1] são entidades comumente empregadas em álgebra matricial, podendo revelar muitas informações importantes sobre o que se está modelando com matrizes. No contexto de detecção de tráfego malicioso, as matrizes podem ser usadas para representar a quantidade de tráfego associado a cada porta de comunicação em um determinado instante de tempo, por exemplo.

3.1 – DEFINIÇÃO

Seja V um espaço linear e S um subespaço de V. Considere a seguinte transformação linear de S em V, $T: S \to V$. Um escalar λ denomina-se autovalor de T se existe um elemento não nulo x em S, tal que:

$$T(x) = \lambda x \tag{3.1}$$

O elemento x se chama autovetor de T, pertencente a λ. O escalar λ se chama autovalor correspondente a x. Existe somente um autovalor correspondente a um dado autovetor x.

Considere de agora os espaços vetoriais $V \in \mathbb{R}^n$ e as operações lineares definidas por:

$$\begin{aligned} T: \mathbb{R}^n &\to \mathbb{R}^n \\ \mathbf{v} &\longmapsto T(\mathbf{v}) = \mathbf{G}\mathbf{v}, \end{aligned} \tag{3.2}$$

onde $\mathbf{G}$ é uma matriz quadrada de ordem n.

Por definição, $\mathbf{v} \neq 0$ é um autovetor de T se $T(\mathbf{v}) = \mathbf{G}\mathbf{v} = \lambda\mathbf{v}$.

Equivalentemente, $\mathbf{G}\mathbf{v} - \lambda\mathbf{v} = 0 \Leftrightarrow \mathbf{G}\mathbf{v} - \lambda\mathbf{I}\mathbf{v} = 0 \Leftrightarrow (\mathbf{G} - \lambda\mathbf{I})\mathbf{v} = 0$

$$(\mathbf{G} - \lambda\mathbf{I})\mathbf{v} = 0, \tag{3.3}$$

onde $\mathbf{I}$ é a matriz identidade de ordem n e $(\mathbf{G} - \lambda\mathbf{I})$ representa uma matriz de ordem n.

Uma solução não nula da Equação (3.3) existe somente se $(\mathbf{G} - \lambda\mathbf{I})$ for singular (não inversível), ou seja, $\det(\mathbf{G} - \lambda\mathbf{I}) = 0$.

$$\det(\mathbf{G} - \lambda\mathbf{I}) = 0 \tag{3.4}$$

A Equação (3.4) denomina-se equação característica da matriz $\mathbf{G}$ e as raízes λ são todos os

[1]As palavras autovalor e autovetor são as traduções das palavras alemãs *eigenvektor* e *eigenwert*. Alguns autores empregam os termos vetor característico ou vetor próprio como sinônimos de autovetor. Os autovalores também se chamam valores característicos ou próprios (Apostol).

autovalores da matriz $\mathbf{G}$. O determinante $\det(\mathbf{G} - \lambda\mathbf{I})$ é um polinômio na variável λ, denominado polinômio característico e denotado por $f(\lambda) = \det(\mathbf{G} - \lambda\mathbf{I})$.

Uma vez os autovalores computados, então os autovetores podem ser determinados através da Equação (3.5).

$$\mathbf{Gv} = \lambda\mathbf{v} \tag{3.5}$$

Só é possível obter autovalores e autovetores de matrizes quadradas. A quantidade de autovalores é definida pela ordem da matriz.

3.2 – POSTO DE UMA MATRIZ

Antes de apresentar a definição de posto de uma matriz, é necessário apresentar a noção de dependência e independência linear.

Diz-se que um conjunto de vetores $\{\boldsymbol{v_1}, \boldsymbol{v_2}, \boldsymbol{v_3} \ldots \boldsymbol{v_n}\}$ é linearmente dependente (l.d.) caso exista um conjunto de escalares $k_1, k_2, k_3 \ldots k_n$, tal que:

$$k_1\boldsymbol{v_1} + k_2\boldsymbol{v_2} + k_3\boldsymbol{v_3} + \cdots + k_n\boldsymbol{v_n} = 0, \tag{3.6}$$

onde $k_1, k_2, k_3 \ldots k_n$ não podem ser todos iguais a zero.

Caso não seja possível encontrar um conjunto de escalares $k_1, k_2, k_3 \ldots k_n$ (nem todos nulos) que satisfaçam (3.6), o conjunto de vetores $\mathrm{v}_1, \mathrm{v}_2, \mathrm{v}_3 \ldots \mathrm{v}_\mathrm{n}$ é dito linearmente independente (l.i.).

Dessa forma, posto (*rank*) de uma matriz (quadrada ou retangular) é definido como o número de colunas (linhas) linearmente independentes.

Caso uma matriz tenha um único elemento diferente de zero, com todos os demais elementos iguais a zero, então o posto dessa matriz é igual a um. A matriz nula possui posto igual a zero. O posto de matriz é sempre $\leq \min(m, n)$, considerando uma matriz com *m* linhas e *n* colunas. É dito que essa matriz possui posto completo se o seu posto $= \min(m, n)$.

3.3 – MATRIZES SEMELHANTES

Duas matrizes $n_x n$ $\mathbf{G}$ e $\mathbf{B}$ são definidas como semelhantes se existir uma matriz não singular $\mathbf{F}$, tal que:

$$\mathbf{B} = \mathbf{F}^{-1}\mathbf{GF} \tag{3.7}$$

Duas matrizes $n_x n$ são semelhantes se e somente se representam a mesma transformação linear. Além disso, matrizes semelhantes possuem o mesmo polinômio característico e, portanto, os mesmos autovalores. Outra propriedade de matrizes semelhantes é que elas possuem o mesmo determinante.

3.4 – DIAGONALIZAÇÃO DE MATRIZES

Sistemas complexos podem ser representados por matrizes que, por vezes, são de difícil manipulação, exigindo um grande esforço computacional, longo tempo de processamento, etc. Encontrar formas alternativas de representar tais matrizes, simplificando-as, a fim de se reduzir o tempo de operação, é um dos objetivos da diagonalização de matrizes.

Um caso particular da Equação (3.7) seria encontrar a matriz $\mathbf{F}$, a fim de diagonalizar a matriz $\mathbf{G}$. $\mathbf{B}$ seria a matriz diagonal.

Dessa forma, se $\mathbf{B}$ é uma matriz diagonal semelhante à matriz $\mathbf{G}$ e, como visto na Seção 3.2, os autovalores são mantidos os mesmos quando as matrizes são semelhantes, então a matriz $\mathbf{B}$ é formada pelos autovalores da matriz $\mathbf{G}$.

$$\mathbf{B} = \text{diag}\,\{[\lambda_1, \lambda_2, \lambda_3 \ldots \lambda_n]\}, \tag{3.8}$$

onde $\lambda_1, \lambda_2, \lambda_3 \ldots \lambda_n$ representam os autovalores da matriz $\mathbf{G}$.

A matriz que diagonaliza a matriz $\mathbf{G}$ é formada pelos autovetores da própria matriz $\mathbf{G}$, onde cada coluna da matriz $\mathbf{F}$ é formada por um autovetor da matriz $\mathbf{G}$, de acordo com a representação em (3.9).

$$\mathbf{F} = [\boldsymbol{v_1}|\boldsymbol{v_2}|\boldsymbol{v_3}| \ldots |\boldsymbol{v_n}], \tag{3.9}$$

onde $\boldsymbol{v_1}, \boldsymbol{v_2}, \boldsymbol{v_3} \ldots \boldsymbol{v_n}$ representam os autovetores correspondentes à matriz $\mathbf{G}$.

Importante salientar que devem existir *n* autovetores, iguais à ordem da matriz $\mathbf{G}$ e devem ser necessariamente linearmente independentes, pois a matriz $\mathbf{F}$ deve ser inversível.

Quando $\mathbf{F}^{\mathrm{T}} = \mathbf{F}^{-1}$, a matriz que diagonaliza $\mathbf{G}$ é uma matriz ortogonal. Isso quer dizer que os autovetores $\boldsymbol{v_1}, \boldsymbol{v_2}, \boldsymbol{v_3} \ldots \boldsymbol{v_n}$ da matriz $\mathbf{G}$ são mutuamente ortogonais; isto é:

$$\boldsymbol{v_i}.\boldsymbol{v_j} = 0, \text{ para } i \neq j \tag{3.10}$$

Para saber se $\mathbf{G}$ pode ser diagonalizada ortogonalmente, deve-se observar se $\mathbf{G}$ é uma matriz simétrica, ou seja, se $\mathbf{G}^{\mathrm{T}} = \mathbf{G}$. Isso implica em $\mathbf{F}^{\mathrm{T}} = \mathbf{F}^{-1}$.

$$\mathbf{G} = \mathbf{F}\mathbf{B}\mathbf{F}^{\mathrm{T}} \tag{3.11}$$

3.5 – DECOMPOSIÇÃO EM VALORES SINGULARES

Decomposição em valores singulares (SVD) é uma ferramenta de álgebra linear que pode ser utilizada tanto para matriz quadrada quanto retangular, decompondo-a como o produto de matrizes ortogonais e matriz diagonal.

Seja $\mathbf{G} \in \mathbb{R}^{m \times n}$ uma matriz com posto d. Então, existem matrizes ortogonais $\mathbf{U} \in \mathbb{R}^{m \times m}$ (cujas colunas são formadas pelos autovetores de $\mathbf{G}\mathbf{G}^{\mathrm{T}}$) e $\mathbf{V} \in \mathbb{R}^{n \times n}$ (cujas colunas são formadas pelos autovetores de $\mathbf{G}^{\mathrm{T}}\mathbf{G}$) e uma matriz diagonal $\mathbf{\Sigma} \in \mathbb{R}^{m \times n}$, tal que:

$$\mathbf{G} = \mathbf{U}\mathbf{\Sigma}\boldsymbol{V}^{\mathrm{T}}, \tag{3.12}$$

onde a matriz $\mathbf{\Sigma}$ é representada por:

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 & 0 \\ 0 & \sigma_2 & 0 & \vdots & 0 \\ \vdots & 0 & \ddots & 0 & \vdots \\ 0 & \vdots & 0 & \sigma_{n-1} & 0 \\ 0 & 0 & \cdots & 0 & \sigma_n \end{bmatrix} \tag{3.13}$$

$$\mathbf{\Sigma}_{i,j} = \begin{cases} 0 & \text{se } i \neq j \\ \sigma_i & \text{se } i = j \end{cases}, \tag{3.14}$$

com $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \cdots \geq \sigma_d \geq \sigma_{d+1} \geq \cdots \geq \sigma_n$

Os escalares σ_i são chamados valores singulares de $\mathbf{G}$, enquanto os vetores $\boldsymbol{u}_i$ e $\boldsymbol{v}_i$ são chamados vetores singulares à esquerda e à direita de $\mathbf{G}$, respectivamente.

Os valores singulares de $\mathbf{G}$ são iguais às raízes quadradas dos r autovalores de $\mathbf{G}^{\mathrm{T}}\mathbf{G}$ e $\mathbf{G}\mathbf{G}^{\mathrm{T}}$, e são denotados por:

$$\sigma_i = \sqrt{\lambda_i} \tag{3.15}$$

As matrizes $\mathbf{G}^{\mathrm{T}}\mathbf{G} \in \mathbb{R}^{n \times n}$ e $\mathbf{G}\mathbf{G}^{\mathrm{T}} \in \mathbb{R}^{m \times m}$ são simétricas. Devido a isso, os autovetores de $\mathbf{G}^{\mathrm{T}}\mathbf{G}$ são ortogonais entre si, bem como os autovetores de $\mathbf{G}\mathbf{G}^{\mathrm{T}}$ são ortogonais entre si.

Os autovalores de $\mathbf{G}^{\mathrm{T}}\mathbf{G}$ e $\mathbf{G}\mathbf{G}^{\mathrm{T}}$ são não-negativos e os r autovalores positivos das matrizes $\mathbf{G}^{\mathrm{T}}\mathbf{G}$ e $\mathbf{G}\mathbf{G}^{\mathrm{T}}$ são os mesmos e com mesma multiplicidade. Para autovalores iguais a zero, vale a seguinte relação de multiplicidade: $(n - r)$ para $\mathbf{G}^{\mathrm{T}}\mathbf{G}$ e $(m - r)$ para $\mathbf{G}\mathbf{G}^{\mathrm{T}}$.

A Figura 3.1 ilustra geometricamente um exemplo genérico de como se interpretar os autovalores e os autovetores. Os autovetores (**E**) indicam as direções onde se podem obter os maiores valores de variância dos dados que estão sendo analisados. As medidas desses valores de variância são representadas pelos autovalores ($\mathbf{\Lambda}$) obtidos.

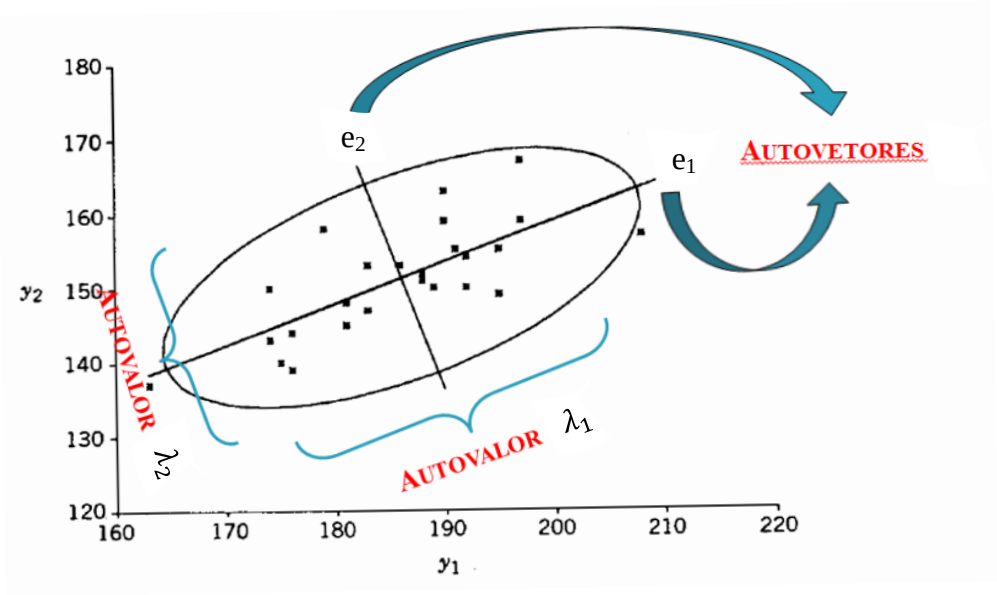


Figura 3.1 - Representação geométrica dos autovalores e autovetores.

3.6 – ESTIMATIVA DA MATRIZ DE COVARIÂNCIA

Estimativa de matrizes de covariância é uma técnica utilizada em diversos ramos do conhecimento, como por exemplo: processamento de sinal, genômica, matemática financeira, reconhecimento de padrões, análise funcional geométrica e geometria computacional (Vershynin, 2012). Um clássico e simples estimador da matriz de covariância é a matriz covariância por média amostral (*sample covariance matrix*), definida por (Krim; Viberg, 1996):

$$\hat{\mathbf{S}}_{xx} = \frac{1}{N}\sum_{t=1}^{N} \mathbf{y}(t)\,\mathbf{y}^{\mathrm{T}}(t), \tag{3.16}$$

ou na forma matricial:

$$\hat{\mathbf{S}}_{xx} = \frac{1}{N}\,\mathbf{Y}\mathbf{Y}^{\mathrm{T}}, \tag{3.17}$$

onde $\boldsymbol{y}(t)$ representa um conjunto de dados em cada instante de tempo t, t = 1,2,3, ..., N. Por questão de simplificação, esta dissertação referir-se-á à matriz $\hat{\mathbf{S}}_{xx}$ apenas por $\mathbf{S}_{xx}$.

4 – ANÁLISE DE COMPONENTES PRINCIPAIS

Análise de Componentes Principais (PCA) é uma técnica de análise multivariada que tem sido amplamente utilizada em diferentes áreas de pesquisa, como: análise de tráfego de *Internet*, economia, processamento de imagem e genética. PCA é utilizada principalmente para reduzir a dimensão de um conjunto de dados, utilizando para isso variáveis não correlacionadas, denominadas de componentes principais (PC). Essa transformação em outro conjunto de variáveis ocorre com a menor perda de informação possível, eliminando apenas algumas variáveis originais que possuam pouca informação (Jollif, 2002).

As componentes principais resultantes são uma combinação linear das variáveis originais, são ortogonais e ordenadas de forma que a primeira componente principal tenha a maior parte da variância dos dados originais. Embora o número resultante de componentes principais seja igual ao número original de variáveis, grande parte da variação no conjunto original pode ser retida pelas primeiras componentes principais, reduzindo dessa forma a dimensão do problema. Assim, pode-se resumir os principais objetivos da PCA: redução de dimensionalidade, determinação de combinações lineares de variáveis, selecionar padrões através da seleção de variáveis mais características e vizualização de dados de multi-dimensão (Cichocki; Zdunek; Phan; Amari, 2009).

Seja $\mathbf{X}$ uma matriz de dados amostrais constituída de p variáveis, observadas simultaneamente n vezes. Dessa forma, pode-se representar a matriz da seguinte forma:

$$\mathbf{X} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{p,1} & \cdots & x_{p,n} \end{bmatrix}_{p \times n} \tag{4.1}$$

Tal matriz também pode ser representada também por um conjunto de vetores, como segue:

$$\mathbf{X} = [\mathbf{x}_1 | \, \mathbf{x}_2 \, | \, \mathbf{x}_3 \, | \, \ldots \, | \, \mathbf{x}_p]^{\mathrm{T}}, \tag{4.2}$$

onde $\mathbf{x}_i^{\mathrm{T}} = (\boldsymbol{x}_{i,1}, \boldsymbol{x}_{i,2}, \boldsymbol{x}_{i,3}, \ldots, \boldsymbol{x}_{i,n})$.

A partir da matriz $\mathbf{X}$ construída é possível se utilizar diversos recursos matemáticos a fim de se fazer inferências sobre a amostra observada.

Seja o vetor $\mathbf{1}_n^{\mathrm{T}} = [1, 1, 1, \ldots, 1]$, então:

$$\mathbf{d}_i = \mathbf{x}_i - \bar{\mathbf{x}}_i\mathbf{1} = \begin{bmatrix} \boldsymbol{x}_{i,1} - \overline{\boldsymbol{x}}_i \\ \boldsymbol{x}_{i,2} - \overline{\boldsymbol{x}}_i \\ \boldsymbol{x}_{i,3} - \overline{\boldsymbol{x}}_i \\ \vdots \\ \boldsymbol{x}_{i,n} - \overline{\boldsymbol{x}}_i \end{bmatrix}, \tag{4.3}$$

onde os elementos de $\mathbf{d}_i$ (vetor desvio) representam os desvios das medidas das *p* variáveis em relação à respectiva média amostral, indicada por $\overline{\boldsymbol{x}}_i$.

Através de manipulações algébricas é possível encontrar a expressão que relaciona o comprimento dos vetores desvio $L_{\boldsymbol{d}_i}$ com os próprios vetores desvio ($\mathbf{d}_i$):

$$L_{\boldsymbol{d}_i}^2 = \mathbf{d}_i{}^{\mathrm{T}}\mathbf{d}_i = \sum_{j=1}^{n}\left(\boldsymbol{x}_{i,j} - \overline{\boldsymbol{x}}_i\right)^2 \tag{4.4}$$

É possível observar em (4.4) que o comprimento ao quadrado do *i*-ésimo vetor desvio é proporcional à variância dos valores de cada *i*-ésima variável. De forma equivalente, o comprimento é proporcional ao desvio padrão.

Para quaisquer dois vetores desvio, $\mathbf{d}_i$ e $\mathbf{d}_k$:

$$\mathbf{d}_i{}^{\mathrm{T}}\mathbf{d}_k = \sum_{j=1}^{n}\left(\boldsymbol{x}_{i,j} - \overline{\boldsymbol{x}}_i\right)\left(\boldsymbol{x}_{k,j} - \overline{\boldsymbol{x}}_k\right) \tag{4.5}$$

Denotando $\theta_{i,k}$ o ângulo formado entre os vetores $\mathbf{d}_i$ e $\mathbf{d}_k$, pode-se aplicar o produto escalar entre eles e encontrar:

$$\mathbf{d}_i{}^{\mathrm{T}}\mathbf{d}_k = L_{\boldsymbol{d}_i}L_{\boldsymbol{d}_k}\cos(\theta_{i,k}) \tag{4.6}$$

Utilizando (4.4) e (4.5) em (4.6), obtém-se:

$$\sum_{j=1}^{n}\left(\boldsymbol{x}_{i,j} - \overline{\boldsymbol{x}}_i\right)\left(\boldsymbol{x}_{k,j} - \overline{\boldsymbol{x}}_k\right) = \sqrt{\sum_{j=1}^{n}\left(\boldsymbol{x}_{i,j} - \overline{\boldsymbol{x}}_i\right)^2}\sqrt{\sum_{j=1}^{n}\left(\boldsymbol{x}_{k,j} - \overline{\boldsymbol{x}}_k\right)^2}\cos(\theta_{i,k}) \tag{4.7}$$

Então:

$$\cos(\theta_{i,k}) = \frac{\frac{\sum_{j=1}^{n}\left(\boldsymbol{x}_{i,j} - \overline{\boldsymbol{x}}_i\right)\left(\boldsymbol{x}_{k,j} - \overline{\boldsymbol{x}}_k\right)}{N}}{\sqrt{\frac{\sum_{j=1}^{n}\left(\boldsymbol{x}_{i,j} - \overline{\boldsymbol{x}}_i\right)^2}{N}}\sqrt{\frac{\sum_{j=1}^{n}\left(\boldsymbol{x}_{k,j} - \overline{\boldsymbol{x}}_k\right)^2}{N}}} = \frac{s_{i,k}}{\sqrt{s_{i,i}}\sqrt{s_{k,k}}} = r_{i,k}, \tag{4.8}$$

onde $r_{i,k}$ representa o coeficiente de correlação amostral e $s_{i,k}$ representa a covariância amostral, e é igual a:

$$s_{i,k} = \frac{1}{N}\sum_{j=1}^{n}\left(\boldsymbol{x}_{i,j} - \overline{\boldsymbol{x}}_i\right)\left(\boldsymbol{x}_{k,j} - \overline{\boldsymbol{x}}_k\right) \tag{4.9}$$

O cosseno do ângulo é o coeficiente de correlação ($r_{i,k}$) entre dois vetores de amostra. Assim, se dois vetores desvio têm quase a mesma orientação, a correlação da amostra será próxima de 1. Se os dois vetores são quase perpendicular, a correlação da amostra será de aproximadamente zero. Se os dois vetores são orientados em direções quase opostas, a correlação da amostra será próxima de -1 (Johnson; Wichern, 2007).

Generalizando, a matriz de covariância é expressa da seguinte forma:

$$\mathbf{S}_{xx} = \begin{bmatrix} s_{1,1} & \cdots & s_{1,p} \\ \vdots & \ddots & \vdots \\ s_{p,1} & \cdots & s_{p,p} \end{bmatrix}_{p\times p} \tag{4.10}$$

A matriz de correlação é expressa por:

$$\mathbf{R}_{xx} = \begin{bmatrix} \dfrac{s_{1,1}}{\sqrt{s_{1,1}}\sqrt{s_{1,1}}} & \cdots & \dfrac{s_{1,p}}{\sqrt{s_{1,1}}\sqrt{s_{p,p}}} \\ \vdots & \ddots & \vdots \\ \dfrac{s_{p,1}}{\sqrt{s_{p,p}}\sqrt{s_{1,1}}} & \cdots & \dfrac{s_{p,p}}{\sqrt{s_{p,p}}\sqrt{s_{p,p}}} \end{bmatrix}_{p\times p} \tag{4.11}$$

Matricialmente pode-se obter a matrizes de correlação fazendo inicialmente uma troca de variável, obtendo uma nova matriz $\mathbf{C}$ a partir de $\mathbf{X}$, da seguinte forma:

$$\boldsymbol{c}_i = \frac{\boldsymbol{x}_i - \overline{\boldsymbol{x}}_i}{\sigma_i}, \tag{4.12}$$

onde σ_i denota o desvio padrão associado ao vetor $\mathbf{x}_i$.

Assim, a matriz $\mathbf{C}$ pode ser representada como segue:

$$\mathbf{C} = [\mathbf{c}_1 | \, \mathbf{c}_2 \, | \, \mathbf{c}_3 \, | \, \ldots \, | \, \mathbf{c}_p]^{\mathrm{T}} = \begin{bmatrix} \dfrac{\boldsymbol{x}_{1,1} - \overline{\boldsymbol{x}}_1}{\sigma_1} & \cdots & \dfrac{\boldsymbol{x}_{1,n} - \overline{\boldsymbol{x}}_1}{\sigma_1} \\ \vdots & \ddots & \vdots \\ \dfrac{\boldsymbol{x}_{p,1} - \overline{\boldsymbol{x}}_p}{\sigma_p} & \cdots & \dfrac{\boldsymbol{x}_{p,n} - \overline{\boldsymbol{x}}_p}{\sigma_p} \end{bmatrix}_{p\times n} \tag{4.13}$$

A matriz de correlação $\mathbf{R}_{xx}$ pode ser obtida através da seguinte expressão:

$$\mathbf{R}_{xx} = \frac{1}{N}\,\mathbf{C}\mathbf{C}^{\mathrm{T}} = \begin{bmatrix} 1 & \cdots & \frac{s_{1,p}}{\sigma_1\sigma_p} \\ \vdots & \ddots & \vdots \\ \frac{s_{p,1}}{\sigma_p\sigma_1} & \cdots & 1 \end{bmatrix}_{p\times p} \tag{4.14}$$

Para a matriz de covariância, basta fazer $\boldsymbol{z}_i = \boldsymbol{x}_i - \overline{\boldsymbol{x}}_i$, sem dividir pelo desvio padrão:

$$\mathbf{Z} = [\boldsymbol{z}_1 \mid \boldsymbol{z}_2 \mid \boldsymbol{z}_3 \mid \ldots \mid \boldsymbol{z}_p]^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{x}_{1,1} - \overline{\boldsymbol{x}}_1 & \cdots & \boldsymbol{x}_{1,n} - \overline{\boldsymbol{x}}_1 \\ \vdots & \ddots & \vdots \\ \boldsymbol{x}_{p,1} - \overline{\boldsymbol{x}}_p & \cdots & \boldsymbol{x}_{p,n} - \overline{\boldsymbol{x}}_p \end{bmatrix}_{p\times n} \tag{4.15}$$

A matriz de covariância $\mathbf{S}_{xx}$ pode ser obtida através da seguinte expressão:

$$\mathbf{S}_{xx} = \frac{1}{N}\,\mathbf{Z}\mathbf{Z}^{\mathrm{T}} = \begin{bmatrix} s_{1,1} & \cdots & s_{p,1} \\ \vdots & \ddots & \vdots \\ s_{p,1} & \cdots & s_{p,p} \end{bmatrix}_{p\times p} \tag{4.16}$$

Geralmente, a extração de componentes a partir de $\mathbf{S}_{xx}$, em vez de $\mathbf{R}_{xx}$, permanece perto do espírito e da intenção da análise de componentes principais. No entanto, em alguns casos, as componentes principais serão mais interpretáveis se $\mathbf{R}_{xx}$ for usado. Por exemplo, se as variâncias diferem muito umas das outras ou se as unidades de medida não são comensuráveis, as componentes principais de $\mathbf{S}_{xx}$ serão dominadas pelas variáveis com maiores variâncias. As demais variáveis contribuirão muito pouco. Para uma representação mais equilibrada, em tais casos, podem ser utilizadas as componentes de $\mathbf{R}_{xx}$. Assim, deve-se escolher entre as componentes principais de $\mathbf{S}_{xx}$ ou $\mathbf{R}_{xx}$, sabendo que haverá interpretação distinta, dependedo da escolha (Rencher, 2002).

Tal como acontece com qualquer mudança de escala, em que as variáveis são padronizadas na transformação de $\mathbf{S}_{xx}$ para $\mathbf{R}_{xx}$, a forma da representação dos pontos no espaço mudará. No entanto, depois de transformar para $\mathbf{R}_{xx}$, quaisquer futuras mudanças de escala sobre as variáveis não afetaria as componentes, pois as alterações de escala não mudam $\mathbf{R}_{xx}$. Assim, as componentes principais de $\mathbf{R}_{xx}$ são invariantes quanto à escala (Rencher, 2002).

Observa-se de (4.14) e (4.16), fazendo referência ao abordado na Seção 3.5, que os autovetores de (4.14) e (4.16) são ortogonais e, por isso, descorrelacionados. Tais autovetores representam as componentes principais e uma pequena parte desses retém a maior variância dos dados originais, atentando para as diferenças entre se escolher a matriz de covariância ou a matriz de correlação. Os autovalores de (4.14) e (4.16) são não-negativos e representam a variância associada a cada componente principal.

5 – SELEÇÃO DE ORDEM DO MODELO

Em muitas aplicações de processamento digital de sinais, incluindo radar, sonar, comunicações, modelagem do canal, imagiologia médica, dentre outros, a seleção de ordem do modelo é um passo fundamental. Ela permite separar, por exemplo, os componentes de ruído dos componentes principais, aplicando redução do posto dos dados analizados. Para muitas técnicas de estimativa de parâmetros como um modelo de redução de posto, esta abordagem é crucial (da Costa; Thakre; Roemer; Haardt, 2009).

Seleção de ordem do modelo é uma parte importante dos estudos matemáticos de inferência. A partir de alguns dados, escolhe-se um modelo em que se acredita que o mesmo é o correto para descrever os dados em questão. O procedimento de seleção de modelos escolhe o "melhor" modelo de um conjunto finito de modelos tal que algum critério seja satisfeito (Rajan; Rayner, 1997).

A determinação do número de parâmetros de um modelo, a fim de representar um conjunto de dados, ou a seleção ordem do modelo (MOS), é uma tarefa fundamental em processamento de sinais. Há diversos esquemas de MOS, dentre eles: Akaike (AIC), *Minimum Description Length* (MDL), *Bayesian Information Criterion* (BIC), *Maximum a Posteriori* (MAP), *Kullback Information Criterion* (KIC), *Conditional Model Order Estimator* (CME) e *Exponentially Embedded Families* (EEF) (Dan; Ming; Tao; Rong, 2009).

O estado da arte referente a técnicas de estimativa de ordem do modelo com base em autovalores incluem: *Akaike's Information Theoretic Criterion*, AIC (Akaike, 1974) (Wax; Kailath, 1985); *Minimum Description Length*, MDL (da Costa; Thakre; Roemer; Haardt, 2009) (Wax; Kailath, 1985); *Efficient Detection Criterion*, EDC (Zhao; Krishnaiah; Bai, 1986); *Stein's Unbiased Risk Estimator*, SURE (Ulfarsson; Solo, 2008); RADOI (Radoi; Quinquis, 2004); e *Exponential Fitting Test*, EFT (Quinlan; Barbot; Larzabal; Haardt, 2007) (da Costa; Roemer; Castro Junior; Ramos; Schwarz; Sabirova, 2011).

Nos esquemas AIC, MDL e EDC, o critério de informação é uma função da média geométrica $g(k)$ e da média aritmética $a(k)$, relacionados aos k menores autovalores de (4.14) ou (4.16) e k é um valor candidato para d (ordem do modelo) (da Costa; Thakre; Roemer; Haardt, 2009).

Basicamente, a diferença entre os esquemas AIC, MDL e EDC é a função de penalidade $p(k, N, \alpha)$, por isso essas técnicas podem ser escritas de uma forma geral como (da Costa; Thakre; Roemer; Haardt, 2009):

$$\hat{d} = \arg\min_k J(k), \tag{5.1}$$

onde

$$J(k) = -N(\alpha - k)\log\left(\frac{g(k)}{a(k)}\right) + p(k, N, \alpha), \tag{5.2}$$

onde $\hat{d}$ representa uma estimativa da ordem do modelo d; N representa a quantidade de amostras; $\alpha = M$, a quantidade de variáveis do problema; $0 \leq k \leq \min[M, N]$; e as funções de penalidades para AIC, MDL e EDC são dadas por, respectivamente: $p(k, N, \alpha) = k(2\alpha - k)$, $p(k, N, \alpha) = \frac{1}{2}k(2\alpha - k)\log(N)$ e $p(k, N, \alpha) = \frac{1}{2}k(2\alpha - k)\sqrt{N\ln(\ln N)}$.

Tabela 5.1 - Funções de penalidade para os esquemas AIC, MDL e EDC.

Esquema	**Função de penalidade** $p(k,\ N, \alpha)$
AIC	$k(2\alpha - k)$
MDL	$0.5k(2\alpha - k)\ log(N)$
EDC	$0.5k(2\alpha - k)\sqrt{N\ ln(lnN)}$

O esquema RADOI é um modelo empírico, representado por:

$$\hat{d} = \arg\min_k \text{RADOI}(k), \tag{5.3}$$

onde

$$\text{RADOI}(k) = \lambda_{k+1}\left(\sum_{i=2}^{M}\lambda_i\right)^{-1} - \xi_k\left(\sum_{i=1}^{M-1}\xi_i\right)^{-1}, \tag{5.4}$$

onde

$$\xi_k = 1 - \frac{\alpha(\lambda_k - \mu_k)}{\mu_k}, \mu_k = \frac{1}{M-k}\sum_{i=k+1}^{M}\lambda_i \text{ e } \alpha = \left[\arg\max_k \frac{(\lambda_k - \mu_k)}{\mu_k}\right]^{-1}, \tag{5.5}$$

O esquema Exponential Fitting Test (EFT) pode ser efetivamente usado em casos quando o número de amostras N é pequeno. Esta técnica é baseada em observações que, num caso

apenas de ruído, o perfil dos autovalores pode ser aproximado por uma exponencial decadente (da Costa; Haardt; Römer; Del Galdo, 2007).

Dado λ_i ser o i-ésimo autovalor de (4.14) ou (4.16), o modelo exponencial pode ser expresso por:

$$\mathrm{E}\{\lambda_i\} = \mathrm{E}\{\lambda_1\} \cdot q(\alpha, \beta)^{i-1}, \tag{5.6}$$

onde $\mathrm{E}\{\cdot\}$ é o operador esperança, e considera-se que os autovalores estão ordenados de forma que λ_1 representa o maior autovalor. O termo $q(\alpha, \beta)$ é definido como:

$$q(\alpha, \beta) = \exp\left\{-\sqrt{\frac{30}{\alpha^2+2} - \sqrt{\frac{900}{(\alpha^2+2)^2} - \frac{720\alpha}{\beta(\alpha^4+\alpha^2-2)}}}\right\}, \tag{5.7}$$

de modo que: $0 < q(\alpha, \beta) < 1$. De acordo com (Quinlan; Barbot; Larzabal; Haardt, 2007), supondo que $M \leq N$, então $\alpha = M \; e \; \beta = N$.

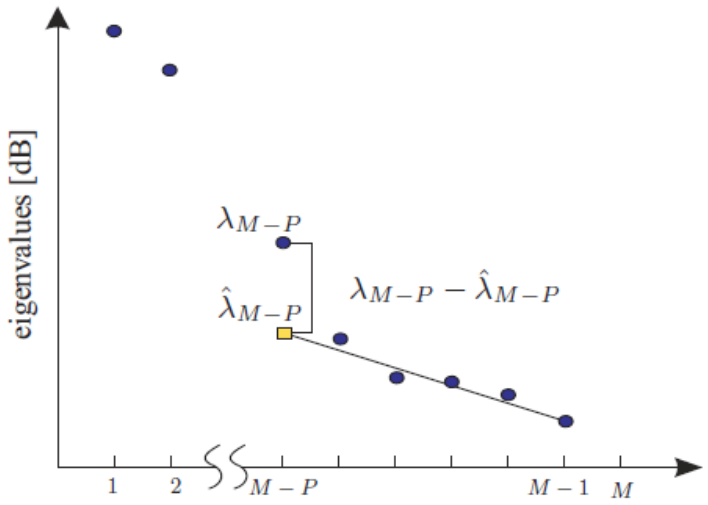


Figura 5.1 - Exemplo de aplicação do esquema EFT (da Costa; et al, 2007).

A Figura 5.1 apresenta um perfil típico de autovalores. Os últimos P – 1 autovalores são usados para estimar o (M – P)-ésimo autovalor, denotado pelo retângulo amarelo. O método EFT considera a discrepância entre o valor real obtido e o valor estimado (J. P. C. L. da Costa; M. Haardt; A. Thakre; F. Röemer; G. D. Galdo, 2007).

No esquema SURE, o risco $\hat{R}(k)$ deve ser minimizado de acordo com a expressão abaixo.

$$\hat{d} = \arg\min_k \hat{R}(k), \tag{5.8}$$

onde

$$\hat{R}(k) = (M-k)\hat{\sigma}_k^2 + 2\sigma^2 k + \tag{5.9}$$

$$+\left(\hat{\sigma}_{k_k}{}^4 - 2\hat{\sigma}_k^2\sigma^2 + \frac{4\hat{\sigma}_k^2\sigma^2}{N}\right)\sum_{i=1}^{k}\frac{1}{\lambda_i} + \frac{4\sigma^2}{N}\sum_{i=1}^{k}\sum_{j=k+1}^{M}\frac{\lambda_i - \hat{\sigma}_k^2}{\lambda_i - \lambda_j} + \frac{2\sigma^2}{N}k(k-1) -$$

$$-\frac{2\sigma^2}{N}(M-1)\sum_{i=1}^{k}\left(1 - \frac{\hat{\sigma}_k^2}{\lambda_i}\right)$$

onde

$$\hat{\sigma}_k^2 = \frac{1}{M-k}\sum_{i=r+1}^{M}\lambda_i \tag{5.10}$$

Um fluxograma genérico pode ser utilizado para se obter a ordem do modelo para problemas envolvendo PCA e consequentemente problemas envolvendo o cálculo de autovalores, conforme apresentado na Figura 5.2.

O fluxograma inicia-se com a obtenção da matriz $\mathbf{X} \in \mathbb{R}^{M\times N}$, onde *M* representa o número de variáveis e *N* a quantidade de amostras.

Dependendo do tipo de problema, a normalização será feita com o intuito de se obter a matriz de covariância $\mathbf{S}_{xx}^{(q)}$ ou a matriz de correlação $\mathbf{R}_{xx}^{(q)}$.

Uma vez obtidas as matrizes $\mathbf{S}_{xx}^{(q)}$ e $\mathbf{R}_{xx}^{(q)}$, procede-se com a decomposição em autovalores (EVD), a fim de se obter os autovalores associados a cada matriz.

Com os autovalores calculados, pode-se utilizar os esquemas de seleção de ordem do modelo e estimar a ordem do modelo $\hat{d}$. Caso a ordem do modelo estimada seja igual ao valor real da ordem do modelo, descobre-se qual esquema de MOS aplica-se ao problema. Pode-se encontrar mais de um esquema que se aplique ao problema, não necessariamente um.

Após se encontrar os esquemas de MOS que se aplicam ao problema em estudo, o fluxograma é finalizado.

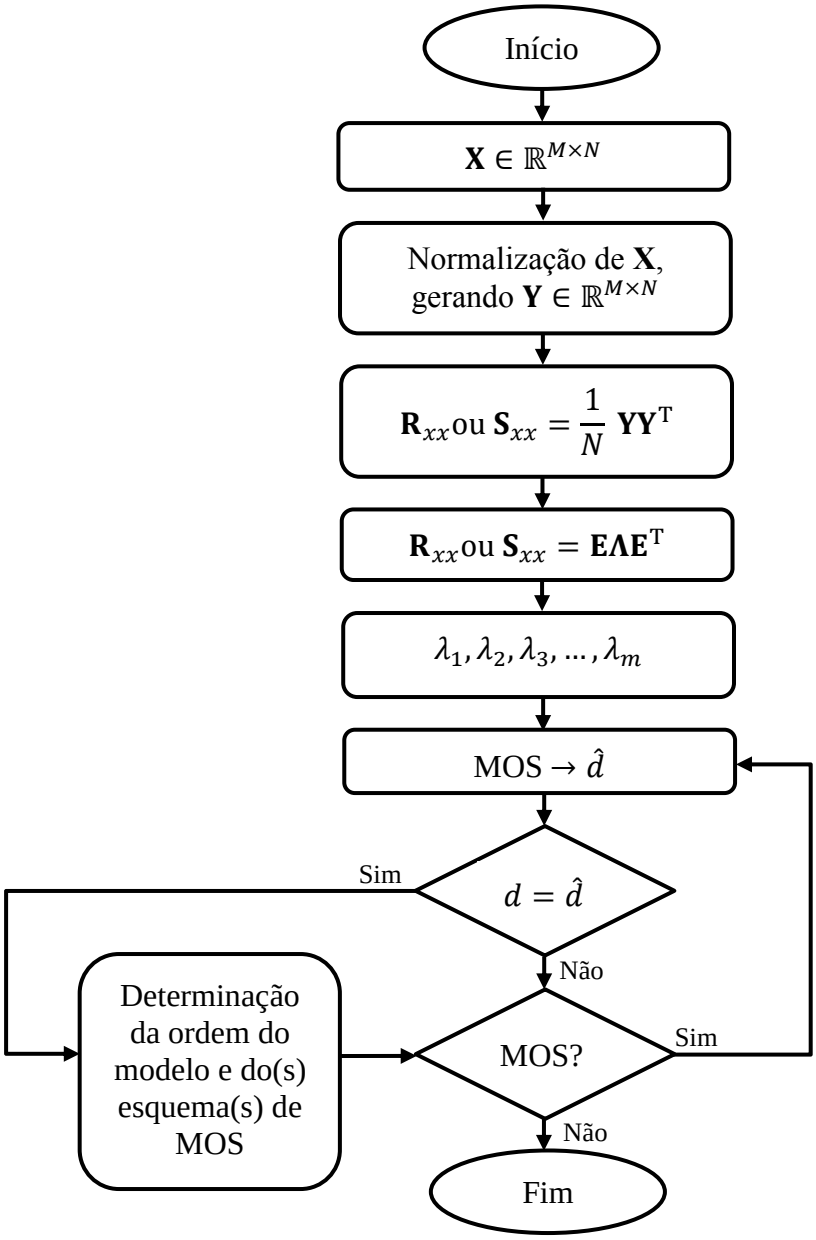


Figura 5.2 - Fluxograma genérico para MOS de um problema.

6 – SOLUÇÃO PROPOSTA

Nesta seção são apresentados detalhes associados aos ataques estudados nesta dissertação, à modelagem dos dados e por fim é mostrada de forma rigorosa toda sequência que foi empregada para a realização da detecção dos ataques. Parte do que foi exposto nas seções anteriores é utilizado como embasamento para esta seção, como por exemplo: conceitos, definições diversas e expressões matemáticas. Devido a isso, frequentemente são feitas referências a essas seções, para situar o leitor melhor em relação a algum conceito ou expressão que se esteja utilizando.

6.1 – ATAQUES DE SYNFLOOD, FRAGGLE E PORTSCAN

Os ataques objetos de estudo deste trabalho são*: synflood*, *fraggle* e *portscan*. Os dois primeiros ataques são classificados como ataques de negação de serviço, discutidos na Seção 2.3.3, enquanto o último voltado para o escaneamento de portas.

6.1.1 – Synflood

Pelo fato do protocolo TCP ser um protocolo orientado à conexão, é configurada uma conexão virtual entre dois computadores, quando o mesmo é utilizado. Esta conexão virtual necessita de um "aperto de mão" (*handshaking*) e ocorre em três vias. Se um computador pretende se comunicar com outro computador, o solicitante da comunicação enviará um pacote de sincronização (SYN) para uma porta específica no destino, que está em um estado de escuta. Se o destino está ativo, funcionando e aceitando solicitações, ele responderá ao solicitante com uma mensagem de confirmação SYN/ACK. Depois de receber essa mensagem, o solicitante enviará uma mensagem de ACK para o destino e a conexão será estabelecida.

A Figura 6.1 representa o ataque de *synflood*, que foi realizado durante as simulações. Num intervalo de tempo de dez minutos houve mais de 210.000 pacotes relacionados ao ataque, um tráfego incomum numa rede de dados, especialmente pelo fato de ser concentrado num curto período de tempo.

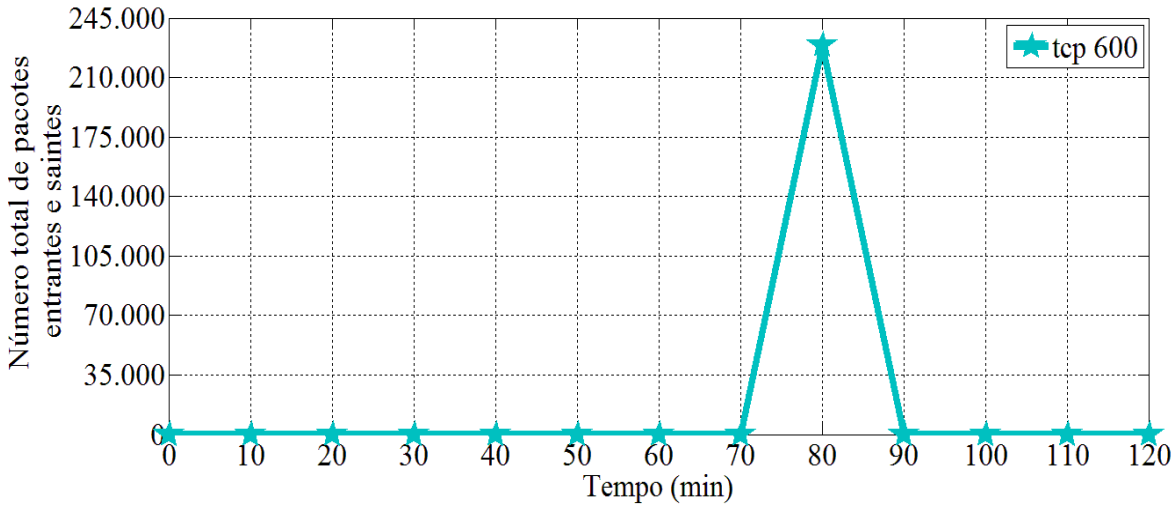


Figura 6.1 - Tráfego malicioso (*synflood*).

O ataque de *synflood* ocorre quando atacantes enviam falsas solicitações SYN para servidores em uma tentativa de esgotar a quantidade de pedidos legítimos que o servidor pode manter. Se o atacante for bem sucedido, o servidor irá esperar por um determinado período de tempo para que os pedidos falsos possam se completar (o que, obviamente, nunca irão ocorrer) e a maioria dos pedidos legítimos será ignorada pelo servidor (Harris, 2010).

6.1.2 – Fraggle

Neste tipo de ataque, um grande tráfego de pacotes com segmentos "UDP echo" é enviado para o endereço IP de *broadcast* da rede, tendo como origem o endereço de IP da vítima (IP *spoofing*). Com o *broadcast*, cada *host* da rede recebe uma enorme quantidade de requisições de "UDP echo", passando todos eles a responderem para o endereço de origem, que é o falsificado, o endereço IP da vítima. Esse ataque consegue afetar toda a rede, pois todos os seus *hosts* recebem diversas requisições "UDP echo" e respondem com o protocolo ICMP, passando cada um a atuar como um "amplificador" do ataque, em relação ao *host* que teve o IP falsificado. Assim, a vítima que teve o seu endereço IP falsificado, recebe os pacotes de todos esses *hosts*, ficando impossibilitada de executar suas atividades normais, sofrendo assim uma negação de serviço. Essa última parte do ataque não será levada em consideração neste trabalho, pois a vítima recebe pacotes ICMP (camada de rede), oriundos dos *hosts* que foram atacados com a inundação de pacotes "UDP echo". Isso devido ao fato do protocolo UDP não ter condições por si só de saber se o segmento enviado atingiu seu destino (modificado – Geus; Nakamura, 2010).

Outro ataque muito semelhante ao *fraggle* é o *smurf*, a única diferença é que com o *smurf* são enviados uma grande quantidade de pacotes "ICMP echo", ao invés de "UDP echo", para o endereço IP de *broadcast*. O *smurf* não é tratado nessa dissertação pelo fato de ser um ataque da camada de rede.

A Figura 6.2 representa o ataque de *fraggle*, que foi realizado durante as simulações. Num intervalo de tempo de dez minutos houve mais de 6.000.000 pacotes relacionados ao ataque, um tráfego incomum numa rede de dados, especialmente pelo fato de ser concentrado num curto período de tempo.

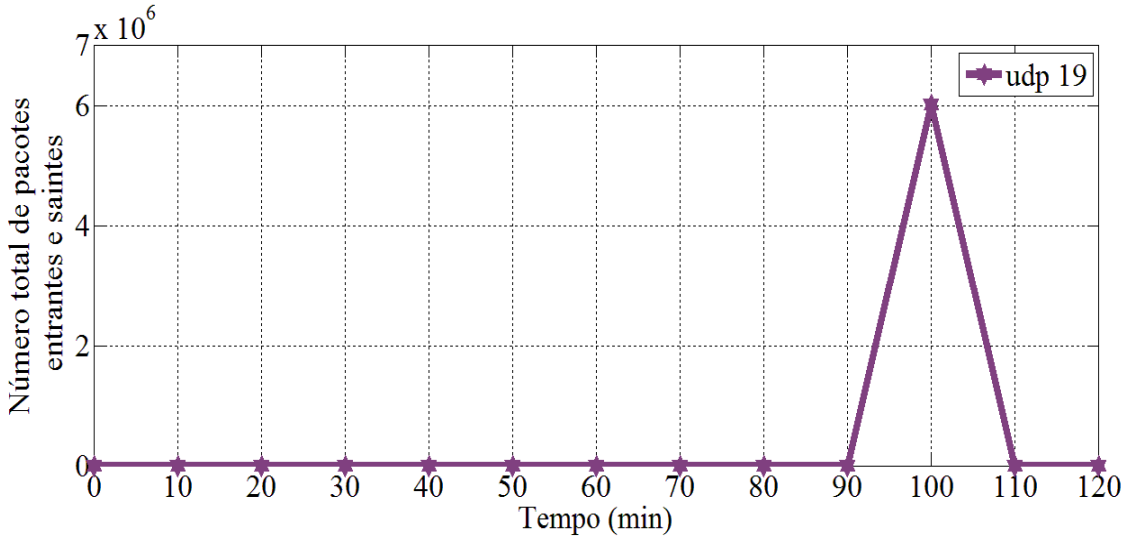


Figura 6.2 - Tráfego malicioso (*fraggle*).

6.1.3 – Portscan

Port scanning ou *portscan* é o processo de se conectar a portas TCP e UDP nos alvos de interesse, a fim de determinar quais serviços estão funcionando ou quais estão no estado de *listening*. Identificar portas no estado de *listening* é crucial para determinar o tipo de sistema operacional da vítima e aplicações em uso (modificado – CERT.br, 2010).

Há diversas técnicas disponíveis de escaneamento, dentre elas: TCP SYN *scan*, TCP ACK *scan*, UDP *scan*, etc. Nesta dissertação é feito o uso dos *scanners* TCP SYN *scan* e UDP *scan*.

A técnica de TCP SYN *scan* é chamada de *half-open scanning*, pois não é feita uma conexão TCP completa. Nesse escaneamento, um pacote SYN é enviado para a porta de destino, podendo ocorrer dois tipos de resposta: um pacote SYN/ACK ser recebido, ou um pacote RST/ACK ser recebido. No primeiro caso, a porta de destino está no estado

listening; no segundo caso, a porta de destino não está no estado *listening*. Neste tipo de escaneamento, um pacote RST/ACK será enviado pelo sistema que está realizando o *portscan*, ao término do escaneamento de cada porta; dessa forma, uma conexão completa nunca é estabelecida. Isso faz com que a origem do ataque seja mais difícil de ser detectada, uma vez que ela não é registrada no sistema de destino (CERT.br, 2010).

A técnica de UDP *scan* envia pacotes UDP para a porta de destino. Caso a porta responda com uma mensagem de "ICMP *port unreachable*", a porta está fechada. Se uma mensagem não é recebida então se deduz que a porta está aberta. UDP é conhecido como um protocolo sem conexão e a eficácia desta técnica é dependente de muitos fatores relacionados à utilização da rede e dos recursos dos sistemas. Esse tipo de escaneamento é também muito lento e pode produzir resultados incertos (CERT.br, 2010).

A Figura 6.3 representa o ataque de *portscan* que foi simulado, onde se pode observar que o mesmo é composto de dois pacotes para cada porta TCP e um pacote para cada porta UDP. Esses resultados práticos estão em perfeita consonância com o que foi explicado no início desta seção. Importante notar a alta correlação dos tráfegos TCP e UDP, em separado, uma vez que os tráfegos referentes às portas TCP são iguais e os tráfegos referentes às portas UDP também são iguais. A igualdade dos tráfegos aqui citada refere-se à quantidade de pacotes entrantes e saintes para cada porta.

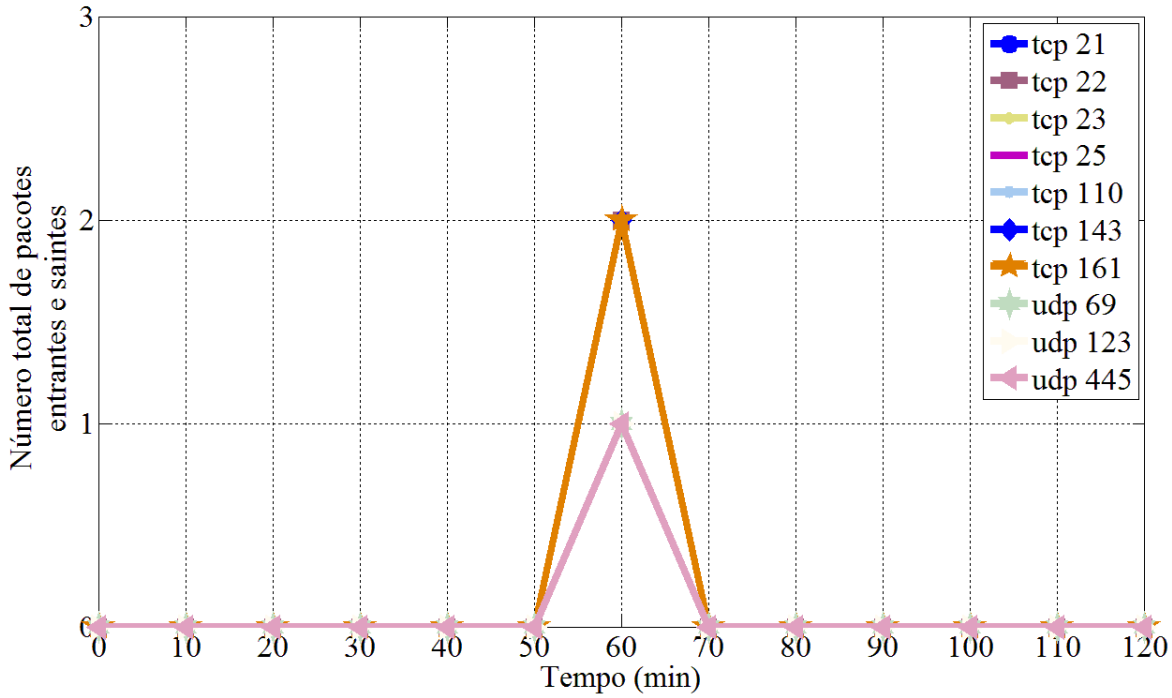


Figura 6.3 - Tráfego malicioso (*portscan*).

6.2 – MODELAGEM DOS DADOS

Antes de se partir para a coleta de milhares de dados, ou tentar aplicar expressões matemáticas a fim de se obter algum resultado, é necessário saber como o problema será modelado, quais são as variáveis, como elas se relacionam, etc. Tal modelagem segue um modelo genérico, podendo ser aplicado a outros problemas também.

O tráfego de rede pode ser caracterizado como a composição de três tráfegos: tráfego legítimo, ruído e tráfego malicioso; de acordo com a expressão abaixo:

$$\mathbf{X} = \mathbf{S} + \mathbf{N} + \mathbf{A}, \tag{6.1}$$

onde **X** é a matriz que representa o tráfego total, **S** é a matriz que representa o tráfego legítimo, **N** representa o ruído e **A** representa o tráfego malicioso.

Pode-se representar (6.1) através de outra notação, com o intuito de definir vários **X**, **S**, **N** e **A**, para intervalos de tempo distintos.

$$\mathbf{X}^{(q)} = \mathbf{S}^{(q)} + \mathbf{N}^{(q)} + \mathbf{A}^{(q)}, \tag{6.2}$$

onde *q* representa o *q*-ésimo período de tempo.

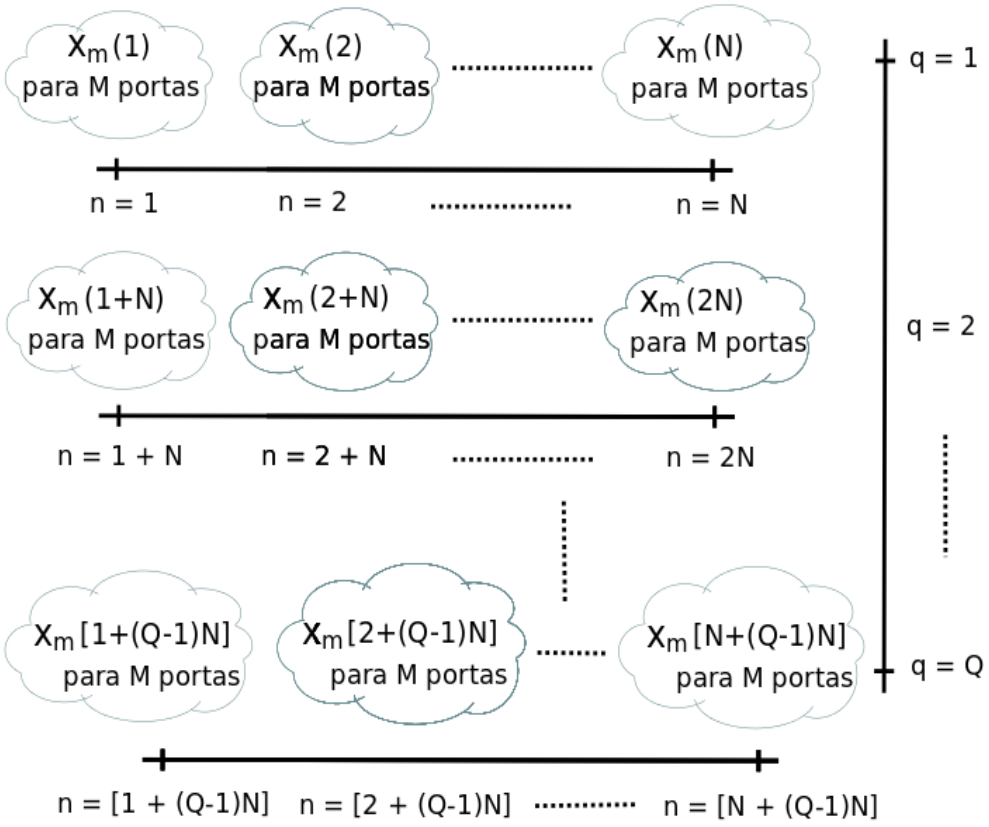


Figura 6.4 - Obtenção da matriz de tráfego $\mathbf{X}^{(q)}$.

Dessa forma, os dados coletados foram divididos em q períodos de tempo de N amostras cada, onde cada amostra é coletada em um determinado instante, segundo um período de amostragem.

A matriz $\mathbf{X}^{(q)} \in \mathbb{R}^{M \times N}$ é formada por M linhas e N colunas, onde cada linha é representada por uma variável, neste caso uma porta de comunicação (porta TCP ou porta UDP), e cada coluna um instante de tempo. Cada elemento $x_{m,n}^{(q)}$ representa o número de vezes que a porta m aparece no n-ésimo instante, dentro do q-ésimo período de tempo.

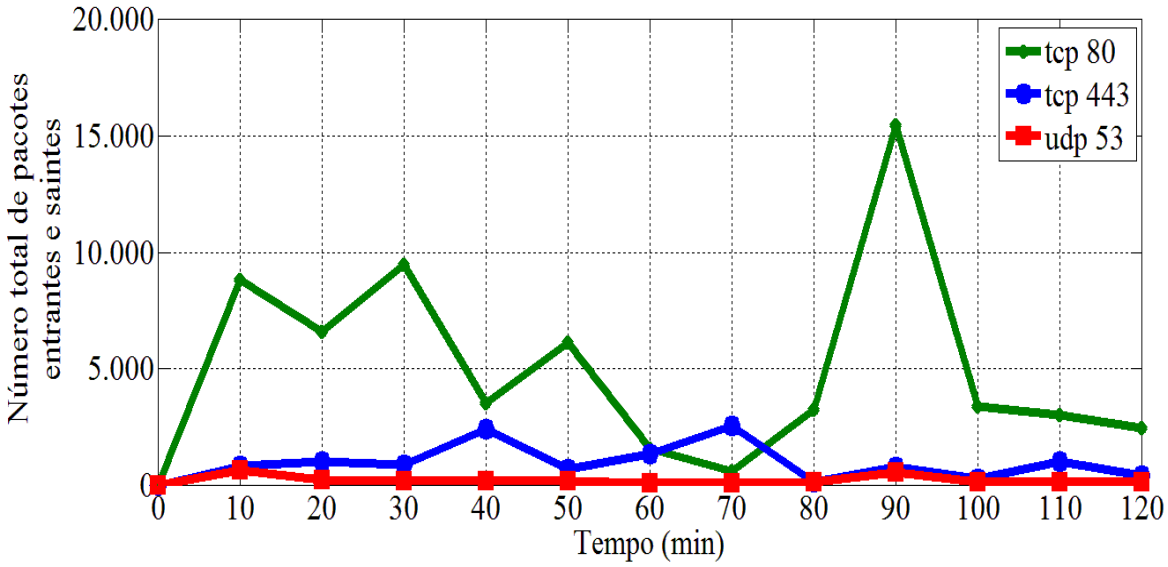


Figura 6.5 - Tráfego do sinal legítimo.

O tráfego legítimo $\mathbf{S}^{(q)}$ é caracterizado pelo tráfego associado diretamente às operações realizadas pelo usuário. Quando um usuário acessa uma página *web*, por exemplo, existe o tráfego TCP/IP relacionado à requisição da página bem como ao tráfego devido à resolução de nomes (DNS). A Figura 6.5 apresenta o tráfego legítimo obtido durante as simulações. Considera-se como ruído $\mathbf{N}^{(q)}$ todo tráfego que não está diretamente associado a operações realizadas pelo usuário, mas que também não é um tráfego malicioso. Um exemplo de ruído é o serviço de aquisição automática de endereço lógico de rede IP (DHCP). Independente de qualquer operação do usuário, a máquina do mesmo receberá um endereço IP, desde que esteja configurada para isso. A Figura 6.6 apresenta o ruído obtido durante as simulações.

Importante salientar o fato de que não se pode considerar neste trabalho ruídos associados a outras camadas do modelo OSI que não sejam da camada de transporte, uma vez que esta

é a camada de estudo dessa dissertação, tanto para o tráfego legítimo, como para o tráfego malicioso, quanto para o ruído.

O tráfego proveniente de uma atividade maliciosa é representado pela matriz $\mathbf{A}^{(q)}$. Neste trabalho é considerado apenas o tráfego oriundo de ataques de inundação, com objetivo de causar negação de serviço e ataques de escaneamento de portas. Se o posto$\{\mathbf{A}^{(q)}\} \neq 0$, existe tráfego malicioso; por outro lado, se posto$\{\mathbf{A}^{(q)}\} = 0$, não há tráfego malicioso. Essa dissertação mostra a forma como se detectar o posto$\{\mathbf{A}^{(q)}\}$ dado apenas a matriz $\mathbf{X}^{(q)}$.

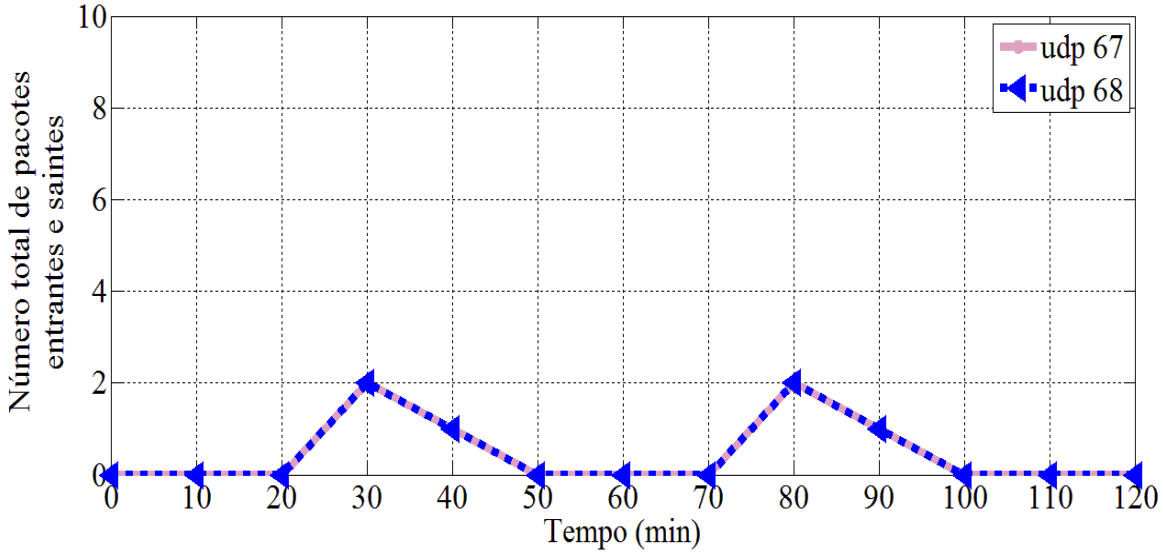


Figura 6.6 - Tráfego do ruído.

6.3 – AMOSTRAGEM DOS DADOS

Nesta seção são discutidas as formas como foram coletados os dados de amostragem, bem como os mesmos foram manipulados a fim de se conseguir as informações de interesse.

6.3.1 – Coleta dos dados

A coleta de dados é fundamental para compor, com amostras, as matrizes apresentadas na Seção 6.2. Tal coleta foi feita utilizando-se a ferramenta de captura e análise de tráfego *tcpdump*.

O *tcpdump* é o melhor analisador de tráfego em modo texto que existe. Ele é baseado na *libpcap*, uma poderosa API para a captura de pacotes de rede durante seu tráfego. Assim, o *tcpdump* mostra as conexões estabelecidas e o tráfego correspondente (Eriberto, 2013).

Tabela 6.1 - Exemplos de uso do *tcpdump* (Eriberto, 2013).

Comando	Resultado
tcpdump -n	Mostra todo o tráfego de rede que passa pela interface listada com #tcpdump -D, sem resolver nomes. Isso permite a visualização do tráfego em tempo real.
tcpdump -nAi eth1 udp	Mostra todo o tráfego UDP no adaptador *eth1*, incluindo o *payload* (área de dados) em ASCII, sem resolver nomes.
tcpdump -n host 10.1.1.25 and udp	Mostra o cabeçalho de todo o tráfego que envolva o *host* 10.1.1.25 e que seja UDP, sem resolver nomes.
tcpdump -n host 10.1.1.25 and udp and port 53	Mostra o cabeçalho de todo o tráfego que envolva o *host* 10.1.1.25, que seja UDP e que tenha como origem ou destino a porta 53, sem resolver nomes.
tcpdump -ne host 10.1.1.25 and udp and port ! 53	Mostra o cabeçalho de todo o tráfego que envolva o *host* 10.1.1.25, que seja UDP e esteja relacionado a qualquer porta, exceto a 53, sem resolver nomes. Também será mostrado o cabeçalho referente à camada 2 do Modelo OSI (enlace).
tcpdump -n tcp and '(port 80 or src port 110)'	Mostra o cabeçalho de todo o tráfego TCP que seja oriundo ou destinado à porta 80 ou apenas oriundo da porta 110, sem resolver nomes.
tcpdump -n icmp and net 10.1.0.0/16	Mostra os pacotes IMCP referentes a qualquer *host* que pertença à rede 10.1.0.0/16, sem resolver nomes.
tcpdump -n ether host 00:ff:31:22:2d:11	Mostra o cabeçalho de todo o tráfego referente ao *host* que possua o endereço MAC especificado. Não resolve nomes.

Tcpdump imprime uma descrição do conteúdo de pacotes em uma interface de rede que corresponde a uma expressão booleana pré-definida, de interesse do usuário. Ele também pode ser executado com o sinalizador *-w*, que faz com que seja salvo os pacotes de dados para um arquivo, para análise posterior; ou com o sinalizador *-r*, que faz com que os pacotes sejam lidos a partir de um arquivo ao invés de uma interface de rede. Ele também pode ser executado com o sinalizador *-v*, que faz com que os pacotes sejam lidos a partir de uma lista de arquivos de pacotes salvos. Em todos os casos, apenas os pacotes que estejam associados à expressão booleana pré-definida serão processados pelo tcpdump (TCPdump[1]).

Como exemplo, as linhas geradas (*logs*) utilizando-se o *tcpdump* podem ser observadas abaixo:

21:00:34.099289 IP 192.168.1.102.34712 > 200.221.2.45.80: Flags [S], seq 2424058224, win 14600, options [mss 1460, sackOK,TS val 244136 ecr 0,nop,wscale 7], length 0

[1]Disponível em: http://www.tcpdump.org/tcpdump_man.html

Assim, as informações de *log* de um computador conectado à rede são formadas por: registro de horário, protocolo, endereço IP de origem, porta de origem, endereço IP de destino, porta de destino e informações adicionais, dependendo do tipo de protocolo de transporte utilizado, TCP ou UDP.

Para se capturar e redirecionar (arquivo *dump*) todos os dados do tráfego que chegam e saem da vítima, foi utilizado o seguinte comando: *tcpdump -n host 192.168.1.102 > dump*.

Nesta dissertação, são considerados apenas as seguintes informações de *log*: registro de horário, tipo de porta e número de porta. Isso pelo fato de apenas essas informações serem suficientes para a detecção dos ataques.

Dessa forma, faz-se necessário não apenas a coleta dos dados, mas também a filtragem dos mesmos, para em seguida serem importados ao MATLAB a fim de se fazer os cálculos necessários.

6.3.2 – Filtragem dos dados

De posse das diversas linhas geradas pelo *tcpdump* é possível extrair apenas as informações de interesse, neste caso: o registro de horário, tipo de porta e número de porta.

Como o sistema operacional utilizado nas simulações foi o Linux (distribuição *Ubuntu* 12.10, *Kernel* 3.8.0-29), então através de comandos em *shell script* foi possível filtrar os *logs* e obter apenas as informações necessárias.

Filtros nada mais são do que comandos que tratam resultados de outros comandos. Os filtros são viabilizados por intermédio dos *pipes* (permite que o resultado de um comando seja passado para outro comando para que este o processe), uma das maiores novidades introduzidas pelo *Unix* (Eriberto, 2012).

De posse do arquivo *dump* é possível se utilizar alguns filtros a fim de separar as informações necessárias. A Tabela 6.2 detalha alguns filtros que foram usados.

Com a Tabela 6.2 é possível se obter então os arquivos de *log* dos tráfegos que estão sendo estudados: dump (tráfego total), noise (ruído), attack_synflood (ataque de *synflood*), attack_fraggle (ataque de *fraggle*), attack_portscan (ataque de *portscan*) e signal (sinal legítimo).

Apesar de já se ter obtido os arquivos de interesse, o que interessa de fato não são o *logs*, mas sim as informações das portas e os instantes em que elas são observadas na simulação.

Tabela 6.2 - Filtros utilizados para a aquisição dos tráfegos.

Comando	Resultado
cat dump | grep –vP 'ARP|ICMP|224.0.0.251' > dump	Retira tudo associado aos protocolos ARP, ICMP e IGMP, pois o protocolo ARP é um protocolo da camada de enlace do modelo OSI e os protocolos IGMP e ICMP são protocolos da camada IP do modelo OSI, camadas que não são objetos de estudo dessa dissertação.
cat dump | grep –vP 'Flags|\.53|\.445|\.69|\.123|\.19' > noise	É criado o arquivo noise, que contém os *logs* associados apenas ao ruído.
cat dump | grep -P '\.600:|\.600 >' > attack_synflood	É criado o arquivo attack_synflood, que contém os *logs* associados ao ataque de *synflood*.
cat dump | grep '\.19:' > attack_fraggle	É criado o arquivo attack_fraggle, que contém os *logs* associados ao ataque de *fraggle*.
cat dump | grep -vP '\.600:|\.600 >' | grep -P '\.21: |\.21 >|\.22: |\.22 >|\.23: |\.23 >|\.25: |\.25 >|\.110: |\.110 >|\.143: |\.143 >|\.161: |\.161 >|\.445: |\.69: |\.123: ' > attack_portscan	É criado o arquivo attack_portscan, que contém os *logs* associados ao ataque de *portscan*.
cat dump | grep -P '\.80: |\.80 >|\.53: |\.53 >|\.443: |\.443 >' > signal	É criado o arquivo signal, que contém os *logs* associados ao sinal legítimo.

O tráfego filtrado referente ao ruído, por exemplo, pode ser visto logo abaixo:

21:24:42.484858 IP 192.168.1.102.68 > 192.168.1.1.67: BOOTP/DHCP, Request from 00:26:9e:b7:82:be, length 300
21:24:42.487652 IP 192.168.1.1.67 > 192.168.1.102.68: BOOTP/DHCP, Reply, length 548
21:32:15.087776 IP 192.168.1.102.68 > 192.168.1.1.67: BOOTP/DHCP, Request from 0c:ee:e6:d5:1e:9a, length 300
22:18:07.494639 IP 192.168.1.102.68 > 192.168.1.1.67: BOOTP/DHCP, Request from 00:26:9e:b7:82:be, length 300
22:18:07.499170 IP 192.168.1.1.67 > 192.168.1.102.68: BOOTP/DHCP, Reply, length 548
22:22:06.058186 IP 192.168.1.102.68 > 192.168.1.1.67: BOOTP/DHCP, Request from 0c:ee:e6:d5:1e:9a, length 300

De posse dos arquivos de tráfego é possível se fazer mais algumas filtragens semelhantes às da Tabela 6.2 e se obter o tráfego para cada porta em particular, como por exemplo: cat signal | grep -P '\.80: |\.80 >' > signal_80. Com isso é possível obter todo o tráfego de entrada e saída (em relação à vítima) referente ao tráfego legítimo na porta 80. De forma semelhante, obtêm-se os arquivos: signal_443, attack_portscan_161, noise_68, etc.

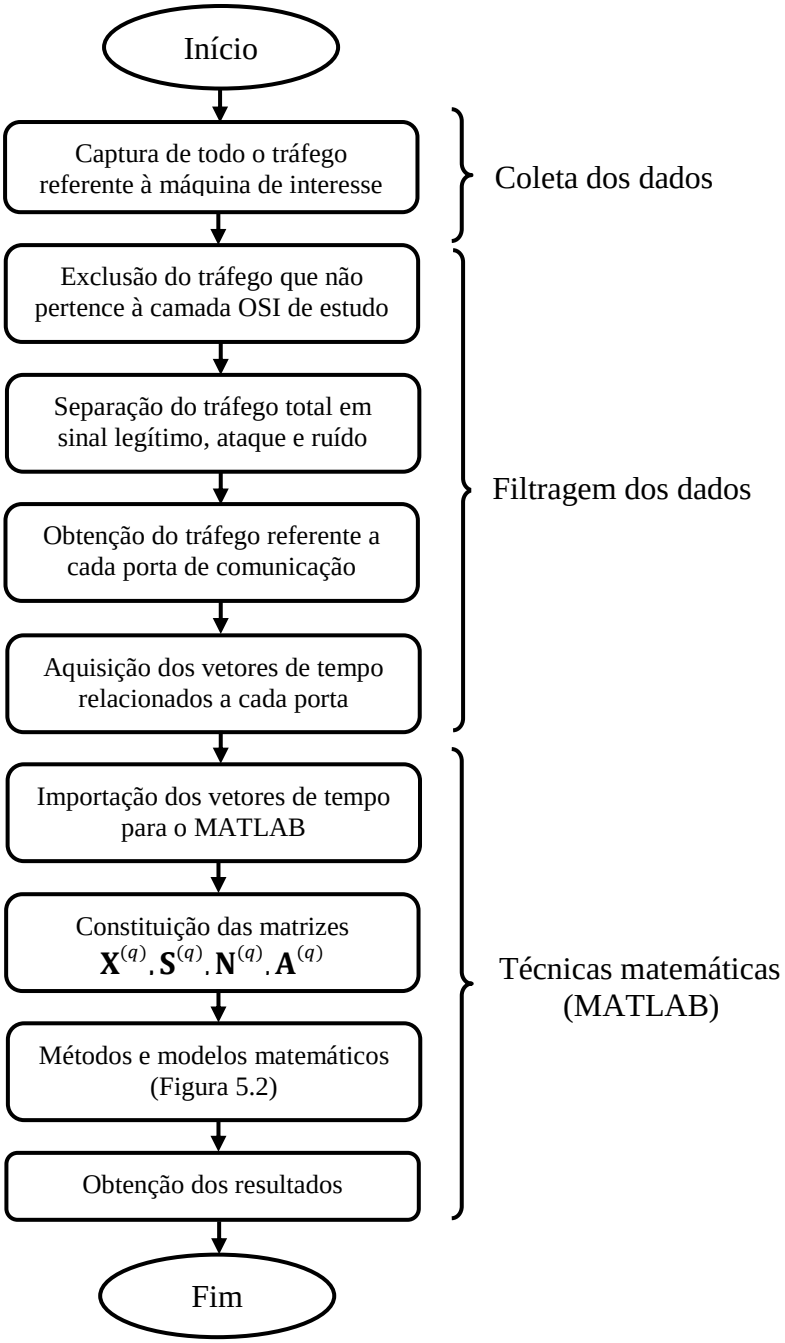


Figura 6.7 - Fluxograma do processo para a obtenção dos resultados.

Nesse momento já se tem as informações individuais de cada porta, sendo necessária agora a última etapa da filtragem, que consiste na redução das informações de tempo constantes nos *logs*, selecionando apenas os minutos.

O procedimento da última etapa da filtragem tomando como exemplo o tráfego do ruído é: cat noise_67 | cut -d: -f2 > noise_67_time. Dessa forma, chega-se ao seguinte resultado:

24
24
32
18
18
22

Finalmente, consegue-se obter então os instantes de tempo (vetores de tempo), em minutos, para cada porta e para cada tráfego: signal_443_time, attack_portscan_161_time, noise_68_time, etc.

Terminada a etapa com os comandos em *shell script*, é necessário agora importar os vetores de tempo para o MATLAB e prosseguir com a composição das matrizes definidas na Seção 6.2 e com as manipulações matemáticas apresentadas na Figura 5.2.

É possível visualizar na forma de diagrama de blocos, conforme apresentado na Figura 6.7, o que discutido na Seção 6.3, sintetizando o processo para a obtenção dos resultados.

6.4 – DETECÇÃO DOS ATAQUES

Observando o fluxograma apresentado na Figura 6.8, o processo de detecção dos ataques inicia-se em $q = 1$, com a obtenção da matriz $\mathbf{X}^{(1)} \in \mathbb{R}^{M \times N}$, passo (1).

Para a detecção dos ataques de *synflood* e *fraggle* (ataques de negação de serviço), é necessário se calcular a matriz de covariância $\mathbf{S}_{xx}^{(q)}$, passo (3), pois neste caso é de interesse que as componentes principais sejam dominadas pelas variáveis de maior variância, de acordo com o que foi discutido na Seção 4. Seguindo (4.15) e (4.16), para se chegar à matriz de covariância $\mathbf{S}_{xx}^{(q)}$ é imprescindível, para cada variável (no caso dessa dissertação, cada porta de comunicação), o cálculo dos desvios dos respectivos elementos em relação à média, passo (2).

Para a detecção do ataque de escaneamento de portas, *portscan*, é necessário se calcular a matriz de correlação $\mathbf{R}_{xx}^{(q)}$, passo (6), ao invés da matriz de covariância $\mathbf{S}_{xx}^{(q)}$, uma vez que neste caso não há interesse que as componentes principais sejam dominadas pelas variáveis de maior variância, até mesmo porque o tráfego associado a este tipo de ataque não gera muitos *logs*, como no ataque de negação de serviço, porém é altamente correlacionado. Seguindo (4.13) e (4.14), para se chegar à matriz de correlação $\mathbf{R}_{xx}^{(q)}$ é imprescindível, para cada variável, o cálculo dos desvios dos resectivos elementos em relação à média dividido pelo desvio padrão, passo (5).

Uma vez as matrizes $\mathbf{S}_{xx}^{(q)}$ e $\mathbf{R}_{xx}^{(q)}$ sendo calculadas, procede-se com a decomposição em autovalores (EVD) - passos (4) e (7), respectivamente - a fim de se obter os autovalores associados a cada matriz, passo (8). É necessário ordenar os autovalores de forma decrescente, passo (9), e selecionar o primeiro autovalor da sequência, que consequentemente é o maior, passo (10).

O processo de obtenção da matriz $\mathbf{X}^{(q)} \in \mathbb{R}^{M \times N}$, q = 1, 2, 3, ..., Q e da matriz $\mathbf{S}_{xx}^{(q)}$ ou $\mathbf{R}_{xx}^{(q)}$, encontrando o maior autovalor referente a cada q-ésimo período de tempo, repete-se até q = Q. O termo "variação temporal do maior autovalor", definido no título dessa dissertação, relaciona-se com o maior autovalor referente a cada q-ésimo período de tempo.

Assim, é possível se construir a matriz $\mathbf{K} \in \mathbb{R}^{M \times Q}$, composta pelos autovalores de $\mathbf{S}_{xx}^{(q)}$ ou $\mathbf{R}_{xx}^{(q)}$. Supondo que $\lambda_1^{(q)} > \lambda_2^{(q)} > \lambda_3^{(q)} > \cdots \lambda_{m-1}^{(q)} > \lambda_m^{(q)}$, a primeira linha da matriz $\mathbf{K}$ contém a Variação Temporal do Maior Autovalor (VTMA), passo (11).

$$\mathbf{K}=\begin{bmatrix} \lambda_1^{(1)} & \lambda_1^{(2)} & \lambda_1^{(3)} & \cdots & \lambda_1^{(Q)} \\ \lambda_2^{(1)} & \lambda_2^{(2)} & \lambda_2^{(3)} & \cdots & \lambda_2^{(Q)} \\ \lambda_3^{(1)} & \lambda_3^{(2)} & \lambda_3^{(3)} & \cdots & \lambda_3^{(Q)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \lambda_m^{(1)} & \lambda_m^{(2)} & \lambda_m^{(3)} & \cdots & \lambda_m^{(Q)} \end{bmatrix} \tag{6.3}$$

Com a obtenção do vetor VTMA, $\lambda_1^{(1)}, \lambda_1^{(2)}, \lambda_1^{(3)}, \ldots, \lambda_1^{(Q)}$, pode-se utilizar os esquemas de seleção de ordem do modelo e estimar a ordem do modelo $\hat{d}$, passo (12). Caso a ordem do modelo estimada ($\hat{d}$) seja igual à ordem real do modelo (d), descobre-se qual esquema de MOS aplica-se ao problema. Pode-se encontrar mais de um esquema que se aplique ao problema, não necessariamente um. Com isso, o processo de detecção dos ataques chega-se ao fim.

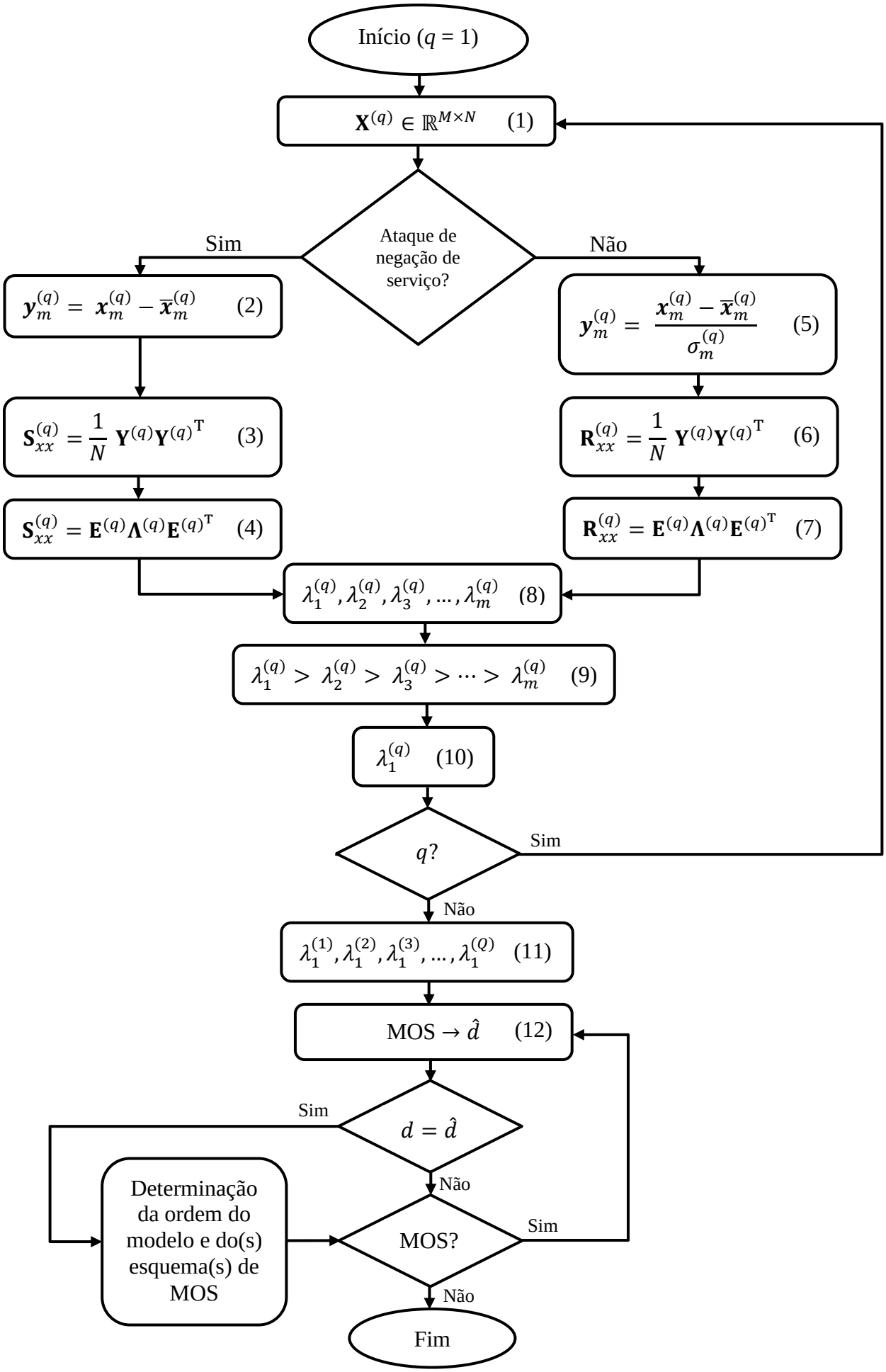


Figura 6.8 - Fluxograma do processo de detecção dos ataques.

7 – RESULTADOS EXPERIMENTAIS

Nesta seção o leitor familiarizar-se-á com o cenário analisado e terá oportunidade de comprovar todos os resultados obtidos, desde as matrizes iniciais de dados, passando pelas matrizes de covariância e correlação, até a obtenção das componentes principais, vetores VTMA e ordem do modelo.

7.1 – CENÁRIO ANALISADO

O ambiente estudado é composto por dois computadores e um roteador com acesso à *Internet* e à rede interna (LAN). Um dos computadores tem o papel de atacante, enquanto o outro de vítima, de acordo com a Figura 7.1.

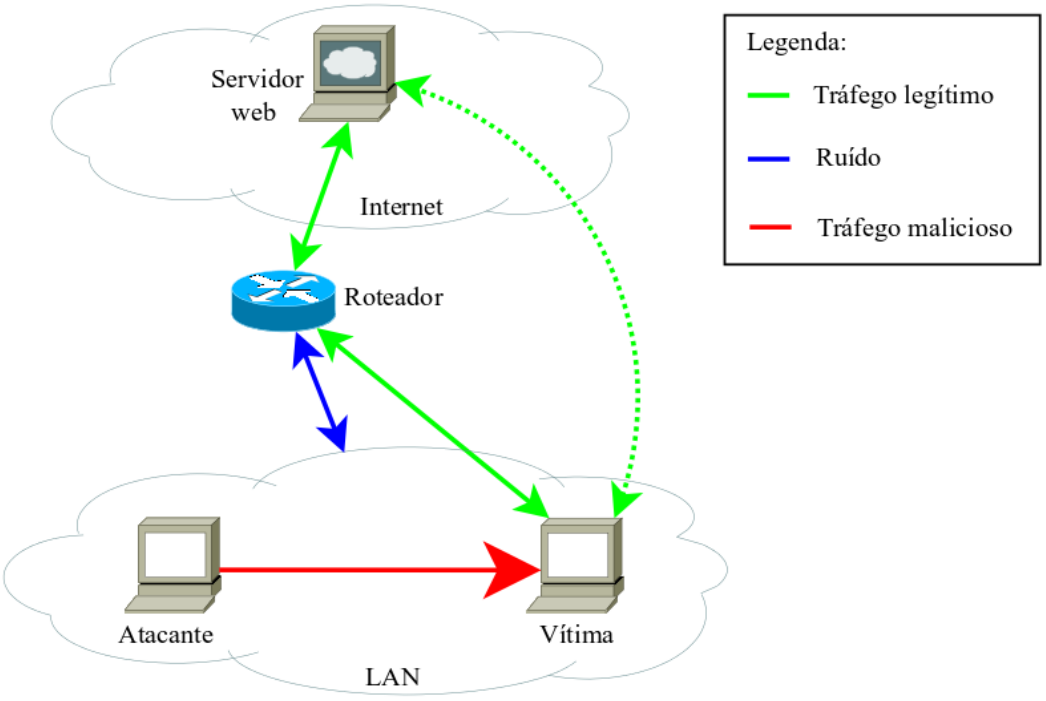


Figura 7.1 - Cenário analisado.

A vítima realizou atividades legítimas, principalmente acesso *web*. Em muitas organizações esse tipo de acesso é feito com muita frequência, pois grande parte dos serviços corporativos é *web*, como por exemplo: acesso ao servidor de *e-mail*, acesso ao protocolo de documentos, acesso à página de intranet, etc.

Como exemplo de ruído associado à camada de transporte do modelo OSI, pode-se citar o tráfego associado ao serviço de DHCP.

No caso do tráfego malicioso, este é composto pelo tráfego associado a três tipos de ataques: *synflood*, *fraggle* e *portscan*, detalhados na Seção 6.1. Os ataques foram simulados utilizando-se ferramentas bem conhecidas por profissionais de segurança de redes. Para o *portscan* utilizou-se o *nmap*, para o ataque de *synflood* o *metasploit* e para conduzir o ataque de *fraggle* utilizou-se o *hping*.

Nmap[1] é um utilitário livre e de código aberto utilizado para a análise de redes e auditoria de segurança. Muitos sistemas e administradores de rede também o acham útil para tarefas como inventário de rede, gerenciamento de agendas de atualização de serviços e monitoramento de *host*s ou serviços. *Nmap* utiliza pacotes IP de forma inovadora para determinar quais *hosts* estão disponíveis na rede, quais serviços (nome da aplicação e versão) os servidores estão oferecendo, quais sistemas operacionais (e versões de OS) eles estão executando, quais tipos de filtros de pacotes/*firewalls* estão em uso e dezenas de outras características. Ele foi projetado para escanear rapidamente grandes redes, mas funciona bem contra redes pequenas. *Nmap* é executado em praticamente todos os principais sistemas operacionais e pacotes binários oficiais estão disponíveis para Linux, Windows e Mac OS X.

O *framework metasploit*[2] é a principal ferramenta presente na distribuição *Linux BackTrack*, sendo praticamente um padrão para testes de penetração. Contém um grande banco de dados de *exploits* conhecidos. É utilizado para o desenvolvimento de assinaturas de IDS em laboratórios, conforme exposto em sua página eletrônica. A ferramenta é desenvolvida em *Ruby*, com componentes em *C* e *Assembly*.

Hping[3] é uma ferramenta de rede muito utilizada por profissionais de segurança, mas ela pode ser usada por muitas pessoas que não estão preocupadas diretamente com segurança, a fim de testar *hosts* e redes. Com o *hping* é possível: executar testes em *firewalls*, escanear portas, traçar rotas, dentre outros. A interface do *hping* é inspirada no comando *unix ping* e suporta protocolos TCP, UDP, ICMP e RAW-IP.

Importante fazer a observaçaõ de que o presente trabalho aplica-se também se o atacante não estiver na rede interna (LAN), mas sim em qualquer lugar da *Internet*. Na prática não faz diferença a localização do atacante.

A Tabela 7.1 apresenta detalhes relacionados aos computadores utilizados nas simulações:

[1] *Network Mapper*. Disponível em: *http://nmap.org/*
[2] *Metasploit Penetration Testing Software.* Disponível em: *http://www.metasploit.com*
[3] Disponível em: *http://www.hping.org/*

sistema operacional, a versão do Kernel e o endereço IP que foi utilizado.

Tabela 7.1 - Características associadas aos computadores nas simulações.

Equipamento	Sistema Operacional /Distribuição	Kernel	Endereço IP
Atacante	*Linux/BackTrack* 5 r3	3.2.6	192.168.1.104
Vítima	*Linux/Ubuntu* 13.04	3.8.0-31	192.168.1.102

O tempo total de simulação foi de cento e vinto minutos, sendo separado em seis períodos, cada período de tempo equivalendo a vinte minutos. Como o tempo de amostragem de cada período é de um minuto, logo $N = 20$.

Para cada período de tempo q, foi obtida uma matriz de tráfego $\mathbf{X}^{(q)} \in \mathbb{R}^{17\times20}$, uma matriz de covariância $\mathbf{S}_{xx}^{(q)} \in \mathbb{R}^{17\times17}$ e uma matriz de correlação $\mathbf{R}_{xx}^{(q)} \in \mathbb{R}^{17\times17}$, sendo no caso dessa dissertação $q = 1, 2, 3, 4, 5$ e 6. A simulação começou às 21:00h, o primeiro período vai de 21:00h até 21:20h ($q = 1$), o segundo de 21:20h até 21:40h ($q = 2$), o terceiro de 21:40h até 22:00h ($q = 3$), o quarto de 22:00h até 22:20h ($q = 4$), o quinto de 22:20h até 22:40h ($q = 5$) e, finalmente, o sexto de 22:40h até 23:00h ($q = 6$).

Durante a simulação, a vítima fez seus acessos legítimos, e o atacante, em determinados instantes, executou os ataques: às 21:54h ($q = 3$) foi realizado o *portscan*; no intervalo de tempo que vai de 22:10h às 22:20h ($q = 4$) o ataque de *synflood* foi simulado; e no intervalo de tempo de 22:30h às 22:40h ($q = 5$) ocorreu o ataque de *fraggle*.

7.1 – MATRIZES $\mathbf{X}^{(q)}$

A matriz de tráfego $\mathbf{X}^{(q)} \in \mathbb{R}^{17\times20}$ está organizada da seguinte forma: as linhas representam as portas de comunicação de estudo e as colunas o tempo de amostragem. Assim, as portas tcp 80, tcp 443, udp 53, tcp 21, tcp 22, tcp 23, tcp 25, tcp 110, tcp 143, tcp 161, udp 69, udp 123, udp 445, tcp 600, udp 19, udp 67 e udp 68, estão representadas nas linhas 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 e 17, respectivamente. A primeira coluna representa o primeiro minuto de amostragem, a segunda coluna o segundo minuto de amostragem e assim por diante, até se chegar na vigésima coluna.

O elemento $x_{m,n}^{(q)}$ representa o número de vezes que a porta m aparece na n-ésima amostra, dentro do q-ésimo período de tempo. Assim, por exemplo, o elemento $x_{1,12}^{(3)}$ representa o número de vezes que a porta tcp 80 apareceu no tráfego de rede (tráfego entrante e sainte)

durante o décimo segundo minuto, do perído que inicia-se às 21:40h e vai até 22:00h. Nesse período, o décimo segundo minuto é inervalo que vai de 21:51h até 21:52h. De acordo com a Figura 7.3 o valor de $x_{1,12}^{(3)}$ = 43.

3244	99	44	3620	4673	1916	192	1877	1416	571	1710	133	2685	51	3000	40	13	1284	1906	2330
281	32	27	144	359	105	75	265	36	302	134	123	636	18	111	48	6	74	571	303
348	8	0	216	251	48	18	306	80	18	76	41	100	16	30	10	0	56	30	54
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figura 7.2 - Matriz $\mathbf{X}^{(1)}$.

2113	5442	1977	3485	774	68	5	1463	3520	50	47	3	1110	10	15	3298	1551	107	637	275
135	75	111	72	34	32	12	221	174	857	171	205	382	677	148	478	405	48	165	2152
16	56	40	54	10	6	0	44	71	48	10	0	46	8	10	96	18	10	29	80
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Figura 7.3 - Matriz $\mathbf{X}^{(2)}$.

511	1114	24	12	65	289	612	346	8239	988	2025	43	43	7	3	5	3	3	3	971
28	36	89	4	337	233	194	233	51	159	771	773	151	176	75	12	113	84	246	253
106	20	0	0	14	37	23	13	16	65	54	22	10	6	0	10	0	0	6	25
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figura 7.4 - Matriz $\mathbf{X}^{(3)}$.

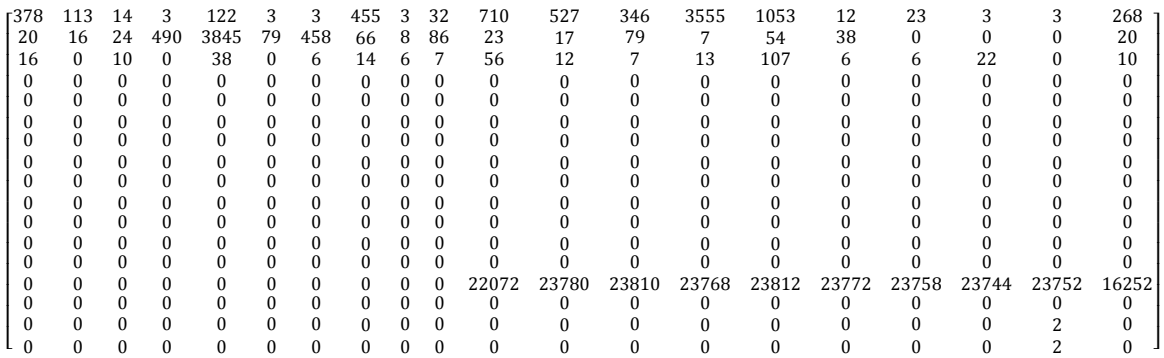


Figura 7.5 - Matriz $\mathbf{X}^{(4)}$.

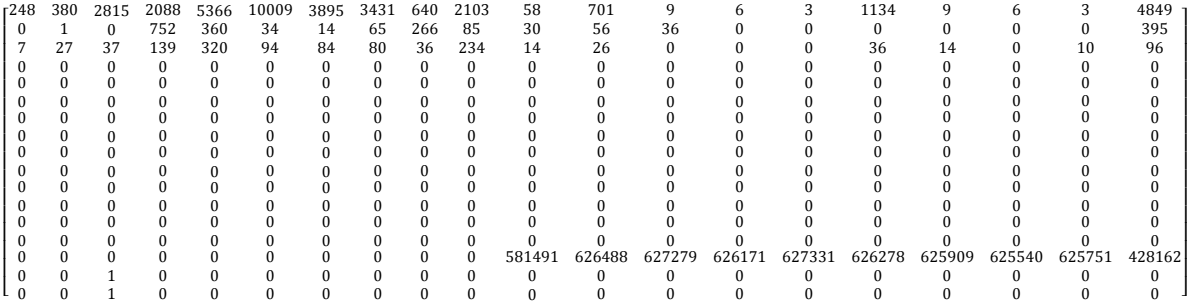


Figura 7.6 - Matriz $\mathbf{X}^{(5)}$.

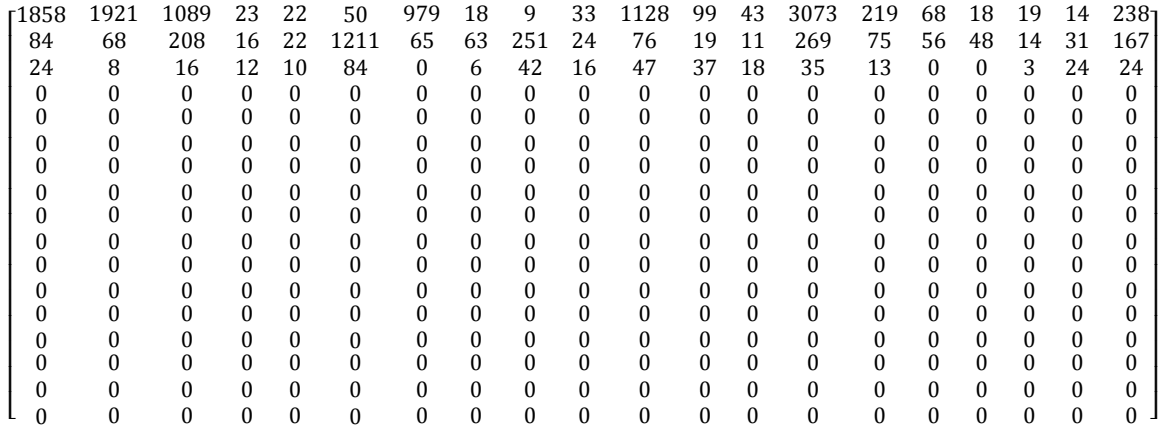


Figura 7.7 - Matriz $\mathbf{X}^{(6)}$.

As linhas nulas apresentadas nas matrizes de tráfego são devidas às ausências de ataques e/ou ruídos nos respectivos períodos de tempo.

7.2 – MATRIZES $\mathbf{S}_{xx}^{(q)}$

Da mesma forma como nas matrizes de tráfego $\mathbf{X}^{(q)}$, as matrizes de covariância que possuem as linhas nulas são devidos às ausências de ataques e/ou ruídos nos respectivos períodos de tempo.

$$
\begin{bmatrix}
18730101 & 130394 & 100017 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
130394 & 30974 & 6213 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
100017 & 6213 & 10698 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

Figura 7.8 - Matriz $\mathbf{S}_{xx}^{(1)}$.

$$
\begin{bmatrix}
2330532 & -149227 & 26126 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -62 & -62 \\
-149227 & 223003 & 6078 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -27 & -27 \\
26126 & 6078 & 766 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & -2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-62 & -27 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0{,}2275 & 0{,}2275 \\
-62 & -27 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0{,}2275 & 0{,}2275
\end{bmatrix}
$$

Figura 7.9 - Matriz $\mathbf{S}_{xx}^{(2)}$.

$$
\begin{bmatrix}
3213785 & -15150 & 5550 & -76 & -76 & -76 & -76 & -76 & -76 & -76 & -38 & -38 & -38 & 0 & 0 & 0 & 0 \\
-15150 & 44317 & 951 & -13 & -13 & -13 & -13 & -13 & -13 & -13 & -63 & -63 & -63 & 0 & 0 & 0 & 0 \\
5550 & 951 & 678 & -2 & -2 & -2 & -2 & -2 & -2 & -2 & -1 & -1 & -1 & 0 & 0 & 0 & 0 \\
-76 & -13 & -2 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}095 & 0{,}095 & 0{,}095 & 0 & 0 & 0 & 0 \\
-76 & -13 & -2 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}095 & 0{,}095 & 0{,}095 & 0 & 0 & 0 & 0 \\
-76 & -13 & -2 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}095 & 0{,}095 & 0{,}095 & 0 & 0 & 0 & 0 \\
-76 & -13 & -2 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}095 & 0{,}095 & 0{,}095 & 0 & 0 & 0 & 0 \\
-76 & -13 & -2 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}095 & 0{,}095 & 0{,}095 & 0 & 0 & 0 & 0 \\
-76 & -13 & -2 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}095 & 0{,}095 & 0{,}095 & 0 & 0 & 0 & 0 \\
-76 & -13 & -2 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}19 & 0{,}095 & 0{,}095 & 0{,}095 & 0 & 0 & 0 & 0 \\
-38 & -63 & -1 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}0475 & 0{,}0475 & 0{,}0475 & 0 & 0 & 0 & 0 \\
-38 & -63 & -1 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}0475 & 0{,}0475 & 0{,}0475 & 0 & 0 & 0 & 0 \\
-38 & -63 & -1 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}095 & 0{,}0475 & 0{,}0475 & 0{,}0475 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

Figura 7.10 - Matriz $\mathbf{S}_{xx}^{(3)}$.

$$
\begin{bmatrix}
609596 & -68873 & 4984 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3210288 & 0 & -38 & -38 \\
-68873 & 692230 & 3490 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2771354 & 0 & -27 & -27 \\
4984 & 3490 & 604 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 83793 & 0 & -2 & -2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3210288 & -2771354 & 83793 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 133102521 & 0 & 1233 & 1233 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-38 & -27 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1233 & 0 & 0{,}19 & 0{,}19 \\
-38 & -27 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1233 & 0 & 0{,}19 & 0{,}19
\end{bmatrix}
$$

Figura 7.11 - Matriz $\mathbf{S}_{xx}^{(4)}$.

$$
\begin{bmatrix}
6363411 & 123655 & 116978 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -404130425 & 46 & 46 \\
123655 & 36039 & 8365 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -19305089 & -5 & -5 \\
116978 & 8365 & 6763 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -13718986 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-404130425 & -19305089 & -13718986 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 92382247564 & -15051 & -15051 \\
46 & -5 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -15051 & 0{,}0475 & 0{,}0475 \\
46 & -5 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -15051 & 0{,}0475 & 0{,}0475
\end{bmatrix}
$$

Figura 7.12 - Matriz $\mathbf{S}_{xx}^{(5)}$.

$$
\begin{bmatrix}
708292 & 5088 & 1413 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
5088 & 66342 & 4039 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1413 & 4039 & 397 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

Figura 7.13 - Matriz $\mathbf{S}_{xx}^{(6)}$.

7.3 – MATRIZES $\mathbf{R}_{xx}^{(q)}$

As matrizes $\mathbf{R}_{xx}^{(q)}$ teoricamente também são formadas por 17 linhas e 17 colunas, como as matrizes $\mathbf{S}_{xx}^{(q)}$, porém para os elementos pertencentes às linhas nulas, o coeficiente de correlação é indeterminado, devido à normalização utilizada na obtenção desse tipo de matriz. Dessa forma, as matrizes de correlação estão representadas apenas com os elementos determinados.

$$\begin{bmatrix} 1 & 0{,}5414 & 0{,}7066 \\ 0{,}5414 & 1 & 0{,}3413 \\ 0{,}7066 & 0{,}3413 & 1 \end{bmatrix}$$

Figura 7.14 - Matriz $\mathbf{R}_{xx}^{(1)}$.

$$\begin{bmatrix} 1 & -0{,}2070 & 0{,}6181 & -0{,}0848 & -0{,}0848 \\ -0{,}2070 & 1 & 0{,}4649 & -0{,}1183 & -0{,}1204 \\ 0{,}6181 & 0{,}4649 & 1 & -0{,}1183 & -0{,}1204 \\ -0{,}0848 & -0{,}1183 & -0{,}1204 & 1 & 1 \\ -0{,}0848 & -0{,}1183 & -0{,}1204 & 1 & 1 \end{bmatrix}$$

Figura 7.15 - Matriz $\mathbf{R}_{xx}^{(2)}$.

$$\begin{bmatrix} 1 & -0{,}0401 & 0{,}1189 & -0{,}0976 & -0{,}0976 & -0{,}0976 & -0{,}0976 & -0{,}0976 & -0{,}0976 & -0{,}0976 & -0{,}0976 & -0{,}0976 & -0{,}0976 \\ -0{,}0401 & 1 & 0{,}1734 & -0{,}1372 & -0{,}1372 & -0{,}1372 & -0{,}1372 & -0{,}1372 & -0{,}1372 & -0{,}1372 & -0{,}1372 & -0{,}1372 & -0{,}1372 \\ 0{,}1189 & 0{,}1734 & 1 & -0{,}1881 & -0{,}1881 & -0{,}1881 & -0{,}1881 & -0{,}1881 & -0{,}1881 & -0{,}1881 & -0{,}1881 & -0{,}1881 & -0{,}1881 \\ -0{,}0976 & -0{,}1372 & -0{,}1881 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -0{,}0976 & -0{,}1372 & -0{,}1881 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -0{,}0976 & -0{,}1372 & -0{,}1881 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -0{,}0976 & -0{,}1372 & -0{,}1881 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -0{,}0976 & -0{,}1372 & -0{,}1881 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -0{,}0976 & -0{,}1372 & -0{,}1881 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -0{,}0976 & -0{,}1372 & -0{,}1881 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -0{,}0976 & -0{,}1372 & -0{,}1881 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -0{,}0976 & -0{,}1372 & -0{,}1881 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -0{,}0976 & -0{,}1372 & -0{,}1881 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Figura 7.16 - Matriz $\mathbf{R}_{xx}^{(3)}$.

$$\begin{bmatrix} 1 & -0{,}1060 & 0{,}2598 & 0{,}3564 & -0{,}1112 & -0{,}1112 \\ -0{,}1060 & 1 & 0{,}1707 & -0{,}2887 & -0{,}0735 & -0{,}0735 \\ 0{,}2598 & 0{,}1707 & 1 & 0{,}2956 & -0{,}1569 & -0{,}1569 \\ 0{,}3564 & -0{,}2887 & 0{,}2956 & 1 & 0{,}2451 & 0{,}2451 \\ -0{,}1112 & -0{,}0735 & -0{,}1569 & 0{,}2451 & 1 & 1 \\ -0{,}1112 & -0{,}0735 & -0{,}1569 & 0{,}2451 & 1 & 1 \end{bmatrix}$$

Figura 7.17 - Matriz $\mathbf{R}_{xx}^{(4)}$.

$$\begin{bmatrix} 1 & 0{,}2582 & 0{,}5639 & -0{,}5271 & 0{,}0801 & 0{,}0801 \\ 0{,}2582 & 1 & 0{,}5358 & -0{,}3346 & -0{,}1265 & -0{,}1265 \\ 0{,}5639 & 0{,}5358 & 1 & -0{,}5488 & -0{,}0717 & -0{,}0717 \\ -0{,}5271 & -0{,}3346 & -0{,}5488 & 1 & -0{,}2272 & -0{,}2272 \\ 0{,}0843 & -0{,}1265 & -0{,}0717 & -0{,}2272 & 1 & 1 \\ 0{,}0843 & -0{,}1265 & -0{,}0717 & -0{,}2272 & 1 & 1 \end{bmatrix}$$

Figura 7.18 - Matriz $\mathbf{R}_{xx}^{(5)}$.

$$\begin{bmatrix} 1 & 0{,}0235 & 0.0843 \\ 0{,}0235 & 1 & 0{,}7875 \\ 0.0843 & 0{,}7875 & 1 \end{bmatrix}$$

Figura 7.19 - Matriz $\mathbf{R}_{xx}^{(6)}$.

7.4 – AUTOVALORES ENCONTRADOS

Com a determinação das matrizes de covariância e correlação, é possível se calcular os autovalores associados a cada uma delas. Teoricamente, a matriz gerada (**K**) é uma matriz de 17 linhas por 6 colunas ($\mathbf{K} \in \mathbb{R}^{17\times6}$), 17 linhas pois representam a quantidade de autovalores gerados a partir de uma matriz $\mathbb{R}^{17\times17}$ e 6 linhas pois nessa dissertação são considerados 6 períodos de tempo (q = 6). Na prática, por uma questão de simplificação, não foram apresentadas todas as 17 linhas pelo fato de, a partir da sexta linha os autovalores serem nulos.

A primeira coluna da matriz **K** representa os autovalores encontrados no primeiro período de tempo (q = 1), a segunda coluna representa os autovalores encontrados no segundo período de tempo (q = 2) e assim por diante. Isso é aplicado independetemente do tipo de matriz que deu origem aos autovalores: covariância ou correlação.

Seguindo o que foi disposto no fluxograma na Figura 6.8, os autovalores em cada coluna de **K** são ordenados de forma decrescente, de tal forma que a primeira linha da matriz **K** apresenta o vetor VTMA.

Serão apresentadas duas matrizes de autovalores oriundas da matriz de covariância, isso devido ao fato de se estar simulando dois ataques de negação de serviço: *synflood* e *fraggle*. A intenção com isso é de se estudar o efeito de cada ataque em separado. Obviamente que o resultado será similar uma vez que ambos possuem o mesmo comportamento.

$$\begin{bmatrix} 2{,}0734 & 2{,}1451 & 10{,}0718 & 2{,}1620 & 2{,}4253 & 1{,}7948 \\ 0{,}6748 & 1{,}5484 & 1{,}1219 & 1{,}6431 & 2{,}0838 & 0{,}9950 \\ 0{,}2518 & 1{,}1882 & 1{,}0365 & 1{,}1072 & 0{,}7187 & 0{,}2101 \\ 0 & 0{,}1183 & 0{,}7698 & 0{,}6655 & 0{,}4409 & 0 \\ 0 & 0 & 0 & 0{,}4222 & 0{,}3314 & 0 \end{bmatrix}$$

Figura 7.20 - Matriz dos autovalores das matrizes de correlação (*portscan*).

$$\begin{bmatrix} 1887545 & 2341327 & 3213867 & 133238294 & 6367983 & 708335 \\ 21831 & 212783 & 44266 & 634322 & 4596480 & 66547 \\ 5306 & 192 & 647 & 531843 & 32547 & 148 \\ 0 & 0{,}4424 & 1{,}3936 & 491 & 3139 & 0 \\ 0 & 0 & 0 & 0{,}3226 & 0{,}0832 & 0 \end{bmatrix}$$

Figura 7.21 - Matriz dos autovalores das matrizes de covariância (*synflood*).

$$\begin{bmatrix} 1887545 & 2341327 & 3213867 & 731229 & 92384021611 & 708335 \\ 21831 & 212783 & 44266 & 634322 & 4596480 & 66547 \\ 5306 & 192 & 647 & 531843 & 32547 & 148 \\ 0 & 0{,}4424 & 1{,}3936 & 491 & 3139 & 0 \\ 0 & 0 & 0 & 0{,}3226 & 0{,}0832 & 0 \end{bmatrix}$$

Figura 7.22 - Matriz dos autovalores das matrizes de covariância (*fraggle*).

A Figura 7.23 representa graficamente a matriz dos autovalores utilizada para a detecção do *portscan*. Nesta figura é possível observar que o maior autovalor, que é relacionado ao *portscan*, se destaca dos demais.

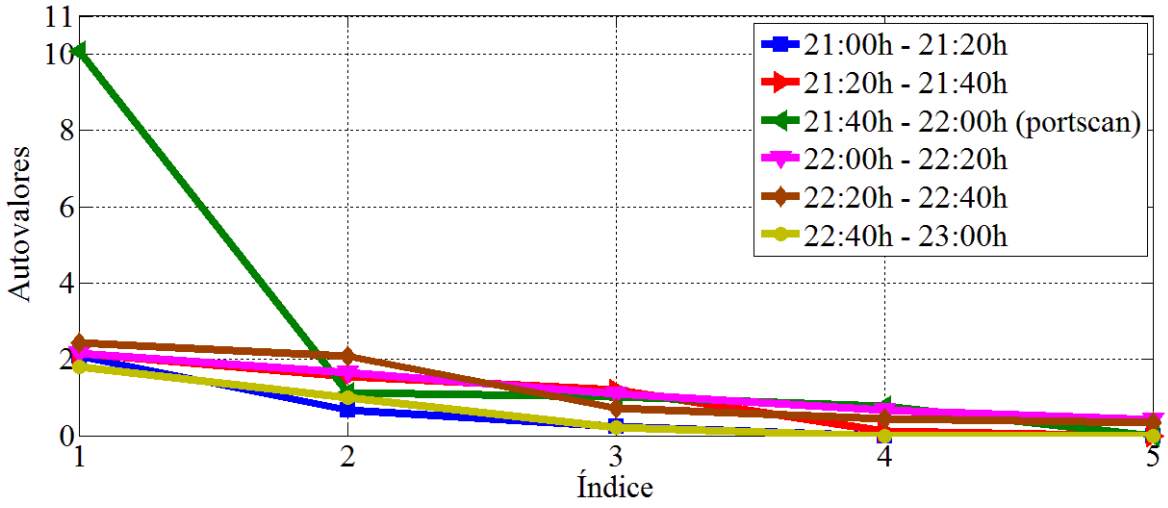


Figura 7.23 - Autovalores associados à matriz de correlação (*portscan*).

A Figura 7.24 representa graficamente a matriz dos autovalores utilizada para a detecção do *synflood*. Nesta figura é possível observar que o maior autovalor, que é relacionado ao *synflood*, se destaca dos demais.

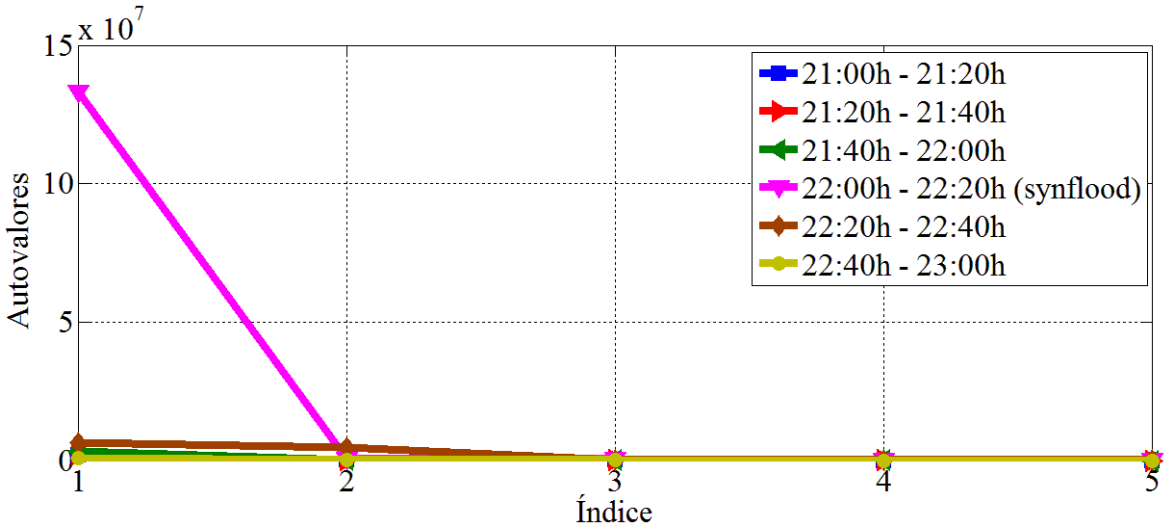


Figura 7.24 - Autovalores associados à matriz de covariância (*synflood*).

A Figura 7.25 apresenta graficamente a matriz dos autovalores utilizada para a detecção do *fraggle*. Nesta figura é possível observar que o maior autovalor, que é relacionado ao *fraggle*, se destaca dos demais.

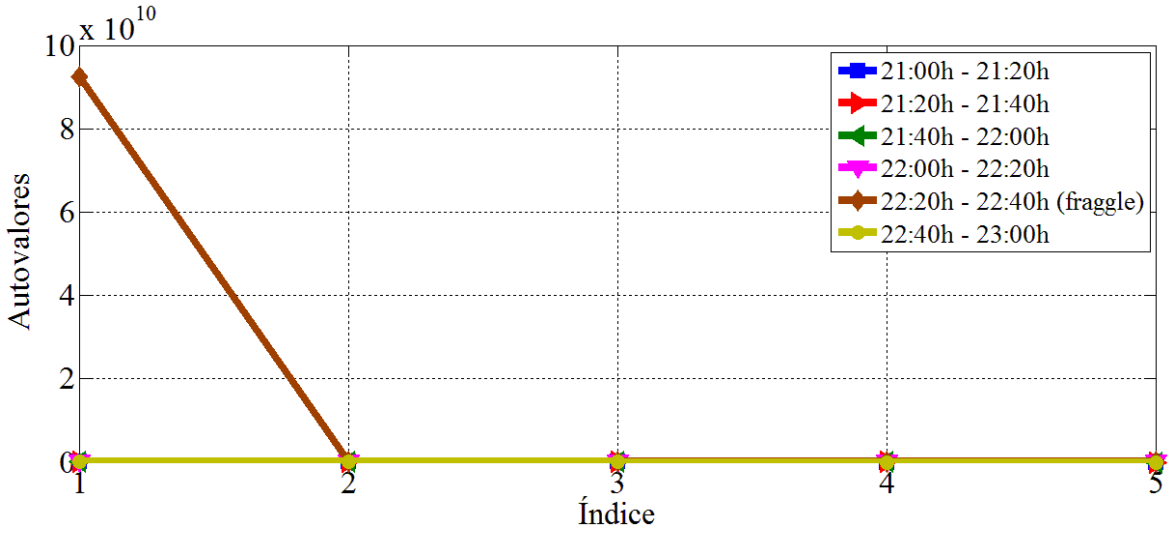


Figura 7.25 - Autovalores associados à matriz de covariância (*fraggle*).

7.5 – VTMA

A Tabela 7.2 sintetiza os autovalores máximos relacionados às matrizes de covariância e de correlação, obtidos em cada período de tempo.

Tabela 7.2 - Máximos autovalores associados às matrizes.

Período de tempo q	**Máximo autovalor** $\lambda_1^{(q)}$	
	$\mathbf{S}_{xx}^{(q)}$	$\mathbf{R}_{xx}^{(q)}$
1	1887545	2,0734
2	2341327	2,1451
3	3213867	10,0718
4	133238294	2,1620
5	92384021611	2,4253
6	708335	1,7948

Como os ataques de negação de serviço apresentam o mesmo comportamento, então, para apurar melhor os resultados relacionados a esse tipo de ataque, serão tomados dois vetores independentes com os máximos autovalores: um vetor considerando apenas o ataque de *synflood* e outro vetor considerando apenas o ataque de *fraggle*. Por fim, para verificar como seria o resultado considerando os dois ataques de negação de serviço, considera-se um terceiro vetor, constando dos ataques de *synflood* e *fraggle*.

A Tabela 7.3 apresenta os vetores formados pela variação temporal do maior autovalor. Vetores esses que foram utilizados como parâmetros para os esquemas de seleção de ordemo do modelo e, consequentemente, para a detecção dos ataques propostos.

Tabela 7.3 - Máximos autovalores associados à detecção dos ataques.

Período de tempo q	**Vetores VTMA**			
	Detecção do *synflood*/*fraggle*	**Detecção do *synflood***	**Detecção do *fraggle***	**Detecção do *portscan***
1	1887545	1887545	1887545	2,0734
2	2341327	2341327	2341327	2,1451
3	3213867	3213867	3213867	10,0718
4	133238294	133238294	731229	2,1620
5	92384021611	6367983	92384021611	2,4253
6	708335	708335	708335	1,7948

Com a Tabela 7.3 é possível se observar o quão destoante são os autovalores associados aos ataques. Em q = 4, onde ocorreu o ataque de *synflood*, o máximo autovalor obtido

nesse período é aproximadamente 21 vezes maior que o segundo maior autovalor do vetor. Em q = 5, onde ocorreu o ataque de *fraggle*, o máximo autovalor obtido nesse período é aproximadamente 29.000 vezes maior que o segundo maior autovalor do vetor. E em q =3, onde ocorreu o ataque de *portscan*, o máximo autovalor obtido nesse período é aproximadamente 4 vezez maior que o segundo maior autovalor do vetor. Neste último caso, apesar do autovalor não ser tão grande, comparado aos ataques de inundação, é perfeitamente suficiente detectá-lo, uma vez que destoa claramente do restante do conjunto.

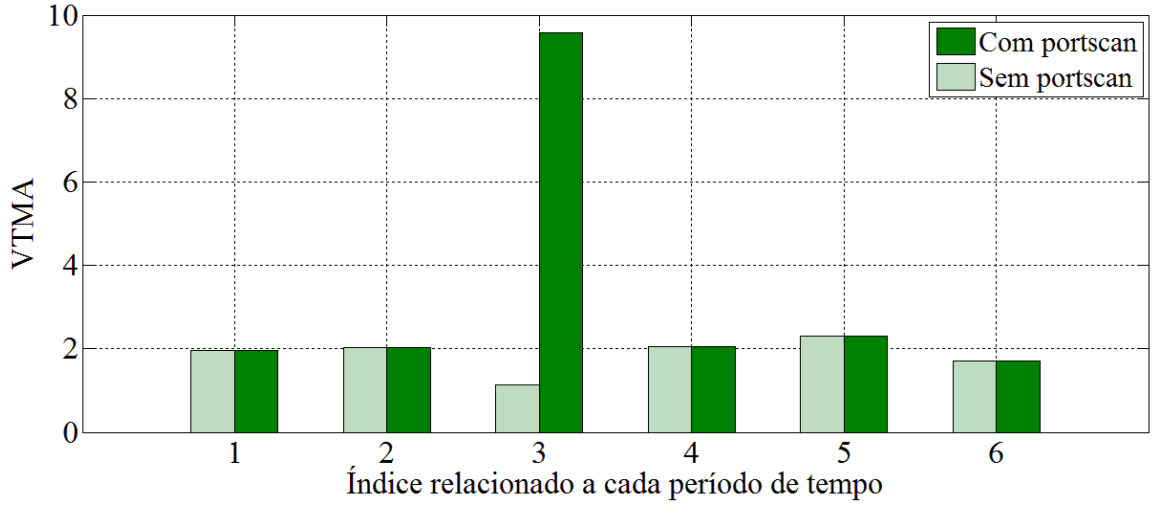


Figura 7.26 - Variação temporal do maior autovalor (*portscan*).

A Figura 7.26 faz um comparativo entre dos vetores VTMA com e sem o ataque de *portscan*.

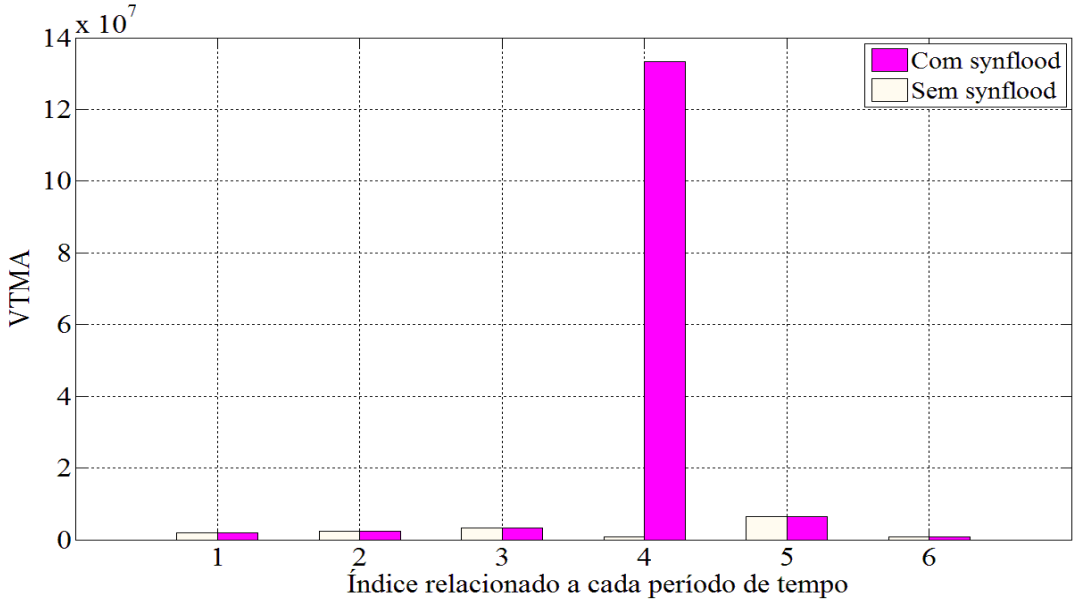


Figura 7.27 - Variação temporal do maior autovalor (*synflood*).

A Figura 7.27 faz um comparativo entre dos vetores VTMA com e sem o ataque de *synflood*, e a Figura 7.28 faz um comparativo entre dos vetores VTMA com e sem o ataque de *fraggle*.

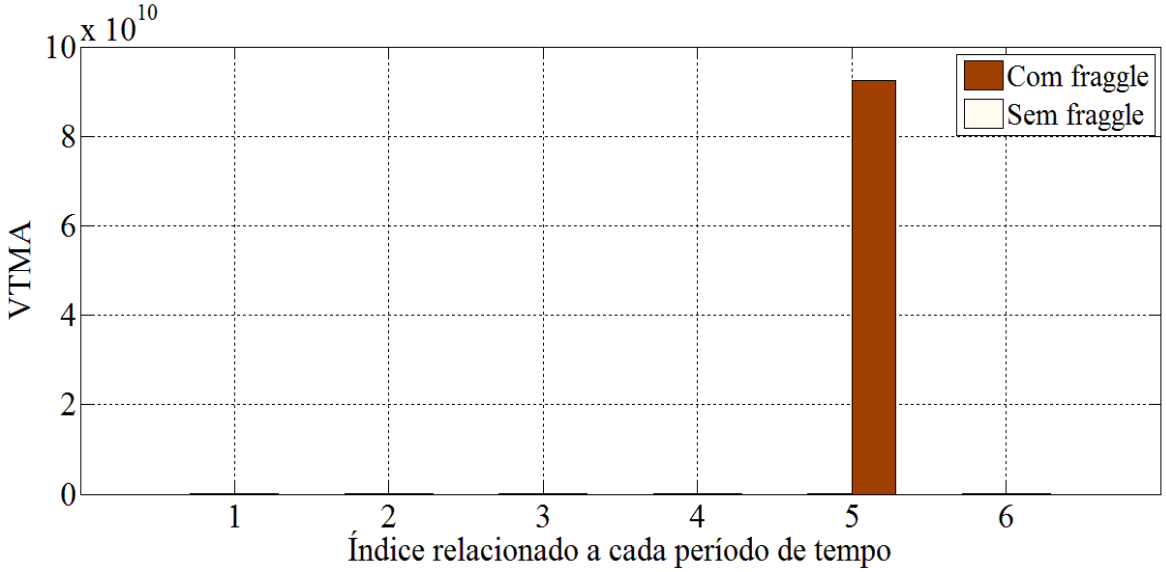


Figura 7.28 - Variação temporal do maior autovalor (*fraggle*).

7.6 – COMPONENTES PRINCIPAIS

Conforme apresentado na Seção 4, a análise de componentes principais é utilizada principalmente para reduzir a dimensão de um conjunto de dados, utilizando para isso variáveis não correlacionadas, denominadas de componentes principais (PC). Essa transformação em outro conjunto de variáveis ocorre com a menor perda de informação possível, eliminando apenas algumas variáveis originais que possuam pouca informação.

As componentes principais resultantes são uma combinação linear das variáveis originais, são ortogonais e ordenadas de forma que a primeira componente principal tenha a maior parte da variância dos dados originais. Embora o número resultante de componentes principais seja igual ao número original de variáveis, grande parte da variação no conjunto original pode ser retida pelas primeiras componentes principais, reduzindo dessa forma a dimensão do problema.

De acordo com o cenário dessa dissertação, as variáveis de estudo são as portas de comunicação: tcp 80, tcp 443, udp 53, tcp 21, tcp 22, tcp 23, tcp 25, tcp 110, tcp 143, tcp 161, udp 69, udp 123, udp 445, tcp 600, udp 19, udp 67 e udp 68. Dessa forma, as componentes principais são formadas por combinações lineares dessas variáveis.

Como se tem 17 variáveis, logo a dimensão do conjunto é 17-dimensional. Com as componentes principais pode-se reduzir tal conjunto em, por exemplo, duas dimensões, apresentado as duas primeiras componentes principais. Com isso, consegue-se reduzir a dimensão do conjunto sem perda siginificativa de informações.

As componentes principais são obtidas a partir dos autovetores das matrizes de covariância ou de correlação. Como serão selecionadas apenas as duas primeiras componentes principais, então, tem-se que selecionar os dois autovetores referentes aos dois maiores autovalores obtidos da decomposição em autovalores das matrizes de covariância ou de correlação.

Como a intenção é mostrar que os ataques apresentam um comportamento diferente e predominante em relação aos outros tráfegos, obviamente serão selecionados, para análise, os períodos de tempo referentes aos ataques: q = 3 para o ataque de *portscan*, q = 4 para o ataque de *synflood* e q = 5 para o ataque de *fraggle*.

7.6.1 – Componentes principais para análise do ataque de synflood

A partir da decomposição em autovalores da matriz $\mathbf{S}_{xx}^{(4)}$, é possível se obter os autovalores, bem como os autovetores, referentes a essa matriz.

$$\begin{bmatrix} 0{,}0242 & -0{,}0164 \\ -0{,}0209 & 0{,}9996 \\ 0{,}0006 & 0{,}0082 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0{,}9995 & 0{,}0213 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Figura 7.29 - Autovetores dos dois maiores autovalores da matriz $\mathbf{S}_{xx}^{(4)}$.

A primeira coluna da matriz representada na Figura 7.29 representa o autovetor correspondente ao maior autovalor de $\mathbf{S}_{xx}^{(4)}$, enquanto a segunda coluna representa o

autovetor correspondente ao segundo maior autovalor de $\mathbf{S}_{xx}^{(4)}$. Dessa forma:

$$\mathrm{PC1}_{synflood} = 0{,}0242 * \mathrm{tcp80} - 0{,}0209 * t\mathrm{cp443} + 0{,}0006 * \mathrm{udp53} + 0{,}9995 * \mathrm{tcp600}, \quad (7.1)$$

onde tcp80, tcp443, udp53 e tcp600 representam as variáveis referentes às respectivas portas de comunicação.

$$\mathrm{PC2}_{synflood} = -0{,}0164 * \mathrm{tcp80} + 0{,}9996 * \mathrm{tcp443} + 0{,}0082 * \mathrm{udp53} + 0{,}0213 * \mathrm{tcp600} \quad (7.2)$$

Teoricamente, as componentes principais seriam representadas como uma combinação linear de todas as variáveis, porém nas Equações 7.1 e 7.2 não constam as demais variáveis pelo fato dos coeficientes das variáveis serem nulos, oriundos das informações dos autovetores. Isso é facilmente explicado pelo fato de, nesse período de tempo não haver tráfego relacionado às portas dos ataques de *fraggle*, tampouco *portscan*. O tráfego referente ao ruído é irrisório, fato que não aparece sequer nas duas primeiras componentes principais.

Com a obtenção das Equações 7.1 e 7.2, a próxima etapa é calcular o valor de cada PC para cada tempo de amostragem. Para isso, deve-se selecionar cada coluna da matriz $\mathbf{X}^{(4)}$ e aplicar os respectivos valores de tráfego relacionados às portas nas expressões 7.1 e 7.2. Com isso, serão obtidos 20 valores para $\mathrm{PC1}_{synflood}$ e 20 valores para $\mathrm{PC2}_{synflood}$, gerando um total de 20 pontos. Na figura abaixo é possível observar tais pontos sob os eixos perpendiculares que representam as componentes principais.

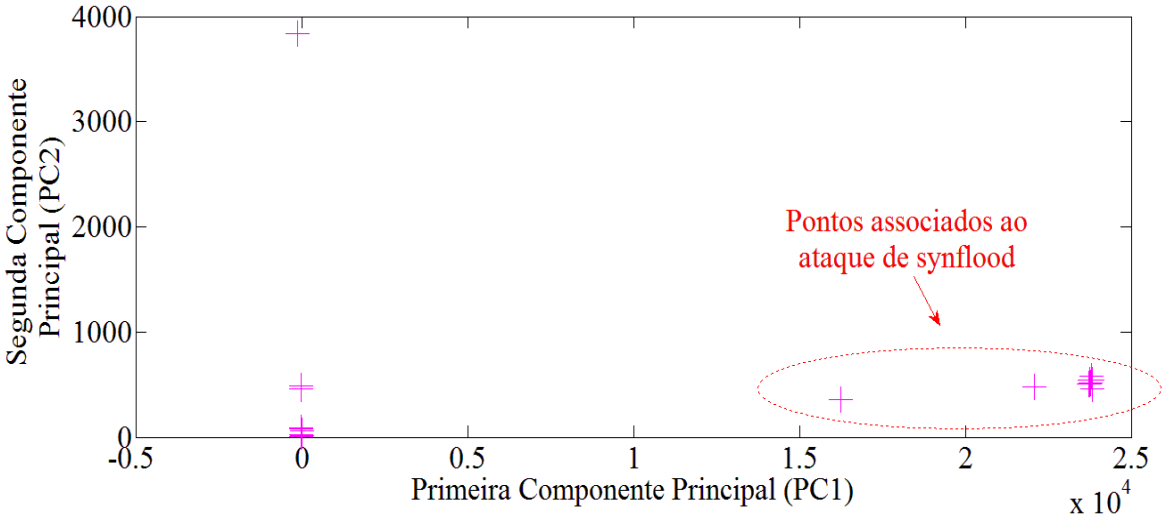


Figura 7.30 - As duas primeiras componentes principais (*synflood*).

Na Figura 7.30 é possível observar que a variância de PC1 é totalmente dominada pelas

componentes do ataque, ou seja, essas componentes são as responsáveis pelo elevado valor de autovalor associado a esta componente principal, consequentemente ao conjunto de valores do período q = 4. Além disso, conforme discutido na Seção 4, a variância de PC1 é igual ao maior autovalor referente à matriz $\mathbf{S}_{xx}^{(4)}$.

7.6.2 – Componentes principais para análise do ataque de fraggle

A partir da decomposição em autovalores da matriz $\mathbf{S}_{xx}^{(5)}$, é possível se obter os autovalores, bem como os autovetores, referentes a essa matriz.

$$\begin{bmatrix} 0{,}0044 & 0{,}9999 \\ 0{,}0002 & 0{,}0086 \\ 0{,}0001 & 0{,}0124 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -1{,}0000 & 0{,}0044 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Figura 7.31 - Autovetores dos dois maiores autovalores da matriz $\mathbf{S}_{xx}^{(5)}$.

A primeira coluna da matriz representada na Figura 7.31 representa o autovetor correspondente ao maior autovalor de $\mathbf{S}_{xx}^{(5)}$, enquanto a segunda coluna representa o autovetor correspondente ao segundo maior autovalor de $\mathbf{S}_{xx}^{(5)}$. Dessa forma:

$$\begin{aligned} \mathrm{PC1}_{fraggle} = {} & 0{,}0044 * \mathrm{tcp80} + 0{,}0002 * t\mathrm{cp443} + \\ & 0{,}0001 * \mathrm{udp45} - 1{,}000 * \mathrm{udp19}, \end{aligned} \tag{7.3}$$

onde tcp80, tcp443, udp53 e tcp600 representam as variáveis referentes às respectivas portas de comunicação.

$$\begin{aligned} \mathrm{PC2}_{fraggle} = {} & 0{,}9999 * \mathrm{tcp80} + 0{,}0086 * \mathrm{tcp443} + \\ & 0{,}0124 * \mathrm{udp53} + 0{,}0044 * \mathrm{udp19} \end{aligned} \tag{7.4}$$

Teoricamente, as componentes principais seriam representadas como uma combinação linear de todas as variáveis, porém nas Equações 7.3 e 7.4 não constam as demais variáveis pelo fato dos coeficientes das variáveis serem nulos, oriundos das informações dos autovetores. Isso é facilmente explicado pelo fato de, nesse período de tempo, não haver tráfego relacionado às portas dos ataques de *synflood*, tampouco *portscan*. O tráfego referente ao ruído é irrisório, fato que não aparece sequer nas duas primeiras componentes principais.

Com a obtenção das Equações 7.3 e 7.4, a próxima etapa é calcular o valor de cada PC para cada tempo de amostragem. Para isso, deve-se selecionar cada coluna da matriz $\mathbf{X}^{(5)}$ e aplicar os respectivos valores de tráfego relacionados às portas nas expressões 7.3 e 7.4. Com isso, serão obtidos 20 valores para $\mathrm{PC1}_{synflood}$ e 20 valores para $\mathrm{PC2}_{synflood}$, gerando um total de 20 pontos. Na figura abaixo é possível observar tais pontos sob os eixos perpendiculares que representam as componentes principais.

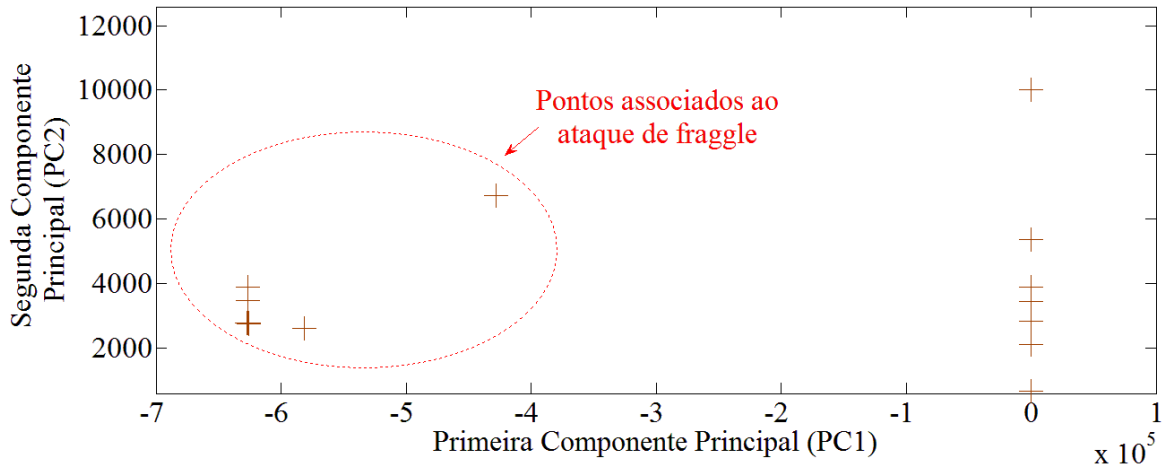


Figura 7.32 - As duas primeiras componentes principais (*fraggle).*

Na Figura 7.32 é possível observar que a variância de PC1 é totalmente dominada pelas componentes do ataque, ou seja, essas componentes são as responsáveis pelo elevado valor de autovalor associado a esta componente principal, consequentemente ao conjunto de valores do período q = 5. Além disso, conforme visto na Seção 4, a variância de PC1 é igual ao maior autovalor referente à matriz $\mathbf{S}_{xx}^{(5)}$.

7.6.3 – Componentes principais para análise do ataque de portscan

A partir da decomposição em autovalores da matriz $\mathbf{R}_{xx}^{(3)}$, é possível se obter os autovalores, bem como os autovetores, referentes a essa matriz.

$$\begin{bmatrix} 0{,}0345 & 0{,}2770 \\ 0{,}0488 & 0{,}6243 \\ 0{,}0667 & 0{,}7251 \\ -0{,}3150 & 0{,}0281 \\ -0{,}3150 & 0{,}0281 \\ -0{,}3150 & 0{,}0281 \\ -0{,}3150 & 0{,}0281 \\ -0{,}3150 & 0{,}0281 \\ -0{,}3150 & 0{,}0281 \\ -0{,}3150 & 0{,}0281 \\ -0{,}3150 & 0{,}0281 \\ -0{,}3150 & 0{,}0281 \\ -0{,}3150 & 0{,}0281 \end{bmatrix}$$

Figura 7.33 - Autovetores dos dois maiores autovalores da matriz $\mathbf{R}_{xx}^{(3)}$.

A partir dos autovetores da Figura 7.33, obtém-se:

$$\begin{gathered} \mathrm{PC1}_{portscan} = 0{,}0345 * \mathrm{tcp80} + 0{,}0488 * \mathrm{tcp443} + \\ 0{,}0667 * \mathrm{udp53} - 0{,}3150 * \mathrm{tcp21} - 0{,}3150 * \mathrm{tcp22} - 0{,}3150 * \mathrm{tcp23} \\ -0{,}3150 * \mathrm{tcp25} - 0{,}3150 * \mathrm{tcp110} - 0{,}3150 * \mathrm{tcp143} - 0{,}3150 * \mathrm{tcp161} \\ -0{,}3150 * \mathrm{udp69} - 0{,}3150 * \mathrm{udp123} - 0{,}3150 * \mathrm{udp445}, \end{gathered} \tag{7.5}$$

onde tcp80, tcp443, udp53, tcp21, tcp22, tcp23, tcp 25, tcp110, tcp143, tcp161, udp69, udp123 e udp445 representam as variáveis referentes às respectivas portas de comunicação.

$$\begin{gathered} \mathrm{PC2}_{portscan} = 0{,}2770 * \mathrm{tcp80} + 0{,}6243 * \mathrm{tcp443} + \\ 0{,}7251 * \mathrm{udp53} + 0{,}0281 * \mathrm{tcp21} + 0{,}0281 * \mathrm{tcp22} + 0{,}0281 * \mathrm{tcp23} \\ +0{,}0281 * \mathrm{tcp25} + 0{,}0281 * \mathrm{tcp110} + 0{,}0281 * \mathrm{tcp143} + 0{,}0281 * \mathrm{tcp161} \\ +0{,}0281 * \mathrm{udp69} + 0{,}0281 * \mathrm{udp123} + 0{,}0281 * \mathrm{udp445} \end{gathered} \tag{7.6}$$

Teoricamente, as componentes principais seriam representadas como uma combinação linear de todas as variáveis, porém nas Equações 7.5 e 7.6 não constam as demais variáveis pelo fato dos coeficientes das variáveis serem nulos, oriundos das informações dos autovetores. Isso é facilmente explicado pelo fato de, nesse período de tempo, não haver tráfego relacionado às portas dos ataques de *synflood*, tampouco *fraggle*. O tráfego referente ao ruído é inexistente nesse período de tempo, dessa forma também não há contribuição das portas udp 67 e udp 68.

Com a obtenção das Equações 7.5 e 7.6, a próxima etapa é calcular o valor de cada PC para cada tempo de amostragem. Para isso, deve-se selecionar cada coluna da matriz normalizada de $\mathbf{X}^{(5)}$ e aplicar os respectivos valores de tráfego relacionados às portas nas

expressões 7.5 e 7.6. Com isso, serão obtidos 20 valores para $PC1_{portscan}$ e 20 valores para $PC2_{portscan}$, gerando um total de 20 pontos. Na figura abaixo é possível observar tais pontos sob os eixos perpendiculares que representam as componentes principais.

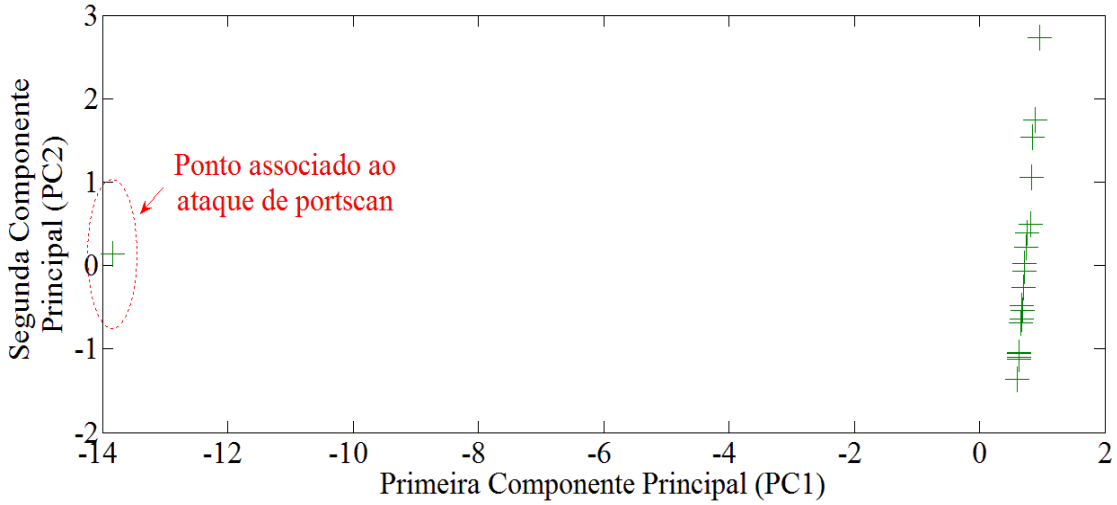


Figura 7.34 - As duas primeiras componentes principais (*portscan*).

Na Figura 7.34 é possível observar que a variância de PC1 é totalmente dominada pelas componentes do ataque (representado por apenas um ponto), ou seja, essas componentes são as responsáveis pelo elevado valor de autovalor associado a esta componente principal, consequentemente ao conjunto de valores do período q = 3. Além disso, conforme visto na Seção 4, a variância de PC1 é igual ao maior autovalor referente à matriz $\mathbf{R}_{xx}^{(3)}$.

7.7 – APLICAÇÃO DOS ESQUEMAS DE MOS

Apesar de se poder observar visualmente o efeito do ataque através do PCA, é relevante se fazer a aplicação dos esquemas de MOS, a fim de tornar o processo automatizado, levando em conta o perfil dos autovalores.

De acordo com o fluxograma disposto na Figura 6.8, uma vez obtido o vetor VTMA é possível se aplicar os esquemas de MOS, a fim de se estimar a ordem do modelo. A Tabela 7.4 apresenta os resultados obtidos com a utilização dos seguintes esquemas de MOS: AIC, MDL, EDC, RADOI, EFT e SURE.

Com os resultados obtidos, apresentados abaixo na Tabela 7.4, observa-se que dois esquemas se destacaram dos demais. O *Efficient Detection Criterion* (EDC) e o *Exponential Fitting Test* (EFT) foram os mais coerentes, retornando valor maior ou igual a um, indicando que houve um ataque e valor nulo na ausência de ataque.

Os métodos AIC e MDL mostram-se satisfatórios apenas na detecção do *portscan*. Os métodos RADOI e SURE não mostraram resultados coerentes para nenhum dos casos.

O valor da ordem do modelo igual a um é esperado quando há o ataque, pois o mesmo representa a presença da componente principal, a componente que se destaca das demais, sendo neste estudo referenciada pelo ataque.

Quando o esquema retorna valor maior que um é porque foram detectados mais de um ataque. Um exemplo disso pôde ser observado quando se agrupam os autovalores referenciados aos ataques de *synflood* e *fraggle* num mesmo vetor VTMA, conforme mostrado na segunda coluna da Tabela 7.3. Este vetor carrega informações de dois ataques de negação de serviço. Devido a isso, os esquemas EDC e EFT retornaram valor igual a 2, indicando a presença dos dois ataques. De acordo com a modelagem do problema isso pode ser interpretado como um único ataque de negação de serviço que perdurou por mais de um período de tempo.

Tabela 7.4 - Aplicação dos esquemas de MOS nos vetores VTMA.

Tipo de análise	**Esquema de MOS**					
	AIC	**MDL**	**EDC**	**RADOI**	**EFT**	**SURE**
Detecção do *synflood* (presença do ataque)	2	1	1	5	1	4
Detecção do *synflood* (ausência do ataque)	1	1	0	1	0	3
Detecção do *fraggle* (presença do ataque)	1	1	1	5	1	4
Detecção do *fraggle* (ausência do ataque)	1	1	0	1	0	3
Detecção do *portscan* (presença do ataque)	1	1	1	1	1	9
Detecção do *portscan* (ausência do ataque)	0	0	0	1	0	1
Detecção do *synflood*/*fraggle* (presença do ataque)	2	2	2	5	2	5
Detecção do *synflood*/*fraggle* (ausência do ataque)	1	1	0	1	0	3

8 – CONCLUSÃO E TRABALHOS FUTUROS

Uma vez que a definição do problema desta dissertação trata de assunto atinente à detecção de ataques, foram apresentados inicialmente os principais conceitos e definições referentes à segurança da informação, para que se pudesse situar melhor o tema proposto no contexto atual.

O produto final deste trabalho é a criação de uma nova técnica, chamada de Variação Temporal do Maior Autovalor (VTMA), para a detecção dos ataques de *portscan*, *synflood* e *fraggle*. De uma forma mais genérica, a técnica pode ser aplicada a ataques envolvendo escaneamento de portas e ataques de negação de serviço, onde mostrou ser bastante eficaz.

Para isso, alguns conceitos matemáticos foram também introduzidos, dentre eles: autovalores e autovetores, matrizes, análise de componentes principais e seleção de ordem do modelo.

Antes de apresentar os resultados obtidos, foi destacada de forma detalhada a solução proposta, onde o leitor pôde se familiarizar com o cenário da simulação, detalhes de funcionamento dos tráfegos maliciosos em estudo, modelagem dos dados, como os dados foram obtidos e tratados e a forma como a técnica é aplicada para a detecção dos ataques. Todo o processo, desde a coleta dos dados até a detecção dos ataques, pôde ser visto na forma de um fluxograma, sintetizado.

Foi possível então aplicar a teoria desenvolvida e verificar os resultados experimentais, onde foram obtidas diversas matrizes de tráfego, de covariância, correlação, bem como autovalores e autovetores. Além disso, foi mostrado que, com as componentes principais referentes ao tráfego contendo cada ataque, é possível observar os pontos relativos aos ataques, conseguindo de forma visual detectá-los. Através do cálculo da variância na direção da componente principal obteve-se o valor relativo ao autovalor da matriz de tráfego na presença do ataque.

Com o intuito de fazer com que a detecção seja automática, foram empregados os esquemas de seleção de ordem do modelo, a fim de se estimar a ordem do modelo do problema. Através de algumas experimentações conclui-se que os esquemas EDC e EFT apresentaram os resultados mais coerentes para o problema proposto.

Este trabalho proporciona a base para diversos outros, tanto de cunho mais prático, voltado para o ambiente profissional, como de cunho acadêmico, voltado para pesquisas.

Visando pesquisas em detecção de ataques, a técnica do VTMA poderia ser testada em outras camadas do modelo OSI, uma vez que esta dissertação aborda tal técnica apenas na camada de transporte. Além disso, poder-se-ia combinar a técnica da VTMA com outras técnicas, como por exemplo mineração de dados e análise regular de arquivos, com o intuito de detectar ataques que fogem um pouco o comportamento dos mostrados neste trabalho.

Importante ressaltar que esta técnica pode ser empregada não só na detecção de tráfego malicioso, mas também em outros estudos, das mais diversas áreas do conhecimento. Assim, seria uma idéia interessante aplicá-la também não apenas para perícia computacional e segurança da informação, verificando assim o quão eficiente a técnica se comporta.

Do ponto de vista profissional, seria interessante a aplicação desta teoria na prática, através de testes contínuos em ambientes reais de produção, com o objetivo de comparar esta técnica com outras associadas a ferramentas já existentes no mercado e também para aumentar a confiança do método quando o mesmo é submetido a uma quantidade maior de dados. Para isso, alguns passos prévios seriam necessários, gerando diversos trabalhos novos, como por exemplo: programação na linguagem C do código de detecção dos ataques; elaboração de um *script* a fim de que se crie um serviço de rede (serviço de detecção de intrusão); integração do código em C com o *script* citado; e testes deste novo serviço em um ambiente real corporativo, verificando o quão eficaz é em relação a outros sistemas bem conhecidos no mercado e na literatura.

REFERÊNCIAS BIBLIOGRÁFICAS

E. T. Nakamura and P. L. de Geus, "*Segurança de Redes em Ambientes Cooperativos*," Novatec Editora Ltda, Brasil, 2010.

K. Salah, K. Elbadawi and R. Boutaba, "*Performance Modeling and Analysis of Network Firewalls*," in IEEE Transactions on Network and Service Management, vol. 9, pp. 12-21, March 2012.

D. Mudzingwa and R. Agrawal, "*A study of Methodologies used in Intrusion Detection and Prevention Systems (IDPS)*," in Proceedings of IEEE Southeastcon, pp. 1-6, March 2012.

W. He, G. Hu, X. Yao, G. Kan, H. Wang and H. Xiang, "*Applying multiple time series data mining to large-scale network traffic analysis*," in IEEE Conference on Cybernetics and Intelligent Systems, pp. 394-399, September 2008.

A. Ghourabi, T. Abbes and A. Bouhoula, "*Data analyzer based on data mining for honeypot route*r," in International Conference on Computer Systems and Applications (AICCSA), pp. 1-6, May 2010.

F. Raynal, Y. Berthier, P. Biondi and D. Kaminsky, "*Honeypot forensics*," in Proceedings from the Fifth Annual IEEE SMC on Information Assurance Workshop, pp. 22-29, June 2004.

S. Almotairi, A. Clark, G. Mohay and J. Zimmermann, "*A Technique for Detecting New Attacks in Low-Interaction Honeypot Traffic*," in Fourth International Conference on Internet Monitoring and Protection, pp. 7-13, May 2009.

R. Puttini, M. Hanashiro, F. Miziara, R. T. Sousa Júnior, L. J. García-Villaba and C. J. Barenco, "*On the Anomaly Intrusion-Detection in Mobile Ad Hoc Network Environments*," Personal Wireless Communications, Lecture Notes in Computer Science, Volume 4217, pp. 182-193, January 2006.

B. M. David, J. P. C. L. da Costa, A. C. A. Nascimento, M. D. Holtz, D. Amaral and R. T. Sousa Júnior, "*Blind automatic malicious activity detection in honeypot data*," in International Conference on Forensic Computer Science (ICoFCS), pp. 142-152, October. 2011.

J. P. C. L. da Costa, E. P. de Freitas, B. M. David, A. M. R. Serrano, D. Amaral and R. T. Sousa Júnior, "*Improved blind automatic malicious activity detection in honeypot*

data," in International Conference on Forensic Computer Science (ICoFCS), pp. 46-45, Sepember. 2012.

W.Z.A. Zakaria and M.L.M. Kiah, "*A review on artificial intelligence techniques for developing intelligent honeypot*," in 8th International Conference on Computing Technology and Information Management (ICCM), vol. 2, pp. 696,701, 24-26, April 2012.

D.X. Dan, D.Y. Ming, Y. Tao and L. Rong, "*Evaluation of AR model order selection approaches*," in International Forum on Information Technology and Applications (IFITA), vol. 1, pp. 704-707, May 2009.

H. F. Lipson and D. A. Fisher, "*Survivability – A New Technical and Business Perspective on Security*," in Internet Computing, IEEE, vol. 3, pp. 55-63, August1999.

W. Stallings, "*Cryptography and Network Security – Principles and Practices*," Fourth Edition, Prentice Hall, USA, 2005.

J. Gadge and A. A. Patil, "*Port Scan Detection*," in 16th *IEEE* International Conference on Networking (ICON), pp. 1-6, December 2008.

N. Y. Hamisi, N.H. Mvungi, D. A. Mfinanga and B. M. M. Mwinyiwiwa, "*Intrussion detection by penetration test in an organization network*," in 2nd International Conference on Adaptive Science and Technology (ICAST), pp. 226-231, January 2009.

A. Sengupta, C. Mazumdar and A. Bagchi, "*A Formal Methodology for Detection of Vulnerabilities in an Enterprise Information System*," in Conference on Risks and Security of Internet and Systems (CRiSIS), pp. 74-80, October 2009.

D. M. Martins, "*Uma Estratégia para Sistemas de Detecção e Prevenção de Intrusão Baseada Em Software Livre*," Dissertação de Mestrado Acadêmico pelo Departamento de Ciência da Computação da Universidade Federal do Ceará, 2012.

S. Harris, "*All in One*," Fifth Edition, McGraw-Hill, USA, 2010.

N. Pavkovic and L Perkov, "*Social Engineering Toolkit – A Systematic Approach To Social Engineering*," in 34th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 1485-1489, May 2011.

S. Ansari, S. G. Rajeev and H. S. Chandrashekar, "*Packet Sniffing: A Brief Introduction*," in Potentials, IEEE, vol. 21, pp. 17-19, January 2003.

D. Goldsmith and M. Schiffman, "*Firewalking – A Traceroute-Like Analysis of IP Packet Responses to Determine Gateway Access Control Lists*," Cambridge Technology Partners Enterprise Security Services, Inc., October 1998.

R. Robinson and C. Thomas, "*Evaluation of Mitigation Methods for Distributed Denial of Service Attacks*," in 7th IEEE Conference on Industrial Electronics and Applications (ICIEA), pp. 713-718, July 2012.

M. F. Avolio, "*Information Security. Cover Story. Firewalls: Are we Asking Too Much?*,"
May 1999. Disponível em: http://www.avolio.com/papers/FirewallsAsking2Much.html

W. R. Cheswick and S. M. Bellovin, "*Repelling the Wily Hacker*," Addison-Wesley, April 1994.

E. D. Zwicky, S. Cooper and D. B. Chapman, "*Building Intenet Firewalls*," O'Reilly Media, Second Edition, 2000.

S. Thukral, R. Maqsood and D. Upadhyay, "*To Design an Intrusion Detection System based on Honeypot using Mobile Agent and IP Traceback Technique*," in International Journal of Science and Research (IJSR), vol. 2, April 2013.

D. Stiawan, A. H. Adbullah and M. Y. Idris, "*Characterizing Network Intrusion Prevention System*," in International Journal of Computer Applications, vol. 14, January 2011.

S.X. Wu and W. Banzhaf, "*The use of computational intelligence in intrusion detection systems: A review*," in Applied Soft Computing, vol. 10, pp. 1-35, January 2010.

I. Mukhopadhyay, M. Chakraborty and S. Chakrabarti, "*A Comparative Study of Related Technologies of Intrusion Detection & Prevention Systems*," in Journal of Information Security, vol. 2, pp. 28-38, January 2011.

M. J. Ranum, "*NFR Security: Coverage in Intrusion Detection Systems*," August 2001. Disponível em: http://bandwidthco.com/whitepapers/compforensics/ids/ Benchmarking IDS.pdf

K. Scarfone and P. Mell, "*Guide to Intrusion Detection and Prevention Systems (IDPS)*," in National Institute of Standards and Technology, February 2007.

J. Allen, A Christie, W Fithen, J. McHugh, J. Pickel and E. Stoner, "*State of the Practice of Intrusion Detection Technologies*," in Software Engineering Institute at Carnegie Mellon, January 2000.

Z. L. Juan, "*Honeypot-based Defense System Research and Design*," in 2nd IEEE International Conference on Computer Science and Information Technology (ICCSIT), pp. 466-470, August 2009.

C. Hoepers, K. S. Jessen and M. H. P. C. Chaves, "*Honeypots e Honeynets: Definições e Aplicações*" in Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil, ver. 1.1, August 2007. Disponível em: http://www.cert.br/docs/whitepapers/honeypots-honeynets/

S. Northcutt and J. Novak, "*Network Intrusion Detection*," Third Edition, New Riders Publishing, USA, 2002.

I. Kotenko, O. Polubelova, and I. Saenko, "*The Ontological Approach for SIEM Data Repository Implementation*," in IEEE International Conference on Green Computing and Communications (GreenCom), pp. 761-766, November 2012.

M. Afzaal, C. D. Sarno, S. D'Antonio and L. Romano, "*An Intrusion and Fault Tolerant Forensic Storage for a SIEM System*," in 8th International Conference on Signal Image Technology and Internet Based Systems (SITIS), pp. 579-586, November 2012.

D. R. Miller, S. Harris, A. A. Harper, S. VanDyke and C. Blask, "*Security Information and Event Management (SIEM) Implementatio*," McGraw-Hill, USA, 2011.

T. M. Apostol, "*Calculus*," Segunda Edición, Editorial Reverté S. A., Spain, 1975.

R. Vershynin, "*How Close is the Sample Covariance Matrix to the Actual Covariance Matrix?,"* in J Theor Probab, pp. 655-686, 2012.

H. Krim and M. Viberg, "*Two Decades of Array Signal Processing Research: the parametric approach*," in Signal Processing Magazine, IEEE, vol. 13, pp. 67-94, July 1996.

A. Cichocki, R. Zdunek, A. H. Phan and S. I. Amari, "*Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*," First Edition, Wiley, USA, 2009.

R. A. Johnson and D. W. Wichern, "*Applied Multivariate Statistical Analysis*," Sixth Edition, Prentice Hall, USA, 2007.

A.C. Rencher, "*Methods of Multivariate Analysis*," Second Edition, Wiley, USA, 2002.

J. P. C. L. da Costa, A. Thakre, F. Röemer and M. Haardt, "*Comparison of model order selection techniques for high-resolution parameter estimation algorithms*," in 54th International Scientific Colloquium (IWK), pp. 07-10, October 2009.

J. J. Rajan and P. J. W. Rayner, "*Model order selection for the singular value decomposition and the discrete Karhunen-Loève transform using a Bayesian approach*," in IEEE Proceedings Vision, Image and Signal Processing, vol. 144, pp. 116-123, April 1997.

D.X. Dan, D. Y. Ming, Y. Tao and L. Rong, "*Evaluation of AR Model Order Selection Approaches*," in International Forum on Information Technology and Applications (IFITA), vol. 1, pp. 704-707, May 2009.

J. P. C. L. da Costa, M. Haardt, A. Thakre, F. Röemer and G. D. Galdo, "*Enhanced Model Order Estimation Using Higher-Order Arrays*," in Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers (ACSSC), pp. 412-416, November 2007.

A. Quinlan, J. P. Barbot, P. Larzabal and M. Haardt, "*Model order selection for short data: An exponential fitting test (EFT)*," in EURASIP Journal on Applied Signal Processing, 2007.

H. Akaike, "*A New Look at the Statistical Model Identification*," in IEEE Transactions on Automatic Control, vol. 19, pp. 716-723, December 1974.

M. Wax and T. Kailath, "*Detection of signals by information theoretic criteria*," in IEEE Transactions on Acoustics, Speech and Signal Processing (ASSP), vol 33, pp. 387-392, April 1985.

L. C. Zhao, P. R. Krishnaiah and Z. D. Bai, "*On detection of the number of signals in* presence of white noise," in Journal of Multivariate Analysis, 1986.

M. O. Ulfarsson and V. Solo, "*Rank selection in noisy PCA with SURE and random matrix theory*," in International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3317-3320, April 2008.

E. Radoi and A. Quinquis, "*A new method for estimating the number of harmonic components in noise with application in high resolution radar*," in EURASIP Journal on Applied Signal Processing, pp. 1177 – 1188, 2004.

J. P. C. L. da Costa, F. Roemer, F. A. de Castro Junior, R. F. Ramos, S. Schwarz and L. Sabirova, "*Ilmenau package for model order selection and evaluation of model order estimation scheme of users of MIMO channel sounders*," in XXIX Simpósio Brasileiro de Telecomunicações (SBrT), October 2011.

J. P. C. L. da Costa, "*Parameter estimation techniques for multidimensional array signal processing*," First Edition, Shaker Publisher, March 2010.

J. E. M. Filho, "*Análise de Tráfego em Redes TCP/IP: Utilize tcpdump na análise de tráfegos em qualquer sistema operacional*," Novatec Editora, Brasil, 2013.

TRABALHOS PUBLICADOS PELO AUTOR

D. F. Tenório, J. P. C. L. da Costa, and R. T. Sousa Júnior, “Greatest Eigenvalue Time Vector Approach for Blind Detection of Malicious Traffic,” in International Conference on Forensic Computer Science (ICoFCS), Brasília, Brazil, 2013, best paper award.

D. F. Tenório, J. P. C. L. da Costa, E. P. de Freitas, e R. T. Sousa Júnior, “Detecção Cega de Tráfego Malicioso Através da Variação Temporal do Maior Autovalor,” em XIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg), Manaus, Brazil, 2013.

APÊNDICES

A – CÓDIGOS DO MATLAB

No apêndice são apresentados os códigos em MATLAB desenvolvidos pelo autor, a fim de auxiliar com os cálculos matemáticos.

A.1 – Amostras

O objetivo deste código é de se obter cada elemento $x_{m,n}$, que representa o número de vezes que a porta *m* aparece no *n*-ésimo período de amostragem.

1	`j=0;`	Inicialização de variáveis
2	`w=0;`	
3	`z=1;`	
4	`k=1;`	
5	`[a,b]=size(x);`	Cálculo do tamanho do vetor x
6	`for k = 1:120`	Início da rotina *for*
7	`    t=1;`	Inicialização de variáveis
8	`    y=0;`	
9	`    if j == 60`	Condicional caso o tempo de amostragem seja de 60 minutos
10	`        j=0;`	Inicialização de variável
11	`        while z <= a && x(z) == j`	Rotina *while* executada enquanto ainda existirem amostras
12	`        y(t) = x(z);`	Mudança de variável
13	`        z=z+1;`	Incremento de variáveis
14	`        t=t+1;`	
15	`        end`	Fim da rotina *while*
16	`        j=j+1;`	Incremento de variável
17	`        w(k)=t-1;`	Formação do vetor *w*
18	`        k=k+1;`	Incremento de variável
19	`    else`	Caso o tempo de amostragem seja diferente de 60 minutos
20	`        while z <= a && x(z) == j`	Rotina *while* executada enquanto ainda existirem amostras
21	`        y(t) = x(z);`	Mudança de variável
22	`        z=z+1;`	Incremento de variáveis
23	`        t=t+1;`	
24	`        end`	Fim da rotina *while*
25	`        j=j+1;`	Incremento de variável
26	`        w(k)=t-1;`	Formação do vetor *w*
27	`        k=k+1;`	Incremento de variável
28	`end`	Fim da rotina *for*
29	`w`	Resultado para cada tempo de amostragem

A.2 – Amostras dentro de cada período de tempo

O objetivo deste código é de se obter cada elemento $x_{m,n}^{(q)}$, que representa o número de vezes que a porta *m* aparece no *n*-ésimo instante, dentro do *q*-ésimo período de tempo.

1	`j=0;`	Inicialização de variável
2	`for i=1:6`	Início da rotina *for*
3	`    j=j+20;`	Incremento de variável
4	`    Z(i,:)= w(j-19:j);`	Mudança de variável
5	`end`	Fim da rotina *for*
6	`A=Z(1,:);`	$x_{m,n}^{(1)}$
7	`B=Z(2,:);`	$x_{m,n}^{(2)}$
8	`C=Z(3,:);`	$x_{m,n}^{(3)}$
9	`D=Z(4,:);`	$x_{m,n}^{(4)}$
10	`E=Z(5,:);`	$x_{m,n}^{(5)}$
11	`F=Z(6,:);`	$x_{m,n}^{(6)}$

A.3 – Gráficos do sinal, ruído e ataque

O objetivo deste código é de criar os gráficos do sinal legítimo, ruído e ataques.

1	`b=1;`	Inicialização de variáveis
2	`y=0;`	
3	`j=1;`	
4	`m(1)=0;`	
5	`while j~=121;`	Rotina *while* executada enquanto ainda existirem amostras
6	`      for i=j:(j+9)`	Início da rotina *for*
7	`          y = w(i)+ y;`	Incremento de variável
8	`      end`	Fim da rotina *for*
9	`      m(b+1)=y;`	Formação da variável *m*
10	`      b=b+1;`	Incremento de variáveis
11	`      j=j+10;`	
12	`      y=0;`	Inicialização de variável
13	`end`	Fim da rotina *while*
14	`d(1) = 0;`	Inicialização de variável
15	`for n = 1:12`	Início da rotina *for*
16	`    d(n+1)= n*10;`	Estabelecimento dos intervalos de tempo (de 10 em 10 minutos)
17	`end`	Fim da rotina *for*
18	`plot(m,d)`	Desenho do gráfico

A.4 – Parânetros associados à correlação

O objetivo deste código é de se calcular de uma só vez os parâmetros de interesse relacionados à correlação dos dados: matriz de correlação (R), autovetores da matriz de correlação (E), autovalores da matriz de correlação (V), máximo autovalor associado à matriz de correlação (M) e normalização de dados (Y).

```
function [R,E,V,M,Y] = correlacao(X)
         for i=1:17
             Y(i,:)=(X(i,:)-mean(X(i,:)))/std(X(i,:),1);
         end
R=1/20*Y*Y';
[V,E]=eig(R);
M=max(diag(E));
```

A.5 – Parâmetros associados à covariância

O objetivo deste código é de se calcular de uma só vez os parâmetros de interesse relacionados à covariância dos dados: matriz de covariância (S), autovetores da matriz de covariância (E), autovalores da matriz de covariância (V), máximo autovalor associado à matriz de covariância (M) e normalização de dados (Y).

```
function [S,E,V,M] = covariancia(X)
   for i=1:17
       Y(i,:)=(X(i,:)-mean(X(i,:)));
   end
S=1/20*Y*Y';
[V,E]=eig(S);
M=max(diag(E));
```

ANEXOS

A – CÓDIGOS DO MATLAB

No anexo são apresentados os códigos em MATLAB relacionados aos esquemas de MOS utilizados no auxílio da estimação da ordem do modelo[1].

A.1 – Akaike's Information Theoretic Criterion - AIC

```
function [usernumber] = akaike_short2(eig_vec,N);
[temp,M] = size(eig_vec);
M = max(temp,M);
eig_vec = eig_vec.';
eig_vec = sort(eig_vec,'ascend');
for ii = 1:M
    kk = M-ii+1;
    trecho = eig_vec(1:kk);
    gii = geomean(trecho);
    aii = mean(trecho);
    mm = M-kk;
    AIC_values(ii) = -N*(M-mm)*log10(gii/aii)+mm*(2*M-mm);
end
[AIC_min,position] = min(AIC_values(1:M));
usernumber = position - 1;
```

A.2 – Minimum Description Length - MDL

```
function [usernumber] = mdl_short2(eig_vec,N);
[temp,M1] = size(eig_vec);
M1 = max(temp,M1);
M = M1;
eig_vec = eig_vec.';
eig_vec = sort(eig_vec,'ascend');
for ii = 1:M1
    kk = M1-ii+1;
    trecho = eig_vec(1:kk);
    gii = geomean(trecho);
    aii = mean(trecho);
    mm = M1-kk;
    MDL_values(ii) = -N*(M-mm)*log10(gii/aii)+0.5*mm*(2*M-
mm)*log10(N);
end
[MDL_min,position] = min(MDL_values(1:M1));
usernumber = position-1;
```

[1] J. P. C. L. da Costa, F. Roemer, F. A. de Castro Junior, R. F. Ramos, S. Schwarz, and L. Sabirova, "*Ilmenau Package for Model Order Selection and Evaluation of Model Order Estimation Scheme of Users of MIMO Channel Sounders*," XXIX Simpósio Brasileiro de Telecomunicações (SBrT'11), Curitiba, Brazil.

A.3 – Efficient Detection Criterion - EDC

```
function [usernumber] = edc_short2(eig_vec,N);
[temp,M] = size(eig_vec);
M = max(temp,M);
eig_vec = eig_vec.';
eig_vec = sort(eig_vec,'ascend');
c_N = sqrt(N*log(log(N)));
for ii = 1:M
    kk = M-ii+1;
    trecho = eig_vec(1:kk);
    gii = geomean(trecho);
    aii = mean(trecho);
    mm = M-kk;
    EDC_values(ii) = -N*(M-mm)*log10(gii/aii)+0.5*mm*(2*M-mm)*c_N;
End
[EDC_min,position] = min(EDC_values(1:M));
usernumber = position-1;
```

A.4 – RADOI

```
function d_est = ranoi_app(eig_values);
eig_values = sort(eig_values,'descend');
eig_num = length(eig_values);
for kk = 1:(eig_num-1)
    ii = eig_num - kk + 1;
    trecho = eig_values(1:ii);
    gii = geomean(trecho);
    aii = mean(trecho);
    mu_fct(kk) = geomean(trecho)/mean(trecho);
end
for kk = 1:(eig_num-1)
    trecho = eig_values(1:kk);
    alpha_var(kk) = (geomean(trecho) - mu_fct(kk))/mu_fct(kk);
end
[temp,alpha_const] = max(alpha_var);
for kk = 1:(eig_num-1)
    trecho = eig_values(1:kk);
    ksi(kk) = 1-alpha_const*(geomean(trecho)-mu_fct(kk))/mu_fct(kk);
end
for kk = 1:(eig_num-1)
    trecho = eig_values(1:kk);
    discr_1(kk) = geomean(trecho)/mean(trecho);
    discr_2(kk) = ksi(kk)/sum(ksi(1:(eig_num-1)));
    discr_crit(kk) = discr_1(kk) - discr_2(kk);
end
[temp2,d_est]= min(discr_crit);
```

A.5 – Exponential Fitting Test - EFT

Cálculo dos coeficientes: EFT_CALC_COEF

```
function [local_coeff,prob_found]=calc_coef_paper(M,N,Pfa,coeff,q);
max_eig_numb = min([M N]);
for pp = 1:(max_eig_numb-1)
    pp_aux = pp+1;
    a_mi_ff = 225/((pp_aux^2+2)^2);
    a_mi_fs = 180*pp_aux/(N*(pp_aux^2-1)*(pp_aux^2+2));
    a_mi = (a_mi_ff-a_mi_fs)^0.5;
    a_ex_fr = 15/(pp_aux^2+2);
    a_diff = a_ex_fr-a_mi;
    a_final(pp_aux) = (0.5*a_diff)^0.5;
    r_final(pp_aux) = exp(-2*a_final(pp_aux));
    J_final(pp_aux)=(1-r_final(pp_aux))/(1-r_final(pp_aux)^pp_aux));
End
barra = waitbar(0,'Please wait... Calculating thresholds for EFT');
for ii = 1:q
    noise = randn(M,N);
    noise_eig = (svd(noise).^2)/N;
    noise_eig = sort(noise_eig,'descend');
    for pp = 1:(max_eig_numb-1)
        pp_aux = pp+1;
        sum_eig = sum(noise_eig((max_eig_numb-pp):max_eig_numb));
        comp_q = noise_eig(max_eig_numb-pp)/sum_eig;
        comp_q_tot = comp_q/J_final(pp_aux)-1;
        prob_q(pp,:) = coeff<comp_q_tot;
        if (ii == 1)
            prob_fim(pp,:) = 1*prob_q(pp,:);
        else
            prob_fim(pp,:)=((ii-1)*prob_fim(pp,:)+prob_q(pp,:))/ii;
        End
    end
    waitbar(ii/q,barra);
end
close(barra);
for pp = 1:(max_eig_numb-1)
    [temp,local_Pfa(pp)] = min(abs(prob_fim(pp,:)-Pfa));
    local_coeff(pp) = coeff(local_Pfa(pp));
    if (local_Pfa(pp) == 1)
        prob_found(pp) = 0;
    Else
        prob_found(pp) = prob_fim(pp,local_Pfa(pp)-1);
    end
end
```

```
function usernumber = eft_short(eig_values,eft_coeff,M,N);
numb_eig = min([M N]);
for pp = 1:(numb_eig-1)
    pp_aux = pp + 1;
    a_mi_ff = 225/((pp_aux^2+2)^2);
    a_mi_fs = 180*pp_aux/(N*(pp_aux^2-1)*(pp_aux^2 +2));
    a_mi = (a_mi_ff-a_mi_fs)^0.5;
    a_ex_fr = 15/(pp_aux^2+2);
    a_diff = a_ex_fr-a_mi;
    a_final(pp_aux) = (0.5*a_diff)^0.5;
    r_final(pp_aux) = exp(-2*a_final(pp_aux));
    J_final(pp_aux)=(1-r_final(pp_aux))/(1-r_final(pp_aux)^pp_aux));
end
flag_noise = 1;
usernumber = 0;
eig_values = sort(eig_values,'descend');
for pp = 1:(numb_eig-1)
    if (flag_noise == 1)
        pp_aux = pp+1;
        sum_eig = sum(eig_values((numb_eig-pp):numb_eig));
        comp_q = eig_values(numb_eig-pp)/sum_eig;
        comp_q_tot = comp_q/J_final(pp_aux)-1;
        if  (eft_coeff(pp) >= comp_q_tot )
        Else
            usernumber = numb_eig-pp;
            flag_noise = 0;
        end
    end
end
```

A.6 – Stein's Unbiased Risk Estimator - SURE

```
function d_est = sure_method(X_mat,M,N);
sing_values = svd(X_mat);
eig_values = sing_values.^2;
[sigma_rmt] = noise_est_sure(eig_values,M,N);
max_d = min(M,N);
other_dim = max(M,N);
for ii = 1:(max_d-1)
    sigma_r_sqr = (1/(max_d-ii))*sum(eig_values((ii+1):max_d));
    C_part_one = 0;
    C_part_three = 0;
    for jj = 1:ii
        for kk = (ii+1):max_d
            C_part_one = (4*sigma_rmt/other_dim)*(eig_values(jj)-
sigma_r_sqr)/(eig_values(jj)-eig_values(kk))+C_part_one;
        End
        C_part_three = -(2*sigma_rmt/other_dim)*(max_d-1)*(1-
sigma_r_sqr/eig_values(jj))+C_part_three;
    end
    C_part_two = (2*sigma_rmt/other_dim)*ii*(ii-1);
    C_total = C_part_one+C_part_two+C_part_three;
    Risk_part_one = (max_d-ii)*sigma_r_sqr;
    Risk_part_two = 0;
    Risk_part_three = 2*sigma_rmt*ii;
    Risk_part_four = 0;
    Risk_part_five = 0;
    for jj = 1:ii
        Risk_part_two = (sigma_r_sqr^2)/eig_values(jj)+
Risk_part_two;
        Risk_part_four = (-2*sigma_rmt*sigma_r_sqr)/eig_values(jj)+
Risk_part_three;
        Risk_part_five =
(4*sigma_rmt*sigma_r_sqr/other_dim)/eig_values(jj)+Risk_part_four;
    end
    Risk_total(ii) = Risk_part_one+Risk_part_two+Risk_part_three+
Risk_part_four+Risk_part_five+C_total;
end
[value,d_est] = min(Risk_total);
```