UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

APLICANDO A TRANSFORMADA WAVELET BIDIMENSIONAL NA DETECÇÃO DE ATAQUES WEB

DISSERTAÇÃO DE MESTRADO

Bruno Augusti Mozzaquatro

Santa Maria, RS, Brasil

2012

APLICANDO A TRANSFORMADA WAVELET BIDIMENSIONAL NA DETECÇÃO DE ATAQUES WEB

por

Bruno Augusti Mozzaquatro

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do grau de
Mestre em Computação

Orientador: Prof. Dr. Raul Ceretta Nunes (UFSM)

Co-orientadora: Profª. Drª. Alice de Jesus Kozakevicius (UFSM)

Santa Maria, RS, Brasil

2012

Universidade Federal de Santa Maria
Centro de Tecnologia
Programa de Pós-Graduação em Informática

A Comissão Examinadora, abaixo assinada,
aprova a Dissertação de Mestrado

APLICANDO A TRANSFORMADA WAVELET BIDIMENSIONAL NA DETECÇÃO DE ATAQUES WEB

elaborada por
Bruno Augusti Mozzaquatro

como requisito parcial para obtenção do grau de
Mestre em Computação

COMISSÃO EXAMINADORA:

Raul Ceretta Nunes (UFSM), Dr.
(Presidente/Orientador)

Christian Emilio Schaerer Serra, Prof. Dr. (UNA)

Osmar Marchi dos Santos, Prof. Dr. (UFSM)

Santa Maria, 27 de fevereiro de 2012.

.

Dedico aos meus pais Claurindo e Aneli.

AGRADECIMENTOS

Agradeço a minha família pelo apoio e confiança durante toda a minha vida acadêmica, sempre me motivando e apoiando nos momentos difíceis da minha caminhada. Também à minha namorada Karine pela companhia, compreensão e motivação durante o desenvolvimento deste trabalho.

A CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) pelo apoio financeiro fundamental para a realização dos trabalhos.

Aos integrantes dos grupos de pesquisa Gestão e Tecnologia em Segurança da Informação (GTSeg), Grupo de Microeletrônica (GMicro) e Grupo de Sistemas de Computação Móvel (GMob) pela acolhida.

Em especial, ao meu orientador Prof. Dr. Raul Ceretta Nunes, que desde o primeiro momento demonstrou um enorme esforço e disposição para ajudar, além da orientação e apoio para a realização deste trabalho.

Em especial, à co-orientadora deste trabalho Profa. Dra. Alice de Jesus Kozakevicius pela orientação no decorrer do desenvolvimento dos trabalhos realizados.

Aos colegas Francisco Vogt, Giani Petri, Renato Preigschadt de Azevedo, Ricardo Macedo e Victor Alves pela amizade e companheirismo durante o andamento dos trabalhos realizados. Agradeço ao colega Cristian Cappo pelas incansáveis ajudas e trocas de conhecimentos durante o período de seu intercâmbio em Santa Maria e também pela amizade e orientações até os últimos dias da finalização deste trabalho.

Meu sincero agradecimento à todos, que de alguma forma se fizeram presentes durante a minha qualificação acadêmica e profissional!

Muito obrigado!

RESUMO

Dissertação de Mestrado
Programa de Pós-Graduação em Informática
Universidade Federal de Santa Maria

APLICANDO A TRANSFORMADA WAVELET BIDIMENSIONAL NA DETECÇÃO DE ATAQUES WEB

AUTOR: BRUNO AUGUSTI MOZZAQUATRO
ORIENTADOR: RAUL CERETTA NUNES (UFSM)
CO-ORIENTADORA: ALICE DE JESUS KOZAKEVICIUS (UFSM)
Local da Defesa e Data: Santa Maria, 27 de fevereiro de 2012.

O aumento do tráfego web vem acompanhado de diversas ameaças para a segurança das aplicações web. As ameaças são decorrentes das vulnerabilidades inerentes dos sistemas web, sendo a injeção de código ou conteúdo malicioso uma das vulnerabilidades mais exploradas em ataques web, pois permite que o atacante insira uma informação ou programa em locais indevidos, podendo causar danos aos clientes e organizações. Esse tipo de ataque tem sido caracterizado pela alteração na distribuição da frequência dos caracteres de algumas requisições dentro de um conjunto de requisições web. Sistemas de detecção de intrusão baseados em anomalias têm sido usados para procurar conter tais tipos de ataques, principalmente em função da diversidade e da complexidade dos ataques web. Neste contexto, o trabalho propõe um novo algoritmo para detecção de anomalias que aplica a transformada wavelet bidimensional na detecção de ataques web e elimina a necessidade de uma fase de treinamento com dados confiáveis de difícil obtenção. O algoritmo pesquisa por anomalias nas frequências dos caracteres de um conjunto de requisições web através da análise em múltiplas direções e resoluções. Os resultados obtidos nos experimentos demonstraram a viabilidade da técnica para detecção de ataques web e também que com ajustes entre diferentes parâmetros foram obtidas taxas de detecção de até 100%, eliminando a ocorrência de falsos positivos.

Palavras-chave: Detecção de Anomalias; Detecção de Intrusão; Wavelet; Ataques Web.

ABSTRACT

Master's Dissertation
Computer Science Graduate Program
Federal University of Santa Maria

APPLYING TWO-DIMENSIONAL WAVELET TRANSFORM FOR THE DETECTION OF WEB ATTACKS

AUTHOR: BRUNO AUGUSTI MOZZAQUATRO
ADVISOR: RAUL CERETTA NUNES (UFSM)
COADVISOR: ALICE DE JESUS KOZAKEVICIUS (UFSM)
Presentation Place and Date: Santa Maria, February 27th, 2012.

With the increase web traffic of comes various threats to the security of web applications. The threats arise inherent vulnerabilities of web systems, where malicious code or content injection are the most exploited vulnerabilities in web attacks. The injection vulnerability allows the attacker to insert information or a program in improper places, causing damage to customers and organizations. Its property is to change the character frequency distribution of some requests within a set of web requests. Anomaly-based intrusion detection systems have been used to break these types of attacks, due to the diversity and complexity found in web attacks. In this context, this paper proposes a new anomaly based detection algorithm that apply the two-dimensional wavelet transform for the detection of web attacks. The algorithm eliminates the need for a training phase (which asks for reliable data) and searches for character frequency anomalies in a set of web requests, through the analysis in multiple directions and resolutions. The experiment results demonstrate the feasibility of our technique for detecting web attacks. After some adjustments on different parameters, the algorithm has obtained detection rates up to 100%, eliminating the occurrence of false positives.

Keywords: Anomaly Detection, Intrusion Detection, Wavelet, Web Attack.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CERN	*Conseil Européen pour la Recherche Nucléaire*
CSRF	*Cross-Site Request Forgery*
CVE	*Common Vulnerabilities and Exposures*
FCD	*Frequency Character Distribution*
HTML	*Hypertext Markup Language*
HTTP	*Hypertext Transfer Protocol*
HTTPS	*HyperText Transfer Protocol Secure*
ICD	*Idealized Character Distribution*
IDS	*Intrusion Detection System*
IP	*Internet Protocol*
IPS	*Intrusion Prevention System*
MRA	*Multiresolution Analysis*
ROC	*Receiver Operation Characteristic*
SQL	*Structured Query Language*
SCC	*Same Character Comparison*
TCP	*Transmission Control Protocol*
TW	*Transformada Wavelet*
TWD	*Transformada Wavelet Discreta*
TW1D	*Transformada Wavelet Unidimensional*
TW2D	*Transformada Wavelet Bidimensional*
URI	*Uniform Resource Identifier*
URL	*Uniform Resource Locator*
WWW	*World Wide Web*

SUMÁRIO

1 INTRODUÇÃO

A Internet está em constante progresso, de modo que se tornou um sistema global de interconexão entre redes de computadores e tem disponibilizado uma diversidade de recursos através de aplicações web. Essas aplicações vêm ganhando múltiplas funcionalidades e geralmente o projeto de software visa somente o desenvolvimento de novas funções, desconsiderando o uso de boas práticas de segurança. Portanto, a carência na utilização de técnicas de segurança no desenvolvimento das aplicações acaba deixando lacunas, chamadas de vulnerabilidades (OWASP, 2011). Desta forma, permite que ataques explorem as falhas com o intuito de violar a segurança e afetar disponibilidade das aplicações (FONSECA; VIEIRA; MADEIRA, 2010). O aumento no número de vulnerabilidades descobertas nas aplicações web não é uma surpresa, como pode ser visto em CVE (2011). Baseado no estudo realizado por Robertson (2009), as vulnerabilidades nas aplicações web são responsáveis por mais de 49% do número total das falhas relatadas no ano de 1999 a 2009. Ainda, segundo uma análise sobre as tendências da segurança em aplicações web (NEUHAUS; ZIMMERMANN, 2010), foram identificadas 39.393 vulnerabilidades únicas até o final de 2009. E, segundo Symantec (2011), o volume de ataques web teve um aumento de 93% entre os anos de 2009 a 2010, conforme pode ser observado na Figura 1.1.

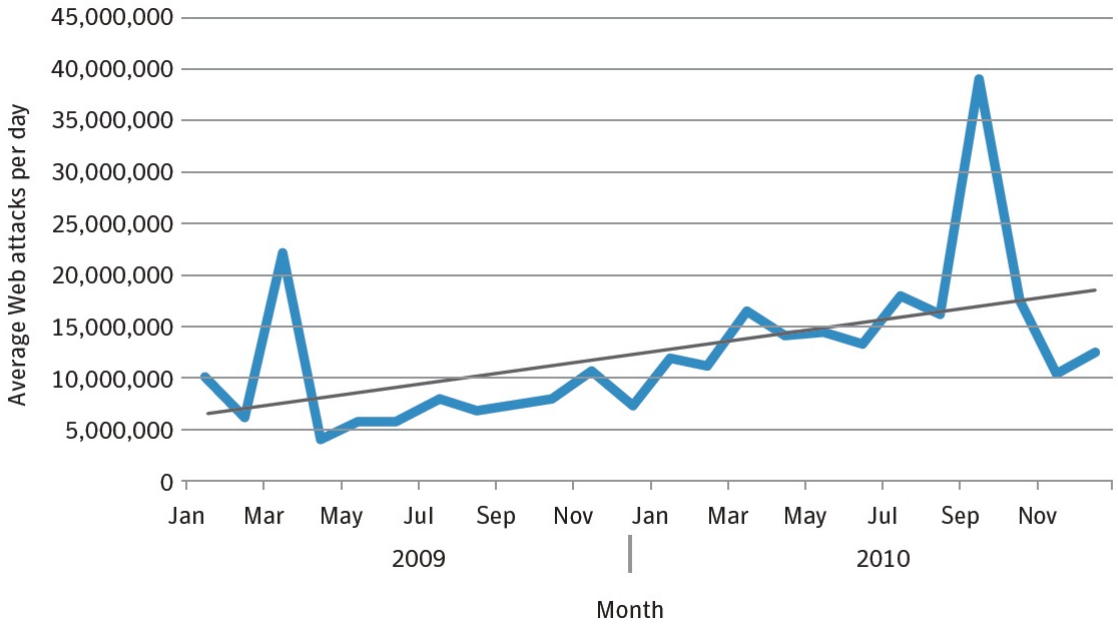

Figura 1.1: Média de ataques web por mês entre o ano de 2009 e 2010 (SYMANTEC, 2011).

Diante deste cenário, o uso de ferramentas para a detecção de ataques web tornou-se indispensável para ambientes computacionais. Portanto, este trabalho propõe um novo algoritmo de

detecção de ataques web.

1.1 Motivação

Soluções tradicionais de segurança, tais como *firewalls*, não foram projetadas para a identificação de ações que comprometem a segurança de um sistema de informação na Internet, denominados ataques web. Essas soluções não inspecionam as características do tráfego e com isso não são capazes de detectar ataques web através de uma análise refinada dos dados. Por outro lado, o surgimento dos Sistemas de Detecção de Intrusão (*Intrusion Detection System* - IDS) proporcionaram ferramentas potenciais que visam identificar e proteger a exploração de vulnerabilidades em redes de computadores, assim como em aplicações web (ROBERTSON, 2009). Ao contrário dos mecanismos tradicionais, os IDS capturam o tráfego, analisam o comportamento a fim de detectar eventuais ameaças para a rede ou sistema. Neste sentido, os IDS possuem duas abordagens, relacionadas ao método de detecção, para proteger os sistemas computacionais: baseadas em assinaturas e baseadas em anomalias (NORTHCUTT; NOVAK, 2002).

A abordagem baseada em assinatura identifica ataques através de um conjunto de informações contendo padrões de ataques (assinaturas) previamente definidas (KRUEGEL; VALEUR; VIGNA, 2004). A vantagem dessa abordagem é a baixa taxa de falsos positivos na detecção dos ataques, sendo considerada como um nível para medir a eficiência dos IDS. Conforme aumenta o número de falsos positivos, menor a eficiência do IDS. Entretanto, essa abordagem apenas detecta ataques conhecidos, e para isso necessita de frequentes atualizações do conjunto de assinaturas para manter a confiabilidade na detecção. Por outro lado, a abordagem baseada em anomalias identifica ataques através da observação do sinal e suas variações quanto às características (CHANDOLA; BANERJEE; KUMAR, 2009). Além disso, essa abordagem identifica o comportamento que não segue a definição considerado usual. Com isso, proporciona a detecção de ataques desconhecidos (também chamados ataques *zero-day*[1]) e também ataques conhecidos que possuem alteração na assinatura utilizada para comprometer a segurança de um sistema, modificando a assinatura do ataque. De tal forma, ao contrário da abordagem baseada em assinaturas, a abordagem baseada em anomalias pode detectar os seguintes tipos de ataques: novos ataques, *zero-day* e variações na assinatura dos existentes. Entretanto, essa abordagem apresenta duas desvantagens, tais como a geração de altas taxas de falsos positivos (alarmes falsos)

[1] Um ataque *zero-day* explora uma vulnerabilidade que ainda não foi publicada.

e a necessidade de dados de uma fase de treinamento para obter o comportamento usual do sistema.

Abordagens baseadas em anomalias têm sido utilizadas em diversos trabalhos propostos na literatura para a detecção de ataques web. Essa abordagem é promissora neste contexto devido ao surgimento de novos ataques a cada dia (SYMANTEC, 2011). A detecção baseada em anomalia no contexto web iniciou com o trabalho de Kruegel e Vigna (2003), que explorou a análise das seções de consultas contidas nas requisições web, buscando identificar anomalias. Ao longo dos anos, diversos outros trabalhos foram propostos aplicando técnicas baseadas em aprendizagem, com a utilização das fases de treinamento e detecção, para a identificação de intrusões, como (KRUEGEL; VIGNA, 2003) (INGHAM et al., 2007) (KIANI; CLARK; MOHAY, 2008), (SRIRAGHAVAN; LUCCHESE, 2008). Esses trabalhos utilizam um perfil normal do comportamento (conhecido como *normal profile*) para a detecção, que é definido na fase de treinamento através de um conjunto de dados sem a ocorrência de ataques (dados de treinamento). Porém, em sistemas de detecção de intrusão baseados em anomalias, a definição do perfil normal é um desafio, principalmente porque além de ser difícil obter um conjunto de dados sem ataques que represente todas as atividades reais, também possui um alto custo financeiro e temporal (JAMDAGNI et al., 2010).

Ataques provenientes da comunicação HTTP causam perturbações nas requisições entre cliente e servidor web, as quais são consideradas anomalias do tráfego. Os ataques web que provocam a manipulação de requisições resultando na inserção de informações com intuito de prejudicar são considerados problemas críticos para as organizações, pois colocam em risco as funcionalidades ou disponibilidade dos serviços na Internet (ALVAREZ; PETROVIC, 2003). Portanto, ataques web que exploram a injeção de código ou conteúdo malicioso são um dos mais utilizados em aplicações web (OWASP, 2011). Além disso, esse tipo de ataque web causa variações significativas na frequência dos caracteres em requisições web (SU; WASSERMANN, 2006). Por exemplo, os ataques *SQL Injection* e o *Cross-Site Scripting* (XSS) provocam consequências para as aplicações e clientes. Outro exemplo é o *Path Traversal*, caracterizado pela utilização de uma grande quantidade de caracteres "." e "/" em relação aos outros caracteres (ROBERTSON et al., 2006). Portanto, a variabilidade dos caracteres contidos nos ataques web poderiam ser detectados através do uso de ferramentas que proporcionam uma análise entre os caracteres de uma mesma requisição e também entre um grupo de requisições web.

Conforme a necessidade de identificar as variações abruptas na frequência dos caracteres

oriunda de ataques web, é indispensável o uso de ferramentas eficientes e ágeis para auxiliar na detecção dessas ameaças. A motivação diante dessa característica está relacionada com as ferramentas atuais que necessitam processar milhões de conexões e possuem alto custo computacional para isso (MARQUES; BAILLARGEON, 2005), logo, possuem desempenho ruim de detecção dos IDS em tempo hábil de execução para identificar a ameaça, também denominado detecção em tempo real. Neste contexto, a transformada wavelet bidimensional é uma ferramenta de processamento de sinais que potencializa a aplicação em diferentes contextos. Sua principal característica é a identificação de picos e a correlação entre diferentes fontes de detalhes para a análise do sinal (DAUBECHIES, 1992). Ainda, a aplicação da transformada wavelet de Haar (STOLLNITZ; DEROSE; SALESIN, 1995) possui baixo custo computacional devido ao uso exclusivo de operações de adição e subtração. Com base na propriedade da transformada wavelet bidimensional, variações significantes entre os caracteres nas requisição web podem ser identificadas através da análise com grupo de requisições.

Neste contexto, a motivação deste trabalho visa explorar a detecção de ataques web baseados em anomalia através da aplicação da transformada wavelet bidimensional para análise da frequência dos caracteres contidos nas requisições web.

1.2 Objetivos e contribuições

O objetivo deste trabalho é apresentar um novo algoritmo para sistemas de detecção de ataques web baseado em anomalias através do uso de técnicas de processamento de sinais (wavelets). A transformada wavelet (TW) é uma ferramenta matemática que permite realizar análise em multirresolução (*Multiresolution Analysis* - MRA), através da decomposição de um sinal em diferentes níveis, devido aos seus componentes localizados em diferentes escalas (WALKER, 2008). Estas características permitem identificar a variabilidade de um sinal analisado, tornando a análise wavelet adequada para detecção de mudanças abruptas. Desta forma, este trabalho permite analisar o comportamento das requisições web analisando a variabilidade na frequência dos caracteres.

A principal contribuição deste trabalho é uma nova abordagem para detecção de ataques web, a qual teve sua eficiência validada através da experimentação com duas diferentes bases de dados contendo registro de ataques. O tempo de processamento foi similar quando comparado com outras técnicas de detecção de ataques web e sua eficácia na detecção chegou a 100%. Outra contribuição, importante para aplicação da técnica, é a eliminação da necessidade de se

ter uma fase de treinamento com dados confiáveis para aferir o IDS. A construção de bases de dados para treinamento é considerada um dos desafios no para sistemas de detecção de intrusão baseados em anomalias.

1.3 Organização do Texto

O texto deste trabalho esta organizado da seguinte forma. O Capítulo 2 apresenta uma revisão bibliográfica sobre os principais conceitos da segurança em aplicações web, tais como as características das vulnerabilidades e ataques web, e os Sistemas de Detectores de intrusão. Na sequência, a descrição teórica referente à Transformada Wavelet Discreta e a estratégia de truncamento dos dados são vistas no Capítulo 3. O Capítulo 4 apresenta a proposta para detecção de ataques web. Por fim, são realizados experimentos no Capítulo 5 e o Capítulo 6 apresenta as conclusões e sugestões de trabalhos futuros.

2 DETECÇÃO DE INTRUSÃO NA WEB

Com a diversidade de ameaças na Internet, a segurança da informação tem observado o surgimento de novas vulnerabilidades nas aplicações web (OWASP, 2011). Neste sentido, este capítulo apresentará assuntos a respeito da segurança da informação, assim como um esboço da arquitetura web, que irá servir de sustentação para a contextualização deste trabalho.

Na Seção 2.1 é apresentada uma breve descrição sobre a segurança das aplicações web e da arquitetura web. Na sequência, a Seção 2.2 apresenta as vulnerabilidades web e ataque web utilizados no contexto deste trabalho. Na seção 2.3 os sistemas de detecção são detalhados. A seção 2.4 apresenta os algoritmos de detecção de anomalias para ataques web. E por fim, a Seção 2.5 apresenta as considerações parciais do capítulo.

2.1 Visão Geral da Web

A *World Wide Web* (WWW), também conhecida como Web, é uma estrutura de interconexão que permite o acesso a documentos hipertexto espalhados por bilhões de máquinas na Internet (TANENBAUM, 2003). A Web teve início em 1989, quando grupos de pesquisadores do centro europeu de pesquisa nuclear identificaram a necessidade de realizar a troca de informações, como relatórios, imagens entre outros documentos no desenvolvimento de pesquisas na área de física de partículas. Então, um protótipo foi proposto pelo físico Tim Berners-Lee, que com o passar do tempo recebeu publicidade atraindo atenção devido a acessibilidade da grande diversidade de informações sobre todos os assuntos.

O acesso aos documentos é realizado por um navegador web através de aplicações web. A base da web é a transferência de páginas de aplicações web entre clientes e servidores. As páginas são compostas de documentos que podem ser estático como documentos HTML (*Hypertext Markup Language*), arquivos multimídias tais como imagens, vídeos e conteúdo dinâmico gerado por demanda, ao invés de armazenado em disco. Para isso, a web possui uma arquitetura que utiliza o modelo cliente-servidor, ou seja, a troca de informação é realizada entre clientes web e servidores web utilizando o protocolo HTTP (*Hypertext Transfer Protocol*) (FIELDING et al., 1999) sobre um conjunto de protocolos TCP/IP.

A web é baseada na troca de mensagens do protocolo HTTP entre clientes e servidores, também conhecidas como requisições web. Desta forma, uma sessão HTTP típica possui a comunicação através do envio de requisições entre clientes e servidores web, na qual os clientes

solicitam recursos para os servidores web. A Figura 2.1 apresenta o fluxo de informações da web através do protocolo HTTP.

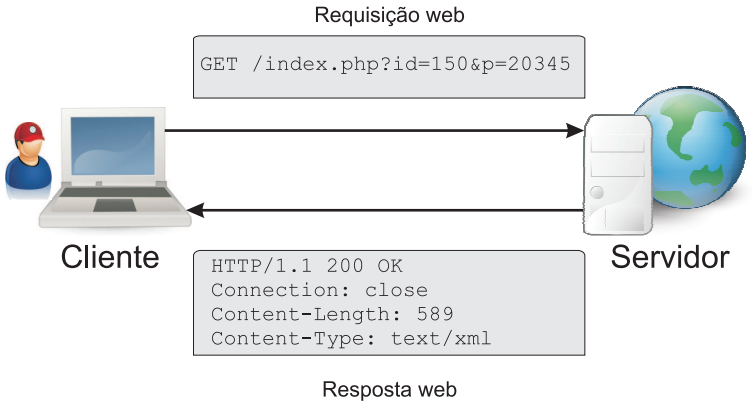

Figura 2.1: Troca de informações utilizando protocolo HTTP.

No fluxo de informações do modelo cliente-servidor, o cliente envia um pedido solicitando um recurso localizado na aplicação web *page.php* que está hospedada no servidor web. Ainda, as informações enviadas são transmitidas através de um pacote HTTP que possui alguns parâmetros. Ao chegar ao servidor, as informações são processadas e o servidor retorna um cabeçalho comunicando que o recurso existe. Logo após, as informações solicitadas são enviadas para o cliente.

O protocolo HTTP tem como objetivo especificar as mensagens que os clientes enviam para os servidores web e cada interação consiste numa solicitação ASCII (TANENBAUM, 2003). As requisições web são mensagens formadas com blocos de dados. Cada mensagem contém uma requisição web de um cliente ou uma resposta de um servidor. Em síntese, elas são compostas em três partes (GOURLEY; TOTTY, 2002): um linha inicial que descreve a mensagem, um bloco de cabeçalhos contendo atributos e um conjunto de informações (opcional). A requisição web e o cabeçalho da mensagem HTTP são apenas texto com caracteres ASCII. A Figura 2.2 apresenta uma mensagem HTTP de solicitação e reposta, separada conforme a estrutura da mensagem.

Na requisição de solicitação, o campo de informações opcional não é utilizado pela mensagem de solicitação. Neste tipo de mensagem HTTP, onde o “GET” é o método utilizado, informações não são enviadas para o servidor. A linha inicial é composta pela requisição web, a qual possui o caminho para o recurso desejado. Porém, informações podem serem enviadas para, através da requisição web, efetuar consultas em aplicações web. Além disso, a requisição web possui a versão do protocolo HTTP utilizado, nesse exemplo a versão é HTTP/1.1.

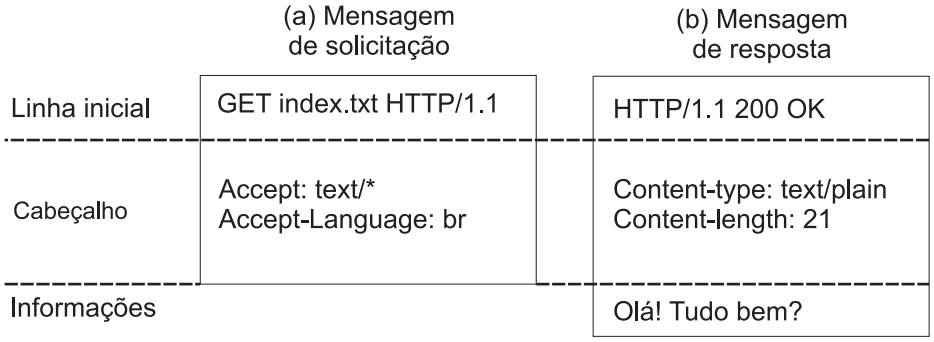

Figura 2.2: Mensagens HTTP divididas por partes. (a) Corresponde a mensagem enviada pelo cliente. (b) Corresponde a resposta do servidor.

Originalmente o protocolo foi desenvolvido para a disseminação e recuperação de informações baseadas em texto, mas ao longo do tempo recebeu atualizações para suportar aplicações complexas, com adição de conexões persistentes. Inicialmente, a conexão TCP era estabelecida somente para o envio de uma solicitação e após receber uma única resposta a conexão era finalizada (HTTP 1.0). No entanto, com o uso de conteúdos web com a diversidade de ícones, vídeos, imagens, entre outros recursos, o estabelecimento de uma conexão TCP para o transporte se tornou um modo de operação muito dispendioso (TANENBAUM, 2003). Esse motivo levou ao lançamento da versão 1.1 do protocolo HTTP, admitindo uso de conexões persistentes, ou seja, estabelecer uma conexão TCP com inúmeras interações entre cliente e servidor. Isto é, possibilitou para o cliente enviar além da solicitação principal e também solicitações adicionais, minimizando o custo com a instalação e liberação de conexões TCP.

A simplicidade da operação de solicitação de uma página web deu lugar as aplicações orientadas a objetos, através do uso de métodos (TANENBAUM, 2003). Os métodos disponíveis são utilizados como comando para clientes e enviados para servidores. Na Tabela 2.1 estão descritos os métodos do protocolo HTTP.

O método GET utilizado na Figura 2.2 solicita ao servidor o arquivo “index.txt” na solicitação. Logo, a página é codificada e, após, a requisição possui a versão do protocolo HTTP utilizado. A codificação é realizada de forma adequada em MIME (*Multipurpose Internet Mail Extensions*) (N. BORENSTEIN; N. FREED, 1992). O MIME é um padrão de mensagens com formato flexível que representa as mensagens enviadas na Internet. Além disso, possui mecanismos que permitem a transmissão de mensagens contendo outros formatos, isto é, usando codificação diferente do ASCII, assim como mensagens com formato contendo áudio, vídeo e outros arquivos binários.

Entre os métodos disponíveis pelo protocolo HTTP, o método GET é o mais utilizado nas

Tabela 2.1: Métodos utilizados pelo protocolo HTTP.

Método	Descrição
GET	Solicita um recurso para leitura
HEAD	Solicita meta-informações através do cabeçalho de uma página web
PUT	Solicita o armazenamento de um recurso
POST	Solicita o envio de um recurso
DELETE	Solicita a remoção de uma página web
TRACE	Reflete uma a solicitação recebida de retorno para o cliente
CONNECT	Reservado para uso com *proxy* destinado a criação de uma conexão segura (túnel SSL)
OPTIONS	Solicita opções da comunicação disponível

requisições para servidores web (TANENBAUM, 2003). Para mais detalhes dos métodos utilizados pelo protocolo HTTP, ver (FIELDING et al., 1999).

Além do método utilizado, cada requisição possui o caminho do recurso, chamado de *Request-Uniform Resource Identifier* (URI). Quando se refere ao protocolo HTTP, URI é um termo geral usado para a identificação de um recurso através de nomes, localização ou outras características (BERNERS-LEE; MASINTER; MCCAHILL, 1994). No entanto, o termo URI é aconselhado pela especificação RFC 3986 para uso em definições formais de futuras especificações. O termo *Uniform Resource Locator* (URL) é mais específico para identificar um recurso através da descrição da localização.

Dentre os definições adotada pela especificação (BERNERS-LEE; MASINTER; MCCAHILL, 1994), o esquema *http URL* usado para localizar um recurso na Internet através do protocolo HTTP é definido com a sintaxe específica na Figura 2.3.

```
http://[servidor[:porta]/[caminho_absoluto[?consulta]]
```

Figura 2.3: Sintaxe específica do esquema http URL.

Segue a descrição dos diferentes componentes do esquema *http URL*:

- Servidor: é o nome do domínio ou endereço IP de um host na rede.
- Porta: o número da porta para estabelecer a conexão. No caso da porta não ser especificada, a porta padrão é a 80.
- Caminho absoluto: é um campo opcional, que indica o caminho de seleção HTTP e normalmente é usado em conjunto com a consulta.

- Consulta: assim como o caminho absoluto, a consulta também é um campo opcional e possui um caractere “?” precedente. Um conjunto de parâmetros no formato-chave-valor são utilizados na definição. Dentro desse componente os caracteres “/”, “;” e “?” são reservados.

As requisições processadas com sucesso no servidor web retornam uma mensagem de resposta com o recurso desejado. Porém, uma série de problemas pode ocorrer no processamento de uma requisição, e para cada problema um código identifica o status do processamento. A informação de situação do processamento consiste em uma linha contendo o código com 3 dígitos. O primeiro dígito é utilizado para a divisão das respostas em cinco grupos, como pode ser observado na Tabela 2.2.

Tabela 2.2: Códigos de *status* das requisições web

Código	Identificação das requisições web
1XX	Código da informação
2XX	Sucesso no processamento
3XX	Redirecionamento
4XX	Erro no cliente
5XX	Erro no servidor web

Por exemplo, o sucesso de uma requisição é identificado pelo código 200, assim como uma página não encontrada pode ser identificada através do código 404. Mais detalhes dos códigos de status do protocolo HTTP podem ser vistos em (FIELDING et al., 1999).

2.1.1 Evolução da Web

Inicialmente, a Internet utilizava web *sites* para a disseminação de informações através de repositórios contendo documentos estáticos. Essas informações eram acessadas pelos clientes através de um navegador web. Ainda, o fluxo da informação era somente para uma direção, ou seja, a informação armazenada no servidor era acessível publicamente com a utilização de navegadores pelos clientes.

Com a evolução da Internet, houve uma alteração na característica dos web *sites*, que começaram a ser chamados de aplicações web (STUTTARD; PINTO, 2007). Essas aplicações receberam outro fluxo para a informação, permitindo a comunicação tanto do servidor com o navegador web ou vice-versa. Além disso, os conteúdos das aplicações tornaram-se dinâmicos e específicos para cada cliente das aplicações web. Desta forma, as aplicações tornaram-se

essenciais para a realidade dos clientes, sendo adotadas pelas organizações para realização de funções essenciais, tais como gerenciamento de recursos e funções administrativas. De fato, essa tendência modificou também os programas de desktop, tais como editores e planilhas de texto, que migraram para as aplicações web, permitindo o acesso através de um navegador web conectado a internet.

Com essa diversidade de aplicações web sendo desenvolvidas, muitas vezes nos deparamos com desenvolvedores com pouco conhecimento sobre segurança. Esses problemas podem surgir nos códigos que estão sendo produzidos resultando em vulnerabilidades das aplicações web (FOGIE et al., 2007). Assim, percebemos que as aplicações web possuem diferenças e consequentemente podem conter vulnerabilidades únicas. Portanto, a segurança tornou-se um grande problema devido à confiabilidade das aplicações para os clientes. Isto é, manter a segurança de uma aplicação web para os clientes tornou-se indispensável, pois nenhum cliente deseja que suas informações particulares possam ser divulgadas para pessoas não autorizadas.

2.1.2 Segurança das Aplicações Web

Aplicações web inseguras já são frequentes para organizações bancárias, saúde, energia e outras infraestruturas críticas (OWASP, 2011). Com o aumento na complexidade e na interconexão das redes de computadores, os obstáculos para alcançar a segurança nas aplicações web tendem a maximizar exponencialmente.

Os aspectos de segurança considerados na Internet têm como principal preocupação a negação de acesso a informações restritas para pessoas não autorizadas. Entretanto, alguns serviços (STALLINGS, 2005) são considerados os pilares da segurança da informação contra diversas ameaças, tais como roubo de informações, fraude, interrupção, entre outros, conforme a seguir:

- Autenticação: assegura a certificação da comunicação segura. Para isso, deve-se garantir que uma determinada aplicação web na Internet é realmente da organização que buscamos. Desta forma, tem-se a garantia da identidade das partes envolvidas na comunicação realizada.

- Controle de Acesso: consiste na capacidade de limitar e restringir o acesso para determinados hosts e aplicações através de links de comunicação. No entanto, esse serviço somente é alcançado depois que o usuário estiver autenticado no sistema.

- Confidencialidade: consiste na proteção dos dados, impedindo que usuários não autoriza-

dos obtenham acesso à informações contidas na comunicação entre a origem e o destino. Desta forma, normalmente são utilizadas técnicas de criptografia para cifrar mensagens e impossibilitar a leitura das informações, mesmo que os usuários consigam interceptar o fluxo de dados não será possível identificar o conteúdo.

- Integridade: consiste na garantia de que uma mensagem depois de enviada não pode ser alterada entre a comunicação de origem e destino.

- Não repúdio: assegurar que o usuário não negue ter realizado uma determinada operação, oferecendo provas inegáveis que a ação específica foi realizada.

As aplicações web tornaram-se um componente essencial na Internet para a economia moderna, fornecendo serviços para organizações empresariais e usuários comuns. Como resultado, a conectividade dessas aplicações web tornou-se crucial, tanto que profissionais da segurança da informação estão atentos para a proteção da infraestrutura mantendo a disponibilidade das informações sem ocorrência de falhas.

No entanto, as falhas nas aplicações web são decorrentes de vulnerabilidades expostas no desenvolvimento dessas aplicações e, além disso, o uso de novas tecnologias introduzem a possibilidade para a exploração de novas vulnerabilidades (STUTTARD; PINTO, 2007). Algumas categorias de falhas têm sido largamente exploradas como resultado de mudanças feitas no software de navegadores web. No decorrer da evolução da web, problemas de segurança nas aplicações têm sido divulgados e a tendência é que esses problemas continuem em ascensão, como é apresentado pela Figura 1.1, na introdução deste trabalho. A segurança das aplicações web tornou-se um campo de batalha entre atacantes e aqueles que possuem recursos e informações para serem protegidas, e é provável que esse desafio permaneça ao longo de um futuro previsível (STUTTARD; PINTO, 2007).

A disponibilidade de uma vulnerabilidade dentro de uma aplicação web pode comprometer a segurança do sistema de uma organização, que através da submissão de dados maliciosos possam colocar em risco a integridade das informações restritas apenas utilizando um navegador web (VACCA, 2009). Assim, através da identificação da vulnerabilidade, atacantes podem fazer uso de técnicas para explorar informações sensíveis e obter controle de acesso à dados restritos.

2.2 Ameaças Web

A cada dia são descobertas novas falhas em aplicações web na Internet (CVE, 2011), permitindo o surgimentos de novas ameaças. Uma vulnerabilidade é uma lacuna ou uma falha de uma aplicação web, que muitas vezes pode ser uma falha no projeto ou no desenvolvimento da aplicação. Por outro lado, os ataques web são técnicas que usuários maliciosos utilizam para explorar as vulnerabilidades das aplicações web buscando obter acesso não autorizado ou informações restritas. A distinção entre os termos vulnerabilidade web e ataque web deve ser considerada.

Na Subseção 2.2.1 serão descritas as principais vulnerabilidades reportadas. A Subseção 2.2.2 apresenta os ataques que exploram a alteração e inserção de informações maliciosas nas requisições web.

2.2.1 Vulnerabilidades Web

O projeto "*The Open Web Application Security*" (OWASP, 2011) publica periodicamente as tendências das vulnerabilidades web nos últimos anos com o intuito de tornar as aplicações mais seguras e permitir que pessoas e organizações possam tomar decisões sobre os riscos para a segurança das aplicações web.

A Tabela 2.3 apresenta a lista dos dez maiores riscos de segurança das aplicações web até o ano de 2010.

Tabela 2.3: Lista dos dez maiores riscos de segurança das aplicações web (OWASP, 2011)

Ataque	Localização
A1: Injeção	Requisição web
A2: *Cross-Site Scripting* (XSS)	Requisição web
A3: Falha na autenticação e gerenciamento de sessão	Requisição web
A4: Referências diretas inseguras a objetos	Requisição web
A5: *Cross-Site Request Forgery* (CSRF)	Requisição web
A6: Configuração incorreta da segurança	-
A7: Armazenamento inseguro de criptografia	Requisição web
A8: Falha de acesso URL restrito	Requisição web
A9: Insuficiente proteção na camada de transporte	-
A10: Redirecionamento e encaminhamento invalidado	Requisição web

A seguir as vulnerabilidades mais comuns serão brevemente descritas conforme OWASP

(2011).

- A1 - Injeção: está relacionada com falhas de injeção, tal como inserção de comandos SQL, OS, LDAP, entre outros. Além disso, dados não confiáveis são enviados com intuito de serem interpretados como parte de um comando ou consulta. Dessa forma, atacantes executam comandos maliciosos buscando obter informações não autorizadas.

- A2 - *Cross-Site Scripting* (XSS): falhas XSS são provocadas devido à ausência de validação na entrada de dados dos usuários nas aplicações web. Assim, usuários mal intencionados inserem scripts para serem executados dentro do navegador da vítima, roubando sessões, modificando conteúdo ou redirecionando para sites maliciosos.

- A3 - Falha na autenticação e gerenciamento de sessão: falhas no funcionamento das funções de autenticação e gerenciamento de sessão permitem que usuários mal intencionados comprometam senhas, chaves e *tokens* de sessões. Além disso, podem explorar falhas de autenticação para assumir identidade dos usuários.

- A4 - Referências diretas inseguras a objetos: atacantes podem comprometer aplicações web através de falhas expostas pelos desenvolvedores sobre referências diretas de objetos internos, tais como arquivos, diretórios e chaves de banco de dados. Assim, a ausência de proteções com controle de acesso possibilita que usuários maliciosos acessem informações não autorizadas.

- A5 - *Cross-Site Request Forgery* (CSRF): usuários maliciosos forçam os usuários vítimas enviarem uma requisição web incluindo sua sessão ou qualquer outra informação de autenticação para uma aplicação web vulnerável. A técnica permite que a aplicação vulnerável pense que são requisições legítimas da vítima e forneça informações restritas para os atacantes.

- A6 - Configuração incorreta da segurança: aplicação web segura requer um configuração definida e implantada em servidores web, banco de dados e plataforma. Configurações padrões devem ser rejeitadas e alteradas. Além disso, manter os softwares atualizados, assim como bibliotecas de códigos usadas na aplicação web. Desconsiderar essas regras pode levar a falhas na segurança da aplicação web, tornando-a vulnerável a ataques.

- A7 - Armazenamento inseguro de criptografia: informações restritas e sensíveis requerem proteção ostensiva em aplicações web. Não utilizar técnicas apropriada de criptografia ou

hashing pode comprometer a segurança da aplicação, permitindo que atacantes realizem ataques de roubo de identidade, fraudes de cartão de crédito, entre outros crimes.

- A8 - Falha de acesso URL restrito: controle de acesso em aplicações web é necessário tanto para verificar se está correta a URL antes do acesso como também a cada vez que a URL é acessada. Caso contrário, usuários maliciosos podem forçar URL para acessar páginas ocultas.

- A9 - Insuficiente proteção na camada de transporte: aplicações seguras utilizam técnicas de criptografia e autenticação para proteger a confidencialidade e integridade das informações no tráfego de rede. No entanto, geralmente utilizam algoritmos fracos e usam certificados inválidos, expirados ou muitas vezes não utilizam.

- A10 - Redirecionamento e encaminhamento invalidado: falha na validação apropriada de redirecionamentos das aplicações web permite que usuários vítimas sejam encaminhados através de informações não confiáveis para sites maliciosos.

As classes de vulnerabilidades A1 e A2 podem ser afetadas através de alterações dos parâmetros das requisições web, injetando informações maliciosas. Neste sentido, este trabalho explora essas classes, na qual as falhas na aplicação web possibilita a utilização de ataques de injeção de código malicioso.

2.2.2 Ataques Web

Entre os ataques publicados nos últimos anos, 93% está relacionada a ataques em aplicações web. Além do surgimento dos novos ataques, os ataques considerados polimórficos[1] chegaram a quantidade de 286 milhões de variações únicas (SYMANTEC, 2011). Esses ataques, também chamados de ataques web, possuem alta diversidade na localização da aplicação. Porém, alteram a aparência da estrutura das requisições web consideradas usuais (INGHAM et al., 2007), conforme a especificação do protocolo HTTP.

Segundo a taxonomia proposta por Alvarez e Petrovic (2003), um atacante segue etapas de um ciclo para confirmar a execução de um ataque web. Considerando isso, uma série de etapas são efetuadas para a realização de diferentes atividades maliciosas. A base para a aplicação de um ataque web é a identificação de uma falha no sistema, tornando-o vulnerável. Posteriormente, diferentes maneiras podem ser utilizadas pelos atacantes para causar algum dano

[1] São variações de ataques já identificados, mas em circulação.

ao sistema. Neste sentido, a taxonomia sobre os ataques web, apresentada na Figura 2.4, é caracterizada pela seguinte definição:

Definição 1 *Um ataque web é definido através de um Ponto de Entrada que possui uma Vulnerabilidade. Essa falha ameaça um Serviço que é explorado por uma Ação e utiliza um Tamanho de Entrada contra um Alvo. O ataque tem certo Escopo visando obter Privilégios.*

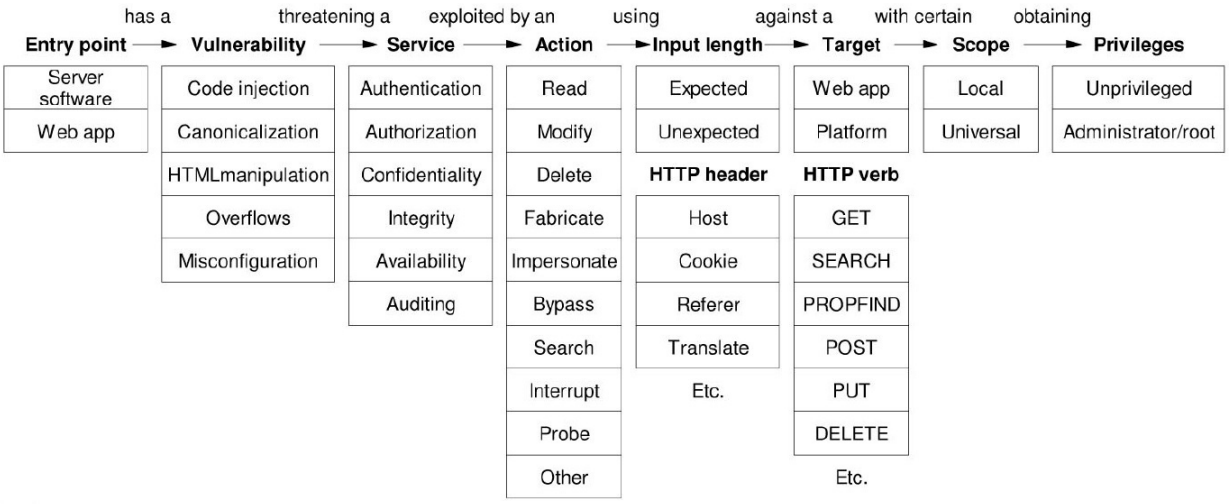

Figura 2.4: Taxonomia de ataques web (ALVAREZ; PETROVIC, 2003).

Assim, alguns critérios de classificação foram definidos para cada estágio do ciclo de vida para o atacante (ALVAREZ; PETROVIC, 2003), conforme apresentados na Figura 2.4.

- Ponto de entrada (*Entry point*): significa a localização explorada devido a uma falha na aplicação web vulnerável. Esse ponto de entrada pode ser em servidores web ou em aplicações web.

- Vulnerabilidade (*Vulnerability*): consiste na falha de um sistema permitindo a realização de uma atividade não autorizada. Dentre os diversos tipos de vulnerabilidades existentes, estão: injeção de código, *canonicalization*, manipulação de código HTML, *flooding* e configuração incorreta.

- Serviço (*Service*): a segurança de um sistema considera serviços ou requisitos que devem ser assegurados. Esses serviços incluem autenticação, integridade, confidencialidade, disponibilidade, controle de acesso e auditoria. Um ataque web quanto realizado com sucesso possui a intenção de afetar um ou mais serviços.

- Ação (*Action*): consiste nas ações realizadas durante um ataque pelo usuário malicioso. Essas ações foram classificadas pela taxonomia em três categorias conforme os objetivos. A primeira categoria são as ações diretas contra as informações: leitura, escrita, remoção, e criação. A segunda é relacionada contra a autenticação de usuários: uso da representação de um perfil legítimo, ignorar o mecanismo de autenticação e busca de informações de um usuário legítimo. Já a terceira são ações contra servidor web: interrupção, monitoramento das atividades e outras tais como execução de código e comandos.

- Tamanho (*Length*): ataques que utilizam de grandes quantidades de dados são baseados no tamanho dos dados resultando em consequências para o sistema, indisponibilizando o host atacado. Segundo a taxonomia, esse critério é diferenciado em ataques com tamanho comum e tamanho não usual.

- Elemento HTTP (*HTTP element*): todo ataque web utiliza o protocolo HTTP/HTTPS. Uma requisição web utilizando esse protocolo consiste em um método qualquer entre *GET*, *POST*, *HEAD*, *SEARCH*, *PUT*, *DELETE*, entre outros, e um grupo de cabeçalho tais como *Host*, *Cookie*, *Referer*, *Translate*, etc.

- Alvo (*Target*): os ataques web possuem uma diversidade de alvos e muitas vezes cada ataque possui um determinado foco. Isto é, cada ataque possui um alvo que pode ser um servidor web, obtenção de registros de banco de dados, etc. O alvo pode ser diferenciado entre ataques em aplicações web ou em plataforma.

- Escopo (*Scope*): o escopo de um ataque é baseado no impacto da sua aplicação que são geradas consequências para um só usuário local ou todos usuários de um determinado serviço web.

- Privilégios (*Privileges*): o objetivo de um atacante quando realiza um ataque é principalmente obter privilégio de administrador/*root* sobre um servidor web. No entanto, nem sempre é possível obter privilégio total. Portanto, esse critério distingue-se entre ataques sem privilégios e com privilégios administrativos.

Com base nesses critérios da taxonomia, um exemplo do ataque *Cross-site Scripting* (XSS) tem como ponto de entrada uma aplicação web com uma vulnerabilidade de injeção de código. Diante da falha no sistema, a confidencialidade é ameaçada explorando através de uma ação de leitura de dados restritos, e para isso utiliza uma requisição de tamanho esperado com o método

GET. Além do ponto de entrada, o ataque visa obter acesso a informações de uma aplicação web gerando um impacto local sem intenção de obter privilégios.

Neste contexto, este trabalho concentra-se em ataques que efetuam injeção de código malicioso em requisições web. Esse tipo de ataque explora vulnerabilidades conhecidas como *code injection* (STUTTARD; PINTO, 2007). Globalmente, injeção de código é suscetível a aplicação com sucesso nas aplicações vulneráveis que não validam corretamente todas as entrada que recebem dados dos usuários (SU; WASSERMANN, 2006).

Neste trabalho, um ataque de injeção de código refere-se ao ataque web que visa explorar uma vulnerabilidade de injeção de código. Os ataques da classe de vulnerabilidade de injeção são considerados severos devido ao impacto causado em aplicações web (OWASP, 2011), resultando na perda e disponibilização de informações restritas e negação de serviço. De maneira simples esses ataques são explorados por usuários maliciosos, que facilmente inserem um conjunto de informações para provocar uma ação mal intencionada. A seguir serão descritos os ataques mais comuns pertencentes a esta classe (WALDEN et al., 2009).

- Cross-Site Scripting (XSS)

O ataque *Cross-site Scripting* (XSS) (WURZINGER et al., 2009) (KIEYZUN et al., 2009) (KIRDA et al., 2009) consiste na injeção de códigos em aplicações web e, portanto, faz uso de técnicas que forçam a aplicação vulnerável apresentar códigos (scripts) para os usuários, os quais são executados nos navegadores web. Geralmente, o ataque XSS explora as vulnerabilidades das aplicações web através de código escritos em HTML/*JavaScript*, que são inseridos através das requisições web. Uma vez que o ataque XSS foi realizado de maneira bem sucedida, o navegador web do usuário vítima está sobre controle do atacante, permitindo que o mesmo realize ações mal-intencionadas (FOGIE et al., 2007).

Segundo Fogie et al. (2007), alguns cenários são comuns para a execução de ataques XSS em aplicações web:

- O desenvolvedor da aplicação web fez upload de um código malicioso propositalmente.
- A aplicação web foi explorada com códigos *Javascript* maliciosos submetidos como parte do ataque através de uma vulnerabilidade de rede ou do próprio sistema operacional.
- Uma aplicação web foi explorada com a vulnerabilidade XSS permanente e foi injetado códigos malicioso em *Javascript* dentro de uma área pública.

- Uma vítima acessou um link de uma aplicação web infectada com informações maliciosas.

Dentre os cenários citados, o ataque XSS é classificado em três categorias conforme a abrangência na execução do ataque (FOGIE et al., 2007). A primeira categoria consiste em ataques XSS não persistente (*Non-persistent*), os quais acontecem quando usuários são enganados a acessar um link geralmente criado com intuito de explorar as informações restritas dos usuários. A Figura 2.8 ilustra o fluxo do ciclo utilizado pelo atacante para aplicação do ataque XSS não persistente.

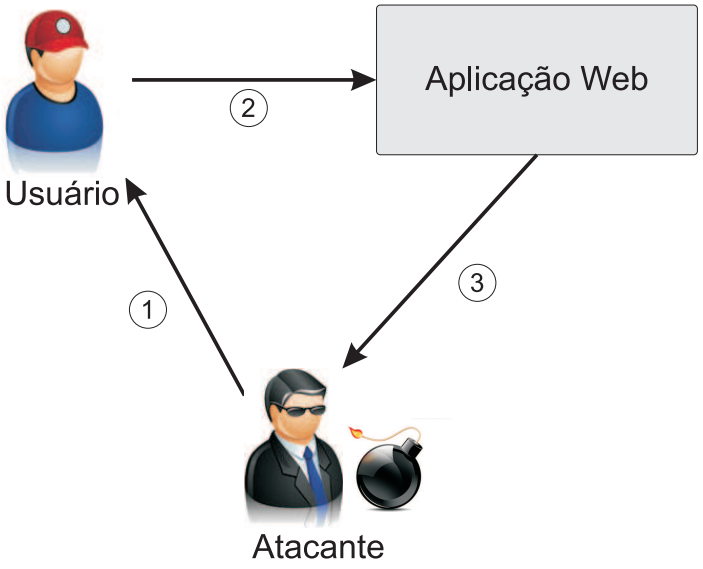

Figura 2.5: Ataque XSS não persistente.

No rótulo 1, o atacante compartilha um link para o usuário, com a injeção de informações maliciosas na requisição web. Desta forma, a requisição possui informações de total controle do atacante, visando determinada vulnerabilidade na aplicação web (rótulo 2). Por fim, no rótulo 3, são disponibilizadas informações não autorizadas para o atacante, conforme os dados inseridos na requisição.

A segunda categoria pertence aos ataques XSS persistentes (*Persistent*) (FOGIE et al., 2007). Essa categoria de ataques XSS explora a aplicação de forma transparente para os usuários, que são prejudicados sem consciência quando realizam o acesso às aplicações web com código malicioso agregado.

A Figura 2.6 ilustra o fluxo do ataque persistente. A linha pontilhada ilustra a inserção de informações maliciosas em uma área pública na aplicação web que armazena no banco de dados da aplicação. A seguir, no rótulo 1, o usuário realiza o acesso a uma aplicação web

que obtém informações armazenadas na base de dados. Com isso, a aplicação web efetua a busca das informações solicitadas pela aplicação que serão executadas no navegador do usuário. Conforme a figura, um atacante inseriu códigos em determinada área pública que o usuário realizou o acesso. Desta maneira, as informações com código malicioso são enviadas para do banco de dados para a aplicação web (rótulo 3). Por fim, no rótulo 4, a aplicação web envia os dados para o usuário e são executados no navegador web da vítima causando danos ao usuário, tais como roubo de informações.

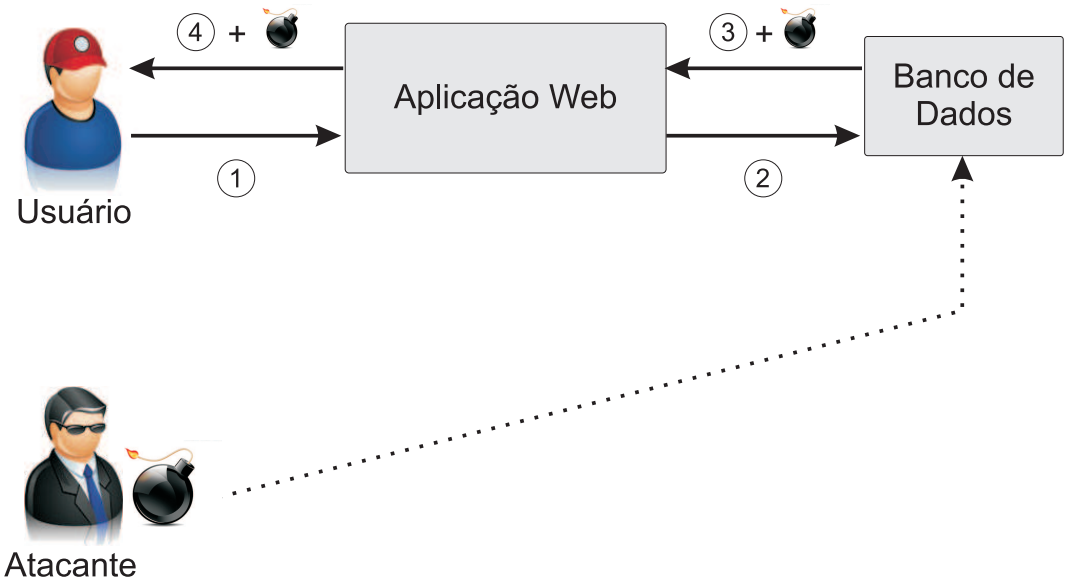

Figura 2.6: Ataque XSS persistente.

A terceira categoria do ataque XSS é baseado em DOM, uma especificação W3C (*World Wide Web Consortium*). Esta categoria difere das duas citadas anteriormente, sendo considerada menos frequente (ROBERTSON, 2009). O ataque XSS baseado em DOM explora uma vulnerabilidade de validação de entrada causada pelo cliente. Ou seja, não é resultado de uma vulnerabilidade dentro de um servidor web, causada por uma injeção de códigos pelo atacante (ROBERTSON, 2009). No entanto, esse ataque pode ser utilizado para obter informações restritas ou roubo de contas de usuário (FOGIE et al., 2007). A aplicação do ataque é através de um código do lado do cliente disponibilizado pelo navegador web que dinamicamente atualiza o documento através de funções DOM, tais como *document.write()*, ou seja, pode ocorrer se um código *Javascript* utiliza valores controlados pelo usuário, permitindo a manipulação de conteúdo HTML da aplicação web.

Considerando que um atacante deseja explorar o ataque XSS não persistente sobre um usuário, o primeiro passo é identificar a vulnerabilidade na aplicação web. Para isso, atacantes sub-

metem informações através de um campo de entrada de dados, com o objetivo de mostrá-las na tela. A Figura 2.7 apresenta um exemplo da inserção do código HTML/*Javascript*, testando a existência de uma vulnerabilidade. Assim, uma caixa de texto deverá ser gerada pelo navegador web mostrando a seguinte mensagem "Ataque XSS não persistente". Uma vez que o teste foi realizado com sucesso, a alteração no conteúdo inserido deverá ser efetuada, adicionando ações maliciosas.

```
'><script>alert('Ataque XSS não persistente')</script>
```

Figura 2.7: Exemplo de ataque XSS não persistente.

Observam-se cuidadosamente os primeiros caracteres (“’>”) que são destinados a fechar o parâmetro sobre qualquer atributo de elemento, o qual é seguido pela informação contendo o script que será processado pelo navegador. Isso é suficiente para determinar se uma aplicação web é vulnerável a ataques XSS. Embora esse seja um teste simples, identifica-se a facilidade na identificação de vulnerabilidades em aplicações web (FOGIE et al., 2007). Isto é, o código *Javascript* foi executado pelo navegador como parte da pesquisa. Esse simples teste é muito utilizado por atacantes para identificar vulnerabilidades em aplicações web.

Entretanto, não se pode garantir a vulnerabilidade na aplicação web baseado neste teste. Pressupõe-se que a aplicação web possui uma limitação com relação a quantidade de caracteres, impossibilitando a injeção de um número limitado de caracteres. Como a injeção da quantidade de caracteres utilizando o ataque XSS é superior a quantidades de caracteres usuais pela aplicação, a vulnerabilidade não pode ser explorada usando esse tipo de ataque. Embora existam outras maneiras de aplicar o ataque XSS com pequenas quantidades de caracteres, as quais não são muito utilizadas devido à discrição e anonimato pela aplicação web. Para mais detalhes ver (FOGIE et al., 2007).

Ainda sobre o ataque XSS não persistente, também são utilizadas técnicas para deslumbrar pessoas para obter acesso a informações explorando a confiança das mesmas (engenharia social). Dessa forma, os usuários são convencidos de acessar um link de uma aplicação web contendo código malicioso. Como resultado, informações sigilosas tais como *Cookies* ou outros dados sensíveis podem ser disponibilizados para o atacante. A Figura 2.8 apresentada um exemplo de uma requisição com ataque XSS não persistente.

Na requisição web, o atacante explora o roubo do *Cookie* do usuário através da inserção do código *Javascript*. Desta forma, o parâmetro “*username*” é utilizado para transmitir o có-

```
GET /index.php?sessionid=12312312\&username=<script>
document.location=`http://attackerhost.example/cgibin/
cookiesteal.cgi?'+document.cookie</script>
```

Figura 2.8: Exemplo de ataque XSS não persistente.

digo para ser executado na máquina do usuário. Além disto, esse ataque é caracterizado pelo uso de uma quantidade superior de caracteres, devido a inserção de informações maliciosas na requisição.

A categoria de ataques XSS persistente geralmente exploram aplicações web que possuem informações transmitidas pelos usuários, tais como fóruns, comentários de blogs, *chat*, entre outros (FOGIE et al., 2007). Portanto, a execução desse ataque é realizada através da submissão de um código malicioso em um local onde usuários não necessitam de autenticação para obter as informações, resultando na transmissão de informações não autorizadas pela aplicação web. Após a submissão pelo atacante, o código é transmitido para todo usuário que acessar o local infectado, de maneira transparente, e a execução do ataque é automática sem o conhecimento da vítima. Ao contrário do ataque XSS não persistente, esse ataque não necessita de link para que os usuários sejam afetados, portanto, a abrangência da sua aplicação bem sucedida é potencialmente superior.

- SQL Injection

Grande parte das aplicações disponibilizadas na Internet operam com banco de dados para o armazenamento de informações. Neste sentido, as aplicações obtêm os dados através de consultas SQL (*Structured Query Language*), permitindo a leitura, atualização, adição e remoção de informações dentro do banco de dados (STUTTARD; PINTO, 2007). O SQL é uma linguagem interpretada e muito comum na construção de consultas que incorporam dados fornecidos pelos usuários. Portanto, quando a construção de consultas é realizada de maneira insegura, afeta a segurança da aplicação tornando-a vulnerável a ataques *SQL Injection*.

O ataque *SQL Injection* (SU; WASSERMANN, 2006) explora as consultas SQL das aplicações web que utilizam bancos de dados para armazenamento de informações. Para isso, realiza modificações na ação desejada dentro de uma consulta através da manipulação de palavras-chaves e operadores (CVE-2011-2703, 2010). Conforme pode ser observado na Figura 2.9, o fluxo da execução do ataque é através da injeção de código malicioso pelo atacante (rótulo 1). Após a injeção, a aplicação web envia a consulta SQL ao banco de dados para ser executada,

no rótulo 2. Deste modo, uma aplicação web vulnerável a ataques *SQL Injection* podem causar problemas para organizações, tais como leitura e modificação de todos as informações armazenadas no banco de dados e também obter controle sobre o servidor que o banco de dados está armazenado (STUTTARD; PINTO, 2007).

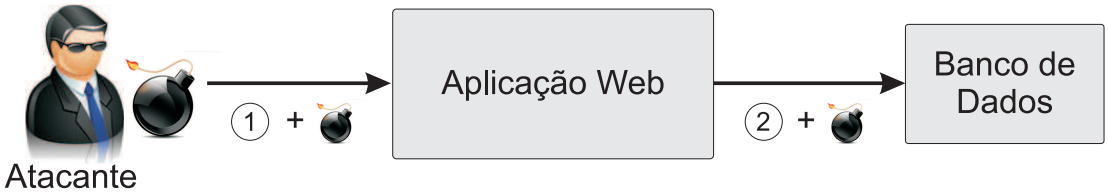

Figura 2.9: Ataque *SQL Injection*.

O ataque *SQL Injection* pode ser aplicado através de diversas formas de injeção (HALFOND; VIEGAS; ORSO, 2006), tais como: entrada de dados de usuários, *cookies*, variáveis do servidor e segunda ordem. Aplicações web vulneráveis a esse tipo de ataque muitas vezes não realizam validação dos dados incorporados pelos usuários. Deste modo, possibilita que atacantes manipulem as consultas SQL para obter acesso a informações sigilosas das aplicações. Esse ataque é caracterizado pela alteração da estrutura das consultas SQL, possibilitando que toda a informação pertencente a base de dados seja manipulada.

Devido à facilidade da aplicação do ataque *SQL Injection*, tornou-se muito comum esse ataque em aplicações web. Através da modificação da lógica, semântica ou sintaxe de uma instrução SQL original, permite que os dados armazenados no banco de dados sejam alterados (HALFOND; VIEGAS; ORSO, 2006). A Figura 2.10 apresenta um exemplo de uma aplicação sem ataque *SQL Injection* que recebe o número de identificação de um usuário, através de um parâmetro da requisição web.

```
GET /page.php?id=14
```

Figura 2.10: Exemplo de aplicação que recebe um número de identificação como parâmetro.

Para a aplicação do ataque, um usuário malicioso insere um valor no parâmetro “id” (Figura 2.11). Como resultado da inserção do comando SQL, a aplicação web concede acesso de todos os registros dos usuários para o atacante.

- Path Traversal

O ataque *Path Traversal* (também chamado *Directory Traversal*) explora as aplicações web que possuem vulnerabilidades provocadas por falhas nas funcionalidades de leitura e escrita de

```
GET /page.php?id=\%27/**/+union+/**/select/**/+1,2,
3,version(),user(),6/**/--+
```

Figura 2.11: Exemplo de aplicação do ataque *SQL Injection*.

dados em sistemas de arquivos (ROBERTSON et al., 2006). Os sistemas operacionais permitem especificar o mesmo arquivo de diferentes maneiras. Por exemplo, o caminho "../passwd" e "/etc/passwd" podem apontar para o mesmo caminho.

Os dados fornecidos pelos usuários através de requisições web podem conter operações maliciosas, isto é, um atacante pode submeter dados com intenção de provocar a leitura de arquivos não autorizados. Muitas vezes são arquivos contendo senhas de usuários, histórico de operações da aplicação web (logs) e arquivos de configurações de serviços críticos (STUTTARD; PINTO, 2007).

Esse ataque utiliza os caracteres "." e "/" (em alguns casos "\") para acessar os diretórios acima do diretório raiz do servidor web. Assim, com o uso de múltiplos "../" permite ao usuário malicioso enganar o interpretador localizado no servidor web. Por exemplo, o ataque com o uso dessa técnica possibilita a edição de arquivos arbitrários do sistema, conforme a vulnerabilidade publicada em (CVE-2010-1679, 2011). Esse ataque pode ser observado na Figura 2.12.

```
GET /page.php?p=../../../../../../../../etc/passwd
```

Figura 2.12: Exemplo do ataque *Path Traversal*.

As informações maliciosas são inseridas no parâmetro "p" com a intenção de obter acesso à informações restritas do arquivo de credenciais do sistema UNIX contido no servidor web. No exemplo a aplicação "page.php" é vulnerável a esse ataque, então, as informações serão disponibilizadas conforme a Figura 2.13.

A característica dos ataques da classe *Traversal* possuem a quantidade de caracteres "." e "/" superior aos demais. Esse ataque é caracterizado pela alteração da distribuição da frequência dos caracteres (ROBERTSON, 2009). Usualmente, eles são bloqueados através de validações das expressões e caracteres utilizados. Para isso, são utilizados filtros para identificar uma possível injeção de código significativo para esse ataque. A identificação do ataque pode ser feito através da verificação de parâmetros contendo sequências de caracteres que caracterizam o ataque *Traversal*. Uma maneira de bloquear seria com remoção destes caracteres que potencializam o ataque. Entretanto, tais filtros são vulnerável devido ao uso de técnicas de aplicação de codificação alternativa, isto é, através da utilização de esquemas de codificação, tais como

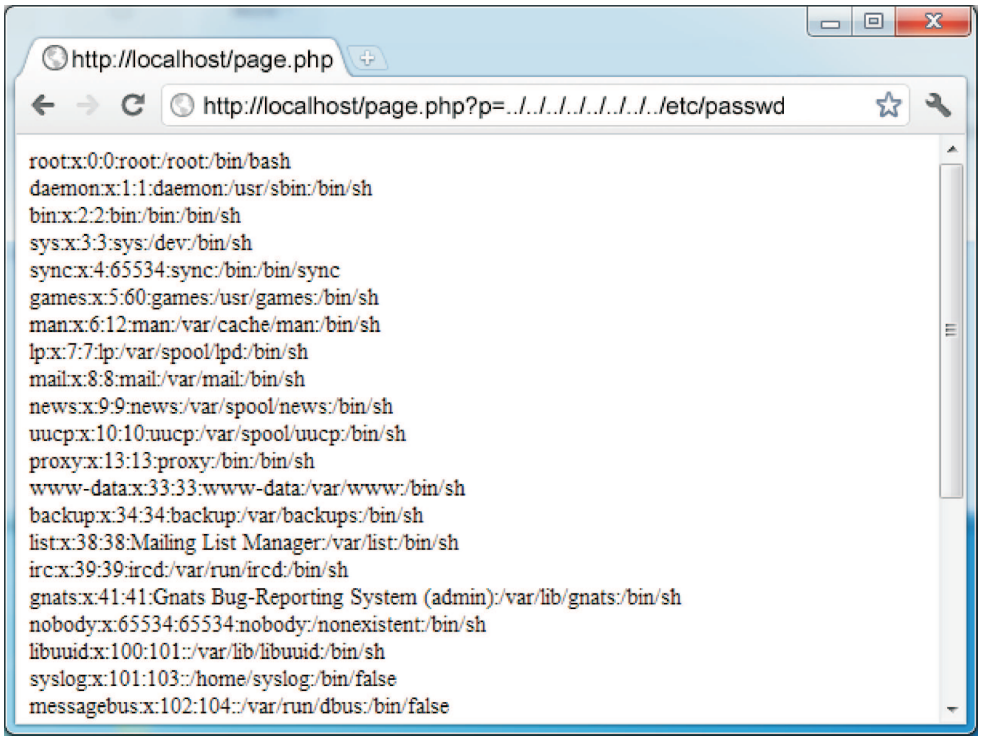

Figura 2.13: Exemplo do ataque *Path Traversal*.

codificação URL, Unicode, HTML, Base64 e Hexadecimal. Além disso, a construção de requisições que escapem de verificações de segurança tem explorado também o *Double Decode* que utiliza caracteres hexadecimais duplamente codificados, ao invés de representações contendo esquemas de codificações simples (STUTTARD; PINTO, 2007).

A Figura 2.14 apresenta o mesmo exemplo da figura 2.12 com codificação hexadecimal. De fato, aplicações web com filtros que buscam pela ocorrência de sequência de caracteres "../" não iriam detectar esse ataque, tornando a aplicação vulnerável.

```
GET /page.php?p=%2e%2e%2f%2e%2e%2f%2e%2e%2f%2e%2e%2fetc/passwd
```

Figura 2.14: Exemplo do ataque *Path Traversal* com codificação hexadecimal.

Com o crescente número de ataques, causando sérios problemas de interrupção de serviços, a detecção de intrusão tornou-se uma área de pesquisa extremamente importante para a segurança dos sistemas computacionais (VACCA, 2009). A complexidade dos ataques tem dificultado a detecção através de soluções tradicionais para a proteção dos sistemas, tornando os sistemas computacionais propensos a intrusões. O estudo e desenvolvimento de novas abordagens de proteção para esses sistemas tem como motivação a busca pela eficiência dos mecanismos de detecção.

2.3 Sistema de Detecção de Intrusão

Na área de segurança da informação a definição de Intrusão tem significados distintos e vem sendo debatido pelos profissionais da área. Muitos consideram uma Intrusão quando um ataque não obteve sucesso, enquanto outros diferenciam ataque de intrusão. Neste trabalho adota-se a definição:

Definição 2 *Uma Intrusão é qualquer tentativa ilegal e deliberada, bem sucedida ou não, de manipulação ou rompimento do funcionamento de um sistema (KIZZA, 2005).*

Detecção de Intrusão é um processo de identificação e diagnóstico de atividades maliciosas em redes de computadores (KRUEGEL; VALEUR; VIGNA, 2004). Esse processo é essencialmente uma combinação de monitoramento, análise e resposta (VACCA, 2009). Além disso, ocorre o envolvimento de pessoas, ferramentas e tecnologias. Portanto, os Sistemas de Detecção de Intrusão (IDS, na sigla em inglês) são softwares projetados para analisar atividades e tomar decisões cabíveis quando uma atividade possui o comportamento característico de um ataque.

No contexto de IDS, o critério de avaliação está relacionado com a capacidade de diferenciar atividades normais de atividades maliciosas (ROBERTSON et al., 2006). A distinção das atividades caracteriza a eficiência de um IDS e algumas medidas são utilizadas para isso. Tanto os desenvolvedores quando os projetistas dos IDS, independente das técnicas de detecção utilizadas, buscam a maximização da taxa de *verdadeiro positivo* e a minimização da taxa de *falso positivo*. O *verdadeiro positivo* que consiste na identificação correta de uma atividade como maliciosa na perspectiva de segurança. Por outro lado, o *falso positivo* consiste na classificação de um ataque ou atividade maliciosa como uma atividade normal. Por outro lado, um falso negativo caracteriza quando uma atividade normal é detectada como anomalia e verdadeiro negativo ocorre quando uma atividade é considerada normal pelo sistema.

Com base nessas medidas é possível definir a taxa de *verdadeiro positivo* e a taxa de *falso positivo*, geralmente utilizadas para avaliação de IDS, conforme deferido a seguir.

$$\text{Taxa de verdadeiro positivo} = \frac{\text{Verdadeiro Positivo}}{\text{Verdadeiro Positivo + Falso Negativo}} \tag{2.1}$$

$$\text{Taxa de falso positivo} = \frac{\text{Falso Positivo}}{\text{Falso Positivo + Verdadeiro Negativo}} \tag{2.2}$$

Além da medição indicada, um método popularmente conhecido como curva ROC (*Receiver Operation Characteristic*) (PROVOST; FAWCETT; KOHAVI, 1998) é utilizado para avaliação da capacidade de detecção dos IDS. A representação gráfica apresenta a relação entre a taxa de *verdadeiro positivo* associados com a taxa de falsos positivo dentro de um intervalo [0,1]. A origem [0,0] corresponde à classificação de atividades normais, enquanto que [1,1] corresponde a classificação de atividades maliciosas. A Figura 2.15 apresenta um exemplo da curva ROC do desempenho da detecção de três IDS.

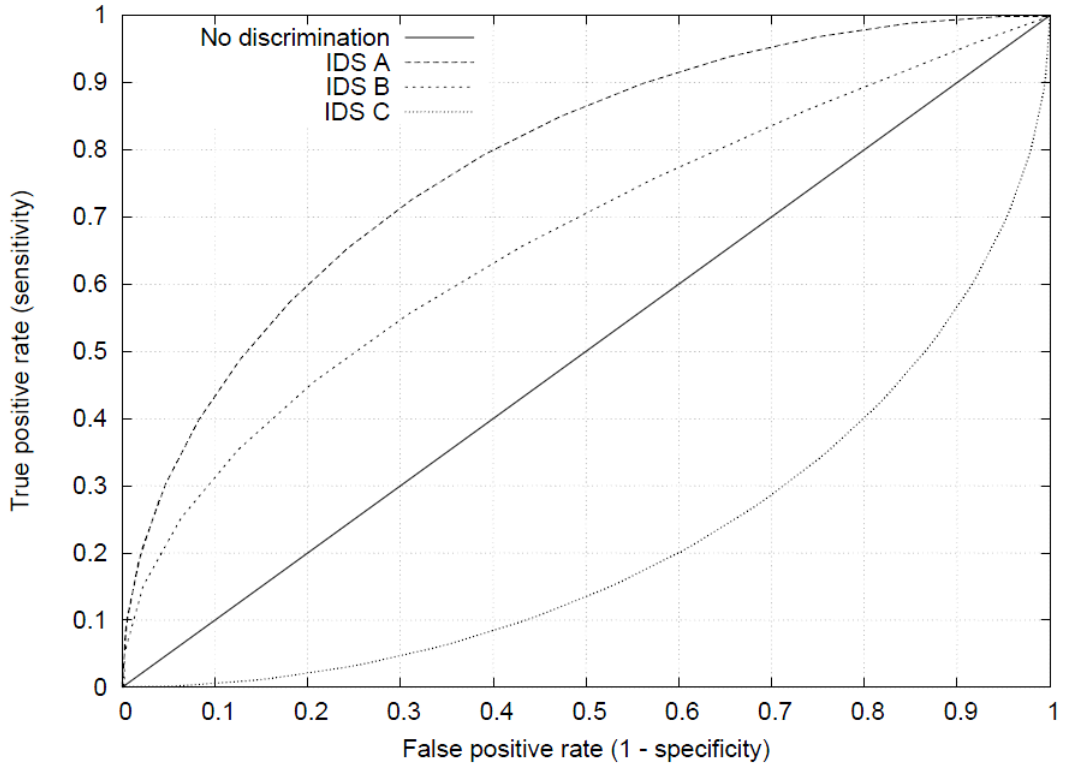

Figura 2.15: Exemplo de representação gráfica utilizando curva ROC (ROBERTSON et al., 2006).

Como se pode observar na figura, o IDS A demonstrou melhor eficiência (maior sensibilidade) com relação à taxa de *falso positivo* que os IDS B e IDS C.

Segundo Debar et al. (1999), os IDS são classificados com base em diferentes características. A Tabela 2.4 apresenta as principais características dos IDS. O método de detecção descreve as características do analisador para identificar intrusões. Ainda no método de detecção os IDS podem ser baseados em anomalia ou baseado em assinatura. IDS baseado em anomalias adotam um comportamento normal e, a partir disso, utilizam esse padrão para monitorar e detectar intrusões no sistema. Já os IDS que utilizam informações sobre os ataques ou conjunto de informações que caracterizam as assinaturas dos ataques são denominados IDS

Tabela 2.4: Diferentes tipos de sistemas de detecção de intrusão.

Critério	Tipos	Descrição
Método de detecção	Baseado em assinatura	Definição do modo de detecção. A abordagem em assinatura utiliza conhecimento prévio dos ataques para detectá-los. A abordagem baseada em assinatura emite um alarme quando existe um desvio substancial do que é considerado usual.
	Baseado em anomalia	
Comportamento de detecção	IDS	Definição do comportamento de detecção. O IDS (passivo) somente gera alerta. IPS (ativo) realiza ações pró-ativas quando detectada uma intrusão.
	IPS	
Local de monitoramento	HIDS	Estabelece localização dos dados a serem analisados. HIDS para dados coletados de um computador host. NIDS para dados coletados de uma rede de computadores.
	NIDS	
Frequência de uso	*Online*	Distinção entre os sistemas que analisam dados em tempo real (*online*) e aquele que analisam a cada intervalo de tempo (*offline*).
	Offline	

baseados em em assinatura.

Além da detecção, os IDS também são classificados de acordo com seu comportamento após a detecção dos ataques, ou seja, realizando ações quando ataques são identificados (KRUEGEL; VIGNA, 2003). No entanto, IDS que somente geram alertas de intrusões para o sistema são chamados de IDS *passivo*. Já os IDS que reagem aos ataques são considerados *ativos* ou também chamados IPS (*Intrusion Prevention System*). Eles efetuam alterações de configurações necessárias para corrigir o problema ou mesmo para bloquear um determinado serviço afetado pela intrusão. A maioria dos IDS são do tipo *passivo*, os quais simplesmente detectam e reportam alertas para entidades responsáveis.

Outra característica dos IDS está relacionada ao local de origem da auditoria ou local de monitoramento (DEBAR; DACIER; WESPI, 1999), distinguindo o tipo de informação de entrada do IDS. Se a informação de análise for coletada em computadores o IDS é baseado em *Host* (HIDS). Caso contrário, se for coletado em redes de computadores o IDS é baseado em *Network* (NIDS). Por fim, podem ser classificados de acordo com a frequência de uso (DEBAR; DACIER; WESPI, 1999): execução contínua (*online*) ou periódica (*offline*).

Os IDS baseada em assinaturas e os baseados em anomalias são similares em termos de funcionamento e composição conceitual, porém possuem uma diferença com relação a metodologia utilizada (GARCIA TEODORO et al., 2009). A seguir na subseção 2.3.1 são descritos os IDS baseados em assinatura, bem como os IDS baseados em anomalia, apresentados na subseção 2.3.2.

2.3.1 Sistema Detecção de Intrusão baseado em Assinatura

Os IDS baseados em assinaturas (KIZZA, 2005) são softwares que monitoram informações coletadas de um computador ou rede de computadores, e identificam ataques comparando os dados com um conjunto de assinaturas de ataques conhecidos. A ocorrência de um evento similar a alguma assinatura dispara um alarme no sistema. Assim, a detecção de intrusão baseada em assinatura leva em consideração que todo ataque é padronizado, isto é, constitui-se de uma assinatura para ser realizado, ou mesmo uma sequência de operações que são postas em prática para colocar em risco a segurança de um sistema.

Essa abordagem tem ótimos resultados em termos de precisão e baixa taxa de *falso positivo*, embora tenha alguns problemas. O primeiro deles é que muitos sistemas não identificam ata-

ques polimórficos[2]. Ou seja, o ataque vai passar despercebido pelo sistema até que o conjunto de assinaturas seja atualizado com essa nova assinatura. Essa abordagem não é capaz de identificar pequenas modificações de um ataque conhecido e necessita de uma nova assinatura para a alteração. O segundo problema é em relação ao crescente surgimento de novos ataques. Para manter a eficiência do IDS, o conjunto de assinaturas necessita de constantes atualizações, pois essa abordagem somente detecta ataques conhecidos.

Por outro lado, a abordagem baseada em assinatura é bastante utilizada pela capacidade de alcançar alta precisão na detecção com baixa taxa de *falso positivo*. Os ataques são identificados quando caracterizam eventos de uma assinatura de ataque conhecido (ROBERTSON, 2009). Portanto, a ocorrência de falsos alarmes só é gerada em eventuais assinaturas incorretas. Entretanto, essa abordagem não é aconselhável para sistemas que possuem ocorrências frequentes de novos ataques ou anomalias.

Dois sistema baseado em assinatura bastante utilizados e difundidos pelos profissionais de segurança da informação são: *Snort* (ROESCH, 1999) e *Bro* (PAXSON, 1999). Esses sistemas se popularizaram por serem disponibilizados livremente, pois ambos adotam um modelo de desenvolvimento de código aberto.

2.3.2 Sistema de Detecção de Intrusão baseado em Anomalia

A abordagem baseada em anomalia tem sido explorada por ser uma classe promissora para mecanismos de detecção de ataques (CHANDOLA; BANERJEE; KUMAR, 2009) (ROBERTSON, 2009). Neste contexto, a detecção de intrusão baseada em anomalias possibilita a identificação de ataques que visam às vulnerabilidades *zero-day*, ou seja, ataques desconhecidos (SRIRAGHAVAN; LUCCHESE, 2008).

Geralmente, os IDS baseado em anomalia monitoram um conjunto de dados considerados livre de ataques e, a fim de modelar o comportamento usual de um sistema. Diversos trabalhos consideram esse modelo de comportamento livre de ataques como *normal profile* (KRUEGEL; VIGNA, 2003) (WANG; STOLFO, 2004) (INGHAM et al., 2007). Quando utilizado com o propósito de detecção de intrusão, esses sistemas analisam os dados de entrada e comparam com o *normal profile* criado previamente. Desta maneira, qualquer desvio de comportamento é considerado como uma anomalia e consequentemente um ataque, onde os dados são marcados como anômalos (ROBERTSON et al., 2006).

[2]São variações de ataques conhecidos, sem perder a eficácia.

Na detecção de anomalias existe o problema de identificar padrões de dados diferentes do comportamento esperado (CHANDOLA; BANERJEE; KUMAR, 2009) e que são frequentemente considerados como anomalias, mas que de fato são usuais. Nesta abordagem de detecção nem toda a identificação de anomalia é realmente um ataque, e essa situação é chamada de falso alarme ou *falso positivo*. Este caso é visto como uma desvantagem da abordagem baseada em anomalia.

Além disso, os IDS baseado em anomalia possui alguns desafios na área de detecção de intrusão. Chandola et al. (2009) salienta alguns deles:

- Definir um comportamento normal ou anormal é muito difícil porque os limites desses comportamentos não são precisos e com isso podem ocorrer confusões quando um comportamento esteja próximo do limite.

- Quando anomalias são resultantes de ações maliciosas, então o indivíduo malicioso adapta suas ações para que a observação anômala aproxime-se de um comportamento normal, dificultando a tarefa da definição de modelos de comportamento normal.

- Disponibilidade de dados rotulados para treinamento e validação de modelos usados pelas técnicas de detecção de anomalias é uma questão importante.

Entretanto, o desafio de obter conjuntos de dados com rótulos de informações normais e anomalias é abordado dentre alguns modos de detecção de anomalias:

- Supervisionado: assume-se a disponibilidade de um conjunto de dados para treinamento com informações de comportamento normal e um conjunto de dados com informações de anomalias. No modo supervisionado, geralmente a quantidade de informações no conjunto de dados de anomalias é muito inferior aos dados normais, dificultando a comparação. Além disso, obter um conjunto de dados confiáveis e representativos é um desafio.

- Semi-supervisionado: assume-se a disponibilidade de um conjunto de dados somente com informações do comportamento normal. O modo semi-supervisionado é indicado para técnicas que constróem um modelo correspondente ao comportamento normal, e utilizam-o para identificar anomalias.

- Não-supervisionado: assume-se que informações normais são mais frequentes que informações anômalas. Desta forma, técnicas operando no modo não-supervisionado utilizam

um conjunto de dados não rotulados para o treinamento dos dados e assume-se que os dados contém poucas anomalias.

2.4 Algoritmos de Detecção de Anomalias para Ataques Web

A detecção de intrusão baseada em anomalia explora a identificação de ataques web conforme a suposição que esses ataques provocam variações sobre o comportamento usual, ou seja, a perturbação causada pelo ataque web descaracteriza o comportamento esperado na análise (CHANDOLA; BANERJEE; KUMAR, 2009). Essa hipótese é considerada pelas técnicas de detecção de anomalias utilizadas para diferenciar entre o comportamento anômalo e normal (KRUEGEL; VIGNA, 2003)(INGHAM; INOUE, 2007).

Portanto, a estratégia para detectar anomalias consiste na definição do comportamento usual e também considerar qualquer anomalia conforme a observação que não corresponde com o comportamento esperado (CHANDOLA; BANERJEE; KUMAR, 2009). Neste contexto, o trabalho de (SONG; KEROMYTIS; STOLFO, 2009) propôs um algoritmo de detecção de anomalias através do modo supervisionado para a identificação de ataques *Cross-site Scripting* (XSS), *Remote File Inclusion* (RFI) e *SQL Injection*. Para isto, os parâmetros das requisições com método GET e POST foram analisados utilizando modelos de cadeias de *Markov* e *n-grams*. Nos testes realizados, os resultados foram bons quando utilizado *2-grams*. Outra forma de distinguir o comportamento anômalo é através de aplicação de técnicas estatísticas para identificar variações na similaridade entre duas distribuições (CHANDOLA; BANERJEE; KUMAR, 2009).

O trabalho de (JAMDAGNI et al., 2010) apresentou um algoritmo para verificar a similaridade entre as frequências dos caracteres dentro do *payload* do tráfego HTTP e para isso utilizou a distância *Mahalanobis Distance Map* (MDM). Outra técnica utilizada com mesmo objetivo é o teste de χ^2 que identifica a diferença entre duas distribuições. Os trabalhos de Kruegel e Vigna (2003), Kiani et al. (2008) e Sriraghavan e Lucchese (2008) apresentaram algoritmos de detecção de ataques web conforme a identificação de variabilidade entre a frequência dos caracteres nas requisições web. Para isso, esses trabalhos exploraram o método de detecção supervisionado, na qual uma fase de treinamento é utilizada para obter uma distribuição com um conjunto de dados sem ataques. Após isto, é aplicada a técnica para distinguir informações que desviam da distribuição ideal.

No trabalho de Ingham et al. (2007) foi proposto um algoritmo de detecção de anomalias supervisionado para a criar o modelo de comportamento usual utilizando DFA (*Deterministic*

Finite Automata). Na etapa de treinamento foram utilizados tokens extraídos das requisições web e armazenado cada tipo de token e seu valor. Algumas heurísticas foram utilizadas para validar entrada de valores bem conhecidas, tais como datas, tipos de arquivos, endereços IP, entre outros. Desta forma, cada estado do modelo DFA representa um único token, e existe uma transição entre dois estados consecutivos. Na fase de detecção, foi verificada a similaridade de cada requisição web com o modelo DFA determinando se a requisição é considerada anômala.

O algoritmo proposto por Song et al. (2009) apresenta um modelo baseado em aprendizagem que utilizada *Mixture-of-Markov-Chains* que incorporada com transições *n-gram* para modelar o comportamento usual das requisições web e identificar anomalias. Além disso, a fase de treinamento é supervisionada, ou seja, o conjunto de dados utilizados possui exemplos de requisições web normais e anormais. Os resultados da detecção de ataques foi satisfatória para ataques XSS, RFI e *SQL Injection*.

Por outro lado, a detecção de ataques web através de técnicas de processamento de sinais não tem sido explorada. Entretanto, para a detecção de ataques em redes de computadores alguns trabalho exploram a aplicação de ferramentas da área de processamento de sinais (LU; GHORBANI, 2009) (LI; LI, 2009). O trabalho de (LU; GHORBANI, 2009) explorou diferentes familias da transformada wavelet para verificar o impacto no desempenho da aplicação. O algoritmo proposto modela o tráfego de rede normal através da aplicação de um modelo *Auto Regressive with eXogenous input* (ARX) para detecção de anomalias. Outro trabalho que examinou a aplicação da transformada wavelet para a detecção de ataques web foi (LI; LI, 2009), que propôs a aplicação da wavelet para minimizar o número de falsos positivos na detecção de ataques (*Distributed Denial of Service*) DDoS.

2.5 Considerações Parciais

Esse capítulo apresentou a contextualização da detecção de intrusão na web, onde foi apresentado o funcionamento da web, assim como os pilares da segurança da informação em aplicações web. Conforme observado, as mensagens do protocolo HTTP com método GET, possuem um componente que permite a submissão de informações para efetuar consultas em aplicações web. De fato, permite a troca de informações através do modelo cliente-servidor, e desta forma, possibilita transmitir dados entre computadores na Internet.

Em contraponto, foram abordadas as principais vulnerabilidades que provocam preocupação para a segurança das aplicações web. No entanto, as vulnerabilidades de injeção de código e

Cross-site Scripting demonstraram maior abrangência conforme a maneira de execução, devido às falhas que podem ser exploradas através da manipulação de requisições web. Adicionalmente, se percebeu que os ataques mais comuns para as aplicações web modificam a estrutura de uma requisição web, ou seja, alteram a frequência usual dos caracteres com relação as requisições sem ataques. Neste sentido, o estudo sobre os ataques web demonstrou a necessidade de uma análise com relação a distribuição de frequência dos caracteres nas requisições web.

Por fim, foram detalhados os sistemas de detecção de intrusão e foram apresentadas as características e a classificação dos IDS. O método de detecção explorado neste trabalho tem como objetivo verificar as características e propriedades que potencializam a detecção de ataques web. A abordagem baseada em anomalias permite a identificação do comportamento usual de uma aplicação web, de modo que possibilita identificar qualquer desvio desse comportamento. A abordagem tem demonstrado potencial, conforme identificado em trabalhos relacionados, embora perceba-se a existência da possibilidade de geração de altas taxas de falso positivo. Além disso, verificou-se que técnicas de processamento de sinais não são utilizados para detecção de ataques web, somente sendo explorados para a detecção de ataques em redes de computadores.

3 WAVELETS

Neste capítulo é apresentada a família de *wavelets* de *Haar* referente às funções *wavelets* existentes, bem como suas transformadas discretas e as técnicas de operação de corte (*threshold*) dos coeficientes *wavelets*. O objetivo é apresentar uma base de conhecimento sobre a técnica de processamento de sinais, considerada como um dos principais itens para esta dissertação. Com base nisso salientar as propriedades da transformada *wavelet* que potencializam a análise do tráfego web, onde são consideradas as requisições web.

Na Seção 3.1 é apresentada um visão geral sobre as funções *wavelets* usadas neste trabalho, assim como os algoritmos da transformada *wavelet*. Já na Seção 3.2 é apresentada a operação de corte utilizada na detecção de anomalias. A Seção 3.3 apresenta as considerações parciais do capítulo.

3.1 Transformada Wavelet

A transformada wavelet é utilizada para decomposição de um sinal em diferentes níveis de resolução com o objetivo de realçar e salientar informações relevantes da composição do sinal. Ela permite a decomposição hierárquica de um sinal em uma representação grosseira e um grupo de detalhes (STOLLNITZ; DEROSE; SALESIN, 1995). Os detalhes são informações complementares da representação grosseira e são necessários para reconstrução do sinal original. Em aplicações de análise de sinais, eles são manipulados para permitir a compressão (USEVITCH, 2001), remoção de ruídos (KOZAKEVICIUS et al., 2005) ou outros extrações de características dos dados analisados.

Entre a diversidade de famílias wavelets, a família wavelet de Haar é a adotada nesta dissertação, principalmente devido a facilidade na preservação da localização e identificação de variações abruptas (MALLAT, 2009) através de suas transformadas associadas. Além disso, a transformada wavelet de Haar não requer tratamento especial para as fronteiras de sinais, comuns em outras famílias. Por se tratar de uma transformação linear e por ser ortogonal, a transformada wavelet mantém a energia do sinal analisado (BAYER; KOZAKEVICIUS, 2010).

3.1.1 Transformada Wavelet Discreta 1D

A transformada wavelet é o processo para obter coeficientes wavelets através da aplicação de filtros passa-alta (H) e passa-baixa (L) de tamanho $2N$, onde N é o número de filtros. Para

isso, o algoritmo piramidal, proposto por Mallat (2009), permite obter dois conjuntos de coeficientes wavelets entre valores de sinal adjacentes: um conjunto de médias (também chamado de coeficiente de aproximação ou escala) $c_{j-1,i}$ que contém valores de baixa frequência; e o conjunto de detalhes (conhecido como coeficientes wavelets) $d_{j-1,i}$ que contém valores de alta frequência. Para calcular os conjuntos de coeficientes, um vetor de dados de uma dimensão no nível mais fino é considerado. Portanto, a decomposição é realizada como segue:

$$c_{j-1,i} = \sum_{k=0}^{2N-1} L_k c_{j,2i+k}, \quad i = 0, ..., M_{j-1} - 1, \tag{3.1}$$

$$d_{j-1,i} = \sum_{k=0}^{2N-1} H_k c_{j,2i+k}, \quad i = 0, ..., M_{j-1} - 1, \tag{3.2}$$

Observe que a diferença entre os conjuntos de coeficientes são obtidos através da aplicação dos filtros. Uma importante propriedade da transformada wavelet é a capacidade de identificar ruídos, pulos e picos sobre o conjunto de detalhes (GRANE; VEIGA, 2009). Essa propriedade permite aplicar a transformada wavelet para a localização e identificação de mudanças abruptas. Neste trabalho são explorados os algoritmos proposto por Stollnitz et al. (1995). A Figura 3.1 apresenta o algoritmo para calcular a transformada wavelet unidimensional (TW1D).

```
 1  procedure DecompositionStep(C: array [1..h] of real)
 2     for i ◄— to h/2 do
 3         C'[i] ◄— (C[2i – 1] + C[2i]) / √2
 4         C'[h/2 + i] ◄— (C[2i – 1] – C[2i] / √2
 5     end for
 6      C ◄— C'
 7  end procedure

 8  procedure Decomposition(C: array[1..h] of real)
 9     while h > 1 do
10         DecompositionStep(C[1..h])
11         h ◄— h/2
12     end while
13  end procedure
```

Figura 3.1: Algoritmo para decomposição da TW1D (adaptado de Stollnitz et al. (1995)).

No procedimento de início, linha 8, os dados de entrada são processados até quando o nível de decomposição mais grosseiro seja alcançado. Na linha 1, cada conjunto de dados associado aos valores médios de um nível é decomposto em um novo conjunto de médias (denotado por

$C'[i]$ na Figura 3.1) e em um novo conjunto de detalhes (denotado por $C'[\frac{h}{2}+1]$), ambos com metade do tamanho do conjunto original das médias. Um aspecto importante é que somente os blocos associados às médias (coeficientes de aproximação) são melhores decompostos. Outro aspecto, no algoritmo proposto, é que todas as decomposições são armazenadas sobre o mesmo vetor de dados C ao invés de utilizar dois diferentes vetores, conforme sugerido pelas Equações 3.1 e 3.2 do algoritmo piramidal. O nível da transformada corresponde ao número de interações do procedimento de decomposição. Cada nível da transformada wavelet Haar é realizada em uma forma diádica, ou seja, os dados de entrada são sempre divididos em conjuntos com a metade do tamanho. Para a execução do algoritmo, o nível mais grosseiro deve conter pelo menos dois valores.

Para apresentar a seguir o funcionamento da decomposição da transformada wavelet, é utilizado o exemplo apresentado por Nielsen (1998). Nesta demonstração, os dados de entrada são compostos por quatros valores que compõem um vetor, que ao final do processo resultará um vetor com valores correspondente a baixa resolução em relação ao original.

$$[9 \quad 7 \quad 3 \quad 5]$$

Durante o processo, algumas informações do vetor original serão descartadas. No entanto, para que possamos recuperar posteriormente o conjunto original de valores é necessário que os coeficientes wavelet sejam armazenados. Estes coeficientes são responsáveis pela recuperação das informações perdidas.

A Tabela 3.1 apresenta o processo de decomposição wavelet. Para fins de conhecimento nesse exemplo foi utilizado o valor 2 ao invés de $\sqrt{2}$ como é proposto nas linhas 3 e 4 do algoritmo da Figura 3.1. Inicialmente, os valores do vetor possuem a resolução máxima (4) sem a presença de detalhes, chamados de coeficientes wavelets. O processo de decomposição da TW1D utiliza os valores adjacentes para as operações.

Tabela 3.1: Exemplo de decomposição wavelet.

Resolução	Coeficientes de aproximação	Coeficientes wavelets
4	[9 7 3 5]	
2	[8 4]	[1 -1]
1	[6]	[2]

No exemplo, os coeficientes de aproximação são obtidos com a adição dos valores 9 e 7 e os resultados são divididos por 2, resultando no valor 8. Da mesma forma ocorre para os

coeficientes wavelet, mas os valores são subtraídos e o resultado é divido por 2, resultando no valor 1. Essas operações são realizadas nos próximos valores adjacentes, 3 e 5, resultando no coeficiente de aproximação com valor 4 e o coeficiente wavelet de valor -1. Esse processo pode ser realizado até quando a quantidade mínima dos coeficientes de aproximação forem maior ou igual a 2.

Nesse exemplo de decomposição da TW1D, através das médias e diferenças dos quatro números ou amostras, foram obtidos os valores $[6 \quad 2 \quad 1 \quad -1]$. A transformada wavelet possui uma propriedade importante que é a possibilidade de reconstrução da sinal original. Com os coeficientes wavelets é possível capturar as informações perdidas. Essa característica é extremamente importante para a aplicação de compressão de imagens, tal como JPEG 2000 (USEVITCH, 2001). Por exemplo, se o coeficiente de aproximação for adicionado ao primeiro valor dos coeficientes wavelets, que é igual a 2, será obtido o valor 8 e depois se for subtraído alcança-se o valor 4. Desta forma, realizando a TW1D inversa é possível reconstruir a imagem original.

No processamento de sinais, a TW1D é explorada em aplicações de compressão de dados, identificação de sinais característicos e eliminação de ruídos. A Figura 3.2 apresenta a decomposição da transformada wavelet unidimensional de um sinal representado pela função $f(x) = sen(\pi * x)$.

Nota-se que na decomposição da TW1D do sinal original são gerados dois novos conjuntos de dados: coeficientes de aproximação e coeficientes wavelets. Além disso, para cada nível de decomposição o número de amostras é dividido pela metade nos dois conjuntos. A identificação de oscilações ou variações abruptas também podem ser identificadas na decomposição da TW1D. Diante disso, a Figura 3.3 apresenta a decomposição de um sinal representado pela função $f(x) = sen(1.2 * \pi * x) + e^{-500*(j-0.5)^2}$, com uma função gaussiana dentro do intervalo [0,1].

Baseado na propriedade da transformada wavelet, a perturbação gerada no sinal original também é vista nos conjuntos de coeficientes até o 3º nível de decomposição. Nos detalhes (coeficientes wavelets), a variação aparece com maior evidência e também quanto maior o nível de decomposição, melhor pode ser identificada a perturbação no sinal. Além disso, no primeiro nível já é possível identificar a perturbação contida no sinal analisado. Desta forma, na análise da decomposição da TW1D, o conjunto dos coeficientes wavelets é fundamental para identificação de variações no sinal original.

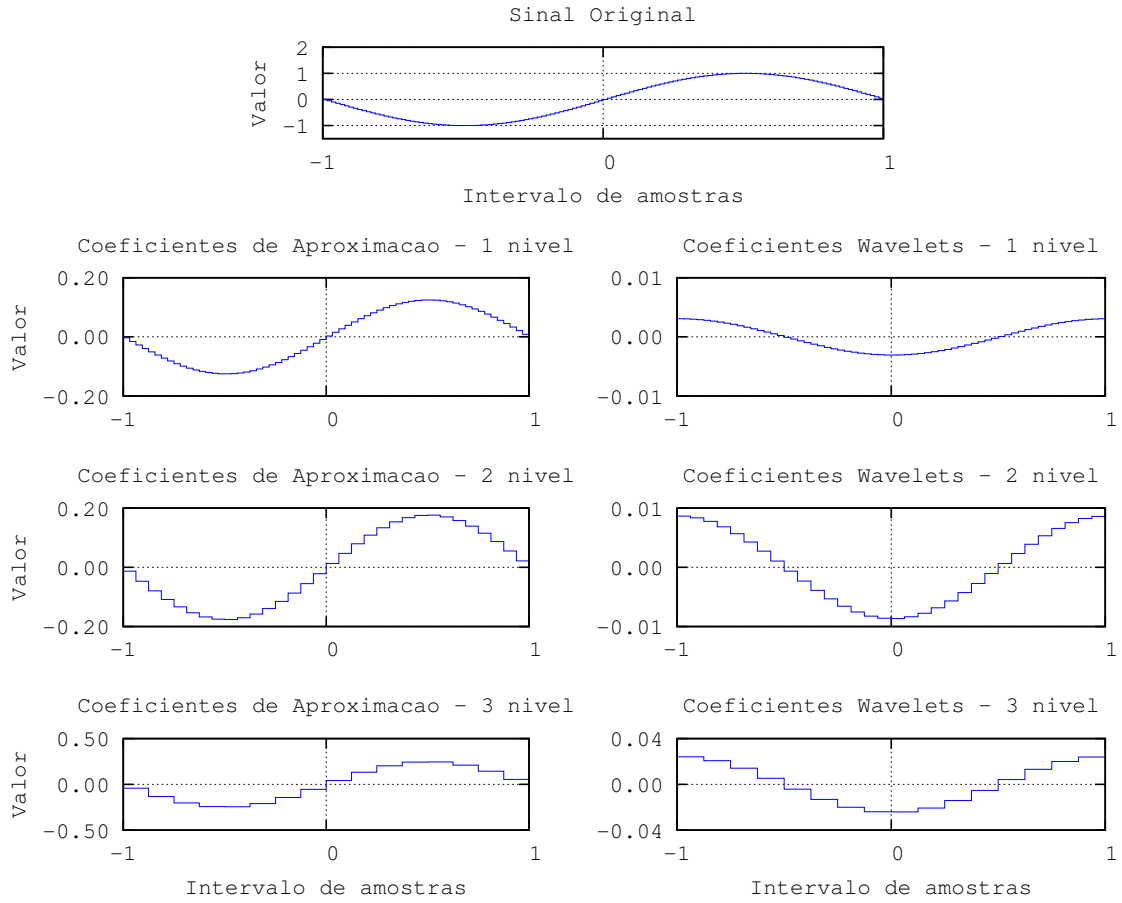

Figura 3.2: Decomposição da TW1D de uma função $f(x) = sen(\pi * x)$ no intervalo [0,1] com 128 amostras.

3.1.2 Transformada Wavelet Discreta 2D

A transformada wavelet bidimensional (TW2D) é a generalização da TW1D, através da aplicação da transformada wavelet unidimensional nas linhas e nas colunas, aplicando os filtros passa-baixa e passa-alta. A TW2D proporciona a realização de uma análise dos dados e identificação de variações entre as linhas e também entre as colunas. Além disso, na decomposição da TW2D outro conjunto de variações está relacionado com a combinação linear de ambos, isto é, além de considerar as variações das direção horizontal e vertical, também considera-se a direção diagonal.

Em outras palavras, a TW2D gera três conjuntos de coeficientes wavelets (denominadas subbandas) para representar as variações dos dados, diferentemente da TW1D, que gera apenas um conjunto de coeficientes wavelets. Esta característica da TW2D é importante na análise do processamento de sinais, pois é no conjunto de detalhes que surgem evidências das perturbações. Portanto, a transformada wavelet bidimensional possibilita uma identificação confiável das va-

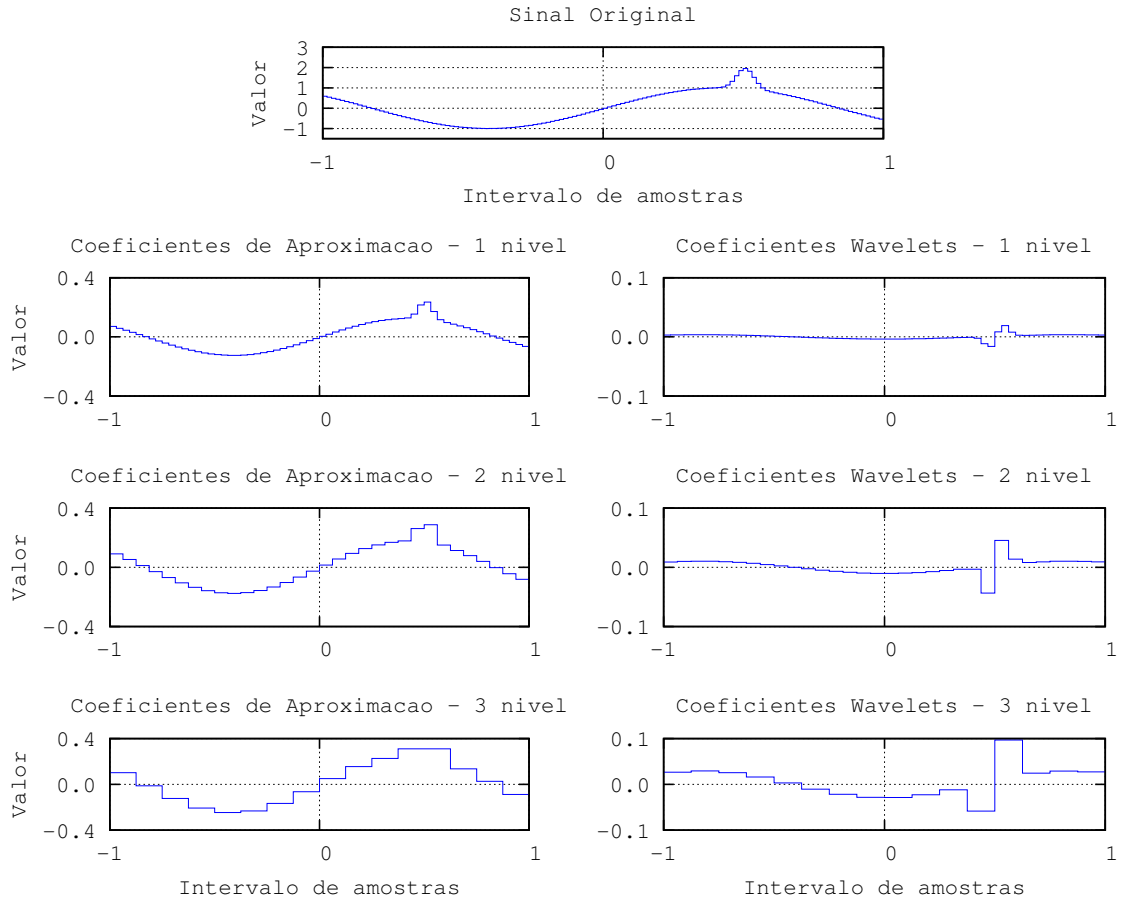

Figura 3.3: Decomposição da TW1D de uma função $f(x) = sen(1.2 * \pi * x) + e^{-500*(j-0.5)^2}$ no intervalo [0,1] com 128 amostras.

riações no sinal com o auxílio de conjuntos de detalhes contendo direções distintas, resultando numa análise refinada das informações.

Na prática, existem alguns caminhos para realizar a TW2D (WANG, 1995), uma vez que a ordem na qual a TW1D é realizada implica em um sistema diferente para representação dos dados. A TW2D é classificada de duas maneiras conforme (STOLLNITZ; DEROSE; SALESIN, 1995), que são as formulações *Standard* e *NonStandard*. Na formulação *Standard*, a TW1D é aplicada em todas as linhas até o final do processo de decomposição, isto é, até que tenha pelo menos dois valores adjacentes. Posteriormente, a TW1D é aplicada novamente até o final do processo, só que agora em todas as colunas. Na outra formulação, *NonStandard*, inicialmente a TW1D é aplicada em todas as linhas da matriz de entrada e, em seguida, aplica-se a TW1D para todas as colunas da matriz resultante. Esse processo é recursivo até o final do processo de decomposição, até que tenha pelo menos dois valores adjacentes. A Figura 3.4 apresenta o algoritmo da formulação *NonStandard*, a qual será utilizada nesta dissertação. A justificativa da escolha está relacionada com a aplicação de 1 nível da transformada, resultando nas variações

desejadas para a análise.

```
1   procedure NonstandardDecomposition(C: array[1..h, 1..h] of real)
2     while h > 1 do
3       for row ◄— 1 to h do
4         DecompositionStep(C[row, 1..h])
5       end for
6       for col ◄— 1 to h do
7         DecompositionStep(C[1..h, col])
8       end for
9       h ◄— h/2
10    end while
11  end procedure
```

Figura 3.4: Algoritmo para decomposição da TW2D: aplicação da TW1D para todas as linhas e depois em todas as colunas (adaptado de Stollnitz et al. (1995)).

O algoritmo é composto por duas chamadas para a TW1D, nas linhas 4 e 7. A primeira chamada para a aplicação da decomposição unidimensional para todas as linhas do conjunto de dados (linha 4) e outra para todas as colunas (linha 7). Ainda, esse processo somente é finalizado quando o conjunto de dados conter pelo menos dois valores, através da restrição da TW1D como pode ser visualizado na linha 2.

No intuito de realizar a decomposição da TW2D em diversos níveis, a processo da transformada nos níveis seguintes sempre será aplicada no conjunto dos coeficientes de aproximação, sendo considerado como nova entrada de dados, gerando quatros novos conjuntos de coeficientes. Em cada etapa da decomposição, os conjuntos de coeficientes wavelets permanecem inalterados.

A cada interação do procedimento da TW2D corresponde ao nível de decomposição, e quanto maior o nível da transformada menor a qualidade dos dados no conjunto de coeficientes de aproximação. Por outro lado, para o conjunto de coeficientes wavelets quanto maior o nível da transformada, maior o realce das informações significantes. Isso significa que conforme aumenta o nível as diferenças menores desaparecem em relação às maiores, permitindo uma melhor visualização das variações no conjunto de detalhes. No entanto, o nível utilizado na transformada implica na qualidade dos dados para análise e dependendo dos dados não é considerado a aplicação de muitos níveis, pois muitas vezes ocorre a perda de informações devido à elevação do nível na decomposição da transformada wavelet.

Na área de processamento de imagens, a TW2D é bastante utilizada devido à simplicidade

de sua aplicação e a capacidade de compactação de dados através da decomposição da transformada. Um exemplo da decomposição de um nível completo da TW2D com a formulação *NonStandard* é apresentado na Figura 3.5.

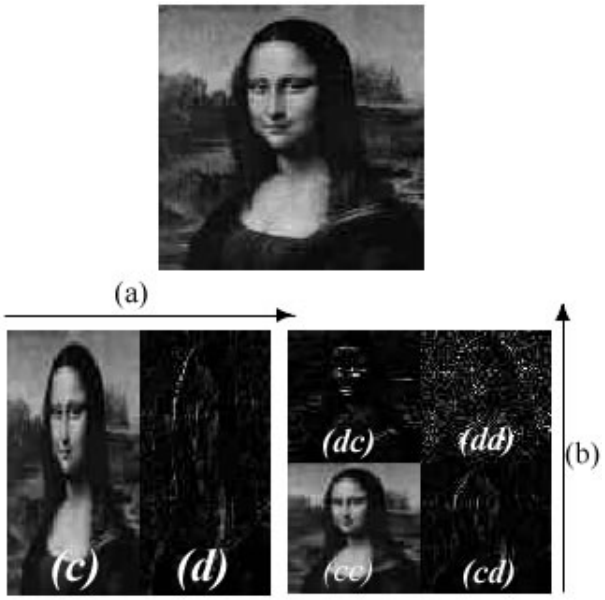

Figura 3.5: Decomposição *NonStandard* de uma imagem (adaptado de Stollnitz et al.(1995)).

Primeiro, no rótulo (*a*) da figura possui o estágio intermediário após a TW1D pelas linhas, onde os dados são decompostos em coeficientes de aproximação e wavelets. No rótulo (*b*) da figura é mostrado o resultado final, quando os três conjuntos de coeficientes wavelets são obtidos, bem como o conjunto de coeficientes de aproximação.

Na interpretação livre, pode-se definir os coeficientes de aproximação como um bloco formado pelas médias (médias das médias), e as três sub-bandas de coeficientes wavelets podem ser definidas como detalhes das médias (*dc*), médias dos detalhes (*cd*) e detalhes dos detalhes (*dd*), como uma tentativa para identificar a origem dos coeficientes wavelets gerados durante o processo de transformação.

A principal propriedade da TW2D permite verificar a variabilidade do sinal em múltiplas direções. Ao contrário da TW1D que analisa a variabilidade somente na horizontal, a TW2D proporciona uma análise do sinal na direção horizontal, vertical e diagonal. No entanto, mais fontes de informações com detalhes são disponibilizadas na analise da TW2D, que com a TW1D não é possível, visto que é gerado somente um conjunto de detalhes.

Por exemplo, a Figura 3.6 ilustra um conjunto de dados com um pico em todas variáveis do eixo Y de uma determinada variável do eixo X. A interseção dos eixos XY é composto de um valor que representa a intensidade daquele ponto do sinal e os picos são identificados devido a

intensidade alta com relação aos demais valores no conjunto de dados. Neste caso, a aplicação da TW1D poderia ser realizada sobre cada variável do eixo X ou Y. Neste exemplo, vamos supor que a TW1D foi aplicada em cada variável do eixo Y contendo todas as variáveis do eixo X. Isto significa, que cada processamento da TW1D irá verificar a variabilidade entre a intensidade dos valores do eixo X. Neste caso, os resultados das análises da TW1D serão a identificação de picos em todas variáveis do eixo Y com relação a variável 13 do eixo X.

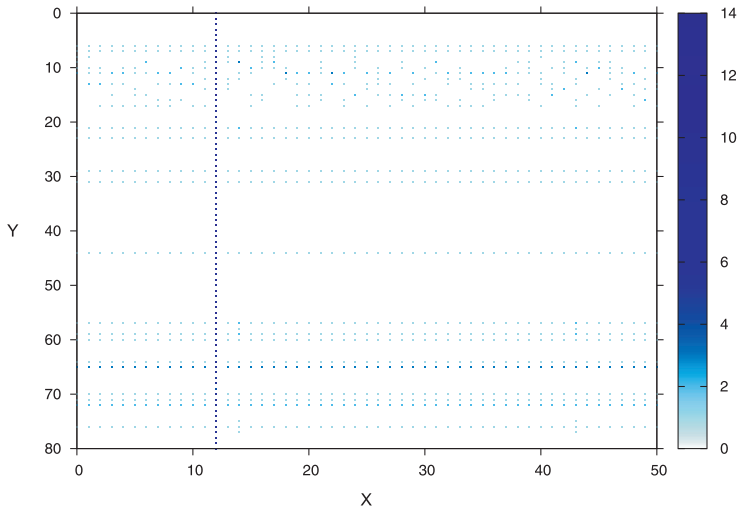

Figura 3.6: Conjunto de dados para análise da propriedade da TW2D.

Por outro lado, devido ao maior conjunto de detalhes da TW2D esse pico não é identificado, pois é considerado uma característica do sinal analisado. Como a TW2D realiza a aplicação da TW1D na linhas e depois nas colunas, o pico existe somente nos detalhes obtidos na análise das intensidades das linhas, da mesma forma que a análise da TW1D. Quando a TW2D aplica a decomposição pelas colunas, a variável 13 no eixo X possuem as intensidades similares, e neste caso não possuem diferenças significantes no sinal. Como resultado, não se identifica picos nessa variável. Com base na principal propriedade da transformada wavelet de identificação de picos, a TW2D possui a vantagem de relacionar os detalhes obtidos em múltiplas direções.

3.2 Operação de Corte

Após a aplicação da transformada wavelet bidimensional, o sinal é representado por quatro sub-bandas diferentes *cc*, *cd*, *dc* e *dd*, conforme explicado na seção anterior. As sub-bandas associadas aos coeficientes wavelets representam as variações dos dados capturados em três diferentes direções. De acordo com uma das principais propriedades da transformada wavelet, as informações grosseiras (médias dos coeficientes) representam com precisão os dados originais.

Portanto, a determinação do tamanho (significado) dos coeficientes wavelets é uma ferramenta para eliminar dados irrelevantes da representação ou para detectar variações relevantes nos dados analisados.

Assim, os critérios para descartar coeficientes wavelets são baseados basicamente na comparação entre os coeficientes de bandas *cd*, *dc* e *dd* com algum valor de referência, denominado valor de truncamento (*threshold*). Um dos valores mais utilizados em aplicações de filtragem é o *threshold* universal proposto por Donoho e Johnstone (1995), que é determinado pelo $\lambda = \sigma\sqrt{2\log(N)}$, onde σ corresponde ao desvio padrão dos coeficientes e N corresponde ao número de amostras e $\frac{N}{2}$ x $\frac{N}{2}$ = o tamanho de cada sub-banda. Quando o valor do *threshold* é escolhido de forma consistente, a operação de *threshold* não destrói as propriedades fundamentais do sinal (DONOHO; JOHNSTONE, 1995). A definição do *threshold* universal é um valor de corte para filtragem de ruídos. Neste sentido, o valor é considerado baixo e necessita de um ajuste fino conforme a aplicação, afim de melhorar a capacidade na identificação de informações significantes para a análise. Esse ajuste é realizado na operação de corte através do parâmetro ρ. Assim, o parâmetro é acrescentado na definição do *threshold* universal, tal como: $\lambda = \rho \cdot \sigma\sqrt{2\log(N)}$. Onde o parâmetro ρ é considerado um fator constante multiplicado pelo *threshold* universal, permitindo a correção do valor conforme a aplicação analisada.

Uma vez definido o valor de truncamento, estratégias de filtragem dos coeficientes são utilizadas. O trabalho de Donoho e Johnstone (1995) propôs duas maneiras de filtrar os coeficientes wavelets: aplicando o *Hard threshold* (Equação 3.3) ou através da aplicação do *Soft threshold* (Equação 3.4). O *Hard threshold* corta os coeficientes wavelet sem modificar os coeficientes maiores do que o valor do *threshold* e, portanto, considera como significativos. Essa estratégia é mais restritiva do que o *Soft threshold* que suaviza todo o conjunto dos coeficientes wavelets. Nesta dissertação os elementos analisados pelas Equações 3.3 e 3.4 são os coeficientes wavelet dos blocos *cd*, *dc* e *dd*, para cada nível e cada posição com relação ao nível.

$$d(k) = \begin{cases} 0 & , \mid d(k) \mid < \lambda \\ d(k) & , \mid d(k) \mid \geq \lambda \end{cases} \tag{3.3}$$

$$d(k) = \begin{cases} 0 & , |d(k)| < \lambda \\ d(k) - \lambda & , |d(k)| \geq \lambda \wedge d(k) \geq 0 \\ d(k) + \lambda & , |d(k)| \geq \lambda \wedge d(k) < 0 \end{cases} \tag{3.4}$$

Nesta dissertação foi adotada a estratégia de *Hard threshold*, pois as principais informações para modelagem da proposta do trabalho é a significância dos coeficientes wavelets, ao invés

do seu valor numérico.

Para uma explicação detalhada sobre as estratégias de definição do valor de truncamento, bem como diferentes critérios para a escolha do valor de truncamento, consulte (DONOHO; JOHNSTONE, 1995).

3.3 Considerações Parciais

Neste capítulo foram abordados os conceitos da transformada wavelet discreta e os algoritmos para a decomposição, bem como estratégias para operação de corte dos coeficiente de detalhes. As características da TW2D apontam que ela pode tornar-se uma ótima ferramenta para identificação de picos e correlação entre informações com detalhes significantes (DAUBECHIES, 1992).

A aplicação da transformada wavelet permite efetuar a decomposição de um sinal em diferentes níveis, onde as informações são descritas em um conjunto de aproximação e um conjunto de detalhes. A decomposição da TW2D tem como principal propriedade a identificação de variações em múltiplas direções no conjunto de dados analisado, assim como a possibilidade de relacionar coeficientes em conjunto de detalhes distintos. Essas variações são identificadas através dos três conjunto de detalhes obtidos na decomposição, que ao contrário da TW1D que somente possui um conjunto de detalhes. Em síntese, a variabilidade do sinal é melhor identificada através da análise utilizando mais de um conjunto de detalhes, o qual é extremamente importante quando se busca por variações nos dados.

A aplicação da operação de corte tem como principal característica a distinção entre os coeficientes mais significantes dentro do contexto aplicado, ou seja, através da estratégia de definição de um valor de truncamento, todo e qualquer valor abaixo deste limiar é descartado não sendo considerado pela operação. A estratégia *Hard threshold* define um valor de truncamento e realiza o processo de corte, onde são filtrados os valores acima do *threshold*.

4 PROPOSTA PARA DETECÇÃO DE ATAQUES WEB

Neste capítulo é apresentada a proposta para detecção de ataques web, baseada na aplicação da transformada wavelet bidimensional e operação de corte. A aplicação da TW2D é considerada uma ótima ferramenta para detecção de ataques web devido a necessidade de identificar a variabilidade nos dados contidos no tráfego HTTP.

A Seção 4.1 apresenta o modelo de sistema para a detecção de ataques web. Na Seção 4.2 são apresentadas as características dos ataques web, assim como o comportamento quando analisado entre um conjunto de requisições. A Seção 4.3 apresenta o algoritmo de detecção de ataques web baseado na TW2D e na Seção 4.4 são apresentados os trabalhos relacionados a proposta.

4.1 Modelo de Sistema

A proposta para detecção de ataques web apresentada neste capítulo considera um sistema distribuído baseado em troca de mensagens, onde os clientes solicitam recursos e serviços enviando requisições web para uma aplicação hospedada num servidor web remoto. A troca de mensagens entre cliente e servidor ocorre através do protocolo HTTP e as requisições são strings contendo endereços e parâmetros para uma aplicação web. Neste trabalho somente são consideradas mensagens com parâmetros enviadas para a aplicação do servidor web.

Assume-se que os ataques web sempre são lançados contra uma dada aplicação web e que provocam perturbação na frequência dos caracteres contidos nas requisições, se considerado a distribuição da frequência de caracteres das requisições web sem ataques. Adicionalmente, assume-se que os ataques não acontecem em rajadas e possuem ocorrências esporádicas.

4.2 Características dos Ataques Web

Anomalias podem ocorrer na comunicação do tráfego HTTP, provenientes de ataques web, causando sérios problemas para clientes e servidores (CHANDOLA; BANERJEE; KUMAR, 2009). Os ataques web de injeção de código, comuns em requisições web, são executados através da submissão de informações maliciosas pela aplicação web. Porém, uma análise entre requisições com ataques demonstra que anomalias são causadas devido ao tamanho superior as requisições sem ataques, uso de caracteres especiais e comandos *Unix* (SRIRAGHAVAN; LUCCHESE, 2008).

Considerando a distribuição das frequências dos caracteres, as requisições web sem ataques possuem uma estrutura regular e tipicamente utilizam caractere legíveis (KRUEGEL; VIGNA, 2003). A característica da estrutura regular consiste na similaridade da frequência dos caracteres entre um conjunto de requisições web. Considerando isso, variações bruscas entre caracteres de um mesmo conjunto podem indicar eventos maliciosos, tais como possíveis ataques. A Figura 4.1 apresenta um conjunto de requisições com um ataque *Path Traversal*.

```
GET /noticia.php?id=33603
GET /noticia.php?id=33606
GET /noticia.php?id=14030&busca=1
GET /noticia.php?id=29463
GET /noticia.php?id=31088
GET /noticia.php?id=../../../../../../../../../../etc/passwd
GET /noticia.php?id=32112
GET /noticia.php?id=30795
```

Figura 4.1: Conjunto de requisições web com ataques *Path Traversal*.

O ataque de injeção de código neste conjunto de requisições web se destaca entre as demais requisições pela quantidade de dois caracteres não usual, o que descaracteriza o comportamento padrão da aplicação web. Isto é, nas requisições web sem ataques os caracteres "." e "/" possuem apenas uma ocorrência, no entanto, a requisição com ataque possui o número de ocorrências consideravelmente superior. A análise do conjunto de requisições através da frequência dos caracteres potencializa a detecção de anomalias, de modo que a perturbação pode ser identificada devido aos caracteres possuirem intensidade superior aos demais.

As requisições não possuem as frequências dos caracteres distribuídas uniformemente com intensidades similares entre elas sobre uma determinada aplicação web (KRUEGEL; VIGNA, 2003). Isto significa que a análise de detecção de ataques em aplicações web pode ser explorada através da identificação de picos na intensidade da frequência dos caracteres nas requisições web. Adicionalmente, a análise em grupo de requisições permite a identificação de alterações na intensidade da frequência dos caracteres, pois os ataques web provocam um modificação quanto à localização e aparência das requisições com ataques (INGHAM, 2007).

Neste contexto, a transformada wavelet bidimensional é uma ferramenta de processamento de sinais que potencializa a identificação de mudanças abruptas no sinal analisado (UKIL; ZIVANOVIC, 2006). Portanto, esta ferramenta é ótima devido a sua principal propriedade que é identificar picos proporcionando uma análise em múltiplas direções entre as frequências dos caracteres do conjunto de requisições. A aplicação da TW2D permite realizar uma análise de

um grupo de requisições através do método de detecção não-supervisionado, o qual não utiliza uma fase de treinamento. Para isso, os dados de entrada devem ser organizados de modo que proporcione a aplicação da TW2D para a identificação de anomalias nas frequências dos caracteres das requisições web. A Figura 4.2 apresenta a modelagem de dados de entrada para a TW2D com o conjunto de requisições web.

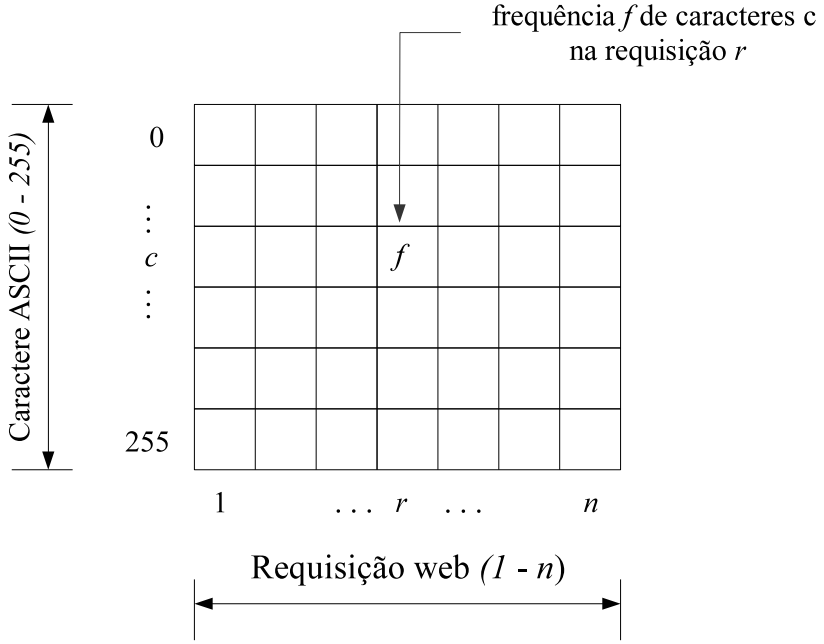

Figura 4.2: Matriz de dados como entrada para a detecção de ataques web.

Os dados foram modelados conforme a matriz $M[m][n]$, onde m é o número de caracteres ASCII e n o número de requisições. As linhas da matriz M são compostas pelos caracteres da tabela ASCII (c) e as colunas por um conjunto de requisições web (r) que estão sendo analisadas. Um requisito para a aplicação da TW2D é com relação ao número de requisições web (n^r) que deve ser múltiplo de 2. Esse requisito é um parâmetro ajustável, o qual neste trabalho é denominado de janela deslizante. A cada conjunto de n requisições web é processada a análise dos dados pela abordagem de detecção. Por exemplo, se o valor de n for definido como 256, a cada passo da janela de processamento com 256 novas requisições serão analisadas, descartando as 256 requisições antigas.

Para ilustrar a representação da modelagem dos dados, um conjunto de 256 requisições de uma determinada aplicação web sem ataques foi apresentado na Figura 4.3. Os caracteres possuem suas intensidades similares entre as requisições web. Outra característica está relacionada com os caracteres não usuais para a aplicação, que na modelagem possuem o valor 0.

A intensidade na frequência dos caracteres possui um nível de regularidade entre as re-

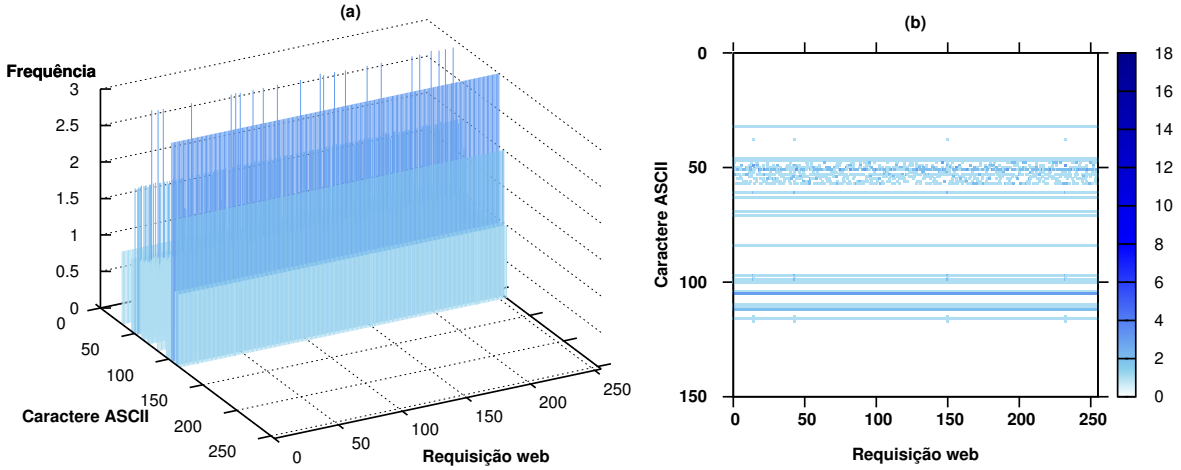

Figura 4.3: Conjunto de requisições web sem ataques.

quisições da aplicação web analisada (Figura 4.3(a)). A Figura 4.3(b) ilustra o gráfico com aproximação entre os valores dos caracteres ASCII 0 e 150 para uma melhor visualização, pois os caracteres com valores superiores não possuem ocorrências na aplicação. A principal causa para afetar a similaridade entre as frequências dos caracteres é a ocorrência de um ataque web.

A injeção de um ataque neste mesmo conjunto de requisições provoca uma alteração na similaridade da intensidade na frequência dos caracteres, identificada como uma perturbação em relação às demais requisições. Esta situação pode ser visualizada na Figura 4.4, com a inserção do ataque *Path Traversal*.

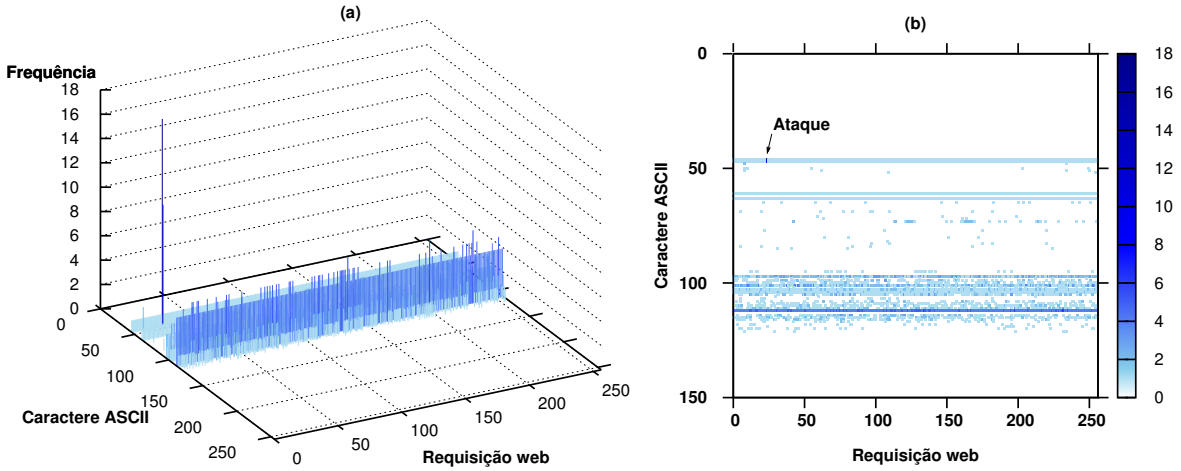

Figura 4.4: Conjunto de requisições web contendo 1 ataque *Path Traversal*.

Pode-se observar claramente a perturbação na frequência dos caracteres 46 (“.”) e 47 (“/”) provocados pela injeção do ataque web, conforme evidencia a Figura 4.4(b) através do rótulo “Ataque”. A quantidade de ocorrência dos caracteres na requisição com ataque é superior em

relação às outras requisições do conjunto. Desta forma, a anomalia provocada pelo ataque web inserido se destaca entre as frequências dos caracteres das demais requisições.

A similaridade da frequência das requisições para uma determinada aplicação web possibilita distinguir a ocorrência de um ataque dentro de um conjunto de requisições web. Levando em consideração a detecção baseada em anomalia, inicialmente tem-se a identificação da característica da aplicação a ser analisada. Isto é, considera-se a característica para a aplicação web um conjunto de caracteres com frequência usual entre as requisições. Desta forma, a ocorrência de uma frequência alta para determinado caractere, no qual este não havia ocorrido entre as demais requisições, pode ser considerada uma anomalia. Esta situação considera-se um desvio de comportamento usual.

Assim, neste trabalho explorou uma nova abordagem para detecção de ataques web através da aplicação da ferramenta de processamento de sinais, na qual se busca identificar picos provocados pela alteração na frequência dos caracteres dentro de um conjunto de dados. Neste contexto, este trabalho propõe um novo algoritmo de detecção de ataque web que elimina a fase de treinamento dos dados através da análise de um conjunto de requisições web.

4.3 Algoritmo de Detecção de Ataques Web

A transformada wavelet é uma ferramenta eficiente para detectar variações abruptas e saltos na frequência do sinal analisado (UKIL; ZIVANOVIC, 2006). Este fator potencializa a análise das requisições web e a aplicação da transformada wavelet bidimensional para detecção de ataques web. Neste contexto, é proposto um algoritmo de detecção de ataques web que utiliza a transformada wavelet bidimensional para analisar a frequência dos caracteres nas requisições web.

A Figura 4.5 apresenta o modelo de processo utilizado na proposta. Inicialmente, na etapa 1, as frequências dos caracteres de um conjunto de requisições web e o parâmetro ρ de limiarização são utilizados como entrada de dados. O parâmetro é ajustável conforme a sensibilidade da aplicação para a distinção refinada das informações consideradas anômalas durante a operação de corte, conforme visto na seção 3.2, no capítulo anterior.

Nesta etapa o algoritmo de detecção de ataques web recebe os dados para serem analisados, conforme pode ser observada na Figura 4.6. Também é inicializado o conjunto de dados A, responsável pelo armazenamento dos ataques web e a variável n recebe o tamanho absoluto da matriz de dados recebida, que será utilizada para o cálculo do valor de truncamento para cada

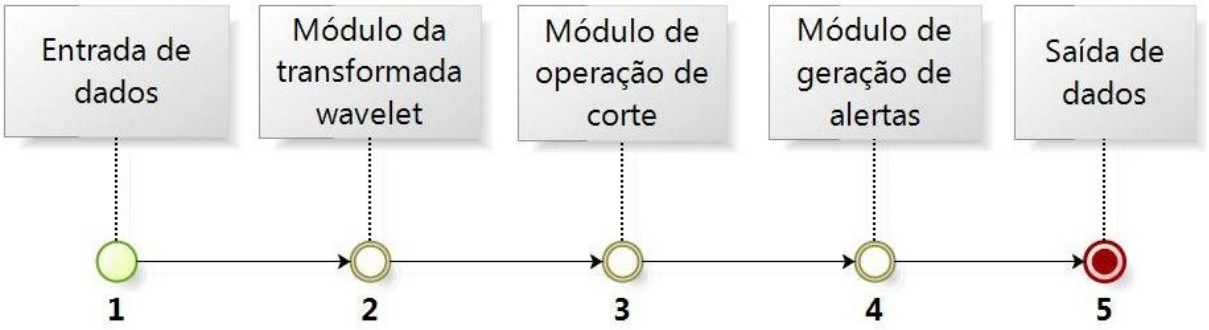

Figura 4.5: Processo do algoritmo de detecção de ataques web.

sub-banda.

Na segunda etapa, é aplicado apenas um nível de decomposição da TW2D na matriz de dados, resultando na geração de quatro sub-bandas de informações (Figura 4.7): um conjunto de coeficientes de aproximação que são as informações grosseiras do sinal analisado (sub-banda *cc*) e os demais conjuntos de coeficientes wavelets que são os detalhes da transformada (sub-banda *cd*, *dc* e *dd*). Essa etapa pode ser observada na linha 3 do algoritmo proposto. A aplicação de um nível da transformada é suficiente para obter os detalhes dos picos causados pelas perturbações dos ataques web (BILEN; HUZURBAZAR, 2002).

Como pode ser observado, as quatro sub-bandas (*cc*, *cd*, *dc* e *dd*) com a intensidade dos coeficientes. A localização dos dados após a transformada não está relacionada com as posições originais dos dados de entrada, mas podem ser re-calculados. Para isso, a transformada wavelet inversa deveria ser aplicada para recuperar as informações originais.

Nas linhas 4 a 6 é calculado o desvio padrão para cada sub-banda de detalhes. Para isso, a média dos coeficientes (d_i) das sub-bandas é utilizada no cálculo conforme a definição μ.

A Figura 4.8(a) representa um conjunto de coeficientes da TW2D contendo um ataque *Path Traversal* e na Figura 4.8(b) mostra as intensidades dos coeficientes através da representação bidimensional. É possível verificar o impacto causado nos coeficientes wavelet pela ocorrência do ataque no conjunto de dados. Além disso, a magnitude dos coeficientes wavelet com ataque é superior aos demais. Os círculos salientam a posição do ataque em cada sub-banda de detalhes.

Depois da aplicação da transformada wavelet bidimensional, a operação de corte é processada com o intuito de distinguir as informações significativas na análise das variações na frequência dos caracteres (etapa 3). Essa operação é um ponto importante deste trabalho, pois possibilita detectar os ataques web através da identificação de informações anômalas. Como podem ser visualizado nas linhas 4-6, os valores de *threshold* são definidos para cada sub-banda de

Entrada: M : Matriz de dados
ρ : Fator constante
Saída : A : Conjunto das posições com ataques

$A \leftarrow \emptyset$
$n \leftarrow |M|$
$(cc, dc, cd, dd) \leftarrow TW2D[M]$ [um nível]
$\sigma_{cd} = \sqrt{\frac{1}{n/2}\sum_{i=1}^{n/4}(d_i - \mu)^2}$
$\sigma_{dc} = \sqrt{\frac{1}{n/2}\sum_{i=1}^{n/4}(d_i - \mu)^2}$
$\sigma_{dd} = \sqrt{\frac{1}{n/2}\sum_{i=1}^{n/4}(d_i - \mu)^2}$
$\lambda_{cd} \leftarrow \rho \cdot \sigma_{cd}\sqrt{2\log(\frac{n}{2})}$
$\lambda_{dc} \leftarrow \rho \cdot \sigma_{dc}\sqrt{2\log(\frac{n}{2})}$
$\lambda_{dd} \leftarrow \rho \cdot \sigma_{dd}\sqrt{2\log(\frac{n}{2})}$
for $i \leftarrow 1$ **to** $\frac{n}{2}$ **do**
  for $j \leftarrow 1$ **to** $\frac{n}{2}$ **do**
    if $(cd_{i,j} >= \lambda_{cd}$ *AND* $dc_{i,j} >= \lambda_{dc})$ *OR*
    $(cd_{i,j} >= \lambda_{cd}$ *AND* $dd_{i,j} >= \lambda_{dd})$ *OR*
    $(dc_{i,j} >= \lambda_{dc}$ *AND* $ddi, j >= \lambda_{dd})$ **then**
      $A \longleftarrow A + (i, j)$
    end
  end
end
return A

Figura 4.6: Algoritmo de detecção de ataques web.

detalhes correspondente (λ_{cd}, λ_{dc} e λ_{dd}) e utilizados como limiarizações dos coeficientes wavelets, onde são selecionados os dados considerados anômalos na operação de corte. O parâmetro ρ também é aplicado pelo algoritmo para proporcionar uma melhor distinção dos coeficientes wavelets significantes. Além disso, o tamanho de cada conjunto de detalhes é definido através da divisão do tamanho absoluto referente a matriz de dados, representado pelo $\frac{n}{2}$, na Figura 4.6. Desta forma, o tamanho dos conjuntos está relacionado com a metade do tamanho da matriz de dados, seja pela quantidade de caracteres utilizados ou pela número de requisições web analisadas.

Após a determinação dos valores de truncamento para a etapa de operação de corte, o módulo de geração de alarmes (etapa 4) verifica a ocorrência de perturbações em mais de duas subbandas de detalhes. Essa verificação é mais uma vantagem que a propriedade da ferramenta da TW2D proporciona para a detecção de ataques web. Ou seja, na decomposição da transformada são gerados 3 sub-bandas de detalhes com informações significantes que permitem identificar

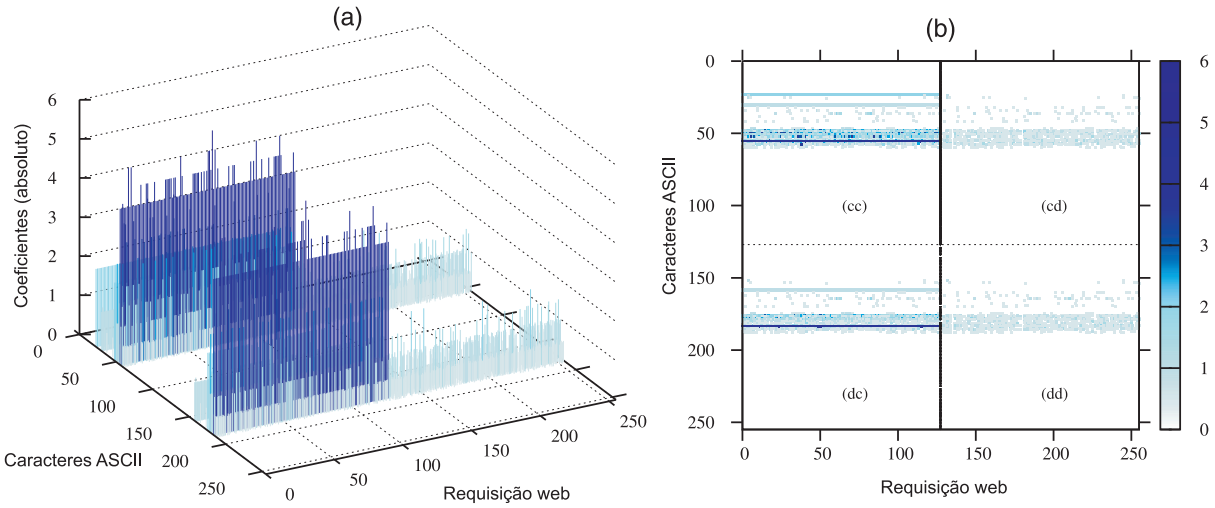

Figura 4.7: Aplicação da transformada wavelet bidimensional no conjunto de 256 amostras sem ataques.

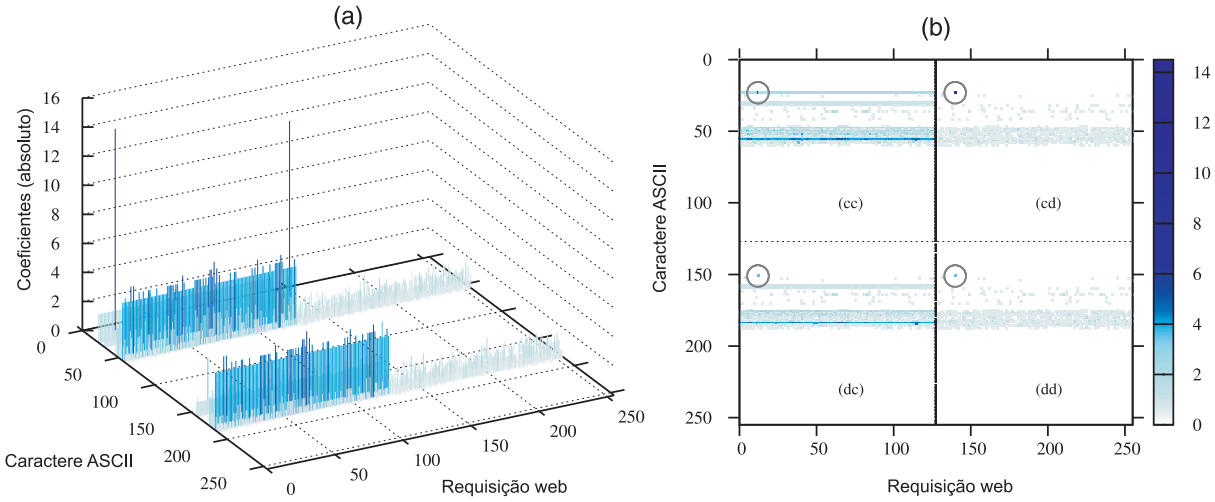

Figura 4.8: Aplicação da transformada wavelet bidimensional no conjunto de 256 amostras com um ataque.

anomalias. No entanto, a identificação de uma perturbação somente em uma sub-banda de detalhes pode ser um falso positivo, pois pode se tratar de um comportamento usual dos dados analisados. Entretanto, se essa perturbação (mesma localização) ocorre em pelos menos duas sub-bandas o módulo de geração de alarmes identifica como uma anomalia dentre um conjunto de dados. Esta situação caracteriza a vantagem da propriedade da TW2D com relação a TW1D, pois a detecção dos ataques web é realizado com base em múltiplas fontes de detalhes.

Portanto, um alerta de ataque web é gerado exclusivamente se pelos menos duas sub-bandas possuem informações significativas. No algoritmo proposto é possível identificar essa verificação, na qual a linha 9 apresenta a validação se uma determinada posição entre duas sub-bandas possuem valores superiores ao valor de truncamento para o respectivo conjunto de detalhes.

Então, se a restrição for atendida, a posição (i, j) é armazenada no conjunto de dados A (linha 10). O relatório de ataques detectados é gerado com base no posicionamento das anomalias no conjunto de dados analisado. Por fim, a análise é concluída e a saída de dados da abordagem contém informações sobre a ocorrência dos ataques web.

4.4 Trabalhos Relacionados

No contexto da análise de frequência de caracteres para a detecção de ataques web, alguns trabalhos são relacionados com a proposta (KIANI; CLARK; MOHAY, 2008) (KRUEGEL; VIGNA, 2003) (SRIRAGHAVAN; LUCCHESE, 2008). O desafio da detecção de intrusão é distinguir o comportamento anômalo das requisições web de uma aplicação web, devido a perturbação causada nos caracteres (ROBERTSON et al., 2006).

Neste sentido, esses trabalhos exploraram a abordagem baseada em anomalias através de técnicas de máquina de aprendizagem, isto é, utilizando uma fase de treinamento para definição do comportamento usual da aplicação web (*normal profile*) e outra fase para a detecção de ataques. Na fase de treinamento as frequências dos caracteres são contabilizadas. O trabalho de Kruegel e Vigna (2003) agrupou as frequências relativas dos caracteres e adicionou em agrupamentos ordenados em ordem decrescente quanto ao valor da frequência, tal como: {[0], [1-3], [4-6], [7-11], [12-15], [16-255]}. A frequência relativa também foi utilizada no trabalho de Sriraghavan e Lucchese(2008), que utilizou um critério de agrupamento separando cada grupo conforme os tipos: caracteres alfabéticos, numéricos e especiais. Já o trabalho de Kiani et al. (2008) explorou a contabilização cumulativa e adicionou em agrupamento seguindo a correção de Yate's (MAMAHLODI, 2006). Isto é, cada grupo contém a frequência máxima igual a 5. Porém, a quantidade de agrupamentos depende dos dados analisados. Da mesma maneira que o trabalho de Kiani (2008), o trabalho desta dissertação explorou a frequência cumulativa dos caracteres, entretanto, não foi utilizada uma fase de treinamento. Deste modo, minimizou o custo para obter um conjunto de dados sem ataques e com ataques rotulados. Deste modo, a definição de um conjunto de dados para determinar o *normal profile* é descartada, pois é considerado um desafio para a detecção de intrusão baseada em anomalias (JAMDAGNI et al., 2010).

A abordagem baseada em aprendizagem possui a fase de detecção para comparar os dados obtidos na fase de treinamento com os dados observados. Assim, o teste de Pearson χ^2 tem sido utilizadas para a comparação da similaridade entre as distribuições de frequências dos caracteres (KRUEGEL; VIGNA, 2003) (KIANI; CLARK; MOHAY, 2008) (SRIRAGHAVAN;

LUCCHESE, 2008). Ao contrário dos trabalhos correlatos, este trabalho buscou propor uma alternativa para a detecção através da aplicação da transformada wavelet, com o objetivo de identificar saltos e picos nas frequências dos caracteres.

Aliada ao processo de detecção, uma importante etapa utilizada na detecção de ataques web é a operação de corte. Portanto, a definição do valor de truncamento é relevante na detecção de intrusão baseada em anomalias, pois é usado para classificar os valores anômalos. Além disso, a definição do valor de truncamento pode minimizar as taxas de falsos positivos, implicando na confiabilidade do sistema. Alguns trabalhos (KIANI; CLARK; MOHAY, 2008) (KRUEGEL; VIGNA, 2003) (SRIRAGHAVAN; LUCCHESE, 2008) utilizam o valor de truncamento com 10% somado ao valor máximo. Porém, alguns trabalhos definem somente uma vez o valor de *threshold* para todo o processo (KRUEGEL; VIGNA, 2003). Isso pode ser um problema porque o tráfego possui variações influenciando a análise. Em contraste com alguns trabalhos que determinam o valor de truncamento global, neste trabalho foi considerado a definição adaptativa sendo determinado localmente um novo valor para cada janela de processamento.

4.5 Considerações Parciais

Este capítulo apresentou as características dos ataques web demonstrando a alteração no comportamento quando as requisições são analisadas em conjunto. Neste estudo percebeu-se que os ataques são caracterizados pela alteração na quantidade de caracteres usuais conforme a aplicação web. Portanto, a aplicação de uma ferramenta com propriedades que potencializam a análise através da identificação das variações na frequência dos caracteres permite a detecção de ataques dentro de um conjunto de requisições web.

Diante da viabilidade de detecção de ataques com base nas variabilidades dos caracteres entre requisições, a aplicação da ferramenta de processamento de sinais TW2D tem potencial para explorar a detecção de ataques web através da análise da distribuição de frequências dos caracteres. Com a TW2D pode-se identificar os picos e variações entre um conjunto de dados, bem como analisar correlação entre múltiplos conjuntos de coeficientes de detalhes, o que é considerado necessário para a detecção de ataque web. Deste modo, foi proposto um novo algoritmo para detecção de ataques web com base na variabilidade entre os caracteres das requisições web.

O algoritmo proposto apresentou uma nova abordagem de detecção de ataques web utilizando a aplicação da TW2D e a operação de corte para selecionar informações significantes. O algoritmo explorou a propriedade da análise em múltiplas direções da TW2D onde são gera-

dos três conjuntos de detalhes. Com base nos coeficientes de detalhes, um alerta de anomalia somente é gerado quando a perturbação ocorre em pelo menos dois conjuntos de detalhes.

5 EXPERIMENTOS

Neste capítulo são apresentados os experimentos realizados para validar a proposta apresentada no Capítulo 4 para detecção de ataques web, pois a precisão na detecção está relacionada com a capacidade de distinguir variações legítimas e evidências de comportamento anômalo na frequência dos caracteres das requisições web. Além disso, serão realizados testes de desempenho para verificar a possibilidade de aplicação da abordagem proposta para detecção de ataques web em tempo real, isto é, em tempo de execução com relação à chegada das requisições ao servidor web.

Na Seção 5.1 são apresentadas as bases de dados utilizadas neste trabalho para validação da abordagem proposta. A Seção 5.2 são realizados experimentos para validar a nova abordagem de detecção. Na Seção 5.3 são realizados testes de desempenho para confirmar a aplicação da abordagem de detecção de ataques web em tempo real.

5.1 Bases de Dados

Para a validação da abordagem proposta foram utilizadas duas base de dados, contendo requisições web de servidores de aplicações web. A primeira base de dados foi coletada a partir do arquivo de logs do servidor web localizado na Universidade Nacional de Assunção (UNA), Paraguai. A segunda base de dados utilizada foi coletada através de uma máquina destinada para captura de tráfego da rede de computadores (*probe network*), localizada no roteador de borda da Universidade Federal de Santa Maria (UFSM), Brasil.

Com o intuito de validar a abordagem proposta, foram removidos todos os ataques web contidos nas bases de dados utilizadas nos experimentos. Para isso, foi realizada uma verificação manual e utilizado um IDS baseado em assinaturas para detectar os ataques existentes, denominado *Snort* (SNORT, 2011).

A injeção dos ataques web foi realizada através da estratégia de definição da localização buscando estimar a eficácia da proposta, registrando o número de ataques detectados, assim como o número de falsos positivos. Deste modo, as posições onde os ataques foram inseridos são gravadas para uma verificação precisa da detecção. Em ambas as bases de dados, amostras de ataques mais comuns foram inseridas, tais como: *Cross-site Scritpting* (XSS), *SQL Injection* e *Path Traversal*. No entanto, a classe dos ataques selecionados são caracterizados pela perturbação no caracteres das requisições web, gerando um comportamento anômalo para o sistema.

Os exemplos de ataques detalhados na Seção 2.2.2:

- *Cross-site Scripting* (XSS)

  <script>document.location='http://attackerhost.example/cgibin/cookiesteal.cgi?'+
  document.cookie</script>

- *SQL Injection*

  AND 5=6 UNION SELECT concat(0x5E252421,count(8),0x2A5B7D2F),2,3,4,5,6,7
  FROM (SELECT 'schema_name','default_character_set_name' FROM
  'information_schema'.'schemata') t - -

- *Path Traversal*

  ../../../../../../../../../etc/passwd

Na Subseção 5.1.1 são apresentadas informações detalhadas da base de dados UNA e na Subseção 5.1.2, da base de dados UFSM.

5.1.1 Base de Dados UNA

A base de dados UNA é composta de informações do tráfego web coletadas durante três meses, Janeiro a Março de 2009, do arquivo de logs do servidor web, no qual somente as requisições enviadas pelos usuários foram armazenadas. Portanto, o período de coleta totalizou 71 dias de tráfego web. O número total de requisições web contidas na base de dados é de 59.248.

A aplicação web utilizada nos experimentos foi desenvolvida pela Faculdade Politécnica da Universidade Nacional de Assunção (FPUNA), e consiste no portal acadêmico[1] da instituição, onde diversos recursos são disponibilizados aos alunos e professores.

A Figura 5.1 apresenta o gráfico das frequências de um conjunto de 256 requisições web da base de dados UNA, sendo analisados 256 caracteres presentes nas requisições. Além disso, esse conjunto de requisições não possui ataques web.

Na Figura 5.1(a) observa-se que o valor máximo para a frequência dos caracteres é igual a 3. Além disso, a similaridade entre as requisições web é uma característica para uma determinada aplicação web. Ainda, apenas uma faixa de caracteres possuem valores diferentes de zero,

[1]Disponível em: `http://www.pol.una.uy`

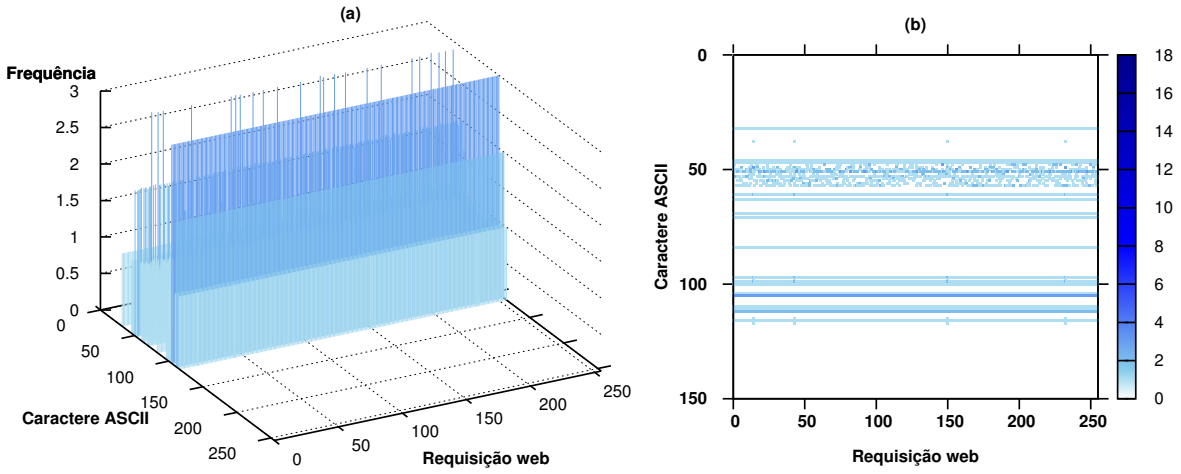

Figura 5.1: Padrão comportamental de um conjunto de requisições da base de dados UNA com 256 amostras sem ataques web.

com relação a frequência dos caracteres nas requisições web, por isto, a Figura 5.1(b) ampliou somente a faixa entre os caracteres ASCII entre 0 e 150.

5.1.2 Base de Dados UFSM

A base de dados UFSM foi coletada através de um servidor destinado a análise do tráfego de redes de computadores (denominado *sniffer*) que captura todo o fluxo de dados que trafegam pela rede da Universidade Federal de Santa Maria. Desta forma, através de uma interface de rede do *sniffer* é coletado o tráfego completo e foram filtrados somente pacotes direcionados a web, descartando os demais. A Figura 5.2 apresenta a arquitetura da rede de computadores da UFSM.

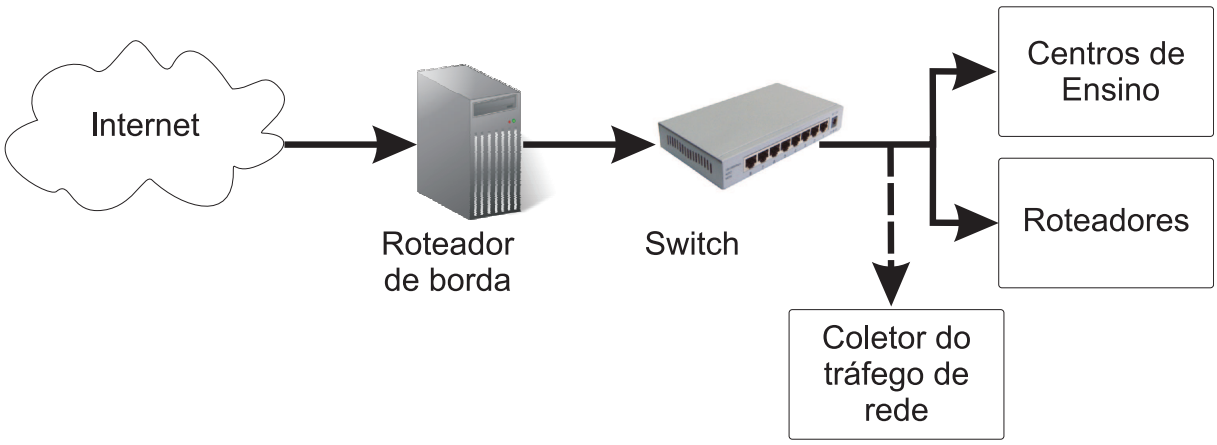

Figura 5.2: Arquitetura da rede de computadores da UFSM.

O *sniffer* utilizado na captura dos dados foi desenvolvido utilizando a linguagem C, com uso

da biblioteca *lipcap 4*[2], destinada a captura de pacotes do tráfego de redes de computadores. A base de dados UFSM é composta pelo tráfego completo de pacotes de rede, também chamado de *payload*. Além disso, foi filtrada uma faixa de endereços IP de servidores web que somente informações do tráfego HTTP foram armazenadas, as quais são significantes para este trabalho.

O período de coleta da base de dados UFSM foi durante os meses de Maio a Junho de 2011. Durante a coleta, informações sobre diversas aplicações web foram capturadas e armazenadas. Posteriormente, foi necessário o tratamento dos dados, selecionando uma aplicação web e filtrando somente suas requisições web para análise. A aplicação web selecionada desta base de dados foi sobre as notícias do portal[3] da Universidade Federal de Santa Maria. O número total de requisições web contidas na base de dados UFSM é de 80.745. A Figura 5.3 mostra o gráfico das frequências de um conjunto de 256 requisições web dos dados da UFSM.

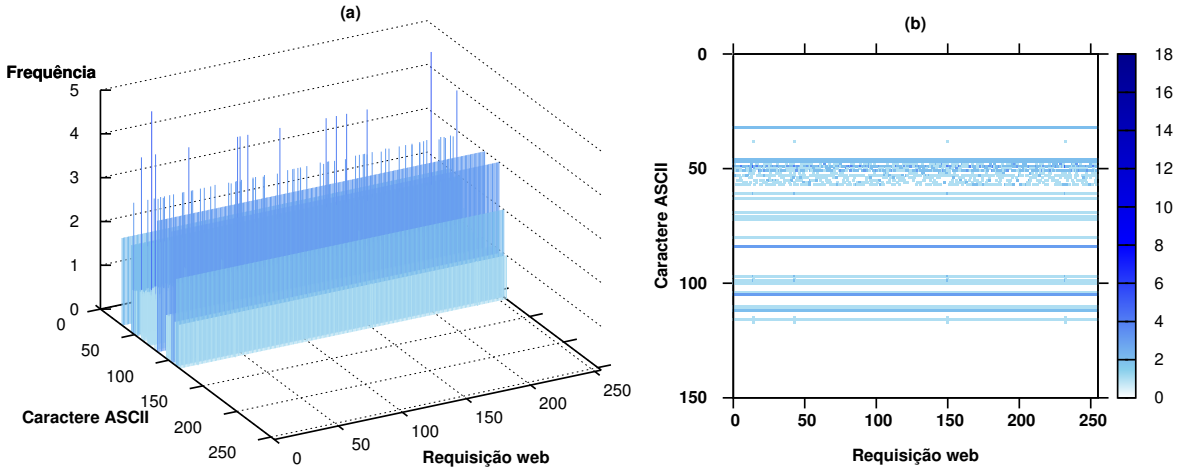

Figura 5.3: Padrão comportamental de um conjunto de requisições da base de dados UFSM com 256 amostras sem ataques web.

5.2 Análise de Detecção

A validação da abordagem proposta é baseada na detecção dos ataques web, inseridos nas bases de dados. A estratégia usada para injeção de ataques possibilitou a medição do potencial da abordagem proposta explorando a análise da frequência dos caracteres entre as requisições web. Levando em consideração que a proposta do trabalho utiliza a abordagem baseada em anomalias, é inviável a validação utilizando ataques desconhecidos. Neste sentido, algoritmos de detecção baseados em anomalias serão testados com ataques conhecidos, embora o algoritmo

[2]Disponível em: `http://www.tcpdump.org/`
[3]Disponível em http://www.ufsm.br

não possua conhecimento prévio dos ataques inseridos (MAHONEY; CHAN, 2003).

Por esse motivo, a detecção baseada em anomalias apenas leva em consideração um comportamento considerado usual e qualquer desvio dessa característica é considerado uma anomalia. Este também é o motivo dessa abordagem gerar alta taxa de falsos positivos, pois uma anomalia nem sempre é um ataque web. Por outro lado, neste trabalho consideramos que um ataque web sempre irá gerar uma perturbação identificada como anomalia.

A validação da proposta deste trabalho é baseada na aplicação da transformada wavelet bidimensional de *Haar* e no *Hard threshold* para a operação de corte. Além disso, a proposta explora o método de detecção não-supervisionado, que não utiliza uma fase de treinamento para treinar os dados (CHANDOLA; BANERJEE; KUMAR, 2009). Desta forma, o objetivo dos experimentos é validar a proposta com base no comportamento da aplicação do algoritmo na presença de ataques web.

Estudo de caso 1

Neste experimento a transformada wavelet bidimensional e a estratégia *Hard threshold* foram aplicadas na base de dados UFSM. O tamanho utilizado para o conjunto de dados na janela de processamento da transformada foi de 256 amostras. Neste teste foi realizada a variação no parâmetro ρ, o qual implica na definição do valor de truncamento. Esse valor é definido através do cálculo do Threshold universal, mas devido à necessidade de um ajuste fino conforme a aplicação a ser analisada, foi necessário a utilização de um fator constante multiplicado ao threshold universal.

No entanto, devido a distinção dos coeficientes wavelets das sub-bandas a operação de corte calcula um valor de truncamento para cada sub-banda de detalhes *cd*, *dc* e *dd*. Esse valor é calculado conforme a variação dos detalhes, sendo considerado adaptativo. Neste sentido, a Figura 5.4 apresenta a variação no parâmetro ρ na base de dados UFSM com 20 ataques. Os valores utilizados para o parâmetro ρ são 1, 2, 3, 4 e 5.

Para a geração do gráfico foram utilizadas 100 janelas de processamento da transformada wavelet e para cada janela foi definido o valor máximo entre os coeficientes wavelets. Os ataque foram inseridos nos dados e em algumas janelas de processamento foram injetados mais de um ataque. As janelas com ataques são: 1, 2, 3, 5, 13, 21, 26, 29, 34, 40, 49, 65, 67, 84, 91, e 99. Na operação de corte, as variações no parâmetro ρ são multiplicadas pelo Threshold universal definindo os valores de truncamento. O valor de truncamento com o parâmetro ρ igual a 1, se utilizado, resulta em alta taxa de falsos positivos, conforme ilustrado no gráfico. Essa situação

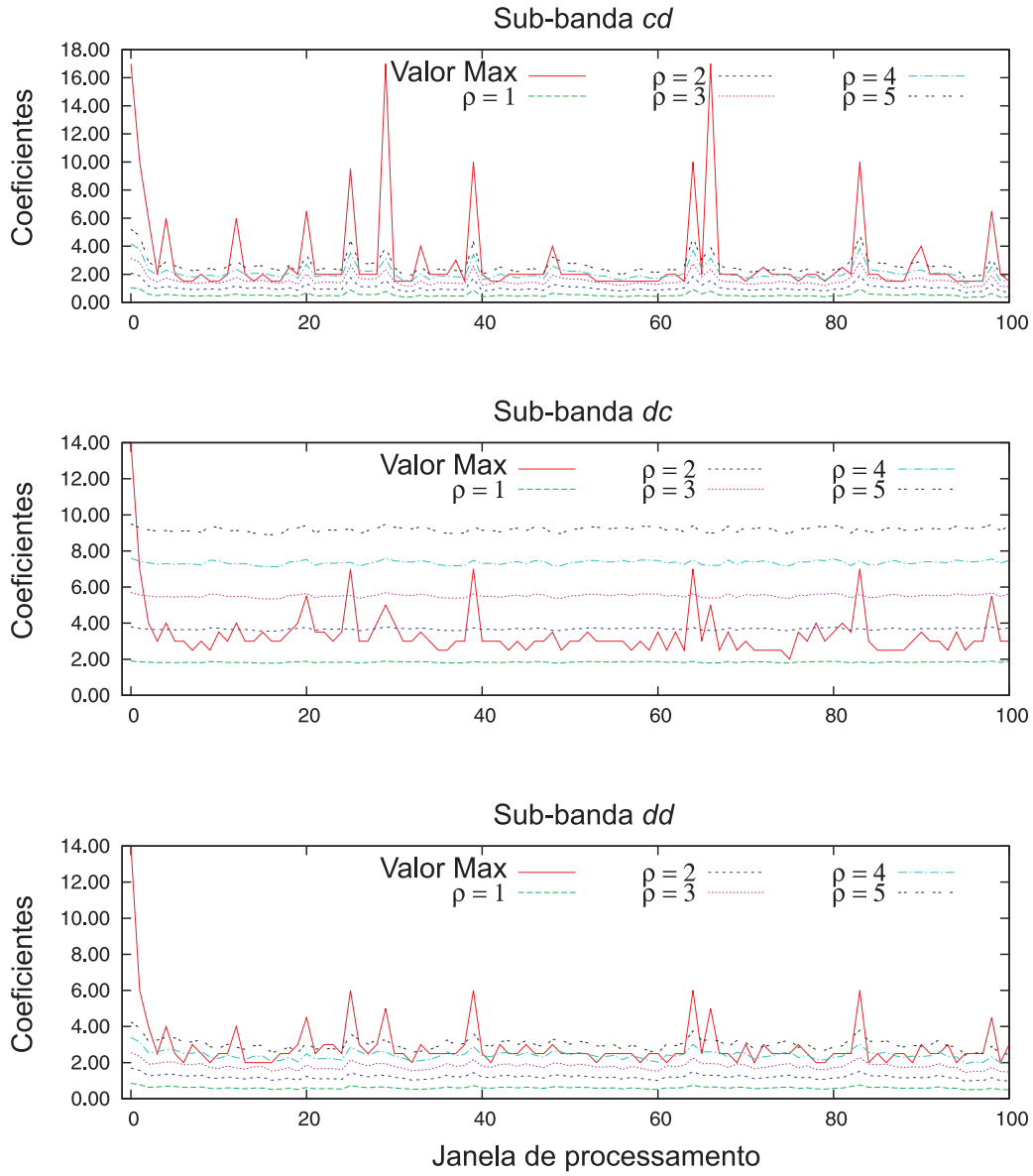

Figura 5.4: Variação do parâmetro ρ na operação de corte.

ocorre devido o baixo valor obtido pelo threshold universal, o qual é destinado para filtragem de ruído em processamento de sinais. Portanto, o ajuste fino nesse valor é obtido através do acréscimo do parâmetro ρ. Nota-se na Figura 5.4 (sub-banda *dd*) que no valor do parâmetro ρ igual a 5, somente os valores dos coeficientes mais significantes são cortados.

Além da variação do parâmetro ρ, também foram executados testes com diferentes quantidades de ataques web. Para comprovar a eficácia da proposta, buscou inserir ataques em diferentes janelas de processamento com o objetivo de explorar a detecção dos ataques para os diferentes valores no parâmetro ρ. Por fim, o algoritmo apresenta a quantidade de falso positivo (FP) e verdadeiro positivo (VP). A Tabela 5.1 apresenta os resultados obtidos com a variação do parâmetro ρ com a base de dados UFSM.

Tabela 5.1: Resultados da análise conforme o parâmetro ρ na base de dados UFSM.

Ataques	# Ataques detectados com diferentes $\rho\lambda$																					
	1λ		1.5λ		2λ		2.5λ		3λ		3.5λ		4λ		4.5λ		5λ		5.5λ		6λ	
	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP
0	6592	0	4098	0	693	0	215	0	53	0	13	0	3	0	2	0	0	0	0	0	0	0
1	6730	1	4019	1	698	1	227	1	47	1	11	1	2	1	0	1	0	1	0	1	0	0
2	6730	2	3981	2	695	2	229	2	49	2	13	2	4	2	0	2	0	2	0	2	0	2
3	6729	3	3981	3	695	3	229	3	49	3	13	3	4	3	0	3	0	3	0	3	0	2
4	6704	4	3939	4	688	4	229	4	49	4	13	4	4	4	0	4	0	4	0	4	0	3
5	6626	5	3914	5	676	5	229	5	49	5	13	5	4	5	0	5	0	5	0	5	0	5
10	6575	10	3804	10	659	10	217	10	44	10	10	10	2	10	0	10	0	10	0	10	0	8
20	6358	20	3597	20	594	20	221	20	44	20	8	20	1	20	0	20	0	20	0	19	0	18

Neste caso, diante da aplicação web selecionada na base de dados UFSM, foi identificada a alta taxa de falso positivo quando considerado o valor de truncamento original ao threshold universal (1λ). Neste caso, o valor de truncamento definido na operação de corte é considerado baixo para a análise, portanto, são identificadas diversas anomalias nos coeficientes wavelets que não possuem ataques. Outra característica importante está relacionada ao acréscimo no valor de truncamento, que conforme aumento o número de falsos positivos diminui. Além disso, percebe-se um degrau na diminuição entre os valores de truncamento 2λ e 3λ, identificando a similaridade das frequências dos caracteres de uma aplicação web específica. Por fim, o valor nulo para a taxa falso positivo foi obtida quando considerado o parâmetro com valor igual a 5λ.

Por outro lado, as taxas de verdadeiro positivo atingiram 100% de detecção até o valor 5λ. Isso se dá devido aos valores das frequência serem superiores ao valor de truncamento, assim, identificando todos os ataques.

A definição do parâmetro ρ na operação de corte está relacionada com a aplicação web, permitindo que o administrador possa ajustar o valor conforme a aplicação. Neste sentido, um segundo experimento foi realizado utilizando a mesma metodologia para a aplicação web da base de dados UNA. A Tabela 5.2 apresenta os resultados obtidos.

Tabela 5.2: Resultados da análise conforme o parâmetro ρ na base de dados UNA.

Ataques	# Ataques detectados com diferentes $\rho\lambda$																					
	1λ		1.5λ		2λ		2.5λ		3λ		3.5λ		4λ		4.5λ		5λ		5.5λ		6λ	
	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP	FP	VP
0	15971	0	4452	0	1939	0	14	0	7	0	2	0	1	0	0	0	0	0	0	0	0	0
1	15970	1	4451	1	1934	1	14	1	7	1	2	1	1	1	0	1	0	1	0	0	0	0
2	15911	2	4447	2	1934	2	14	2	7	2	2	2	1	2	0	2	0	2	0	2	0	2
3	15911	3	4447	3	1934	3	14	3	7	3	2	3	1	3	0	3	0	3	0	3	0	3
4	15909	4	4447	4	1934	4	14	4	7	4	2	4	1	4	0	4	0	4	0	4	0	2
5	15887	5	4382	5	1874	5	28	5	3	5	1	5	1	5	1	5	0	5	0	3	0	3
10	15774	10	4298	10	1930	10	15	10	7	10	2	10	1	10	0	10	0	10	0	8	0	8
20	15419	20	4087	20	1876	20	16	20	3	20	1	20	1	20	0	20	0	20	0	18	0	18

O comportamento deste experimento é similar ao realizado com a base de dados UFSM. O parâmetro ρ que demonstrou melhores resultados foi igual a 5λ.

Estudo de caso 2

Para este experimento foi aplicada a transforma wavelet de *Haar* bidimensional na base de dados UFSM, com variação na quantidade de ataques dentro da janela de processamento com 256 amostras. O parâmetro ρ adotado na definição do valor de truncamento (operação de corte) neste experimento foi 5λ, visto que demonstrou melhores resultados no estudo de caso 1.

A proposta considera a hipótese que mais de uma ocorrência de ataques dentro da mesma janela de processamento pode influenciar na detecção dos ataques web. Considerando isso, este experimento foi executado com intuito de verificar se o aumento na quantidade de ataques inseridos na mesmas janela provoca alteração na taxa de detecção dos ataques web. Portanto, foram injetados até 10 ataques na janela de processamento com 256 amostras de requisições web. Essa variação possibilita verificar a eficiência da proposta diante das ocorrências dos ataques dentro da mesma janela de processamento.

A injeção dos ataques foi realizada de maneira arbitrária dentro da base de dados UFSM. Foram obtidas as taxas de verdadeiro positivo durante o teste e ao final do cálculo de detecção, resultando na taxa de detecção para cada quantidade de ataques dentro da janela de processamento. A Tabela 5.3 apresenta os dados obtidos neste estudo de caso.

Tabela 5.3: Resultados da análise conforme a quantidade de ataques no conjunto de 256 requisições web.

Ataques injetados	Verdadeiro Positivo	Taxa de Detecção
1	1	100%
2	2	100%
3	3	100%
4	4	100%
5	5	100%
6	5	83.3%
7	3	42.8%
8	2	25%
9	2	22.2%
10	2	20%

Os resultados demonstraram que a ocorrência de múltiplos ataques dentro de uma mesma janela de processamento causa alteração na taxa de detecção. Com base nos experimentos, a proposta demonstrou que dependendo da variabilidade dos caracteres contidos nos ataques da mesma janela de processamento a taxa de detecção variou. Portanto, não é possível afirmar que o aumento no número de ataques minimiza a taxa de detecção da proposta. Embora, dependendo dos ataques ocorridos no conjunto de dados isso possa acontecer. Com isso, novos experimentos

devem ser realizados para verificar a ineficiência da proposta quando ocorre aumento no número de ataques dentro da janela de processamento, o qual não é o objetivo deste trabalho. Neste trabalho assume-se o métodos de detecção de anomalias não-supervisionado que considera que as informações normais ocorrem com mais frequência que as informações anômalas, sendo assim diversos ataques não devem ocorrer dentro de uma mesma janela.

5.3 Análise de Desempenho

A análise de desempenho diante do tempo de execução tem como objetivo demonstrar a viabilidade de utilização da abordagem proposta. Neste sentido, foram realizados testes com o servidor web com a coleta dos tempos de execução. A base de dados UNA foi utilizada nos testes com um conjunto de 59.248 requisições web, onde foram usadas as configurações com melhor taxa de detecção na Seção 5.2. Os experimentos foram realizados no servidor:

1. Intel Xeon Quad-Core 2.4GHz
   Ubuntu Server Linux
   Kernel 2.6 - gcc 4.5.2

Considerando que o desempenho pode ser ruim dependendo do tamanho da janela de processamento, foi realizado alguns experimentos com os tempos de processamento para 64, 128, 256 e 512 amostras. Embora o tempo médio de processamento para cada janela de requisições seja importante, também foi considerado o tempo final de execução e a quantidade de janelas processadas. A Tabela 5.4 são apresentados os resultados obtidos.

Tabela 5.4: Resultados da análise do desempenho com variação no tamanho da janela.

Tamanho da Janela	Tempo médio de execução por janela (ms)	Tempo total de execução (ms)	Quantidade de janelas processadas
64	0.3298814	572.9	924
128	0.6657262	572.2	462
256	1.3258301	584.6	231
512	2.6853687	587.4	116

Conforme os resultados obtidos na análise de desempenho, o tempo total de execução é linear com relação ao tamanho da janela de processamento. Adicionalmente, conforme aumenta

a janela, minimiza o número de janelas processadas pela abordagem. Entretanto, observou-se através deste experimento que somente a variação no tamanho da janela não interfere no desempenho da análise.

De maneira geral, somente verificar o tempo de processamento da proposta é trivial, pois não se tem um valor considerado ideal para a execução de algoritmos de detecção de intrusão. Portanto, foi necessário comparar o tempo final de processamento utilizando a mesma base de dados, de modo que os tempos pudessem ser comparados. Embora os algoritmos de detecção de ataques web não realizam a detecção por janela de processamento, a comparação foi considerando o tempo final de execução. Os trabalhos utilizam técnicas estatísticas para verificar a similaridade entre as frequências dos caracteres, aplicando o teste χ^2(KRUEGEL; VIGNA, 2003) e o teste de Mahalanobis (WANG; PAREKH; STOLFO, 2006). Outro algoritmo de detecção explora a técnica *n-grams* (INGHAM; INOUE, 2007).

Como não é possível obter os algoritmos de outros trabalhos foi necessário o desenvolvimento dos algoritmos de detecção de ataques web com este trabalho. Os algoritmos foram desenvolvidos na linguagem C, na qual se desenvolveu o algoritmo proposto. A Tabela 5.5 apresenta o tempo final de processamento entre os algoritmos de detecção de ataques web.

Tabela 5.5: Comparação da análise do desempenho com algoritmos de detecção de ataques web.

Algoritmo	Tempo total de execução (ms)
Algoritmo proposto	572.9
(KRUEGEL; VIGNA, 2003) (χ^2)	1350.0
(INGHAM; INOUE, 2007) (*n-gram*)	564.0
(WANG; PAREKH; STOLFO, 2006) (Mahalanobis)	321.0

Embora a execução do algoritmo proposto seja através de janelas de processamento dos dados, a comparação verificou o tempo de processamento do algoritmo proposto com relação ao tempo de execução final. Por um lado, os tempos das técnicas *n-gram* e *Mahalanobis* obtidos foram similares e a técnica de χ^2 o tempo foi superior devido a necessidade de uma fase de treinamento, aumentando o tempo de execução final.

5.4 Considerações Parciais

Este capítulo apresentou os experimentos para avaliar a eficiência de detecção de ataque web através do algoritmo proposto neste trabalho. Foram utilizadas as bases de dados da UFSM e

UNA contendo requisições web com aplicações selecionadas.

Com a variação do valor de truncamento na operação de corte, se percebeu que aplicação web existe a necessidade de um refinamento através do parâmetro ρ. Para os experimentos realizados, o valor igual a 5 para esse parâmetro apresentou taxa de 100% de detecção para as aplicações selecionadas entre as bases de dados UNA e UFSM. Embora o valor tenha sido o mesmo para as duas bases de dados, os experimentos demonstraram que conforme a variabilidade das frequências dos caracteres existe a necessidade de um refinamento no parâmetro. Uma vez que uma aplicação selecionada possui requisições com maiores variações entre os caracteres, o valor do parâmetro ρ deve ser elevado para filtrar somente os coeficientes anômalos para a detecção de ataques web.

Outros experimentos foram realizados com variação na quantidade de ataques dentro da mesma janela de processamento da TW2D. A análise destes experimentos demonstraram que a capacidade de detecção diminui conforme o aumento no número de ataques. Entretanto, este experimento deve ser explorado com maior abrangência em trabalhos futuros, o qual não foi o objetivo deste trabalho.

Além disto, experimentos com relação ao desempenho da aplicação da ferramenta foram realizados. Com isso, buscou-se verificar a capacidade de detecção da proposta em comparação com técnicas dos trabalhos relacionados. Mesmo que a execução do algoritmo proposto seja realizado por janela de processamento, os tempos finais de execução do conjunto de requisições foram similares.

6 CONCLUSÕES E CONSIDERAÇÕES FINAIS

A Internet tornou-se um componente essencial para a sociedade atual e inúmeras organizações efetuam suas operações diárias através de aplicações Web. No entanto, a segurança da informação na Web, assim como a disponibilidade e a confiabilidade das aplicações, deve ser assegurada. Qualquer ataque que danifique uma aplicação Web ou obtenha acesso a informações restritas podem causar sérios problemas para uma organização, e consequentemente a seus usuários.

Este trabalho propôs uma nova abordagem baseada em anomalias para detecção de ataques web. Para isso um algoritmo foi proposto para analisar a variabilidade entre as frequências dos caracteres provocada pelos ataques web. O algoritmo explora as propriedades da transformada wavelet bidimensional para a identificação de variações nos caracteres nas requisições com ataques através da correlação entre diferentes conjuntos de coeficientes de detalhes. A operação de corte explorou a seleção dos coeficientes com informações significantes para a análise de detecção.

Nos experimentos realizados para validação do algoritmo proposto, foi verificado que o refinamento do valor de *threshold* através do parâmetro ρ tem relação com a geração de falsos positivos. Portanto, para as aplicações selecionadas o valor igual a 5 obteve taxa de detecção de 100% para ambas as base de dados. Embora a hipótese adotada neste trabalho que ataques em rajada não ocorrem com frequência, percebeu-se através dos experimentos que a taxa de detecção de 100% diminui com o aumento no número de ataques na mesma janela de processamento.

Na análise de desempenho o algoritmo proposto demonstrou que o tempo de execução é similar aos trabalhos relacionados, o que caracteriza a possibilidade de detecção de ataques web em tempo real, considerando o tempo de execução da janela de processamento. Entretanto, se observou a possibilidade de aplicação da TW2D como uma ferramenta para identificar variações no conjunto de dados e correlação entre conjuntos de detalhes distintos, o qual é necessário para a identificação de ataques web.

6.1 Trabalhos Futuros

Neste trabalho foi proposto um algoritmo para detecção de ataques web através da aplicação da transformada wavelet de *Haar* bidimensional. A técnica não havia sido explorada ainda, mas outras famílias wavelets podem ser aplicadas e exploradas. Porém o grande desafio é

conseguir eficácia melhor a ponto de compensar o uso de uma transformada mais complexa. A transformada de *Haar* é a mais simples das famílias de wavelets.

De maneira similar, outras técnicas podem ser exploradas para a operação de corte, uma vez que este trabalho apenas explorou o *Hard Threshold* na operação de corte. Acredita-se que a exploração de diferentes thresholds pode resultar em variações não desprezíveis nos resultados, porém uma análise detalhada neste sentido não estava no escopo deste trabalho.

Finalmente, novos experimentos também podem ser realizados para explorar a proposta de uso da TW2D com diversos ataques na mesma janela de processamento. Resultados preliminares demonstram que um aumento no número de ataques na mesma janela de observação, o que pode ser gerado por ataques em rajadas, pode interferir na qualidade da detecção.

REFERÊNCIAS

ALVAREZ, G.; PETROVIC, S. A new taxonomy of Web attacks suitable for efficient encoding. **Computers & Security**, v.22, n.5, p.435–449, Jul. 2003.

BAYER, F. M.; KOZAKEVICIUS, A. d. J. SPC-Threshold: uma proposta de limiarização para filtragem adaptativa de sinais. **TEMA - Tendências em Matemática Aplicada e Computacional.**, v.11, n.2, p.121–132, May 2010.

BERNERS-LEE, T.; MASINTER, L.; MCCAHILL, M. **Uniform Resource Locators (URL)**. Acesso em: 13/03/2011, Disponível em `http://www.ietf.org/rfc/rfc1738.txt`.

BILEN, C.; HUZURBAZAR, S. Wavelet-based detection of outliers in time series. **Journal of Computational and Graphical Statistics**, v.11, n.2, p.311–327, Jun. 2002.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection: a survey. **ACM Comput. Surv.**, New York, NY, USA, v.41, p.15:1–15:58, Jul. 2009.

CVE-2010-1679. **Common Vulnerabilities and Exposures**. Acesso em: 12/04/2011, Disponível em `http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-1679`.

CVE-2011-2703. **Common Vulnerabilities and Exposures**. Acesso em: 10/04/2010, http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2703.

CVE. **Common Vulnerabilities and Exposures**. Acesso em: 05/11/2011, Disponível em `http://www.cve.mitre.org`.

DAUBECHIES, I. **Ten lectures on wavelets**. 1.ed. Philadelphia, PA, USASociety for Industrial and Applied Mathematics, 1992. 357p.

DEBAR, H.; DACIER, M.; WESPI, A. Towards a taxonomy of intrusion-detection systems. **Comput. Netw.**, New York, NY, USA, v.31, p.805–822, Apr. 1999.

DONOHO, D. L.; JOHNSTONE, I. M. Adapting to unknown smoothness via wavelet shrinkage. **Journal of the American Statistical Association**, v.90, n.432, p.1200–1224, Dec. 1995.

FIELDING, R.; GETTYS, J.; MOGUL, J.; FRYSTYK, H.; MASINTER, L.; LEACH, P.; BERNERS-LEE, T. **Hypertext Transfer Protocol – HTTP/1.1**. Acesso em: 13/03/2011, Disponível em `http://www.w3.org/Protocols/rfc2616/rfc2616.html`.

FOGIE, S.; GROSSMAN, J.; HANSEN, R.; RAGER, A.; PETKOV, P. D. **XSS Attacks Cross Site Scripting Exploits and Defense**. 1.ed. Syngress Publishing, 2007. 548p.

FONSECA, J.; VIEIRA, M.; MADEIRA, H. The Web Attacker Perspective - A Field Study. In: SOFTWARE RELIABILITY ENGINEERING (ISSRE), 2010. **Anais...** 2010. p.299–308.

GARCIA TEODORO, P.; DIAZ VERDEJO, J.; MACIA FERNANDEZ, G.; VAZQUEZ, E. Anomaly-based network intrusion detection: techniques, systems and challenges. **Computers & Security**, v.28, n.1-2, p.18–28, Feb. 2009.

GOURLEY, D.; TOTTY, B. **HTTP - the definitive guide**: understanding web internals. O'Reilly, 2002. 635p.

GRANE, A.; VEIGA, H. **Wavelet-based detection of outliers in volatility models**. Universidad Carlos III, Departamento de Estadística y Econometría, 2009. Statistics and Econometrics Working Papers.

HALFOND, W. G.; VIEGAS, J.; ORSO, A. A Classification of SQL-Injection Attacks and Countermeasures. In: INTERNATIONAL SYMPOSIUM ON SECURE SOFTWARE ENGINEERING, 2006. **Proceedings...** 2006.

INGHAM, K. L. **Anomaly Detection for HTTP Intrusion Detection Algorithm Comparisons and the Effect of Generalization on Accuracy**. 2007. Tese (Doutorado) — University of New Mexico.

INGHAM, K. L.; INOUE, H. Comparing anomaly detection techniques for HTTP. In: RECENT ADVANCES IN INTRUSION DETECTION, 10., 2007, Berlin, Heidelberg. **Proceedings...** Springer-Verlag, 2007. p.42–62. (RAID'07).

INGHAM, K. L.; SOMAYAJI, A.; BURGE, J.; FORREST, S. Learning DFA representations of HTTP for protecting web applications. **Comput. Netw.**, New York, NY, USA, v.51, p.1239–1255, Apr. 2007.

JAMDAGNI, A.; TAN, Z.; NANDA, P.; HE, X.; LIU, R. P. Intrusion detection using GSAD model for HTTP traffic on web services. In: INTERNATIONAL WIRELESS COMMUNICATIONS AND MOBILE COMPUTING CONFERENCE, 6., 2010, New York, NY, USA. **Proceedings...** ACM, 2010. p.1193–1197. (IWCMC '10).

KIANI, M.; CLARK, A.; MOHAY, G. Evaluation of Anomaly Based Character Distribution Models in the Detection of SQL Injection Attacks. In: AVAILABILITY, RELIABILITY AND SECURITY, 2008. ARES 08. THIRD INTERNATIONAL CONFERENCE ON, 2008. **Anais...** 2008. p.47–55.

KIEYZUN, A.; GUO, P. J.; JAYARAMAN, K.; ERNST, M. D. Automatic creation of SQL Injection and cross-site scripting attacks. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 31., 2009, Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2009. p.199–209. (ICSE '09).

KIRDA, E.; JOVANOVIC, N.; KRUEGEL, C.; VIGNA, G. Client-side cross-site scripting protection. **Computers & Security**, v.28, n.7, p.592–604, Oct. 2009.

KIZZA, J. M. **Computer Network Security**. 1.ed. New York, NY:Springer, 2005. 538p.

KOZAKEVICIUS, A. J.; RODRIGUES, C. R.; NUNES, R. C.; GUERRA FILHO, R. Adaptive ECG Filtering and QRS Detection using Orthogonal Wavelet Transform. In: INTERNATIONAL CONFERENCE ON BIOMEDICAL ENGINEERING., 2005, Innsbruck, Austria. **Anais...** 2005. p.109–114.

KRUEGEL, C.; VALEUR, F.; VIGNA, G. **Intrusion Detection and Correlation Challenges and Solutions**. 1.ed. Santa Clara, CA, USASpringer-Verlag TELOS, 2004. 118p.

KRUEGEL, C.; VIGNA, G. Anomaly detection of web-based attacks. In: ACM CONFERENCE ON COMPUTER AND COMMUNICATIONS SECURITY, 10., 2003, New York, NY, USA. **Proceedings...** ACM, 2003. p.251–261. (CCS '03).

LI, M.; LI, M. A New Approach for Detecting DDoS Attacks Based on Wavelet Analysis. **Proceedings of 2nd International Congress on Image and Signal Processing, 2009. CISP '09.**, p.1–5, Oct. 2009.

LU, W.; GHORBANI, A. A. Network anomaly detection based on wavelet analysis. **EURASIP J. Adv. Signal Process**, New York, NY, United States, p.4:1–4:16, Jan. 2009.

MAHONEY, M. V.; CHAN, P. K. An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection. In: SIXTH INTERNATIONAL SYMPOSIUM ON RECENT ADVANCES IN INTRUSION DETECTION, 2003. **Anais...** Springer-Verlag, 2003. p.220–237.

MALLAT, S. **A wavelet tour of signal processing**. 3.ed. Elsevier/Academic Press, Amsterdam, 2009. 805p. The sparse way, With contributions from Gabriel Peyré.

MAMAHLODI, M. **What is the chi-square statistic?** Connexions Web site. http://cnx.org/content/m13487/1.2/.

MARQUES, O.; BAILLARGEON, P. A Multimedia Traffic Classification Scheme for Intrusion Detection Systems. In: INFORMATION TECHNOLOGY AND APPLICATIONS, 2005. ICITA 2005. THIRD INTERNATIONAL CONFERENCE ON, 2005. **Anais...** 2005. v.2, p.496–501.

N. BORENSTEIN, B.; N. FREED, I. **MIME (Multipurpose Internet Mail Extensions) Mechanisms for Specifying and Describing the Format of Internet Message Bodies**. Acesso em: 04/01/2012, Disponível em `http://www.rfc-editor.org/rfc/rfc1341.txt`.

NEUHAUS, S.; ZIMMERMANN, T. Security Trend Analysis with CVE Topic Models. In: SOFTWARE RELIABILITY ENGINEERING (ISSRE), 2010 IEEE 21ST INTERNATIONAL SYMPOSIUM ON, 2010. **Anais...** 2010. p.111–120.

NIELSEN, O. M. **Wavelets in scientific computing**. 1998. Tese (Doutorado) — Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby. Vejleder Per Christian Hansen.

NORTHCUTT, S.; NOVAK, J. **Network Intrusion Detection**. 3.ed. New Riders Publishing, 2002. 512p.

OWASP. **The Open Web Application Security Project - Top 10 Web Application Security Risks**. Acesso em: 21/11/2011, Disponível em `https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project`.

PAXSON, V. Bro: a system for detecting network intruders in real-time. **Comput. Netw.**, New York, NY, USA, v.31, p.2435–2463, Dec. 1999.

PROVOST, F. J.; FAWCETT, T.; KOHAVI, R. The Case against Accuracy Estimation for Comparing Induction Algorithms. In: ICML '98 PROCEEDINGS OF THE FIFTEENTH INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 1998, San Francisco, CA, USA. **Anais...** Morgan Kaufmann Publishers Inc., 1998. p.445–453.

ROBERTSON, W. K. **Detecting and Preventing Attacks Against Web Applications**. 2009. 249p. Tese (Doutorado) — University of California, Santa Barbara.

ROBERTSON, W.; VIGNA, G.; KRUEGEL, C.; KEMMERER, R. Using generalization and characterization techniques in the anomaly-based detection of web attacks. In: NDSS '06 PROC. OF 17TH ISOC SYMPOSIUM ON NETWORKS AND DISTRIBUTED SYSTEMS SECURITY, 2006, San Diego, CA. **Anais...** 2006.

ROESCH, M. Snort - Lightweight Intrusion Detection for Networks. In: USENIX CONFERENCE ON SYSTEM ADMINISTRATION, 13., 1999, Berkeley, CA, USA. **Proceedings...** USENIX Association, 1999. p.229–238. (LISA '99).

SNORT. **Snort Network Intrusion Detection System web site.** Acesso em: 30/08/2011, Disponível em `http://www.snort.org`.

SONG, Y.; KEROMYTIS, A.; STOLFO, S. Spectrogram A Mixture-of-Markov-Chains Model for Anomaly Detection in Web Traffic. In: NETWORK AND DISTRIBUTED SYSTEM SECURITY SYMPOSIUM (NDSS), 2009. **Proceedings...** 2009.

SRIRAGHAVAN, R.; LUCCHESE, L. Data processing and anomaly detection in web-based applications. In: MACHINE LEARNING FOR SIGNAL PROCESSING, 2008. MLSP 2008. IEEE WORKSHOP ON, 2008. **Anais...** 2008. p.187–192.

STALLINGS, W. **Cryptography and Network Security**. 4.ed. Upper Saddle River, NJ, USAPrentice-Hall, Inc., 2005. 592p.

STOLLNITZ, E.; DEROSE, A.; SALESIN, D. Wavelets for computer graphics a primer 1. **Computer Graphics and Applications, IEEE**, v.15, n.3, p.76–84, May. 1995.

STUTTARD, D.; PINTO, M. **The web application hacker's handbook discovering and exploiting security flaws**. 1.ed. New York, NY, USAJohn Wiley & Sons, Inc., 2007. 736p.

SU, Z.; WASSERMANN, G. The essence of command injection attacks in web applications. **SIGPLAN Not.**, New York, NY, USA, v.41, p.372–382, Jan. 2006.

SYMANTEC. **Symantec Internet Security Threat Report Trends for 2010**. 2011.

TANENBAUM, A. S. **Redes de Computadores**. 4.ed. Rio de JaneiroElsevier, 2003. 632p.

UKIL, A.; ZIVANOVIC, R. Abrupt change detection in power system fault analysis using adaptive whitening filter and wavelet transform. **Electric Power Systems Research**, v.76, n.9, p.815–823, Jun. 2006.

USEVITCH, B. A tutorial on modern lossy wavelet image compression: foundations of jpeg 2000. **Signal Processing Magazine, IEEE**, v.18, n.5, p.22–35, Sep. 2001.

VACCA, J. R. **Computer and Information Security Handbook**. 1.ed. San Francisco, CA, USAMorgan Kaufmann Publishers Inc., 2009. 844p.

WALDEN, J.; DOYLE, M.; WELCH, G.; WHELAN, M. Security of open source web applications. In: EMPIRICAL SOFTWARE ENGINEERING AND MEASUREMENT, 2009. ESEM 2009. 3RD INTERNATIONAL SYMPOSIUM ON, 2009. **Anais...** 2009. p.545–553.

WANG, K.; PAREKH, J. J.; STOLFO, S. J. Anagram: a content anomaly detector resistant to mimicry attack. In: RAID, 2006. **Anais...** 2006. p.226–248.

WANG, K.; STOLFO, S. Anomalous Payload-Based Network Intrusion Detection. In: RECENT ADVANCES IN INTRUSION DETECTION, 2004. **Anais...** Springer Berlin / Heidelberg, 2004. p.203–222. (Lecture Notes in Computer Science, v.3224).

WANG, Y. Jump and sharp cusp detection by wavelets. In: RECENT RESEARCH ON THE NITINOL ALLOYS AND THEIR POTENTIAL APPLICATION IN OCEAN ENGINEERING, OCEAN ENGINEERING, 1995. **Anais...** 1995. v.82, n.2, p.385–397.

WURZINGER, P.; PLATZER, C.; LUDL, C.; KIRDA, E.; KRUEGEL, C. SWAP Mitigating XSS Attacks using a Reverse Proxy. In: IEEE COMPUTER SOCIETY PRESS, 2009, Canada. **Anais...** ICSE Workshop on Software Engineering for Secure Systems (SESS), 2009.

APÊNDICE A ARTIGOS PUBLICADOS

***MOZZAQUATRO, B. A.**; AZEVEDO, R. P. ; NUNES, R. C. ; KOZAKEVICIUS, A. J. ; CAPPO, C. ; SCHAERER, C. **Detecção de Ataques Web usando Técnicas de Detecção de Anomalias**. In: Escola Regional de Redes de Computadores, São Leopoldo - RS. Anais da ERRC, 2011. v. 9. p. 101-104.

* Escolhido entre os 5 melhores artigos da Escola Regional de Redes de Computadores 2011.

MOZZAQUATRO, B. A.; AZEVEDO, R. P. ; NUNES, R. C. ; KOZAKEVICIUS, A. J. ; CAPPO, C. ; SCHAERER, C. **Anomaly-based Techniques for Web Attacks Detection** In: Journal of Applied Computing Research. v. 2 p. 111-120. ISSN 2236-8434. 2011.