Universidade Federal do Amazonas
Instituto de Computação
Programa de Pós-Graduacao em Informática

Maria Azevedo Bezerra

Uma Investigação do uso de Características na Detecção de URLs

Manaus
Setembro de 2015

Maria Azevedo Bezerra

Uma Investigação do uso de Características na Detecção de URLs

Trabalho apresentado ao Programa de Pós-Graduação em Informática do Instituto de Computação da Universidade Federal do Amazonas como requisito parcial para obtenção do grau de Mestre em Informática.

Orientador: Prof. Dr. Eduardo Luzeiro Feitosa

Manaus
Setembro de 2015

Aos meus, Raimundo Santino (in memorian) e Francisca (in memorian) pelo amor, carinho e estímulo, sem vocês nada disso seria possível. A minha irmã Vanusa e meu cunhado Wolfango por estarem sempre presentes sendo uma referência de força, coragem e determinação.

Agradecimentos

Primeiramente a Deus, que me abençoou nesta jornada e por ter me dado saúde, força e determinação para superar as dificuldades.

Ao meu orientador, Prof. Eduardo Feitosa, pela oportunidade de contribuir e aprender cada vez mais nesta área tão dinâmica; por todos os conhecimentos riquíssimos, ensinamentos e orientações acadêmicas que fizeram uma diferença nos resultados alcançados neste trabalho; por toda a paciência e confiança para que eu superasse os momentos difíceis que a vida impôs e por toda a motivação e palavras de incentivos, tão importantes para que eu concluísse este trabalho.

Agradeço especialmente aos amigos Osvaldo, Janainny, Jonathan, Maiara pela cumplicidade, amizade e momentos inesquecíveis de estudos. A todos que, de alguma forma, contribuíram para a concretização desta conquista.

Resumo

URLs maliciosas tornaram-se um canal para atividades criminosas na Internet, como *spam* e *phishing*. As atuais soluções para validação e verificação de URLs maliciosas se consideram ou são consideradas precisas, com resultados bem ajustados. Contudo, será que realmente é possível ou factível se obter percentuais beirando 100% de precisão nessas soluções? Neste sentido, esta dissertação descreve uma simples e direta investigação de características, bases e formatos de URLs, visando mostrar que os resultados de validação e verificação de URLs são bastante dependentes de certos aspectos/fatores. A ideia é extrair características (léxicas, DNS e outras) que permitam obter o máximo de informação das URLs e empregar algoritmos de aprendizagem de máquina para questionar a influência dessas características em todo o processo. Como forma de provar essa ideia, foram elaboramos quatro hipóteses, que ao final no trabalho, mostraram que é possível discordar do resultado de vários trabalhos já existentes na literatura.

Palavras-chave: URL, Características, Aprendizagem de Máquina, Hipóteses.

Abstract

Malicious URLs have become a channel for criminal activities on the Internet, such as *spam* and *phishing*. Current solutions for validation and verification of malicious URLs are considered or are believed to be accurate, with well-adjusted results. However, is it really possible or feasible to obtain 100% of accuracy in these solutions? This work describes a simple and direct investigation of features, bases and URL formats, aiming to show that the results of validation and verification URLs are highly dependent on certain aspects/factors. The idea is to extract URL features (lexical, DNS and others) for obtain the maximum information from the URLs and employ machine learning algorithms to question their influence throughout the process. In order to prove this idea, were created four hypotheses that showed that it is possible to disagree with the results of several studies from the literature.

Keywords: URL, Features, Machine Learning, Hypotheses.

Sumário

Capítulo 1

Introdução

Desde o início da vida, o ser humano é levado a gerenciar seus riscos pessoais, ou seja, avaliar quais situações podem ser perigosas e evitá-las. Por exemplo, em cidades onde existem "zonas perigosas", de forma geral, as pessoas entendem que é um local arriscado para visitas casuais. No entanto, esta mesma noção não se traduz para o contexto da Internet. Não existem "placas" ou sinais indicativos sobre a periculosidade de uma determinada página, sítio Web e/ou domínio. Além disso, os mecanismos e dispositivos para diferenciar "locais seguros daqueles perigosos" não são tão eficientes. Como consequência, atacantes e criminosos aproveitam-se para invadir computadores, redes e sistemas, tornando a Internet uma plataforma que suporta uma ampla gama de atividades criminosas, tais como geração de *spams*, fraudes financeiras (*phishing*) e propagação de códigos maliciosos (*malware*).

O sucesso de atividades maliciosas na Internet tem como principal ponto de partida a existência de usuários desavisados e despreparados que visitam sites desconhecidos, acessam e-mails não solicitados, ativam links e/ou fazem o download de programas de forma inadvertida [1]. Além disso, todas essas formas de atividades maliciosas tem em comum o uso de URL (*Uniform Resource Locator*) como canal para, por exemplo, *drive-by-downloads*, *spam* e *phishing*. Kaspersky Lab [2] relata que, em 2012, os ataques baseados no navegador (*browser*) passaram de 946.393.693 para 1.595.587.670, onde 87,36% desses usavam URLs maliciosas. O Grupo de Trabalho Anti-Phishing (APWG) relata, também, que ataques de *phishing* usando URLs maliciosas somaram 115.565 incidentes em 2013. Dos milhões de URLs usadas a cada dia, menos que 0,01% são maliciosas e, além disso, são de curta duração, a fim de evitar o bloqueio por listas negras [3].

No intuito de informar os usuários, preferencialmente com antecedência, se uma determinada URL é ou não perigosa, algumas soluções de segurança vem sendo desenvolvidas, dentre as quais pode-se destacar a navegação segura do Google [4] e serviços de reputação Web da Trend Micro [5]. A solução mais

generalista é o uso de listas negras (*blacklist* [1]), cuja função é manter registro de URLs maliciosas (relacionadas a ataques e fraudes) ou abusivas (no caso do *spam*). Contudo, muitos sites maliciosos acabam não fazendo parte de *blacklists* porque ou são novos e nunca foram avaliados ou porque foram avaliados de forma incorreta. Outra solução é a análise de conteúdo e comportamento de páginas Web, que procura indícios que permitam classificar e detectar uma página como maliciosa. Porém, a análise de conteúdo pode gerar problemas de privacidade enquanto a análise de comportamento pode apresentar um alto custo computacional e acarretar atrasos para o usuário final.

1.1 Motivação

Criadas para facilitar a vida dos Internautas, hoje em dia, as URLs não são mais vistas como um local de conteúdo certo ou seguro. Exemplos de URLs para atividades maliciosas não faltam. No Twitter [6] [2], em virtude do limite máximo de 140 caracteres por *tweet*, as URLs são encurtadas para impactar o mínimo possível no tamanho da mensagem. Entretanto, o serviço de encurtamento de URLs não executa qualquer tipo de verificação antes ou depois do encurtamento. Assim, de acordo com Bevenuto et al. [7], atacantes exploram o uso de URLs encurtadas para esconder links maliciosos que direcionam os usuários para página com propaganda, pornografia, disseminação de vírus ou *phishing*.

Já para *proxies* Web, cuja principal finalidade é intermediar e atender requisições de clientes por alguma página Web, URLs não validadas/verificadas podem facilmente ser utilizadas para disseminação de conteúdo malicioso, tendo em vista que vários usuários fazem uso desse tipo de serviço. O mesmo problema de validação de URLs ocorre na verificação de e-mails que apresentam URLs na forma de links. Por fim, o fato é que a não checagem das URLs pode trazer inconvenientes como redirecionar o usuário para sites com conteúdo inapropriado (cenas pornográficas ou chocantes, por exemplo), páginas destinadas a ataques de *phishing*, sites contendo códigos maliciosos e XSS (*Cross-Site Scripting*), sites de difusão de *spam* ou "pegadinhas" (memes, por exemplo), entre outros locais com atividades não desejadas e muitas vezes maliciosas.

Para ilustrar os problemas de segurança envolvendo URLs, a Tabela 1.1 [8] apresenta os cinco (5) principais objetos maliciosos (URLs, scripts, exploits, arquivos executáveis, entre outros) detectados via antivírus Web da empresa Kaspersky Lab, no ano de 2014. Esses cinco programas maliciosos correspondem a

[1] Tipicamente, listas negras e brancas podem ser encontradas no formato de *add-ons* em navegadores Web, APIs e mecanismos de busca em páginas Web

[2] rede social e *microblogging* que permite a comunicação e o compartilhamento de informação em tempo real entre seus usuários através de mensagens chamadas de (*tweets*)

91,77% dos ataques online registrados.

Tabela 1.1: Os 5 principais objetos maliciosos detectados online

País	Percentual
URL Maliciosas	73,70%
Trojan.Script.Generic	9,10%
Adware.Script.Generic	4,75%
Trojan.Script.Iframe	2,12%
Trojan-Downloader.Script.Generic	2,10%

1.2 Justificativa

Atualmente, existe uma grande quantidade e diversidade de trabalhos que objetivam determinar se uma URL é maliciosa ou não. Dentre os vários tipos de soluções encontradas, destacam-se as baseadas em (ou que geram) *blacklists* e as que utilizam técnicas de aprendizagem de máquina.

As abordagens baseadas em *blacklists* podem ser consideradas as principais medidas contra URLs maliciosas na Internet. Contudo, precisam ser constantemente atualizadas para evitar falhas na detecção, uma vez que URLs maliciosas tem vida curta [9]. Assim, o foco dos diferentes trabalhos nesta linha [9, 10, 11, 12] é sempre tentar manter a *blacklist* atualizada.

Diferente das *blacklists*, as abordagens que empregam aprendizagem de máquina tentam aprender certas características da URL para determinar se ela é ou não maligna. De acordo com Eshete et al. [13], tais soluções fazem uso de algum tipo de análise estática, uma vez que inspecionam os artefatos de páginas Web, sem a necessidade de executar a página em um navegador para obter caraterísticas que possam ser avaliadas. A inspeção geralmente envolve extração rápida de recursos discriminativos (características) da URL, tais como os caracteres da URL, a identidade do host, HTML e os códigos dinâmicos como JavaScript. Os valores das características são então codificados para treinar a máquina de aprendizagem, a fim de construir classificadores capazes de distinguir páginas Web não conhecidas. Uma desvantagem das soluções que empregam aprendizagem de máquina na detecção de URLs maliciosas apresentam dificuldades em detectar ataques que requerem renderização de uma página [13].

Apesar dos esforços e das inúmeras contribuições das abordagens que empregam aprendizagem de máquina [1, 13, 14, 15], dois aspectos chamam atenção: a quantidade e a eficácia das características utilizadas para classificar URL. Em

uma rápida pesquisa pela literatura existente, é possível enumerar mais de 75 características extraíveis de uma URL que podem ser aplicadas na sua classificação. Tal fato gera alguns questionamentos:

1. Existem informações valiosas em todas essas características?
2. Todas essas características são necessárias e/ou são realmente utilizadas na classificação de URLs?
3. Todas essas características tem potencial para indicar ameaças?
4. Existe alguma influência da URL (formato, serviço a que se refere, base de onde foi extraída) sobre as características e, consequentemente, sobre a classificação?
5. É possível categorizar características de modo a permitir o uso mais adequado das mesmas no processo de classificação?

1.3 Objetivo

O objetivo desta dissertação é investigar a capacidade de validar/classificar URLs como benignas, suspeitas ou maliciosas. Para tanto, conjuntos de características extraíveis das próprias URLs serão empregados como fontes de informação e diferentes métodos específicos de aprendizagem de máquina serão utilizados para avaliação.

Especificamente, pretende-se:

- Como base na literatura existente, definir conjuntos de características relevantes e extraíveis das URLs que possam ser empregados para aferir se a URL é benigna, suspeita ou maligna;
- Desenvolver um conjunto de mecanismos de extração de características de URL capaz de retirar informações úteis para o processo de avaliação.

1.4 Contribuições

A partir do desenvolvimento dos objetivos definidos neste trabalho foi possível realizar as seguintes contribuições:

1. Elaborar uma taxonomia para classificação das características observáveis em URLs e utilizadas na detecção de atividades suspeitas e/ou maliciosas.

2. Elaborar um conjunto de scripts que possibilitem a extração de características de URLs;

3. Apresentar uma análise comparativa entre os métodos de classificação KNN, Naive Bayes, SVM e Árvore de Decisão, a fim de demonstrar o desempenho geral na classificação de URLs;

4. Provar que embora a análise de características de uma URL seja uma arma eficaz na detecção de páginas Web maliciosas, fatores como o formato da URL e o seu local de extração podem interferir consideravelmente no resultado do processo de classificação.

1.5 Estrutura do Documento

Este documento está organizado em 6 Capítulos.

No Capítulo 2 são apresentados os conceitos básicos necessários para a compreensão desta dissertação.

O Capítulo 3 apresenta as características extraíveis de URLs, encontradas na literatura e empregadas na detecção de atividades maliciosas. Como diferencial, esse Capítulo propõem uma taxonomia baseada na necessidade de conexão com a Internet, para extração das características.

O Capítulo 4 discute alguns trabalhos relacionados que utilizam as características mencionadas no Capítulo anterior.

Já o Capítulo 5 detalha a implementação das características selecionadas e apresenta o protocolo experimental necessário para validação dessas características.

No Capítulo 6 são apresentados os experimentos e resultados que validam a investigação realizada nesta dissertação.

Por fim, no Capítulo 7 são apresentadas as conclusões, as dificuldades encontradas e os trabalhos futuros.

Capítulo 2

Conceitos Básicos

Este Capítulo apresenta os principais conceitos básicos necessários para a compreensão dos temas abordados nesta dissertação. Conceitos necessários para a compreensão do trabalho, como URL e aprendizagem de máquina, são descritos.

2.1 URL

URL (*Uniform Resource Locator*) é um formato universal para representar um recurso na Internet, de modo a ser facilmente lembrado pelos usuários. Definida e especificada na RFC 1738 [16], uma URL é composta por duas seções, conforme descrito a seguir.

<esquema>:<parte-específica-do-esquema>

O **esquema** de uma URL representa a linguagem ou protocolo utilizado para comunicação. O esquema mais comum empregado na Internet é o do protocolo HTTP (*HyperText Transfer Protocol*). Entretanto, uma URL pode utilizar esquemas dos protocolos FTP (*File Transfer Protocol*), News, Mailto, Gopher, entre outros. É importante esclarecer que as partes específicas de uma URL variam de acordo com o esquema e que em alguns protocolos, como HTTP e FTP, as partes específicas podem ser organizadas hierarquicamente.

No âmbito do esquema para o protocolo HTTP, foco deste trabalho, as partes específicas de uma URL são: domínio e caminho (Figura 2.1).

O **domínio**, também chamado de máquina ou host, faz referência ao nome do domínio que hospeda o recurso pedido e pode ser representado tanto por um nome quanto pelo endereço IP do servidor. Um domínio é formado por um ou mais marcadores (camadas) que são concatenados e delimitados por pontos (“.”) e cuja hierarquia de leitura é definida da direita para a esquerda. Assim, um domínio tem em sua primeira camada, na sua parte mais a direita, um TLD (*Top*

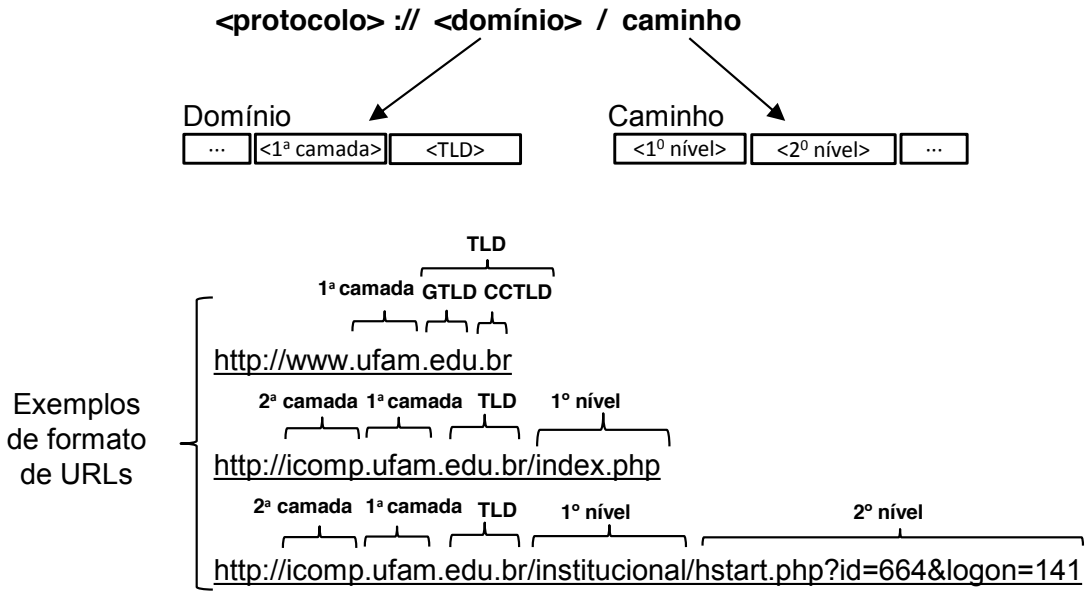

Figura 2.1: Formato de uma URL

Level Domain) para representar seu tipo (*.com*, *.net*, *.org*, *.edu*, entre outros) - chamado de *Generic TLD* (GTLD) - e o país de origem (*.br* para o Brasil, *.uk* para o Reino Unido, *.us* para os Estados Unidos, entre outros) - chamado de *Country Code TLD* (CCTLD). Em seguida, tem-se o segundo nível (*Second Level Domain* - SLD) que representa o nome do domínio propriamente dito. É possível existir ainda outras camadas, cuja finalidade é representar especificidades do nome de domínio. Na Figura 2.1, o segundo exemplo apresenta três (3) camadas no domínio: edu.br (TLD), ufam (nome do domínio) e icomp (parte específica do domínio).

Entre o domínio e o caminho, existe uma parte denominada **porta**, que nada mais é do que um número associado a um serviço que permite ao servidor saber que tipo de recurso está sendo pedido. A porta associada por padrão ao protocolo HTTP é a porta número 80. O número da porta é facultativo.

Já o **caminho** (do inglês *path*) permite ao servidor conhecer o lugar onde o recurso está armazenado, ou seja, o(s) diretório(s) e o nome do recurso pedido, bem como os argumentos empregados para realização de alguma ação. O caminho é delimitado por uma barra ("/") e sua hierarquia de leitura é da esquerda para a direita. Na Figura 2.1, o segundo exemplo mostra um caminho formado por uma parte ou nível (index.php), no caso um arquivo de extensão *php*. O terceiro exemplo apresenta dois níveis (instituição e hstart.php?id=664&logon=141), onde instituição é o recurso desejado (no caso um diretório), hstart.php é um arquivo e o restante é um argumento. A Tabela 2.1 ilustra o terceiro exemplo de URL.

Tabela 2.1: Componentes de uma URL

Componente	Exemplo
URL	http://icomp.ufam.edu.br/inst/hstart.php?id=664&logon=141
Nome do domínio	icomp.ufam.edu.br
Caminho	inst/hstart.php
Sub diretório	inst
Nome do arquivo	hstart
Extensão do arquivo	php
Argumento	id=664&logon=141

Opcionalmente, um caminho pode ter uma **String de consulta** (*Query string*), um conjunto de parâmetros a ser enviado ao servidor, usado para localizar, filtrar, ou mesmo criar o recurso; e um **Fragmento** para referenciar a uma parte ou posição específica dentro do recurso.

2.2 Extração de Características

Segundo o dicionário Aurélio, característica é aquilo que caracteriza algo; uma particularidade. No contexto deste trabalho, característica é uma informação que pode ser extraída de uma URL com o objetivo de classificar um domínio e/ou endereço na Web.

A extração de características consiste na retirada de elementos que caracterizam um conjunto para classificação. É um passo de pré-processamento essencial em problemas que envolvem o reconhecimento de padrões em aprendizagem de máquina.

2.3 Aprendizagem de Máquina

Com o objetivo de entender o processo de aprendizagem e a descoberta de padrões que determinam se uma URL é maliciosa ou não, esta seção apresenta os principais conceitos sobre aprendizagem de máquina e uma breve descrição dos métodos de classificação usados neste trabalho.

2.3.1 Definição

Em linhas gerais, Aprendizado de Máquina é uma área de Inteligência Artificial cujo objetivo é o desenvolvimento de técnicas computacionais sobre o aprendizado bem como a construção de sistemas capazes de adquirir conhecimento de forma automática. De acordo com Alpaydin [17], um sistema de aprendizado são

programas de computador (algoritmos) utilizados para otimizar um critério de desempenho, usando dados de exemplo ou experiência do passado. A definição clássica de Mitchell [18], diz que na aprendizagem de máquina: "Um programa aprende a partir da experiência **E**, em relação a uma classe de tarefas **T**, com medida de desempenho **P**, se seu desempenho em **T**, medido por **P**, melhora com **E**". Nesta dissertação, a tarefa **T** é classificar potenciais novas URLs como boas ou maliciosas/suspeitas; a medida de desempenho **P** é a porcentagem de URLs classificadas corretamente; e a experiência de treinamento **E** é uma base de dados histórica em que as URLs já conhecidas são previamente classificadas como boas ou más.

Embora os estudos sobre aprendizagem de máquina apresentem divergências entre as classificações, as mais empregadas na área de segurança em redes de computadores são a aprendizagem supervisionada e a aprendizagem não-supervisionada. Na aprendizagem supervisionada, conhecida como classificação, o algoritmo de aprendizado recebe um conjunto de exemplos de treinamento para os quais os rótulos da classe associada são conhecidos. Cada exemplo (instância ou padrão) é descrito por um vetor de valores (atributos) e pelo rótulo da classe associada. Seu objetivo é construir um classificador que possa determinar corretamente a classe de novos exemplos ainda não rotulados. Vale deixar claro que esta dissertação faz uso de aprendizagem supervisionada.

Na aprendizagem não-supervisionada, o algoritmo analisa os exemplos fornecidos e tenta determinar se alguns deles podem ser agrupados de alguma maneira, formando agrupamentos ou clusters. Após a determinação dos agrupamentos, em geral, é necessário uma análise para determinar o que cada agrupamento significa no contexto do problema que está sendo analisado.

2.3.2 Métodos e Algoritmos de Classificação

Existem diversos métodos de aprendizagem supervisionada disponíveis e utilizados com grande efetividade em diversas aplicações. A seleção do melhor classificador depende de uma série de variáveis, dentre elas, o tipo de problema a ser tratado, a natureza e a disponibilidade de dados, o desempenho, entre outras [19]. Esta dissertação comparou e avaliou 04 (quatro) classificadores em um mesmo conjunto de dados. Os resultados dos experimentos realizados com esses classificadores podem ser verificados no Capítulo 5 e 6.

Esta seção irá se concentrar numa breve descrição dos classificadores empregados nos experimentos realizados neste trabalho. Maiores detalhes sobre o emprego de aprendizagem de máquina em segurança de redes de computadores poderão ser obtidos em Henke et al. [20].

Naive Bayes

Considerado o classificador mais utilizado em aprendizagem de máquina, o classificador Naive Bayes é uma técnica simples bastante aplicada ao problema de classificação de tráfego Internet. Segundo Buntine [21], o classificador Naive Bayes pode ser entendido como uma forma especializada de uma rede Bayesiana intitulada "Naive" (ingênua) por se sustentar em dois importantes pressupostos: A suposição que os atributos preditivos são condicionalmente independentes dada a classe e que nenhum atributo oculto ou subtendido influência o processo de predição. Assim, um classificador Naive Bayes pode ser representado graficamente conforme a Figura 2.2, na qual todos os enlaces partem do atributo classe para os atributos observáveis e preditivos $(X_1, X_2, ...X_k)$, expressando a independência condicional destes dado ao atributo classe (C). Essas suposições apoiam muitos algoritmos eficientes tanto para classificação quanto aprendizado.

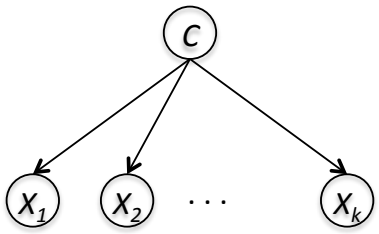

Figura 2.2: Projeção do Classificador Naive Bayes como uma Rede Bayesiana. Fonte: [17]

Considerando que para classificar uma instância de teste x tem-se:

- C como sendo uma variável aleatória que denota a classe de uma instância;
- X como sendo um vetor de variáveis aleatórias representando os valores observados dos atributos;
- c como sendo um rótulo de uma determinada classe, e;
- x como sendo um vetor de valores de atributo.

A classe mais provável será aquela com maior valor para $P(C = c|X = x)$, ou seja, a probabilidade da classe c dada a instância x. A expressão seguinte apresenta a regra de Bayes, aplicada para calcular esta probabilidade, onde $X = x$ corresponde ao evento $X_1 = x_1 \wedge X_2 = x_2 \wedge ...X_k = x_k$ e $P(C = c)$ representa a probabilidade a priori de c, ou seja, a probabilidade de obtenção da classe c sem levar em conta os dados de treinamento.

$$p(C = c|X = x) = \frac{p(C=c)p(X=x|C=c)}{p(X=x)}$$

A hipótese de independência entre características pode parecer restritiva, mas resultados práticos encontrados na literatura em diversas áreas de aplicação mostram que Naive Bayes produz elevada taxa de classificação mesmo quando as características são claramente dependentes [22, 23].

A principal vantagem de Naive Bayes é a baixa complexidade na fase de treinamento, tendo em vista que essa fase envolve apenas o cálculo de frequências para que as probabilidades sejam obtidas. Essa peculiaridade faz com que Naive Bayes seja indicado para aplicações [1] onde o treinamento precisa ocorrer de forma online e com frequência regular. Outra característica positiva é a possibilidade de manipular atributos nominais e numéricos. Atributos nominais são frequentes em detecção de spam, XSS, páginas *phishing*, dentre outros problemas de classificação de documentos textuais.

SVM (*Support Vector Machine*)

SVM é uma técnica de classificação amplamente aplicada em detecção e classificação de URLs [15, 24], fundamentada nos princípios da Minimização do Risco Estrutural (Structural Risk Minimization - SRM) [25]. Sua finalidade é buscar minimizar o erro com relação ao conjunto de treinamento (risco empírico), assim como o erro com relação ao conjunto de teste, isto é, conjunto de amostras não empregadas no treinamento do classificador (risco na generalização). O objetivo de SVM consiste em obter um equilíbrio entre esses erros, minimizando o excesso de ajustes com respeito às amostras de treinamento (*overfitting*[1]) e aumentando consequentemente a capacidade de generalização.

A questão da generalização pode ser melhor avaliada para o caso de duas classes. Assumindo que as amostras de treinamento das duas classes são linearmente separáveis, a função de decisão mais adequada é aquela para a qual a distância entre os conjuntos das amostras de treinamento é maximizada. Neste contexto, a função de decisão que maximiza esta separação é denominada de ótima (Figura 2.3).

O algoritmo original de SVM não encontra a solução desejada quando aplicado a dados não linearmente separáveis, característica presente na maioria dos problemas reais [17]. Por isso, a decisão em SVM é expressa em termos de uma função kernel $k(x, x')$ que calcula similaridade entre dois vetores de características e coeficientes não-negativos $\{\alpha_i\}$ $i^n = 1$, que indicam exemplos de treinamentos que se encontram perto da fronteira de decisão [1]. Maiores detalhes sobre as funções de kernel podem ser obtidas em [20].

[1] O problema denominado de *overfitting* consiste em o classificador memorizar os padrões de treinamento, gravando suas peculiaridades e ruídos, ao invés de extrair as características gerais que permitirão a generalização ou reconhecimento de padrões não utilizados no treinamento do classificador.

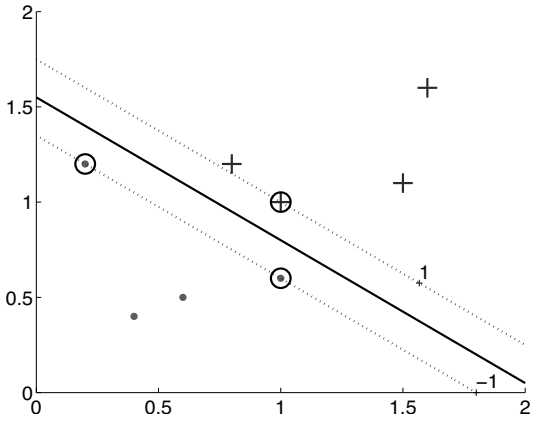

Figura 2.3: Hiperplano de separação ótima para um problema com duas classes. Fonte: [17]

As vantagens desse classificador são: conseguir lidar bem com grandes conjuntos, possuir um processo de classificação rápido e possuir uma baixa probabilidade de erros de generalização. A desvantagens são: precisa definir um bom kernel (função que define a estrutura do espaço de características onde o hiperplano de separação ótima será encontrado) e empregar um tempo de treinamento longo, dependendo do número de dimensionalidade dos dados.

KNN (K-Nearest Neighbor)

KNN é um algoritmo de classificação baseado no vizinho mais próximo, ou seja, depende de medidas de distância usadas para classificar objetos com base em exemplos de treinamento, que estão mais próximos no espaço de caraterísticas. Para determinar a classe de um elemento que não pertença ao conjunto de treinamento, o classificador KNN procura K elementos do conjunto de treinamento que estejam mais próximos deste elemento desconhecido, ou seja, que tenham a menor distância. Estes K elementos são chamados de K-vizinhos mais próximos. Verifica-se quais são as classes desses K vizinhos e a classe mais frequente será atribuída à classe do elemento desconhecido. A métrica mais comum no cálculo de distância entre dois pontos no KNN é a distância Euclidiana, mas pode-se empregar também a distância Manhattan e a distância Minkowski.

Em linhas gerais, o procedimento de execução do KNN consiste em:

1. Calcular a distância entre o exemplo desconhecido e outros exemplos do conjunto de treinamento;

2. Identificar os K vizinhos mais próximos;

3. Utilizar o rótulo da classe dos vizinhos mais próximos para determinar o rótulo da classe do exemplo desconhecido (votação majoritária).

A principal vantagem do classificador KNN é a de ser uma técnica simples e facilmente implementada. Já como desvantagem, o uso de poucas instâncias de treinamento pode gerar resultados errados, já que por padrão o KNN intuitivamente usa mais do que um vizinho mais próximo.

Árvore de Decisão

Árvore de decisão é uma técnica de aprendizagem de máquina composta por três elementos básicos:

- nó raiz, que corresponde ao nó de decisão inicial;
- arestas, que correspondem as diferentes características;
- nó folha, que corresponde a um nó de resposta, contento a classe a qual pertence o objeto a ser classificado.

Em árvores de decisão, duas grandes fases devem ser asseguradas. A primeira refere-se à construção da árvore e tem como base o conjunto de dados de treinamento, sendo dependente da complexidade dos dados. Uma vez construída, regras podem ser extraídas através dos diversos caminhos providos pela árvore para que sejam geradas informações sobre o processo de aprendizagem. A segunda refere-se à classificação, pois para classificar uma nova instância, os atributos são testados pelo nó raiz e pelos nós subsequentes, caso necessário. O resultado deste teste permite que os valores dos atributos da instância dada sejam propagados do nó raiz até um dos nós folhas. Ou seja, até que uma classe seja atribuída à amostra.

De acordo com Nunan et al. [26], vários algoritmos foram desenvolvidos a fim de assegurar a construção de árvores de decisão e seu uso para a tarefa de classificação. O ID3 e C4.5, algoritmos desenvolvidos por Quinlan [27], são provavelmente os mais populares. Vale também mencionar o algoritmo CART de Breiman [28].

A principal vantagem do uso de Árvores de Decisão é a de obter regras que explicam claramente o processo de aprendizagem, podendo ser usadas para uma compreensão mais completa dos dados e dos atributos mais relevantes para o problema de classificação. Vale também ressaltar que esta técnica permite a obtenção de resultados que, em geral, são superados apenas por algoritmos de complexidade muito superior.

Capítulo 3

Características de URLs

Uma breve revisão na literatura de validação e detecção de URLs suspeitas, especialmente quando relacionadas as atividades maliciosas como *phishing* e spam, mostra a existência de diversos trabalhos nesta área [1, 9, 10, 11, 12, 13, 14, 15]. Em comum, todos eles (sejam propostas, ferramentas, soluções e métodos) utilizam características observáveis (extraíveis) das próprias URLs, como, por exemplo, o tamanho da URL e quantidade de determinados caracteres para avaliar e inferir sobre a reputação de uma URL. Contudo, um ponto que chama a atenção sobre esse assunto é que embora seja possível enumerar um grande número de características que podem ser extraídas das URLs, até onde esse trabalho pesquisou, não existem classificações formais (taxonomia) para essas características.

Alguns trabalhos [24, 29, 30] fazem uso de dois grupos de características, sendo, normalmente, (i) características léxicas - aquelas relacionadas aos caracteres que compõem a URL - e (ii) de rede ou relacionadas ao host. Já outros trabalhos agrupam as características de acordo com suas afinidades ou funcionalidades, mas sempre como forma de apresentar os resultados obtidos em suas pesquisas. Choi et al. [15], por exemplo, divide as características em seis (6) grupos: textuais ou léxicas, popularidade do link, conteúdo da página, DNS, redes de fluxo rápido com DNS e tráfego de rede. Sayamber e Dixit [31] apresenta uma divisão em sete (7) grupos de características: léxicas, rede ou host, conteúdo da página, popularidade do link, especiais, DNS e redes de fluxo rápido com DNS.

O fato é que a forma como as características são agrupadas ou é muito simplista ou é generalista ao extremo. Dada essa lacuna, este Capítulo propõe uma forma simples de ordenar (classificar) as características observáveis em URLs (e que são aplicadas na verificação e detecção de atividades suspeitas e maliciosas).

3.1 Taxonomia

Após analisar um grande número de trabalhos que utilizam as mais variadas características de URLs na elaboração de soluções e propostas que visam confirmar a reputação de uma URL como benigna, suspeita ou maliciosa, é fácil perceber que tais características podem ser agregadas sem perdas.

Avaliando os outros trabalhos e com base na experimentação de extração de características, a taxonomia proposta neste trabalho tem a necessidade de comunicação externa como base primária para categorizar características. Além disso, percebe-se também que várias características, embora agregáveis em um único grupo, podem e devem ser separadas de forma a melhor representar suas funcionalidades e o processo de extração de seus valores.

Sendo assim, a taxonomia proposta gera dois grandes grupos de características que podem ser extraídas de uma URL: offline e online. Características offline dependem exclusivamente dos valores encontrados na própria URL (os caracteres, por exemplo) enquanto as características online dependem de conexão com outros serviços da Internet para obter valores/informações. Em termos práticos, o grupo offline é representado pela variada gama de características léxicas. Já o grupo online, formado por características baseadas em informações da URL (seu conteúdo) e do domínio (DNS, por exemplo), popularidade e recursos de rede, depende de acesso a Internet para obter valores/informações. A Figura 3.1 ilustra a classificação proposta.

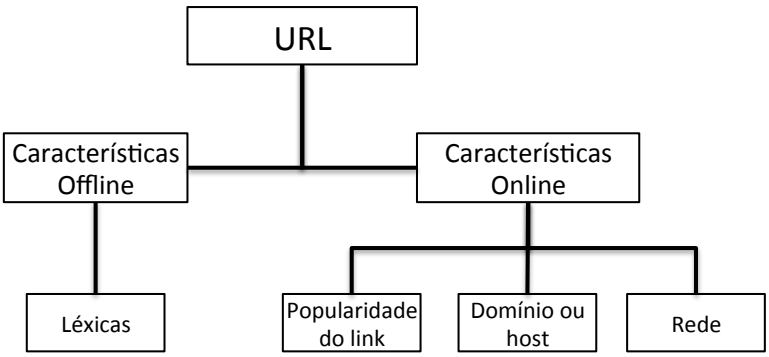

Figura 3.1: Taxonomia proposta

Vale ressaltar dois aspectos sobre a taxonomia apresentada. O primeiro é que as características de conteúdo da página são muito variadas e extensas. Seu uso depende diretamente da solução a ser proposta. Como este trabalho foca apenas na URL, tais características não serão explicadas. Exemplos de características do conteúdo da página podem ser obtidas em [26, 32, 33, 34, 35].

O segundo aspecto é que as características relacionadas ao domínio ou host

apresentam variações no que diz respeito a extração de seus valores. Sendo assim, a classificação proposta pode ser expandida para refletir essa diferenciação. A Figura 3.2 ilustra a taxonomia final proposta nesta dissertação.

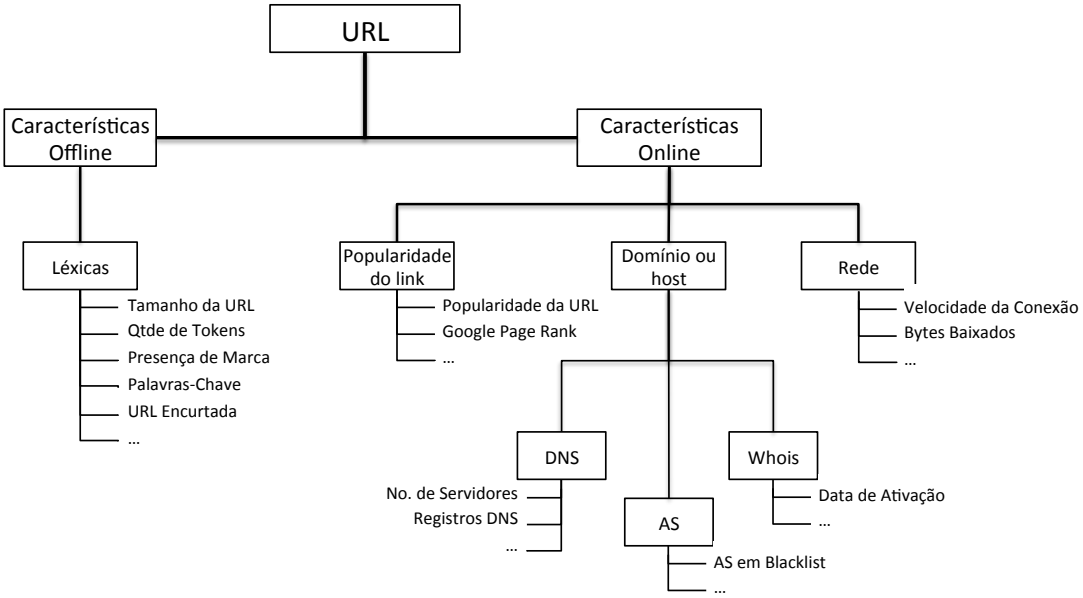

Figura 3.2: Melhoramento na taxonomia proposta

A seguir, cada uma dessas classes são detalhadas.

3.2 Características

3.2.1 Popularidade do Link

Características de popularidade do link tentam estimar, através da contagem do número de ligações (links) recebidas de outras páginas Web, a importância (utilização) de uma determinada página. Em linhas gerais, tais características podem ser consideradas como uma medida da reputação de uma URL. Desta forma, enquanto sites maliciosos tendem a ter um valor pequeno de popularidade do link, sites benignos, principalmente os populares, tendem a ter um grande valor.

A obtenção de características de popularidade do link é feita através da pesquisa em diferentes mecanismos de busca. Sites como Google, Bing, Baidu, Yahoo!, AllTheWeb, Ask e Alexa.com têm sido empregados em diversos trabalhos [13, 15] para extração de características de popularidade do link.

No trabalho de Choi et al. [15], os autores afirmam que a popularidade de link pode ser manipulada por atacantes, através de *link-farming* [36], o uso de

um grupo de páginas Web que se apontam umas para as outras, aumentando assim a sua popularidade.

A Tabela 3.1 detalha as principais características de popularidade do link.

Tabela 3.1: Principais características de popularidade do link

Característica	**Descrição/Funcionalidade**	**Tipo/Valor**
Popularidade da URL	Mede a popularidade da URL em sites de busca.	Real
Popularidade do Domínio	Mede a popularidade do domínio em sites de busca.	Real
Proporção de links distintos do domínio	Gera uma razão entre o número de domínios únicos para o número total de domínios que apontam para a URL alvo.	Real
Proporção máxima de links do domínio	Gera uma razão entre o número máximo de links de um único domínio para o número total de domínios que apontam para a URL alvo.	Real
Razão de links do domínio em *Spam*, *Phishing* e *Malware*	Representam a razão de domínios de um tipo específico que apontam para a URL alvo. Para medir estas três características, os autores usaram as listas de URL maliciosa descritas na Tabela 3.4.	Real
Reputação Social	Mede o número de vezes que uma URL é publicamente compartilhada em redes sociais (Facebook, Twitter e Google Plus). Os autores dessa característica [13] afirmam que uma avaliação experimental mostrou que páginas benignas possuem altas contagens enquanto URLs maliciosas são compartilhadas em menor número.	Inteiro
Domínio nos resultados do Google Search	Verifica se o domínio de uma URL corresponde a quaisquer domínios dos dois principais resultados da pesquisa. De acordo com os autores [11], se o domínio aparecer nos dois primeiros resultados da pesquisa, essa características recebe 1. Senão recebe 0.	Binário
Page Rank[1]	Mede a importância relativa de uma página dentro de um conjunto de páginas da Web, isto é, quanto maior o Page Rank, maior a importância da página. Autores como [33] argumentam que páginas *phishing* são de curta duração e, portanto, devem ter um Page Rank baixo ou mesmo não ter Page Rank. O algoritmo de Page Rank mais famoso é o do Google, por isso é comum encontrar Google Page Rank para representar o conceito.	Inteiro

[1]PageRank é um sistema para dar notas a páginas na Web, desenvolvido por Larry Page e Sergey Brin na Universidade de Stanford em 1998.

3.2.2 Relativas ao Domínio ou Host

Características relacionadas ao domínio ou host são aquelas referentes as informações exclusivas do nome de domínio ou servidor da URL. Basicamente, respondem a perguntas como "onde" está hospedado e localizado o site, "quem" o gerencia e "como" ele é administrado. Essas características são obtidas através de dados do DNS (*Domain Name Service*), do ASN (*Autonomous System Number*) e do registro (*whois*). Sua importância se deve ao fato de que sites maliciosos tendem a serem hospedados em fornecedores de serviço menos respeitáveis, em máquinas não convencionais ou em registros corrompidos. Um bom exemplo de uso dessas características é dado por Ramachandran et al. [37] ao demonstrar que uma parte significativa dos spams veio de um conjunto relativamente pequeno de sistemas autônomos (AS). Neste grupo destacam-se características como o número de servidores DNS, o número de endereços IP, a data de ativação do domínio, entre outros.

Além das características bem conhecidas dessa natureza, Choi et al. [15] propõem novas características relacionadas a DNS, mais especificamente a redes de fluxo rápido (*Fast-Flux Service Network* - FFSN) baseadas em DNS. De acordo com os referidos autores, FFSN estabelecem redes proxy para sediar serviços online ilegais com disponibilidade muito elevada e são normalmente empregadas por atacantes para fornecer conteúdo malicioso como *malware*, sites de *phishing* e campanhas de spam. Para detectar URLs que utilizam redes FFSN, os autores usaram características discriminativas propostas por Holz et al. [38]. Em linhas gerais, a ideia é obter o nome de domínio da URL e realizar consultas consecutivas ao DNS (sempre após o TTL de cada consulta esgotar). Choi et al. [15] é capaz de estimar o *fluxiness* (ϕ) dos endereços IP únicos e dos ASNs, bem como dos servidores de nomes únicos, dos endereços IP de servidores de nomes e os ASN dos servidores de nomes de todas as consultas DNS.

A Tabela 3.2 detalha as características mais usuais relacionadas ao domínio ou host.

Tabela 3.2: Principais características relacionados ao domínio ou host

Característica	**Descrição/Funcionalidade**	**Tipo/Valor**
Número de Endereços IP resolvidos	Conta a quantidade de endereços IP associados ao domínio da URL.	Inteiro
Número de Servidores de Nome	Contabiliza a quantidade de servidores de nome associados ao domínio da URL.	Inteiro
Número de Endereços IP dos Servidores de Nome	Contabiliza a quantidade de números de endereços IP dos servidores de nome.	Inteiro

Tabela 3.2: Principais características relacionados ao domínio ou host (Continuação)

Característica	Descrição/Funcionalidade	Tipo/Valor
Registros DNS	Verifica se os registros DNS do tipo A (endereço), NS (servidor de nomes) e MX (mail exchange) do servidor de domínios pertencem ao mesmo AS. De acordo com Ma et al. [1], URLs maliciosas tendem a residir em IPs e/ou ASNs diferentes.	Binário
Endereço IP do(s) servidor(es) de nome(s) em *blacklist*	Verifica se o endereço IP de um servidor do nome do domínio pertence a alguma *blacklist*.	Binário
ASN malicioso via IP	Verifica se o ASN, obtido através do endereço IP da URL, está presente em alguma relação de ASNs maliciosos (*blacklist*).	Binário
ASN malicioso via Nome do Domínio	Verifica se o ASN, obtido através do nome do domínio, está presente em alguma relação de ASNs maliciosos (*blacklist*).	Binário
Localização geográfica do servidor de nomes	Verifica se a localização geográfica do servidor de nomes do domínio é igual ao TLD da URL.	Binário
Localização geográfica do prefixo de rede (IP)	Verifica se localização geográfica do prefixo de rede (IP) é igual ao TLD da URL.	Binário
Localização geográfica do AS	Verifica se a localização geográfica do AS é igual ao TLD da URL.	Binário
Dados do Whois	Avalia informações como as datas de registro dos servidores de nomes do domínios, o tempo que o domínio está ativo, entre outros dados.	Dependente do uso

3.2.3 Recursos de Rede

Esse grupo abrange aquelas características relacionadas a informações mais diversificadas, que não podem ser categorizadas nos outros grupos. Por exemplo, URLs maliciosas podem redirecionar o usuário até atingir o local da atividade ilícita. Isso pode ocorrer através de redirecionamentos dentro do código HTML ou via o encurtamento das URLs. Além dessa características, informações que precisam contabilizar respostas vindas de serviços Internet também compõem esse grupo.

3.2.4 Léxicas

Características léxicas são as propriedades textuais que compõe uma URL, incluindo os símbolos e marcadores, mas não incluindo o conteúdo da página. Uma vez que estão relacionadas a padrões no texto, essas características são extraídas

Tabela 3.3: Principais características de Rede encontradas na literatura

Característica	Descrição/Funcionalidade	Tipo/Valor
Contagem de redirecionamentos	De acordo com [15], atacantes tentam esconder-se através de redirecionamentos (iframe HTML ou encurtamento de URL). Desta forma, realizar a contagem de redirecionamentos pode ser um útil recurso para detectar URLs maliciosas.	Inteiro
Bytes baixados do campo *content-length*	No pacote HTTP existe um campo (*content-length*) que marca o comprimento total do pacote HTTP. Atacantes costumam colocar valores mal formados (negativos) nesse campo para tentar ataques de buffer estouro. Assim, medir o tamanho do campo *content-length* pode ser usado como um recurso discriminativo de URLs maliciosas [15].	Real
Tempo de pesquisa de domínio	Uma vez que sites benignos tendem a ser mais acessados, seu tempo de resposta a uma consulta de domínio tende também a ser mais rápido do que de sites maliciosos. Choi et al. [15] propõe medir esse tempo de reposta.	Real
Hospedagem da URL	McGrath e Gupta afirma que a verificação da hospedagem de uma URL recente é uma característica válida na detecção de URLs maliciosas. Visto que sites benignos estão, de modo geral, ativos há mais tempo do que sites maliciosos.	Inteiro

através de tokens (símbolos como "/", "**.**", "**,**" , "=" , "**?**", "**-**", "**@**", "**&**" e ou palavras-chave) da URL e empregadas em algum tipo de contabilização.

Para melhorar o entendimento sobre características léxicas, duas das mais relevantes, encontradas em grande parte da literatura, são descritas a seguir:

- **Quantidade de Tokens**: A simples quantificação (contabilização) destes símbolos pode ajudar a mostrar o quão confiável uma URL é. Tomando a quantidade de pontos (".") por exemplo, percebe-se que nas duas URLs entre parênteses (`http://www.bank.com.br.badsite.com/` e `http://badsite.com/www.bank.com.br/`) existe uma quantidade incomum de pontos, o que pode indicar a presença de uma URL maliciosa. A quantidade de tokens pode ser avaliada em toda a URL, mas também pode ser avaliada apenas na FQDN ou apenas para parte do caminho.

- **Tamanho da URL**: A quantidade de caracteres que formam uma URL também é um aspecto interessante. Existem URLs que possuem uma grande quantidade de caracteres, que diverge do número de caracteres de URLs tradicionais. Isso pode desviar a atenção do usuário e torna-se um sinal de URL maliciosa.

A Tabela 3.4 relaciona as principais características léxicas encontradas na literatura em trabalhos relacionados a detecção de URLs maliciosas

Tabela 3.4: Principais características léxicas

Característica	Descrição/Funcionalidade	Tipo/Valor
Quantidade de Tokens	Conta a quantidade de tokens "/", ".", ",", "=" , "?", "-", "@", "&" presentes na URL. Grande parte dos trabalhos [11, 14, 15, 24, 34] contabiliza os tokens no domínio, no caminho e em toda a URL. Já o Trabalho de Anh et al. [32] prova que contabilizar os tokens no domínio, no diretório, no arquivo e nos argumentos de uma URL obtêm melhores resultados, uma vez que evitam técnicas de ofuscação como as apresentadas em [33].	Inteiro
Média de Tokens	Calcula a média de tokens presentes no domínio e no caminho da URL. Os autores dessa característica [15] afirmam que ela ajuda a detectar padrões em grandes coleções de URLs. Nesta característica, os tokens são contabilizados no domínio, no caminho e em toda a URL.	Real
Maior Comprimento entre Tokens	Checa o número de caracteres (string) entre tokens. Proposta por [11], essa característica é aplicada tanto no domínio quanto no caminho da URL. Os autores argumentam que URLs maliciosas tem, relativamente, nomes de domínio ou hosts longos quando comparados com URLs legítimas. Por exemplo, a URL maliciosa `http://31837.hzaseruijintunhfeugandeikisn.com/5/54878/` tem uma sequência com 28 caracteres (*hzaseruijintunhfeugandeikisn*).	Inteiro
Composição do Nome do Domínio	Avalia a frequência dos caracteres que compõem o nome do domínio, excluindo o TLD. Os autores dessa característica [34] provaram que URLs benignas (base DMOZ [39]) tendem a manter uma frequência esperada de vogais e consoantes na língua inglesa, enquanto URLs maliciosas (extraídas, por exemplo, do PhishTank) tendem a usar menos vogais nos nomes de seus domínios.	Inteiro
Tamanho da URL e do Domínio	Conta o comprimento (em caracteres) da URL e do domínio. A maior parte dos trabalhos contabiliza apenas o comprimento da URL e do domínio, mas, assim como na característica Quantidade de Tokens, o trabalho de Anh et al. [32] contabiliza o tamanho do domínio, do diretório, do arquivo, dos argumentos e da URL.	Inteiro
Hífens no Domínio	Verifica o número de traços no nome do host da URL. Segundo os autores [11], muitos sites maliciosos possuem nomes muito compridos, com palavras concatenadas através de hifens (`http://yj4yb6hmb3.boy-cant-get-you-out-of-my-head.cn/yj4yb6hmb3/Oraliao_show_23Y`).	Inteiro

Tabela 3.4: Principais características léxicas (Continuação)

Característica	Descrição/Funcionalidade	Tipo/Valor
Presença de Marca	Verifica a presença da marca (*brand name*) na URL. Em [34], os autores dessa característica afirmam que páginas de *phishing* provavelmente tem como alvo uma marca (produto ou empresa) amplamente confiável, por isso a presença do nome ou da marca em uma URL não relacionada a marca pode significar uma página maliciosa. Essa características é considerada léxica porque todos os trabalhos que a implementam fazem uso de uma lista local para comparação dos nomes e marcas.	Binário
Palavras-chave na URL	Verifica a existência de determinadas palavras-chave *strings* na URL. Em [14], os autores afirmam que palavras-chave como: *account*, *update*, $confirm$, $verify$, *secur*, $notif$, *log*, *click*, *inconveninen*, *ebay* e *paypal* são comuns em ataques de *phishing*.	Binário
Domínio em endereço IP	Verifica se o domínio da URL está no formato de endereço IP como, por exemplo, `http://192.168.0.1/`. Vários autores [11, 35, 40] afirmam que ataques phishing utilizam computadores comprometidos. Como essas máquinas normalmente não tem nenhuma entrada DNS, o único modo de referenciá-las é através do endereço IP.	Binário
Domínio em endereço IP Codificado	Verifica se o domínio da URL está representado por um endereço IP codificado. Ao invés de representar o domínio através de endereço IP, atacantes representam o endereço IP em hexadecimal (por exemplo, `http://0x42.0x1D.0x25.0xC2/`) ou inteiro longo (por exemplo, `http://1037729794/cache`). De acordo com [11], esse é um comportamento típico sites maliciosos.	Binário
Domínio Codificado	Verifica se o domínio da URL está representado de forma codificada (por exemplo, `http://www.%64isc%72%65%74%2done-%6ei%67h%74.%63o%6d`). De acordo com [11], domínios benignos nunca são apresentados desta forma, ou seja, apenas sites maliciosos tentam iludir os usuários com esta artimanha.	Binário
Caracteres Especiais Duplicados	Verifica a existência de uma cadeia de caracteres especiais mal formados, inseridos nas aberturas e fechamentos de tags [41], frequentemente encontrados em ataques XSS (Exemplo: `<"<<sCrIpT>alert(document.cookie)</sCrIpT><"<sCrIpT>alert(document.cookie)<"</sCrIpT>`).	Binário

Tabela 3.4: Principais características léxicas (Continuação)

Característica	Descrição/Funcionalidade	Tipo/Valor
Código Ofuscado	Verifica se os argumentos de uma URL estão ofuscados por cadeias de caracteres nos formatos Hexadecimal, Decimal, Octal, Unicode, Base64, caracteres de referência HTML e HTML Name. De acordo com os autores [26] os exemplos mais comuns envolvem os argumentos ofuscados (http://www.siteconfiavel.com.br/search.html?type=%3C%22%3C%3C%73%43%72%49) e a inserção de scripts no argumento (http://www.siteconfiavel.com.br/search.html?type=<"<<sCripT>alert(document.cookie)</sCripT><"<<sCripT>alert(document.cookie)</sCripT>)).	Binário
Número de Domínios	Verifica o número de domínios encontrados na URL. Certos tipos de ataque utilizam o redirecionamento em cadeia de URLs para redirecionar a vítima a páginas armazenadas em servidores maliciosos. Por exemplo, a URL (www.benignsite.com/redir.php?url=http://www.maliciousssite.com./) tem dois domínios ".com", indicando a presença de outra URL.	Inteiro
Email na URL	Verifica se existe um email na URL (http://username@hotmail.com.fddcol.com, por exemplo). O autor dessa característica [11] afirma que os atacantes tentam inserir endereços de email na URL para simular o acesso a um serviço.	Binário
URL encurtada	Verifica se a URL está encurtada (bit.ly/raCz5i, por exemplo). Vários trabalhos [42, 43, 44] apontam que atacantes tem empregado URL encurtadas para realização de ataques, especialmente no Twitter.	Binário
Spam, *Phishing* e *Malware* SLD	Verifica se o SLD de uma URL é muito frequente em blacklists. Para essas três características, os autores [15] empregaram duas listas de URLs, uma benigna e outra maligna (dividida em URLs de *spam*, *phishing* e *malware*). O SLD de uma dada URL é comparado com ambas as listas e os acertos (comparações positivas) são contados. No fim, é gerada uma razão entre os acertos na lista benigna sobre os acertos nas listas maliciosas. Essa probabilidade é usada para mensurar se uma URL é benigna ou suspeita.	Real

Capítulo 4

Trabalhos Relacionados

Este Capítulo realiza uma breve análise de alguns trabalhos relacionados à detecção de URLs maliciosas. Diferente de outros trabalhos, as pesquisas aqui apresentadas não estão organizadas didaticamente de acordo com o tipo de solução ou análise empregada na abordagem de detecção. Na verdade, os trabalhos discutidos a seguir representam uma série de pesquisas que propuseram novas características extraídas das URLs e utilizaram algumas das técnicas de aprendizagem de máquina, discutidas na Seção 2.3.2, no processo de implementação/validação. Ao final do capítulo é apresentada uma discussão sobre esses trabalhos.

4.1 Trabalhos

4.1.1 Beyond *Blacklists*: Learning to Detect Malicious *Web* Sites from Suspicious URLs

O artigo de Ma et al. [1] descreve uma técnica para identificação de URLs suspeitas em larga escala e online. A ideia é tentar predizer se uma URL associada a um determinado site é ou não maliciosa. Para tanto, os autores categorizaram as características extraídas das URLs como sendo léxicas e baseadas em informações do host (servidor) ligado a URL.

A justificativa para o uso de características léxicas é que URLs de sites maliciosos tendem a "parecer diferente"aos olhos dos usuários que as vêem. Assim, elas permitem capturar, metodicamente, propriedades para fins de classificação e, talvez, permitir inferir padrões de URLs maliciosas. As características léxicas utilizadas no trabalho foram: (1) Comprimento do nome do domínio; (2) Comprimento de toda a URL; (3) Número de pontos (“.”) na URL; (4) Presença de tokens no domínio (delimitado por “.”) e no caminho da URL (delimitada por “/”, “,” , “=” , “?”, “-” e “_”).

Já a razão para o uso de características baseados no host é que sites mal-intencionados podem ser hospedados em serviços de hospedagem pouco respeitáveis, em máquinas que não são provedores de hospedagem convencionais ou por meio de serviços de registro de má reputação. As características baseadas em host utilizadas no trabalho foram: (1) Propriedades do endereço IP; (2) Endereço IP presente em uma *blacklist*; (3) Endereços IP dos registros A, MX ou NS (do DNS) são os mesmos dos sistemas autônomos (AS); (4) Data de registro, atualização e validade do domínio; (5) Quem é o registraste; (6) A entrada no WHOIS está bloqueado; (7) Valor *time-to -live* (TTL) para os registros DNS; (8) Domínio contém as palavras-chave "cliente" ou "servidor"; (9) Domínio é um endereço IP; (10) Existe um registro PTR (DNS) para o servidor de nomes; (11) O registro PTR resolve um dos endereços IP do servidor; (12) Qual o continente/país/cidade o endereço IP pertence; e (13) Qual é a velocidade da conexão de uplink (banda larga, dial-up, entre outros).

Como base nessas características, os autores avaliaram a acurácia de quatro (4) classificadores - Naive Bayes, SVM com kernel linear (SVM-lin), SVM com kernel RBF (SMV-RBF) e regressão logística (RL) - em dados extraídos das bases DMOZ [39] e Yahoo! [45] (benignas) e PhishTank [46] e Spamscatter (malignas). Os autores observaram que os classificadores SVM e RL apresentam menos da metade dos erros encontrados no classificador Naive Bayes, o que não é surpreendente, visto que, os dados modelados não são os mais adequados para o classificador. No final, o classificador RL teve melhor desempenho que os SVMs.

Infelizmente, os resultados disponibilizados pelos autores são confusos e apresentam resultados separados por função/característica.

4.1.2 Binspect: Holistic Analysis and Detection of Malicious *Web* Pages

Em seu artigo, Eshete et al. [13] aborda o projeto, a implementação e a avaliação experimental de um sistema holístico e leve chamado BINSPEC, que aproveita uma combinação de análise estática e emulação minimalista. A análise estática inspeciona artefatos de páginas *Web* sem mostrar a página em um navegador, tais como recursos discriminativos da sequência de URL, identidade do host, código HTML e código JavaScript. O pressuposto desta análise é que a distribuição estatística dos recursos em URLs maliciosas tende a se diferenciar de URLs benignas. Já a emulação minimalista serve para fiscalizar a execução dinâmica de uma URL, podendo ser implantada em ambiente controlado (*Proxy Web*, por exemplo) e assim decidir se é seguro processar a página no navegador do usuário.

As características utilizadas no BINSPECT, baseadas em análise estatística, são divididas em três classes (URL, código fonte e reputação social), o que re-

presenta um total de 39 características. Entre as características relacionadas a URL, os autores afirmam que três (3) são novas: (i) Tamanho do caminho da URL; (ii) Tamanho da consulta na URL (parte do caminho sem o diretório) e (iii) tamanho do arquivo no caminho da URL. Já as características de conteúdo da página somam 25 no total, todas já utilizadas em trabalhos anteriores. Por fim, os autores utilizam três (3) características de reputação social, propostas por eles para avaliar a distribuição estatística da URL quando compartilhada no Facebook, no Twitter e no Google Plus.

No quesito avaliação, os autores utilizaram 71.919 URLs maliciosas, coletadas das blacklists de malware e phishing do Google [47], da base Phishtank [46] e de uma lista de URL de Malware [48]. Também foram utilizadas 414.000 URLs benignas dos sites Alexa [49], Yahoo! [45] e DMOZ [39]. Os autores empregaram sete (7) classificadores J48, Random Tree, Random Forest, Naive Bayes, redes Bayianas, SVM e Regressão Logística.

Para verificar se as novas características são importantes, os autores comparam a precisão da classificação, os falsos positivos e os falsos negativos de todos os classificadores em conjuntos com ou sem as 6 novas características. Como resultado, as novas características de URL melhoraram o desempenho global de 5 dos 7 classificadores (J48, Random Forest, Naive Bayes, redes Bayianas e regressão logística). Já as características de reputação social melhoraram a exatidão de classificação da Random Forest, redes Bayianas e regressão logística. Além da contribuição individual das novas características, os autores também mediram a melhoria global na precisão dos classificadores. De forma resumida, foram obtidos ganhos de precisão em 4 dos 7 classificadores, com melhorias no intervalo de 0,21% a 3,08%.

4.1.3 EINSPECT: Evolution-Guided Analysis and Detection of Malicious Web Pages

Neste artigo, Eshete et al. [50] propõe uma abordagem que faz uso de busca e otimização evolutiva para integrar com modelos de detecção baseados em aprendizagem para uma análise mais precisa de páginas Web maliciosas. Para isso, a abordagem, denominada EINSPECT, inicia com uma população de modelos candidatos treinados usando algoritmos de aprendizagem padrão baseados em características discriminativas extraídos da URL, do código HTML, de código JavaScript e metadados sobre a reputação da página em sites de redes sociais. Em seguida, emprega algoritmos genéticos para automaticamente procurar e otimizar a melhor interação de recursos e algoritmos de aprendizagem. Usando o modelo mais apto, ele detecta páginas Web desconhecidas e as identifica como maliciosas ou benignas. A Figura 4.1 ilustra o funcionamento do EINSPECT.

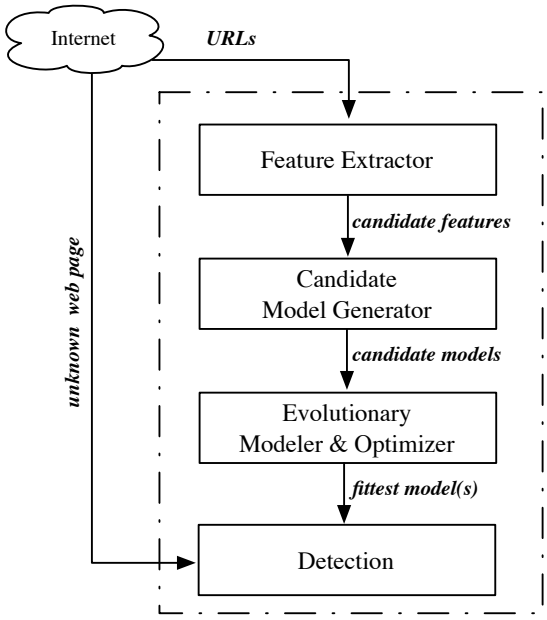

Figura 4.1: Framework do EINSPECT. Fonte: [50]

As características utilizadas no EINSPECT são as mesmas 39 utilizadas no BINSPECT [13], dividas em: URL, código fonte e reputação social. Para avaliação, os autores coletaram 23.791 URLs, sendo 7.221 benignas (das fontes Alexa [49], Yahoo [45] e DMOZ [39]) e 16.570 maliciosas extraídas do Google Safe Browsing [47], PhishTank [46] e MalwareURL [48]. Desse total, 70% foi usado no treinamento e 30% para teste. O classificadores utilizados também foram os mesmos do BINSPECT: J48, Random Tree, Random Forest, Naive Bayes, redes Bayianas, SVM e Regressão Logística.

Como resultado, os melhores classificadores foram Random Forest e J48, usando todas as características. Random Forest obteve 4.9% de falso positivo e 2.7% de falso negativo, enquanto J48 obteve 6.4% de falso positivo e 1.9% de falso negativo. Em relação aos grupos de características, SVM teve o melhor desempenho na URL, com 32.8% de falso positivo e 1.7% de falso negativo, e no código JavaScript, com 38.3% de falso positivo e 11.0% de falso negativo.

Contudo, quando aplicado o algoritmo genético, executado 6 vezes com 20, 30, 40, 60, 80 e 100 gerações, o melhor classificador foi o J48 usando apenas 12 características (4 de URL, 6 de JavaScript e 2 de reputação social) com 96.5% de acurácia, 8.1% de falso positivos e 1.7% de falso negativo. Quando comparado ao resultado sem o algoritmo genético, o falso positivo reduziu de 1.9% para 1.7%, uma queda de 10.5%.

4.1.4 Detecção de *Phishing* em Páginas *Web* Utilizando Técnicas de Aprendizagem de Máquina

Cunha et al. [14] afirmam que construir URLs que aparentam ser legítimas é uma das técnicas que os *phishers* utilizam para convencer os usuários que uma página (URL) é legítima. Assim é impossível identificar, em diversas situações, um site *phishing* apenas olhando para sua URL. Desta forma, os autores propuseram uma metodologia para detecção de *phishing* em páginas *Web* utilizando um conjunto de doze (12) características baseadas na URL, extraídas de bases de dados online e baseadas no conteúdo da página.

Dentre as características empregadas, as mais interessantes envolvem: (i) a utilização da geolocalização da URL, uma vez que a hospedagem de páginas *phishing* se concentra em determinadas regiões do planeta; (ii) Google *PageRank*, visto que as páginas *phishing* tem um curto período de vida, apresentando um *PageRank* muito baixo ou inexistente.

Em relação a avaliação e resultados, foram utilizados os classificadores KNN, SVM e Regressão Logística. Os autores montaram uma base contendo 12.912 páginas, sendo 6.456 amostras de *phishing* e 6.456 de páginas benignas. Como resultado nos diferentes classificadores, os autores observaram que o classificador SVM obteve os melhores valores na classificação de páginas *phishing*, com precisão de 94,20%, taxa de verdadeiros positivos de 97,10%, taxa de falsos positivos de 5,90% e taxa de acerto de 95,60%.

4.1.5 Automatic Classification of Cross-site Scripting in Web Pages Using Document-based and URL-based Features

Nunam et al. [26] apresentam um método de classificação automática de XSS em páginas Web baseado em técnicas de aprendizagem de máquina supervisionadas. Embora não seja um trabalho exclusivamente focado na detecção de URLs maliciosas, e sim de conteúdo Web malicioso, apresenta uma série de características interessantes e extraíveis da URL, bem como faz uso de vários classificadores de aprendizagem de máquina.

O método proposto é organizado em quatro (4) etapas: (i) detecção e extração de características de código ofuscado, decodificação da página Web, extração de características decodificadas e classificação de páginas Web (Figura 4.2).

Dentre as características mais interessantes para detecção de URLs, o trabalho apresenta: (i) Código Ofuscado; (ii) Quantidade de Domínios; (ii) Caracteres Especiais Duplicados (todos explicados na Tabela 3.4. Em relação a avaliação e resultados, foram utilizados os classificadores Naive Bayes e SVM. Os autores

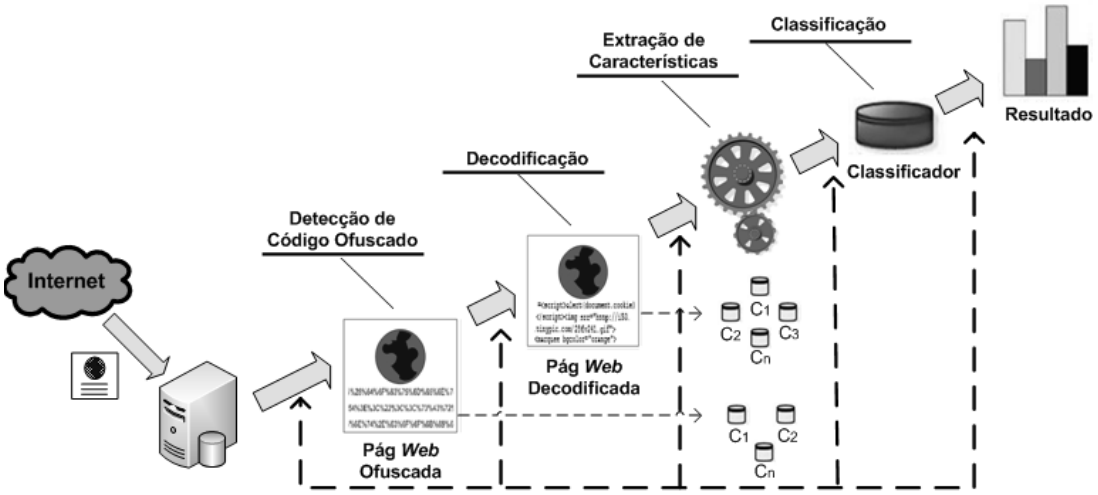

Figura 4.2: Método empregado na classificação automática de XSS. Fonte: [26]

montaram uma base contendo 57.207 páginas benignas da DMOZ [39], 158.847 páginas benignas da ClueWeb09 (http://www.lemurproject.org) e 15.366 páginas infectadas com código XSS da base XSSed, (http://www.xssed.com) referentes a ataques ocorridos de 23 de junho de 2008 a 02 de agosto de 2011.

Como resultado, os autores observaram que o classificador SVM obteve os melhores valores na classificação de páginas XSS, com precisão de 98,58% para base DMOZ/XSSed e 99,89% para base ClueWeb/XSSed. Contudo, os autores comentam que o desempenho do classificador Naive Bayes foi bem próximo ao de SVM, devido ao fato de que as características propostas são aderentes ao conceito de independência condicional, ou seja, o valor de um atributo para uma classe independe dos valores dos outros atributos [26].

4.1.6 Detecting Malicious *Web* Links and Identifying their Attack Types

Choi et al. [15] propõem um método que utiliza a aprendizagem de máquina para detectar se uma URL é maliciosa ou não, classificando-as de acordo com tipos de ataque (*spam*, *phishing* e *malware*). Para tanto, adota um grande conjunto de características discriminativas relacionadas a padrões textuais, estruturas do link, composição do conteúdo, informações de DNS e tráfego de rede. Alguns desses recursos são novos e aparentemente eficazes como consultas de DNS, que além de discriminatórias podem identificar os tipos de ataque. Segundo os autores, a identificação dos tipos de ataque é útil uma vez que o conhecimento da natureza de uma ameaça potencial permite tomar uma reação adequada, bem como uma

pertinente e eficaz medida preventiva contra a ameaça.

Dentre as características léxicas abordadas no trabalho, a que trata da presença de *spam*, *phishing* e *malware* no SLD da URL é a mais inovadora (explicada na Tabela 3.4). Em relação a relacionadas a popularidade do link, os autores propuseram cinco (5) novas características que são indiferentes à manipulação da popularidade por atacantes (explicadas na Tabela 3.1). Os autores também propõem novas características de DNS: (1) Número de endereços IP resolvidos para o domínio da URL; (2) Número de servidores de nomes que servem ao domínio; (3) Número de endereços IP associados aos servidores de nomes; (4) Taxa de ASN maliciosos por IP resolvido; e (5) Taxa de ASN maliciosos por servidor de nome. Essas duas últimas características estão relacionadas ao ASN. Para isso, o método registra os ASN dos endereços IP e dos servidores de nomes resolvidos nas listas de URLs benignas e maliciosas. Além dessas, os autores também utilizam características não aplicáveis a esta proposta, ou porque fogem do escopo ou porque são computacionalmente dispendiosas.

Na avaliação, o método proposto utiliza SVM para classificar a URL como benigna ou não, e os algoritmos *multi-label* RaKEl (*Random k-labelsets*) [51] e ML-KNN (*Multi-label k-Nearest Neighbor*) [52] para identificar os tipos de ataques. No que diz respeito aos resultados, foram avaliadas 72.000 URLs, onde 40.000 foram consideradas benignas e 32.000 maliciosas, com uma precisão de mais de 98% na detecção de URLs maliciosas e uma precisão de mais de 93% na identificação de tipos de ataque. Além disso, estudou-se a eficácia de cada grupo de características discriminativas em ambos, detecção e identificação.

4.1.7 Prophiler: A Fast Filter for the Large-Scale Detection of Malicious Web Pages

A principal forma de reduzir os recursos necessários para a realização de análise em grande escala de páginas Web maliciosas é desenvolver filtros capazes de descartar rapidamente as páginas que são benignas, enviando para ferramentas de análise apenas páginas susceptíveis a conter código malicioso. O trabalho de Canali et al. [23] consiste em construir um filtro, chamado Prophiler, que utiliza técnicas de análise estática para examinar rapidamente uma página Web para averiguar se ela possui ou não conteúdo malicioso. Esta análise leva em consideração características derivadas a partir do conteúdo da página HTML, a partir do código associado JavaScript e a partir da URL. Os autores geraram modelos de detecção que utilizam essas características em técnicas de aprendizado de máquina aplicadas a conjuntos de dados rotulados.

As características avaliadas pelo Prophiler são divididas em duas categorias: conteúdo da página e URL da página (o que engloba características léxicas e do

host). No trabalho, os autores afirmam que além das características já empregadas em outros trabalhos, foram elaboradas mais 48, sendo 19 referentes a códigos HTML, 25 a códigos JavaScript e 4 referentes a URL.

Os algoritmos de aprendizagem de máquina utilizados foram Random Tree, Random Forest, Naive Bayes, Regressão Logística, J48 e redes Baysianas. Todos os classificadores foram testados com bases de treinamento, usando validação cruzada com 10 (dez) partições . Foi empregada uma base de dados contendo 153.115 páginas, sendo 139.321 benignas e 13.794 maliciosas, submetidas ao Wepawet num período de 15 dias. Em média, o Prophiler produziu uma taxa de 10.4% de falsos positivos e 0.54% de falsos negativos, o que na base de validação representa descartar imediatamente 124.906 páginas benignas e poupa recursos no processo de análise.

4.2 Discussão

A discussão feita nesta seção trata dos classificadores. Dos quatro classificadores (SVM, Naive Bayes, Árvore de Decisão - J48 - e KNN) empregados nesta dissertação, pelo menos algum deles é empregado em algum dos trabalhos aqui relacionados, o que torna a escolha por eles acertada.

O SVM e o Naive Bayes são os classificadores mais usados. Os trabalhos de Ma et al. [1], Choi et al. [15], Cunha et al. [14] e Nunan et al. [26] utilizam o SVM, justificando tal escolha pela vantagem do classificador em conseguir lidar bem com grandes conjuntos de dados e possuir um processo de classificação rápido com baixa probabilidade de erros de generalização. O Naive Bayes foi utilizado pelos autores Nunan et al. [26] , Eshete et al. [13], Ma et al. [1] e Canali et al. [23] devido o fato de ser um classificador bastante utilizado em tráfego Internet, além de permitir o uso de em grandes quantidade de dados.

Já a Árvore de Decisão (J48) foi usada por Eshete et al. [13] e Canali et al. [23] com a justificativa de obter regras que explicam claramente o processo de aprendizagem. Por fim, o KNN foi usado por Choi et al. [15] e Cunha et al. [14] devido sua simplicidade e facilidade de implementação.

Além desses classificadores, alguns dos trabalhos relacionados fazem uso de outros classificadores. O primeiro deles é a Regressão Logística (RL), utilizam em Ma et al. [1] , Eshete et al. [13], Cunha et al. [14] e Canali et al. [23]. Em linhas gerais, RL consiste em um modelo paramétrico simples para classificação binária, onde os atributos são classificados de acordo com a sua distância a partir de uma fronteira de decisão do hiperplano. Neste trabalho, esse classificador não foi utilizado, pois é bem adequado para variáveis dependentes, o que não é a realidade na detecção de URLs maliciosas.

O segundo classificador é o Random Forest (RF), usado em [13] e [23]. RF

opera por meio da construção de um grande número de árvores de decisão, durante o período de treinamento, e gera as classes de classificação ou previsão, reajustando o conjunto de treinamento. Uma vez que este trabalho já utiliza um classificador de Árvore de Decisão (J48), o Random Forest não foi avaliado.

O terceiro classificador são as Redes Baysianas. Este classificador, usado por [13] e [23], produz estimativas de probabilidade e ao invés de classificações para cada valor de classe, calcula a probabilidade de uma dada característica pertencer a essa classe. Por ser similar a Naive Bayes, esse classificador não foi utilizado neste trabalho.

O quarto classificador é o CW (*Confidence-Weighted* - Confiança Ponderada), um classificador binário linear que captura a noção de confiança no peso de um recurso, utilizado por Eshete et al [13]. Este classificador também não é adequado uma vez que possui uma percepção das características de forma on-line enquanto neste trabalho a classificação é off-line.

O quinto e último classificador é Random Tree (RT), usado em Canali et al [23], que possui características similares ao Random Forest, mas utiliza somente uma árvore de decisão que pertence a mesma família do algoritmo J48. Por este motivo sua utilização não foi necessária.

Capítulo 5

Implementação e Protocolo Experimental

Este Capítulo descreve, em duas seções, os aspectos essenciais para alcançar o objetivo de investigar a capacidade de validar/classificar URLs como benignas, suspeitas ou maliciosas. A primeira seção descreve as características selecionadas para extração e os mecanismos e ferramentas implementadas para obtenção de seus valores. A segunda seção relata o protocolo experimental necessário (ambiente de teste, base, ajuste nos classificadores, entre outros aspectos) para, efetivamente, validar as características.

5.1 Implementação das Características

Esta seção descreve as características selecionadas para serem extraídas das URLs, já incorporando os aspectos de implementação. Contudo, antes de explicá-las, é preciso descrever um pouco as implementações desenvolvidas. Uma vez que as características variam de acordo com sua funcionalidade, uma série de scripts foi desenvolvida para extração de cada uma das características. Para tanto, a linguagem Perl [53] foi utilizada, pois apresenta simplicidade, portabilidade, versatilidade e a capacidade de lidar com *strings*.

É importante esclarecer que algumas características e suas respectivas implementações necessitam de conexão à Internet, de forma estável, para um correto funcionamento. A seguir as características escolhidas para implementação serão descritas.

5.1.1 Características Léxicas

Nesta dissertação foram escolhidas e implementadas as seguintes características léxicas:

1. **Contagem de Tokens no Domínio, Diretório, Arquivo e Argumento de uma URL**: Os tokens considerados na contagem foram: "/", "**.**", "**,**" , "=" , "**?**", "**-**", "**@**", "**&**", "**_**", "**!**" e "**~**". Todos os componentes de uma URL, denominados segmentos, foram utilizados para a contagem dos tokens (nome de domínio ou FQDN, diretório, arquivo e argumento, conforme descrito em [32]. Sua implementação envolve algumas sub-rotinas simples que utilizam funções Perl e expressões regulares para identificar os segmentos de uma URL e contar a quantidade de cada token predefinido. A saída retornada é um conjunto de variáveis contendo, cada uma, a quantidade encontrada de cada token.

2. **Comprimento do domínio, diretório, arquivo, argumento e URL**: Essa característica refere-se a medição dos segmentos de uma URL. Para tanto, foi utilizada a função *length* do Perl em cada segmento.

3. **Contagem de Parâmetros no Argumento de uma URL**: Essa característica consiste em identificar a quantidade de vezes que o caractere & aparece dentro do argumento de uma URL. Sua implementação emprega funções Perl e expressões regulares e tem como saída um valor inteiro que representa a quantidade de parâmetros.

4. **Contagem de subdiretórios de uma URL**: Essa característica consiste em identificar a quantidade de subdiretórios (separados pelo caracter "/") dentro de uma URL. Sua implementação emprega funções Perl e expressões regulares e tem como saída um valor inteiro que representa a quantidade de diretórios.

5. **Presença de SLD ou FQDN dentro do Argumento**: Não é comum uma URL enviar outra por meio de parâmetros no segmento de argumentos da URL. Nesse sentido, a ocorrência de SLD ou FQDN é dada como atividade suspeita. O propósito dessa característica é verificar a ocorrência de SLD ou FQDN dentro de um argumento da URL.

6. **SPAM SLD, Phishing SLD e Malware SLD**: Essa característica supõe a existência, respectivamente de uma lista de SPAM, Phishing e Malware. Ao se realizar a análise de uma URL será verificado se a mesma se encontra em uma dessas listas. Foi utilizada nesse trabalho a API *Safe Browsing* [47], do Google, para verificação de listas de Phishing e Malware, bem como, o

módulo Net::RBLClient de CPAN2, para SPAM (http://search.cpan.org/~ablum/Net-RBLClient-0.5/RBLClient.pm).

5.1.2 Características de DNS

Nesta dissertação foram escolhidas e implementadas as seguintes características relacionadas ao DNS:

1. **Número de endereços IP resolvidos para o domínio de uma URL**: Essa característica consiste, basicamente, em obter todos os endereços IP associados ao domínio da URL. O retorno é a quantidade de endereços IP encontrados.

2. **Número de servidores de nome que respondem ao domínio de uma URL**: Essa característica consiste em encontrar a quantidade de servidores de nomes (NS) associados ao domínio, sendo seu retorno, essa quantidade.

3. **Tempo de Ativação do Domínio**: Essa característica consiste em obter o tempo (em dias) que um determinado domínio está ativo, uma vez que domínios maliciosos tendem a serem recentes [1]. Para gerar essa característica, foi elaborada uma subrotina que obtém, da lista de endereços IP associados ao domínio, a data de registro do domínio a partir da função "whoisip_query", da ferramenta whois. Essa subrotina usa o módulo "Net::Whois::IP" do Perl. Vale frisar que nem sempre as informações whois fornecem a data de registro e, nesse caso, o retorno será uma string vazia. Já para a obtenção do tempo de ativação, é usada outra subrotina que recebe a data de registro como parâmetro e faz um cálculo a partir da data atual, subtraindo o tempo dessa data atual da data de registro e obtendo, consequentemente, o tempo de ativação em dias.

Essas três (3) características foram elaboradas em diferentes sub-rotinas que utilizam o módulo "Net::DNS::Resolver" do Perl para obtenção dos endereços IP e servidores de nomes. Como mencionado na Taxonomia do Capítulo 3, essas características necessitam de conexão com a Internet para obtenção dos valores.

5.1.3 Características Especiais

Nesta dissertação as características que dependem de serviços Internet ou de conexão a Internet, e não ligadas ao DNS, foram denominadas de características especiais. Foram escolhidas e implementadas as seguintes características especiais:

1. **Presença do Domínio da URL em RBL**: Essa característica permite identificar se um determinado domínio está presente em uma RBL (*Real-time Blackhole List*). Por meio do módulo "Net::RBLClient" do Perl, foi desenvolvida uma subrotina que informa, a partir de um IP obtido da URL, em quantas listas negras o domínio está presente. Dessa forma, sua saída consiste na soma da quantidade de vezes em que cada um dos endereços IP associados aparece em RBLs.

2. **Presença em Listas de Phishing ou Malware**: Essa característica utiliza a API *Safe Browsing* [47] para descobrir se a URL pertence a alguma blacklist de phishing ou malware. Sua implementação usa recursos de socket, a partir do módulo "IO::Socket::SSL", e o método GET do protocolo HTTP. Em linhas gerais, a URL é enviada pela API, processada e a resposta retornada consiste em *mal* ou *phi*, caso seja maliciosa, ou um *OK*, caso seja benigna.

3. **Localização Geográfica do Domínio**: Funciona similarmente à subrotina anterior. A partir da lista de endereços IP associados obtém, para cada um, as informações *whois* referentes à localização geográfica do domínio. Retorna a sigla do país encontrado ou uma string em branco, caso não encontre esse resultado.

4. **Presença de marca**: Essa característica baseia-se em buscar uma marca contida na URL, mas não em seu SLD. Sua implementação é feita com base em uma lista de marcas. Nesta dissertação essa lista foi coletada manualmente da base de *brand names* da empresa Strategic Name Development, Inc. (`http://www.namedevelopment.com/brand-names.html`). Assim, para cada URL analisada é verificada se alguma dessas marcas encontra-se no caminho (diretório, arquivo, argumento) da URL. É importante salientar que não ocorrem buscas dentro do domínio, pois muitas URLs legítimas usam suas próprias marcas em seus domínios (por exemplo, `www.bradesco.com.br`). Por outro lado, marcas muito pequenas (hp, por exemplo) seriam facilmente confundidas (php, por exemplo), gerando falsos positivos. Para evitar esse tipo de erro, expressões regulares são empregadas.

5. **Google PageRank**: Essa característica retorna o Page Rank (do Google) de domínio de uma URL. Na implementação dessa característica, utilizou-se o módulo Perl chamado *www::google::pagerank*, onde o PageRank de um domínio retorna um valor entre 0 e 10, caso exista, e um resultado vazio, quando não é encontrada nenhuma informação. O fato de não existir um

rank, não necessariamente caracteriza uma URL como maliciosa, porém pode ser um indício para uma análise mais detalhada da mesma.

6. **Alexa Ranking**: Essa característica retorna o Ranking global do domínio de uma URL de acordo com o site Alexa.com. Sua implementação consiste em, usando expressões regulares, obter dados do Alexa.com do código HTML dá pagina. Domínios sem ranking global ou com valores muito elevados podem ser considerados suspeitos.

Como mencionado na Taxonomia do Capítulo 3, essas características necessitam de conexão com a Internet para obtenção dos valores. Vale também ressaltar que embora a característica Presença de Marca seja considerada léxica, sua implementação exige um processo de comparação, que dependendo da quantidade de informações a ser comparada, apresenta um tempo superior a uma operação léxica comum. Por isso, ela foi avaliada como característica especial.

A Tabela 5.1 apresenta todas as características implementadas, totalizando 56 artefatos.

5.2 Protocolo Experimental

Para investigar a capacidade de validar/classificar URLs como benignas, suspeitas ou maliciosas é necessária a realização de vários experimentos com os classificadores Naive Bayes, KNN, SVM e Árvore de Decisão empregando todas as 56 (cinquenta e seis) características constantes da Tabela 5.1. Contudo, antes de apresentar esses resultados é preciso descrever o ambiente, as bases de dados, o processo de extração das características, as métricas de avaliação e os ajustes nos classificadores em questão. Esta seção descreve todo o protocolo experimental necessário para se atingir o objetivo proposto.

5.2.1 Ambiente de Experimentação

Os experimentos realizados na elaboração desta dissertação foram executados em duas máquinas. A primeira é um notebook com sistema operacional Windows 7 64 bits, 4 GB de memória RAM, disco de 500 GB e um processador Intel Core i5, 2.3 Ghz. A segunda é uma estação de trabalho Intel Core 7 de 3.4 Ghz, com 8 GB de memória RAM, disco de 500 GB e plataforma Linux, distribuição Ubuntu 14.04. Para a execução dos algoritmos de classificação e análise do conhecimento foi utilizado o ambiente Weka (http://www.cs.waikato.ac.nz/ml/weka/), em sua versão 3.6.10, tanto para Windows quanto Linux.

Tabela 5.1: Características Implementadas

Características Léxicas					
Nome	**Descrição**	**Nome**	**Descrição**	**Nome**	**Descrição**
qt_dom_ponto	Qtde de pontos (.) no domínio	*qt_dom_hifen*	Qtde de hífens (-) no domínio	*qt_dom_underline*	Qtde de underline (_) no domínio
qt_url_ponto	Qtde de pontos (.) na URL	*qt_url_barra*	Qtde de barras (/) na URL	*qt_url_interrog*	Qtde de interrogações (?) na URL
qt_url_igualdade	Qtde de iguais (=) na URL	*qt_url_hifen*	Qtde de hífens (-) na URL	*qt_url_underline*	Qtde de underline (_) na URL
qt_url_arroba	Qtde de arroba (@) na URL	*qt_url_ecomerc*	Qtde de & na URL	*qt_url_exclam*	Qtde de exclamações (!) na URL
qt_url_til	Qtde de til (~) na URL	*comp_dominio*	Comprimento do domínio	*comp_url*	Comprimento da URL
qt_dir_ponto	Qtde de ponto (.) no diretório	*qt_dir_barra*	Qtde de barra (/) no diretório	*qt_dir_interrog*	Qtde de interrogação (?) no diretório
qt_dir_igualdade	Qtde de igual (=) no diretório	*qt_dir_hifen*	Qtde de hífen (-) no diretório	*qt_dir_underline*	Qtde de underline (_) no diretório
qt_dir_arroba	Qtde de arroba (@) no diretório	*qt_dir_exclam*	Qtde de exclamação (!) no diretório	*qt_dir_til*	Qtde de til (~) no diretório
qt_arq_ponto	Qtde de ponto (.) no arquivo	*qt_arq_interrog*	Qtde de interrogação (?) no arquivo	*qt_arq_igualdade*	Qtde de igual (=) no arquivo
qt_arq_hifen	Qtde de hífen (-) no arquivo	*qt_arq_underline*	Qtde de underline (_) no arquivo	*qt_arq_arroba*	Qtde de arroba (@) no arquivo
qt_arq_exclam	Qtde de exclamação (!) no arquivo	*qt_arq_til*	Qtde de til (~) no arquivo	*qt_par_ponto*	Qtde de pontos (.) no parâmetro
qt_par_barra	Qtde de barra (/) no parâmetro	*qt_par_interrog*	Qtde de interrogação (?) no parâmetro	*qt_par_igualdade*	Qtde de igual (=) no parâmetro
qt_par_hifen	Qtde de hífens (-) no parâmetro	*qt_par_underline*	Qtde de underline (_) no parâmetro	*qt_par_arroba*	Qtde de arroba (@) no parâmetro
qt_par_ecomerc	Qtde de & no parâmetro	*qt_par_exclam*	Qtde de exclamação (!) no parâmetro	*qt_par_til*	Qtde de til (~) no parâmetro
qt_params	Qtde de parâmetros na URL	*pres_tld_arg*	Presença de TLD no argumento da URL	*comp_diretorio*	Comprimento do diretório da URL
comp_arquivo	Comprimento do arquivo na URL	*comp_params*	Comprimento dos parâmetros da URL		
Características de DNS					
Nome	**Descrição**	**Nome**	**Descrição**	**Nome**	**Descriçã0**
ip_asssociado	No. de IPs resolvidos	*sn_associado*	No. de servidores de nome resolvidos	*data_tempo_ativo*	Tempo (em dias) de ativação do domínio
Características Especiais					
Nome	**Descrição**	**Nome**	**Descrição**	**Nome**	**Descrição**
mal_phi	Presença em listas de Phishing ou Malware	*presenca_marca*	Presença de marca	*geo_localizacao*	Localização geográfica do domínio
rank_google	Page Rank do Google	*rank_alexa*	Page Rank do Alexa	*rbl_check*	Presença do domínio em RBL (*Real-time Blackhole List*

5.2.2 Base de Dados

Para elaboração dessa dissertação foram utilizadas três (3) bases de dados: DMOZ [39], PhishTank [46] e Shalla‘s Blacklist [54], uma coleção de listas de URL agrupadas em várias categorias destinadas ao uso em filtros de URL. A base DMOZ corresponde a URLs benignas e as bases PhishTank e Shalla's Blacklist correspondem a URLs maliciosas. Para o treinamento e ajuste dos parâmetros dos classificadores, foram utilizadas 20.092 URLs, sendo 10.046 oriundas da base DMOZ e 10.046 da base PhishTank. Já para a etapa de teste, foram utilizadas dois conjuntos de URLs. O primeiro é composto por 20.000 URLs das bases DMOZ e PhishTank, divididas de forma igualitária. O segundo é composto por 20.000 URLs das bases DMOZ e Shalla's, também divididas de forma igualitária.

A Tabela 5.2 apresenta alguns exemplos de URLs de ambas as bases.

Tabela 5.2: Exemplos de URLs das bases DMOZ, PhishTank e Shalla's Blacklist

URL	Base
http://animation.about.com/	DMOZ
http://www.quixium.com/technogirls/brother.htm	DMOZ
http://www.toonhound.com/	DMOZ
http://www.quoddyloop.com/lights.htm	DMOZ
http://www.digitalmediafx.com/Features/animationhistory.html	DMOZ
http://jornaldeuberaba.com.br/Publicidade/	PhishTank
http://riktig-viktnedgang.se/wp-content/Alibaba.html	PhishTank
http://peperonity.com/go/sites/mview/kinaihoroszkop	PhishTank
http://kingdunamis.org/bells/Doc/Google%20Docs.htm	PhishTank
http://nexboved2.pixub.com/online.bncrbank/letred/sys.php	PhishTank
http://memweb.newsguy.com/~twilight/ch.htm	Blacklist
http://h6traiteurservice.be/images/morfeoshow/tafels-7927/BalancePayme.html	Blacklist
http://www.vstecinveste.com/CWB/Coldwell%20Banker/ColdwellBanker/Purchase/index.html	Blacklist
http://macamix.myhealth2u.com/banner/banner1-160X60.gif	Blacklist
http://khits.com/Pics/khits_arbysWorkplace_teaser.jpg	Blacklist

5.2.3 Extração de Características e Classificadores

Como já mencionado, a extração de características é realizada por uma série de scripts em Perl. Os valores obtidos (extraídos) são armazenados em um arquivo texto, salvo no formato CVS (*Comma-Separated Values*) e então submetido aos classificadores. É importante ressaltar que todos os vetores de características foram preenchidos com zero (0) quando informações não puderam ser extraídas da URL. Tal fato ocorre somente para características que dependem de conexão a outros serviços.

Os classificadores selecionados para os experimentos foram: Naive Bayes, KNN, SVM e Árvore de Decisão. A Seção 2.3.2 explica cada um deles.

5.2.4 Medidas de Desempenho

A métrica empregada para a avaliação dos resultados foi a validação cruzada [55] com 10 (dez) partições para os quatro classificadores, mantendo-se a mesma proporção em todos os experimentos a fim de permitir a comparação dos resultados obtidos. As medidas empregadas para a análise de desempenho foram:

1. Taxa de detecção = VP/(VP+FN);
2. Taxa de precisão = (VP+VN) / (VP+VN+FP+FN);
3. Taxa de falso alarme = FP/ (FP+VN).

Onde VN (Verdadeiro Negativo) indica instâncias (URLs) normais classificadas corretamente; FN (Falso Negativo) indica instâncias maliciosas classificadas como normais; FP (Falso Positivo) indica instâncias normais classificadas como maliciosas; e VP (Verdadeiro Positivo) indica instâncias maliciosas classificadas corretamente.

5.2.5 Ajustes dos Classificadores

Para obter o melhor resultado sobre o conjunto de dados para cada classificador, foram realizados treinamentos onde os valores dos principais parâmetros de cada classificador foram ajustados até a obtenção do valor mais adequado. O classificador Naive Bayes foi treinado em sua configuração padrão, tendo em vista que o mesmo não possui parâmetros ajustáveis manualmente.

Classificador SVM

O classificador SVM, além de diferentes funções kernel, apresenta alguns parâmetros que podem ser ajustados durante a fase de treinamento. Dentre eles, o mais importante é o parâmetro de regularização ou penalização, denominado parâmetro C, que determina a rigidez do modelo em relação à tolerância a erros. Quanto maior o valor desse parâmetro, mais rígido e preciso será o modelo, porém mais custoso na fase de treinamento. Por outro lado, quanto menor esse valor, mais tolerante a erros e menos rígida será a margem de separação do hiperplano. Dessa forma, durante o treinamento buscou-se o valor ideal para este parâmetro.

Cabe ressaltar que os resultados apresentados se referem somente ao kernel polinomial de grau 1.0. Outros graus de polinômio e funções kernel, como RBF, por exemplo, foram testados, porém, se mostraram proibitivos por necessitarem de muitas horas de treinamento. Isso se deve basicamente a alguns fatores como: a grande quantidade de dados de treinamento (20.000 instâncias), o uso da técnica de validação cruzada com 10 (dez) partições e a configuração da máquina

empregada nos experimentos. Por outro lado, conforme destaca Karatzoglou et al. [56], a função kernel do tipo linear é a recomendável para conjuntos de dados esparsos, comuns em problemas de categorização de textos.

A Tabela 5.3 mostra o resultado do ajuste de parâmetros para o classificador SVM com função kernel polinomial de grau 1.0.

Tabela 5.3: Ajuste do Parâmetro C para o Classificador SVM

Kernel	**Polinomial Grau 1.0**			
Parâmetro de Regularização (C)	**Taxa de Precisão**	**Taxa de Detecção**	**Falso Alarme**	**Custo de Treinamento (s)**
0,01	89,60%	89,54%	10,50%	0,51
0,1	90,80%	90,80%	9,20%	1,57
1	91,20%	91,23%	8,80%	13,68
2	91,40%	91,37%	8,60%	27,29
3	91,40%	91,39%	8,60%	40,63
4	91,30%	91,34%	8,70%	62,27
5	92,30%	91,33%	8,70%	51,18
6	91,40%	91,36%	8,60%	60,32
7	91,40%	91,37%	8,60%	67,94
8	91,40%	91,38%	8,60%	121,97
9	91,40%	91,38%	8,60%	94,77
10	91,40%	91,42%	8,60%	95,22
20	91,40%	91,39%	8,60%	172,37
30	91,40%	91,42%	8,60%	277,23
40	91,40%	91,41%	8,60%	324,57
50	91,40%	91,43%	8,60%	464,32
60	91,40%	91,43%	8,60%	552,13
70	91,40%	91,42%	8,60%	569,15
80	91,40%	91,41%	8,60%	659,18
90	91,40%	91,42%	8,60%	742,88
100	91,40%	91,42%	8,60%	922,34
200	91,50%	91,45%	8,50%	1853,71
300	91,50%	91,46%	8,50%	2666,29
400	91,50%	91,46%	8,50%	3743,16

Os resultados apresentados na Tabela 5.3 mostram que ao aumentar o valor do parâmetro C, maior a precisão do classificador e maior o custo de treinamento. Ao avaliar o melhor valor de ajuste, percebe-se que o valor para o parâmetro C mais adequado é 50, pois, após este valor, não há aumento significativo na taxa de precisão e detecção que justifique o custo em relação ao tempo de treinamento. A base de treinamento para a realização dos ajustes continha 10.000 instâncias da base DMOZ e 10.000 instâncias da base PhishTank.

Classificador baseado em Árvore de Decisão

Um dos parâmetros ajustáveis em alguns classificadores baseados em árvore de decisão, no nosso caso o J.48 (baseado no algoritmo C4.5), é o Fator de Confiança, que determina a poda de nós descendentes até o nó de decisão, de forma a estabelecer a classe das amostras. Esse fator estabelece a confiança na base de treinamento e na avaliação de erro [27]. Esse parâmetro atua de forma similar ao fator de penalidade no classificador SVM. No caso do classificador baseado em árvore de decisão, quanto menor o valor do Fator de Confiança, maior será a probabilidade do nó ser podado em função dos nós estáveis. A redução do fator de confiança pode reduzir o tamanho da árvore e a quantidade de nós estatisticamente irrelevantes que poderiam levar a erros de classificação [27].

A Tabela 5.4 mostra o resultado do ajuste do Fator de Confiança do classificador J.48.

Tabela 5.4: Ajuste do Fator de Confiança para o Classificador J.48

Fator de Confiança	**Taxa de Precisão**	**Taxa de Detecção**	**Falso Alarme**	**Custo de Treinamento (s)**
0,10	94,70%	94,73%	5,30%	1,15
0,15	95,10%	95,02%	4,90%	1,12
0,20	95,10%	95,08%	4,90%	1,10
0,25	95,10%	95,11%	4,90%	1,07
0,50	95,10%	95,07%	4,90%	1,13
1,00	94,80%	94,84%	5,20%	5,72
1,10	94,80%	94,84%	5,20%	4,91
1,15	94,80%	94,84%	5,20%	4,72
1,20	94,80%	94,84%	5,20%	4,75
1,25	94,80%	94,84%	5,20%	5,73
1,50	94,80%	94,84%	5,20%	5,03

A análise da Tabela 5.4 permite inferir que o melhor ajuste foi para o Fator de Confiança de valor igual a 0.25, que apresenta uma taxa de precisão igual a outros índices, mas apresenta a melhor taxa de detecção (95,11%).

Classificador KNN

A Tabela 5.5 mostra o resultado do ajuste do Fator de Confiança do classificador KNN.

A análise da Tabela 5.5 permite inferir que o melhor ajuste foi para o Fator de Confiança de valor igual 1.

Tabela 5.5: Ajuste do Fator de Confiança para o Classificador KNN

Fator de Confiança	**Taxa de Precisão**	**Taxa de Detecção**	**Falso Alarme**	**Custo de Treinamento (s)**
1	94,90%	94,91%	5,10%	0,01
2	94,20%	93,96%	6,00%	0,00
3	94,40%	94,43%	5,60%	0,00
4	94,20%	94,09%	5,90%	0,01
5	94,60%	94,63%	5,40%	0,00
6	94,40%	94,33%	5,70%	0,00
7	94,40%	94,43%	5,60%	0,00
8	94,20%	94,12%	5,90%	0,00
9	94,20%	94,17%	5,80%	0,00
10	94,10%	94,04%	6,00%	0,00

5.2.6 Escolha do Melhor Classificador

Após o ajuste dos parâmetros foi possível realizar a comparação entre os resultados dos classificadores Naive Bayes, KNN, SVM e Árvore de Decisão (J.48), a fim de determinar o classificador que melhor se ajusta ao conjunto de treinamento.

A Tabela 5.6 apresenta a comparação dos resultados obtidos pelos classificadores Naive Bayes, SVM, Árvore de Decisão e KNN após o ajuste de parâmetros.

Tabela 5.6: Comparação entre os Classificadores

Classificador	**Naive Bayes**	**SVM**	**Árvore de Decisão**	**KNN**
Parâmetros Ajustados	-	C=50, Kernel Polinomial, Grau do Polinômio=1.0	Fator de Confiança=0,25	KNN=1
Taxa de Precisão	76,00%	91,40%	95,10%	94,90%
Taxa de Detecção	66,35%	91,43%	95,11%	94,91%
Falso Alarme	33,60%	8,60%	4,90%	5,10%
Total de Instâncias	20.092	20.092	20.092	20.092

Observando o resultado constante da Tabela 5.6, é possível verificar que o classificador Naive Bayes apresentou uma taxa de precisão muito inferior em comparação aos outros. O mesmo ocorreu com a taxa de detecção obtida, o que reflete uma baixa qualidade na classificação da classe positiva (66,35%), inferindo ao classificador uma baixa capacidade de aprendizagem para o conjunto de dados fornecido. Uma das razões para isso se deve ao tipo de dados processados. O Naive Bayes, em termos gerais, não apresenta bom desempenho para

dados contínuos, lidando melhor com dados discretos. Outro fator que corrobora para o baixo desempenho desse classificador está relacionado aos atributos, que precisam atender à hipótese de independência condicional.

Já os outros três classificadores obtiveram altas taxas de precisão. Entretanto, essa métrica não pode ser avaliada isoladamente, pois o seu resultado expressa o percentual de exemplos classificados corretamente, independentemente da classe a qual pertence; o que pode esconder o erro de classificação da classe minoritária. O classificador SVM apresentou um bom desempenho, obtendo uma taxa de 91,40% de precisão, uma taxa de 91,43% de detecção e uma taxa de 8,60% de falso alarme, mostrando que conseguiu aprender a discriminar as duas classes de forma satisfatória. O mesmo comportamento é visto no classificador KNN, que obteve 94,40% de taxa de precisão, 94,91% de taxa de detecção e 5,10% de falso alarme.

Por fim, o classificador J.48 (Árvore de Decisão) foi o que apresentou o melhor desempenho, obtendo uma taxa de 95,10% de precisão e 95,11% de taxa de detecção, além de um baixo índice de falso alarme (4,90%,), apresentando-se como um classificador que se ajusta bem ao problema pesquisado, considerando o conjunto de dados fornecidos. O sucesso desse classificador, dentre outros fatores, pode estar relacionado ao fato de que as estatísticas obtidas no conjunto de teste se assemelharem às estatísticas do conjunto de treinamento, pois este classificador tem acesso à probabilidade conjunta dos atributos que a árvore considera mais relevante.

Os resultados apresentados nestas análises iniciais ratificam vários trabalhos nessa área que utilizam a aprendizagem de máquina em métodos de detecção de URLs maliciosas. Além disso, corroboram com a relevância das características utilizadas, as quais foram empregadas com classificadores estáveis e de amplo uso na área de aprendizagem de máquina, o que permite inferir que tais características são aderentes e adequadas para a detecção de URLs maliciosas no contexto do conjunto de dados avaliado.

Capítulo 6

Hipóteses e Provas

Como já mencionado várias vezes no decorrer desta Dissertação, o objetivo deste trabalho é investigar a capacidade de validar/classificar URLs como benignas, suspeitas ou maliciosas através da extração de características das URLs e sua análise em diferentes métodos específicos de aprendizagem de máquina. Assim, a partir desse objetivo foram definidas as seguintes hipóteses que precisam de investigação:

- **H1. Existe alguma influência do formato da URL na extração das características e, consequentemente, no processo de avaliação?**
- **H2. Todas as características extraídas são realmente necessárias no processo de detecção de URLs?**
- **H3. Grupos de características permitem resultados adequados, e até melhores, no processo de detecção de URLs se comparados com características individuais**.
- **H4. A importância das características depende da base onde as URLs são coletadas**.

Desta forma, este Capítulo apresenta as provas (experimentos e resultados) para cada uma dessas hipóteses realizadas com os dados (características) extraídos das URLs.

6.1 Provas

6.1.1 Formato da URL

Para avaliar a influência do formato da URL na extração das características (hipótese H1), o primeiro passo é entender como uma URL é composta. Embora

já apresentada na Seção 2.1, uma URL é composta, basicamente, por duas partes: domínio e caminho. Esta última pode ser subdividida em diretório, arquivo e argumento.

É com base nessa possível divisão que a extração das características das URLs, tanto nas bases benignas quanto nas malignas, foi realizada para detecção de URLs maliciosas. Para poder avaliar a hipótese H1, essas características são comparadas. A Tabela 6.1 apresenta as médias de algumas características extraídas nas URLs das duas bases maliciosas PhishTank e Shalla's Blacklist.

Tabela 6.1: Média de algumas Características em bases maliciosas

Características	**Blacklist (Dez. 2014)**	**PhishTank (Nov. 2013)**
comp_url	31,03	54,11
comp_dominio	14,35	19,07
comp_diretorio	6,19	21,66
comp_arquivo	5,38	7,15
comp_params	3,09	6,22
qt_tok_dir_barra	1,75	2,99
sn_assoc	2,49	1,85
rank_alexa	129,06	16.430,63
rank_google	2,08	0,35

Na Tabela nota-se que existe uma certa diferença entre as características das bases. Aquelas ligadas ao tamanho (comprimentos) mostram que a base PhishTank possui URLs maiores e com mais informações a serem extraídas. Tais aspectos se devem ao fato de se tratar de uma base com URLs ligadas a ataques de *phishing*, por isso, por exemplo, o diretório é 3 vezes maior que na base de Blacklist, usada para spam. Essa mesma tendência se repete nas características léxicas que fazem contagem de tokens na URL, domínio, diretório, arquivo e argumento.

Somente na característica *sn_assoc* a base Blacklist tem valor médio maior que na base PhishTank, o que é, na verdade, um ponto positivo, pois significa que as URLs extraídas tem, em média, 2.4 servidores de nomes respondendo pelo domínio, ou seja, é mais confiável que os domínios na base PhishTank. Já características de popularidade como *rank_alexa* e *rank_google* apresentam grandes distorções. Enquanto na base Blacklist o valor médio do *rank_alexa* é de 129, na base PhishTank esse valor é de 16.430,63, o que representa uma diferença de aproximadamente 175%. Vale lembrar que para essa característica, quanto menor o valor, mais conhecida é a URL. No caso de *rank_google*, quanto maior o valor, mais conhecida é a URL.

Desta forma, esta dissertação adota a posição que **o formato da URL possui certa influência na extração de características e, consequentemente,**

no processo de avaliação.

6.1.2 Análise das Características: Individual ou em grupos?

De forma a provar as hipóteses H3 e H4, uma série de experimentos envolvendo os quatro (4) classificadores (SVM, KNN, Naive Bayes e Árvore de Decisão) foi realizada para avaliar a contribuição e o impacto do conjunto de características sobre o resultado final da classificação.

Para tanto, foram gerados três (3) grupos formados por: (i) Características de DNS; (ii) Características Especiais; e (iii) Características Léxicas. Em termos práticos, esses grupos foram separados da seguinte forma para a validação da hipótese:

- **Conjunto A** - Composto pelas 3 características baseadas em informações obtidas do DNS, conforme explicado na Seção 5.1.2;
- **Conjunto B** - Composto pelas 6 características denominadas especiais, que fazem consultas a informações externas sobre domínio, popularidade e presença em blacklist (Seção 5.1.3);
- **Conjunto C** - Composto pelas 15 características léxicas mais comuns [1];
- **Conjunto D** - Composto por 32 características léxicas variáveis (que aparecem em algumas bases de dados, mas não em todas);
- **Conjunto E** - Composto por todas as características léxicas (47 características);
- **Conjunto A+B+C** - Composto pelos 3 grupos de características comuns a qualquer URL, totalizando 24 características.
- **Conjunto A+B+D** - Composto pelos 3 grupos de características comuns, mas considerando o conjunto D (de léxicas variáveis), totalizando 41 características.

É importante enfatizar que foram utilizadas 20.000 URLs, sendo 10.000 da base DMOZ e as outras 10.000 da base PhishTank. Contudo, embora as bases sejam as mesmas da etapa de ajuste de parâmetros, essas URLs foram recolhidas no dia 24 de Novembro de 2014, enquanto as do ajuste foram obtidas em 2013.

[1] Para descobrir essas características, todas as URLs de todas as bases de dados foram avaliadas, incluindo URLs extraídas do Twitter

A Tabela 6.2 apresenta os resultados obtidos individualmente por cada conjunto de características e pela combinação desses conjuntos, através de três tipos de avaliação: Taxa de Precisão (TP), Taxa de Detecção (TD) e Falso Alarme (FA).

Tabela 6.2: Comparação entre Classificadores dos Grupos de Características

Classificadores	**J.48**			**KNN**		
Conjuntos	**TP**	**TD**	**FA**	**TP**	**TD**	**FA**
A	85,50%	85,22%	14,80%	**86,40%**	**85,98%**	14,00%
B	90,00%	89,95%	10,10%	**90,20%**	**90,11%**	9,90%
C	86,60%	86,52%	13,50%	**86,60%**	**86,55%**	13,40%
D	80,20%	79,20%	20,80%	**80,40%**	**79,49%**	20,50%
E	**87,70%**	**87,65%**	12,50%	87,30%	87,20%	12,80%
A+B+C	94,70%	94,67%	5,30%	**95,00%**	**94,97%**	5,00%
A+B+D	94,30%	94,32%	5,70%	**94,80%**	**94,77%**	5,20%
Todos	**95,00%**	**94,99%**	5,00%	94,90%	94,89%	5,10%
Classificadores	**SVM**			**Naive Bayes**		
Conjuntos	**TP**	**TD**	**FA**	**TP**	**TD**	**FA**
A	74,10%	74,10%	25,90%	74,30%	74,23%	25,80%
B	83,20%	82,64%	17,40%	79,50%	71,70%	28,30%
C	77,00%	75,90%	24,10%	73,30%	66,25%	33,70%
D	69,50%	69,22%	30,80%	70,20%	60,94%	39,10%
E	78,40%	77,31%	22,70%	72,80%	64,30%	35,60%
A+B+C	90,00%	89,95%	10,00%	80,00%	73,27%	26,70%
A+B+D	89,80%	89,83%	10,20%	75,30%	64,32%	35,70%
Todos	90,30%	90,30%	9,70%	75,30%	65,86%	34,10%

Comparando-se os resultados apresentados com os da Tabela 5.6 usada na comparação entre os classificadores para o ajuste dos parâmetros, percebe-se ainda que o melhor classificador geral (ou seja, para o grupo com todas as características) é o J.48, seguido pelo KNN. Assim como a Tabela 5.6, os classificadores SVM e Naive Bayes obtiveram os piores resultados.

Contudo, na avaliação individual dos grupos por características, o classificador KNN foi o que apresentou os melhores resultados. Nas métricas de Taxa de Precisão (TP) e Taxa de Detecção (DT), ele obteve 86,40% e 85,98%, respectivamente, para o conjunto A; 90,20% e 90,11% para o conjunto B; 86,60% e 86,55% para o conjunto C e 80,40% e 79,49% para o conjunto D. Além disso, os conjuntos agrupados A+B+C e A+B+D também obtiveram os melhores valores de TP e TD (95,00% e 94,97%; e 94,80% e 94,77%, respectivamente). Já o classificador J.48 obteve os melhores resultados para o conjunto individual E (87,70% e 87,65%) e para o conjunto formado por todas as características (95,00% e 94,99%).

Um aspecto do classificador KNN que pode explicar essa melhor aderência aos dados testados diz respeito a normalização dos atributos. A normalização é usada para evitar que as medidas de distância utilizadas no cálculo do vizinho mais próximo sejam dominadas por um único atributo. E é justamente isso que acontece nesta base de dados, onde todos os atributos são inteiros positivos. Além disso, as características dos conjuntos A e B, intrinsicamente dependentes de serviços Internet (DNS, whois, popularidade), muitas vezes não retornam valores e, assim, são preenchidos com valores zero (0). Só para esclarecer melhor essa influência, as três características do conjunto A, *ip_associado*, *sn_associado* e *data_tempo_ativo*, obtiveram, respectivamente, 612, 4.584 e 7.328 valores sem resposta, substituídos por zero, ou seja, dos 60.000 valores esperados (20.000 de cada característica), 12.524 tiveram o zero atribuído. Para o conjunto B, as seis (6) características tiveram praticamente 1/3 de seus valores totais zerados (39.894 valores de um total de 120.000 esperados foram zerados).

De volta ao foco desta seção que é avaliar as hipóteses H3 e H4, fica claro que comparando as métricas de TP e TD do grupo A+B+C (95,00% e 94,97%) com as métricas de TP e TD do conjunto Total (95,00% e 94,99%), o uso de todas as características não é relevante na obtenção dos melhores resultados. Mesmo que se argumente que os resultados do conjunto A+B+C são do classificador KNN e os do conjunto Total são do J.48, e por isso não devem ser comparados, ao se analisar os mesmos conjuntos (A+B+C e Total) somente entre os mesmos classificadores percebe-se que: no KNN, o conjunto A+B+C é melhor que o Total, e que no J.48, a diferença entre o Total e o A+B+C é de apenas 0,30%.

Assim, pode-se afirmar que a hipótese H3 é falsa e que a hipótese H4 é verdadeira. Em outras palavras, **não é necessário utilizar todas as características no processo de detecção de URLs maliciosas** e **os grupos de características permitem resultados adequados, e até melhores, no processo de detecção de URLs se comparados com características individuais**.

Em relação a hipótese H4, os resultados mostram um melhor ajuste do conjunto A+B+C do que o conjunto Total, o que é semelhante ao exposto no trabalho de Nunan et al. [26] onde os autores propõem um método de classificação automática de XSS em páginas Web, fazendo uso de três (3) agrupamentos de características e mostrando que essa solução é mais eficiente que o uso de todas as características, apresentando ganhos de 1,46% na precisão e 3,69% na detecção.

6.1.3 Diferenças nas bases de dados

Com a função de validar a hipótese H2, dois novos experimentos foram realizados. O primeiro deles avalia a relevância da característica para a análise e o segundo avalia o resultado dos quatro (4) classificadores (com os ajustes definidos na Seção 5.2.5), mas em uma base diferente composta por URLs ligadas a blacklist.

Para avaliar a relevância de cada característica individualmente e assim medir sua qualidade no processo de análise de uma URL, foi empregada a técnica de seleção de atributos, chamada Information Gain, disponível na ferramenta Weka [57] como InfoGain. De modo geral, o Information Gain é um algoritmo de "ranking" fundamentado no conceito de entropia (H) [58]. A ideia é medir a redução na Entropia quando o conjunto de dados é subdividido de acordo com o valor das características. Em outras palavras, qual o resultado geral do conjunto quando uma características é retirada.

No Weka, o InfoGain é independente de classificador. Valores altos no InfoGain significam que a característica é mais adequada no processo de predição. Tipicamente, esses valores variam entre 0 e 1.

A Tabela 6.3 apresenta os resultados do InfoGain aplicado às 24 primeiras características mais relevantes nas bases DMOZ/PhishTank e DMOZ/Blacklist.

Tabela 6.3: Comparação do InfoGain nas Bases DMOZ/PhishTank e DMOZ/Blacklist

DMOZ/PhishTank		**DMOZ/Blacklist**	
Característica	**InfoGain**	**Característica**	**InfoGain**
rank_google	0.390339	*qt_tok_url_barra*	0.70944
data_tempo_ativo	0.348147	*data_tempo_ativo*	0.452587
geo_localizacao	0.228261	*qt_tok_dir_barra*	0.276703
qt_tok_dom_ponto	0.158864	*geo_localizacao*	0.2612
qt_tok_url_barra	0.14602	*qt_tok_dom_ponto*	0.237529
qt_tok_dir_barra	0.118791	*comp_arquivo*	0.233644
comp_diretorio	0.115648	*rank_alexa*	0.180241
comp_url	0.105754	*comp_dominio*	0.176853
qt_tok_url_ponto	0.096124	*ip_assoc*	0.162585
rbl_check	0.082557	*qt_tok_url_ponto*	0.133253
comp_arquivo	0.074864	*rank_google*	0.120826
comp_dominio	0.073538	*presenca_marca*	0.102384
sn_assoc	0.068434	*sn_assoc*	0.067489
presenca_marca	0.056488	*rbl_check*	0.067109
rank_alexa	0.051651	*comp_diretorio*	0.06236
ip_assoc	0.038537	*comp_url*	0.052208
qt_tok_dir_ponto	0.035866	*comp_params*	0.037848
comp_params	0.029835	*qt_tok_arq_ponto*	0.032119
qt_tok_url_hifen	0.029591	*qt_tok_dir_hifen*	0.03101
qt_tok_dir_hifen	0.028234	*qt_tok_url_igualdade*	0.02972
qt_tok_url_ecomerc	0.024787	*qt_tok_par_igualdade*	0.029292
qt_tok_dom_hifen	0.02455	*qt_tok_arq_til*	0.022689
qt_tok_par_ecomerc	0.022983	*qt_tok_url_hifen*	0.018402
qt_params	0.022983	*qt_tok_url_interrog*	0.014041

Comparando-se os resultados do InfoGain para ambas as bases, nas 24 primeiras características, percebe-se claramente que existem grandes diferenças. A primeira delas diz respeito a quais são essas características. Na base DMOZ/PhishTank existem cinco (5) características ($qt\_tok\_dir\_ponto$, $qt\_tok\_url\_ecomerc$, $qt\_tok\_dom\_hifen$, $qt\_tok\_par\_ecomerc$ e qt_params) que não estão presentes no InfoGain da outra base. Neste mesmo ponto de vista, na base DMOZ/Blacklist também existem cinco (5) características ($qt\_tok\_arq\_ponto$, $qt\_tok\_url\_igualdade$, $qt\_tok\_par\_igualdade$, $qt\_tok\_arq\_til$ e $qt\_tok\_url\_interrog$) que não aparecem no InfoGain da primeira base. Em comum, ambas as bases tem 19 características.

O segundo aspecto que as diferencia são os valores (importância) das características. Enquanto na base DMOZ/PhishTank, a característica mais relevante é $rank\_google$ com um valor de 0.390339 (39% de relevância), na base DMOZ/Blacklist, $qt\_tok\_url\_barra$ é a característica mais relevante com 0.70944 (70%). Já o segundo elemento em ambas as bases é $data\_tempo\_ativo$, mas sua relevância é maior na base DMOZ/Blacklist (0.452587) do que na base DMOZ/PhishTank (0.348147). Esse comportamento (maior relevância das características) é visto nos outros elementos da base DMOZ/Blacklist.

Desta forma, esta dissertação segue a posição que **a importância das características depende da base onde as URLs são coletadas**.

Capítulo 7

Conclusões

Esta Dissertação apresentou uma investigação sobre a capacidade de validação/classificação de URLs como benignas, suspeitas ou maliciosas, através de determinadas características extraídas das próprias URLs, empregando técnicas de aprendizagem de máquina.

Primeiramente, um estudo sobre as características extraíves de URLs foi realizado e uma taxonomia foi proposta, visto que nenhuma classificação oficial foi encontrada na revisão bibliográfica. Em função dessa taxonomia, as principais características foram agrupadas e apresentadas. Em seguida, trabalhos relacionados foram apresentados visando mostrar os classificadores mais comuns empregados na classificação de URLs, bem como as características utilizadas.

A extração das características ocorreu em grupos e foi descrita em termos de sua implementação. Também foi apresentado o protocolo experimental envolvendo ajustes nos quatro (4) classificadores utilizados, bases de dados e outros dados.

Para alcançar o objetivo proposto, vários experimentos foram realizados envolvendo URLs maliciosas e legítimas. Neste processo, foi imprescindível a elaboração de quatro (4) hipóteses para investigar a importância das características, de forma individual ou em grupo, a relevância das características e do local (base) de onde as URLs são coletadas.

Todas as hipóteses foram submetidas aos classificadores ou filtros e os resultados provam que:

1. O formato da URL possui certa influência na extração de características e, consequentemente, no processo de avaliação;

2. Não é necessário utilizar todas as características no processo de detecção de URLs maliciosas;

3. Os grupos de características permitem resultados adequados, e até melho-

res, no processo de detecção de URLs se comparados com características individuais;

4. A importância das características depende da base onde as URLs são coletadas.

Além disso, essa Dissertação comprova que a maioria dos trabalhos envolvendo a classificação de URLs podem ter seus resultados questionados, pois não informam dados essenciais como a base, o formato da URL e a importância das características utilizadas no processo de avaliação.

7.1 Trabalhos Futuros

As pesquisas em relação a detecção de URLs maliciosas estão sendo atualizadas constantemente, pois novos experimentos surgem, trazendo para a comunidade científica inovações que aprimoram processos focados na área de Segurança de Rede. Os resultados obtidos nessa investigação podem ser considerados satisfatórios, uma vez que ratificam vários trabalhos.

Contudo, é fato que novos estudos para identificar outras características relevantes ainda podem contribuir para o desenvolvimento de novas abordagens de detecção de páginas Web maliciosas. Os classificadores Naive Bayes, SVM, KNN e Árvore de Decisão foram utilizados neste trabalho e, executados com seus devidos ajustes de parâmetros, foram relevantes, embora devam ser empregados outros classificadores para obtenção de novos resultados.

Além disso, o uso de novas bases de dados de treinamento e pesquisa para identificação de novas características pode exigir a aplicação de experimentos inovadores.

Referências Bibliográficas

[1] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond blacklists: Learning to detect malicious web sites from suspicious urls," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, (New York, NY, USA), pp. 1245–1254, ACM, 2009.

[2] D. Maslennikov and Y. Namestnikov, "Kaspersky security bulletin statistics 2012," 2012. http://www.securelist.com/en/analysis/204792255/Kaspersky_Security_Bulletin_2012_The_overall_statistics_for_2012.

[3] Aaron, Greg and Rasmussen, Rod, "Global phishing survey: Trends and domain name use in 2h2013," 2014. http://docs.apwg.org/reports/APWG_GlobalPhishingSurvey_2H2013.pdf.

[4] Google, "Google safe browsing," 2013. https://developers.google.com/safe-browsing/.

[5] Trendmicro, "Trend micro web reputation service," 2013. http://cloudsecurity-apac.trendmicro.com/solutions-and-services/spn-feature/web-reputation-service.aspx.

[6] Twitter, "Twitter," 2014. http://twitter.com/.

[7] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, "Detecting spammers on twitter," in *Proceedings of the 7th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS)*, 2010.

[8] Garnaeva, Maria and Chebyshev, Victor and Makrushin Denis and Unuchek, Roman and Ivanov, Anton, "Kaspersky security bulletin 2014," 2014. http://securelist.com/analysis/kaspersky-security-bulletin/68010/kaspersky-security-bulletin-2014-overall-statistics-for-2014/.

[9] M. Akiyama, T. Yagi, and M. Itoh, "Searching structural neighborhood of malicious urls to improve blacklisting," in *Proceedings of the 2011*

IEEE/IPSJ International Symposium on Applications and the Internet, SAINT '11, (Washington, DC, USA), pp. 1–10, IEEE Computer Society, 2011.

[10] P. Prakash, M. Kumar, R. R. Kompella, and M. Gupta, "Phishnet: Predictive blacklisting to detect phishing attacks," in *Proceedings of the 29th Conference on Information Communications*, INFOCOM'10, (Piscataway, NJ, USA), pp. 346–350, IEEE Press, 2010.

[11] J. bin Lin, "Anomaly Based Malicious URL Detection in Instant Messaging," Master's thesis, Department of Computer Science and Engineering, National Sun Yat-Sen University, 2008.

[12] J. Zhang, P. Porras, and J. Ullrich, "Highly predictive blacklisting," in *Proceedings of the 17th Conference on Security Symposium*, SS'08, (Berkeley, CA, USA), pp. 107–122, USENIX Association, 2008.

[13] B. Eshete, A. Villafiorita, and K. Weldemariam, "Binspect: Holistic analysis and detection of malicious web pages," in *Security and Privacy in Communication Networks* (A. Keromytis and R. Pietro, eds.), vol. 106 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pp. 149–166, Springer Berlin Heidelberg, 2013.

[14] F. Cunha, E. Santos, and E. Souto, "Detecção de Phishing em Páginas Web Utilizando Técnicas de Aprendizagem de Máquina," in *Workshop de Trabalhos de Iniciação Científica e de Graduação (WTICG), XII Simpósio Brasileiro de SeguranÃ§a da Informação e Sistemas Computacionais (SBSEG 2012)*, 2012.

[15] H. Choi, B. B. Zhu, and H. Lee, "Detecting malicious web links and identifying their attack types," in *Proceedings of the 2Nd USENIX Conference on Web Application Development*, WebApps'11, (Berkeley, CA, USA), pp. 11–11, USENIX Association, 2011.

[16] B. T. Lee, L. Masinter, and M. Mccahill, "RFC 1738: Uniform resource locator (URL)." http://www.ietf.org/rfc/rfc1738.txt, 1994.

[17] E. Alpaydin, *Introduction to Machine Learning*. The MIT Press, 2nd ed., 2010.

[18] T. M. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, Inc., 1 ed., 1997.

[19] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, pp. 15:1–15:58, July 2009.

[20] M. Henke, E. Nunan, C. Santos, E. Souto, E. M. Dos Santos, and E. Feitosa, "Aprendizagem de maquina para seguranca em redes de computadores: Metodos e aplicacoes," in *Livro dos Minicursos do XI Simposio Brasileiro em Seguranca da Informacao e de Sistemas Computacionais* (SBC, ed.), pp. 53–103, SBC, Novembro 2011.

[21] W. L. Buntine, "Operations for learning with graphical models," *Journal of Artificial Intelligence Research*, vol. 2, pp. 159–225, 1994.

[22] L. Kuncheva and Z. Hoare, "Error-dependency relationships for the naÃ¯ve bayes classifier with binary features.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 4, pp. 735–740, 2008.

[23] D. Canali, M. Cova, G. Vigna, and C. Kruegel, "Prophiler: A fast filter for the large-scale detection of malicious web pages," in *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, (New York, NY, USA), pp. 197–206, ACM, 2011.

[24] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Identifying suspicious urls: An application of large-scale online learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, (New York, NY, USA), pp. 681–688, ACM, 2009.

[25] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995.

[26] A. E. Nunan, E. Souto, E. M. dos Santos, and E. Feitosa, "Automatic classification of cross-site scripting in web pages using document-based and url-based features," in *Proceedings of the 2012 IEEE Symposium on Computers and Communications (ISCC)*, ISCC '12, (Washington, DC, USA), pp. 702–707, IEEE Computer Society, 2012.

[27] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[28] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks, 1984.

[29] M. D and A.-S. R, "Malurls: Malicious urls classification system," Annual International Conference on Information Theory and Applications Canning, GSTF Digital Library (GSTF-DL), 2011. The best paper award.

[30] S. Egan and B. Irwin, "An evaluation of lightweight classification methods for identifying malicious urls," in *Information Security South Africa (ISSA), 2011*, pp. 1–6, Aug 2011.

[31] A. B. Sayamber and A. M. Dixit, "On url classification," *International Journal of Computer Trends and Technology (IJCTT)*, vol. 12, pp. 235–241, June 2014.

[32] A. Le, A. Markopoulou, and M. Faloutsos, "Phishdef: Url names say it all," in *INFOCOM*, pp. 191–195, IEEE, 2011.

[33] S. Garera, N. Provos, M. Chew, and A. D. Rubin, "A framework for detection and measurement of phishing attacks," in *Proceedings of the 2007 ACM Workshop on Recurring Malcode*, WORM '07, (New York, NY, USA), pp. 1–8, ACM, 2007.

[34] D. K. McGrath and M. Gupta, "Behind phishing: An examination of phisher modi operandi," in *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, LEET'08, (Berkeley, CA, USA), pp. 4:1–4:8, USENIX Association, 2008.

[35] I. Fette, N. Sadeh, and A. Tomasic, "Learning to detect phishing emails," in *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, (New York, NY, USA), pp. 649–656, ACM, 2007.

[36] Z. Gyöngyi and H. Garcia-Molina, "Web spam taxonomy," in *AIRWeb*, pp. 39–47, 2005.

[37] A. Ramachandran and N. Feamster, "Understanding the network-level behavior of spammers," *SIGCOMM Comput. Commun. Rev.*, vol. 36, pp. 291–302, Aug. 2006.

[38] T. Holz, C. Gorecki, F. Freiling, and K. Rieck, "Detection and mitigation of fast-flux service networks," in *Proceedings of the 15th Annual Network and Distributed System Security Symposium*, 2008.

[39] DMOZ, "Netscape open directory project," 2014. http://www.dmoz.org.

[40] Y. Zhang, J. I. Hong, and L. F. Cranor, "Cantina: A content-based approach to detecting phishing web sites," in *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, (New York, NY, USA), pp. 639–648, ACM, 2007.

[41] CAPEC, "Capec-245: Cross-site scripting using doubled characters," 2010. http://capec.mitre.org/data/definitions/245.html.

[42] D. Wang, S. Navathe, L. Liu, D. Irani, A. Tamersoy, and C. Pu, "Click traffic analysis of short url spam on twitter," in *Collaborative Computing: Networking, Applications and Worksharing (Collaboratecom), 2013 9th International Conference Conference on*, pp. 250–259, Oct 2013.

[43] S. Lee and J. Kim, "Warningbird: A near real-time detection system for suspicious urls in twitter stream," *IEEE Trans. Dependable Secur. Comput.*, vol. 10, pp. 183–195, May 2013.

[44] A. Amleshwaram, N. Reddy, S. Yadav, G. Gu, and C. Yang, "Cats: Characterizing automation of twitter spammers," in *Communication Systems and Networks (COMSNETS), 2013 Fifth International Conference on*, pp. 1–10, Jan 2013.

[45] Yahoo, "Yahoo!," 2014. https://dir.search.yahoo.com.

[46] PhishTank, "Free community site for anti-phishing service," 2014. http://www.phishtank.com.

[47] Google, "Google: Google safe browsing api," 2014. http://code.google.com/apis/.

[48] MalwareURL, "Malwareurl: Malware urls," 2014. http://www.malwareurl.com/.

[49] Amazon, "Alexa," 2014. http://www.alexa.com/topsites.

[50] B. Eshete, A. Villafiorita, K. Weldemariam, and M. Zulkernine, "Einspect: Evolution-guided analysis and detection of malicious web pages," in *Computer Software and Applications Conference (COMPSAC), 2013 IEEE 37th Annual*, pp. 375–380, July 2013.

[51] G. Tsoumakas and I. Vlahavas, "Random k-labelsets: An ensemble method for multilabel classification," in *Proceedings of the 18th European Conference on Machine Learning*, ECML '07, (Berlin, Heidelberg), pp. 406–417, Springer-Verlag, 2007.

[52] M.-L. Zhang and Z.-H. Zhou, "Ml-knn: A lazy learning approach to multi-label learning," *Pattern Recognition*, vol. 40, no. 7, pp. 2038 – 2048, 2007.

[53] E. Raymond, "Book review: The essential perl books," *Linux J.*, vol. 1998, Feb. 1998.

[54] Shalla Secure Services, "Shalla's blacklists," 2014. http://www.shallalist.de.

[55] S. Chakrabarti, *Mining the Web: Discovering Knowledge from HyperText Data*. Science and Technology Books, 2002.

[56] A. Karatzoglou, D. Meyer, and K. Hornik, "Support vector machines in r," *Journal of Statistical Software*, vol. 15, pp. 1–28, 4 2006.

[57] Machine Learning Group at University of Waikato, "Weka 3: Data mining software in java," 2014. http://www.cs.waikato.ac.nz/ml/weka.

[58] C. E. Shannon, "A mathematical theory of communication," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 5, pp. 3–55, Jan. 2001.