Detecção de Canvas Fingerprinting em Páginas Web baseada no Modelo Vetorial

Pablo A. da P. Elleres[1], **Adria M. de Oliveira** [1], **Eduardo L. Feitosa**[1]

[1]pabloelleres14@gmail.com, adriah.menezes@gmail.com, efeitosa@icomp.ufam.ed.br

Abstract. *This paper presents a method for evaluating Canvas Fingerpriting scripts in Web pages. The method consists in calculating the similarity between a database with 100 queries related to Canvas Fingerpriting and databases with pages considered as benign and malignant. The results show high levels of similarities with a canvas base (99%), a base of phishing pages (92%), a base with pages from the DMOZ directory (91%), and a base with pages of Alexa.com (97%).*

Resumo. *Este trabalho apresenta um método para realizar avaliação de scripts Canvas Fingerpriting em páginas Web. O método consiste em realizar o cálculo da similaridade entre uma base com 100 consultas reconhecidamente ligadas à Canvas Fingerpriting e bases de dados com páginas tidas como benignas e malignas. Os resultados mostram altos níveis de similaridades com uma base de Canvas (99%), uma base de páginas phishing (92%), uma base com páginas do diretório DMOZ (91%) e uma base com páginas do site Alexa.com (97%).*

1. Introdução e Motivação

Embora recente, o uso de técnicas de *Website Fingerprinting* tornou-se comum como forma de identificação ou re-identificação de usuários na *Web*. Parte desse sucesso se deve as iniciativas de limitar o uso de cookies, mas sua popularização pode ser atribuída as várias possibilidades (formas e tecnologias) de implementação e execução. Isso porque *Website Fingerprinting* recolhe um conjunto de propriedades de um dispositivo (endereço IP, tamanho da tela, hardware, entre outros) e dos softwares instalados (versões, tipos de fontes, entre outros), a partir de um navegador [Saraiva et al. 2016], para caracterizar um usuário. A questão é que esse "perfil" é obtido tipicamente sem o consentimento dos usuários, o que gera uma série de preocupações. Mesmo que alguns usuários não vejam problemas, a posse de tais informações podem trazer consequências potencialmente desastrosas para a privacidade das pessoas e, em casos mais graves, pode envolvê-las em furtos, fraudes e ataques maliciosos.

Inicialmente, os artefatos de *fingerprinting* exploravam as possibilidades (métodos e propriedades) acessíveis apenas via linguagens como JavaScript e ActionScript (Flash). *Timezone*, *user-agent* e *geolocation* são bons exemplos de propriedades utilizadas nos *fingerprintings* elaborados nessas linguagens. Entretanto, com o tempo, novas tecnologias passaram a ser empregadas para identificar um usuário/dispositivo. É o caso do *fingerprinting* baseado na tecnologia HTML 5 Canvas, que faz uso de propriedades e métodos gráficos. A ideia de um artefato Canvas *fingerpriting* é instruir o navegador a desenhar linhas, textos, figuras geométricas, entre outros, que a posteriori são convertidos em um identificador único.

Mas, se elaborar um *fingerprinting* é relativamente fácil e simples, sua detecção não é. A questão não é, simplesmente, ao encontrar uma característica (um método, uma propriedade ou um atributo) de *fingerprinting* afirmar que o site realiza *Website Fingerprinting*, mas sim quão intrusiva ou perigosa essa característica é. Tomando Canvas como exemplo, o simples fato do site desenhar imagens ou textos não o caracteriza como *fingerprinting*, mas sim a posteriori captura da imagem desenhada e o envio para um servidor [Ximenes et al. 2016].

É neste cenário de incerteza que este artigo propõe um método capaz de detectar e avaliar scripts Canvas *fingerprinting* em páginas Web, com base na extração de características relevantes do conteúdo estático do documento *Web*, por meio de técnicas de recuperação da informação, a fim de apresentar um ranqueamento destes scripts por níveis de similaridade. Desta forma, é possível não só conhecer os scripts originais, mas também o seu grau de periculosidade, informando aos usuários quando acessarem páginas *Web* com este tipo de mecanismo. O método proposto também analisa um conjunto de características relevantes que podem ser extraídas de páginas *Web*, de acordo com sua capacidade de gerar ataques ou ameaças à privacidade dos usuários *Web*, permitindo sua classificação por níveis de similaridade. Esta classificação por nível de similaridade é interessante não só por oportunizar uma comparação entre os scripts, mas principalmente porque poderá servir para uma futura implementação de um plug-in.

Para validar a proposta, uma prova de conceito foi implementada e fez uso de scripts obtidos em bases reais da Internet, como, por exemplo, da Universidade de Princeton (base Canvas), do Phishtank, do Dmoz e do site Alexa.com. Os resultados mostraram altos níveis de similaridade nas bases avaliadas, comprovando que o método criado, baseado no cálculo de similaridade do modelo vetorial, é capaz de detectar Canvas *fingerprinting* em páginas *Web*.

2. Fundamentação Teórica

Esta seção apresenta os conceitos e fundamentos empregados neste trabalho.

2.1. *Website Fingerprinting*

No âmbito *Web*, o termo *fingerprinting* veio a tona em 2009 quando Mayer [Mayer 2009] observou que as características de um navegador e seus *plugins* podiam ser identificados e o usuário rastreado. Em 2010, Eckersley [Eckersley 2010] mostrou que informações (atributos) fornecidas pelo navegador dos usuários eram suficientes para identificar a grande maioria das máquinas que navegam na Internet. Para tanto, desenvolveu um algoritmo para investigar o grau em que os navegadores modernos estão sujeitos às técnicas de *fingerprinting*. Dos mais de 470.000 usuários que participaram de seu projeto público (Panopticlick[1]), 84% tiveram seus navegadores identificados.

Formalmente, é importante salientar a diferença entre os termos *Fingerprint* e *Fingerprinting*. De acordo com a RFC 6973 [Cooper et al. 2013], o primeiro é definido como "um conjunto de elementos de informação que define um dispositivo ou uma instância de uma aplicação" e o segundo como "o processo pelo qual um observador ou atacante identifica, de maneira única e com alta probabilidade, um dispositivo ou uma instância de um aplicativo com base em um conjunto de múltiplas informações".

[1] http://panopticlick.eff.org

Neste artigo, os termos *Website Fingerprinting* e *Fingerprinting* serão usados para representar essas técnicas de identificação com foco na *Web*. Mais detalhes sobre o assunto podem ser encontrados nos trabalhos de [Saraiva et al. 2014] e [Saraiva et al. 2016].

2.2. HTML5 Canvas

Canvas é um elemento da HTML5 que fornece uma área da tela que pode ser utilizada via programação, proporcionando acesso a um conjunto completo de funções de desenho e geração dinâmica de gráficos [W3C 2015].

No que diz respeito a *fingerprint*, um típico script Canvas primeiro desenha um texto com fontes, tamanho e cores de fundo variados [Acar et al. 2014]. Em seguida, chama o método *toDataURL*, da API Canvas, para obter os dados de pixel da tela em formato *DataURL* - uma representação codificada em Base 64 dos dados de pixel binários. Por fim, obtém o *hash* dos dados de pixel codificado de texto, que serve como *fingerprint*. Mowery e Shacham [Mowery and Shacham 2012] observaram que usando Canvas é possível relacionar o navegador com o hardware e o sistema operacional.

2.3. Modelo Vetorial

O modelo vetorial, também conhecido como espaço vetorial, trata documentos e consultas como características num espaço vetorial n-dimensional, sendo a distância vetorial usada como medida de similaridade. Formalmente, um documento d_j e uma consulta de usuário q são representados como vetores com t dimensões. O modelo calcula o grau de similaridade ente o documento e a consulta sob forma de correlação entre os vetores d_j e q, podendo ser medida, por exemplo, pelo cosseno do ângulo entre esses dois vetores [Baeza-Yates 2013].

A Figura 1 ilustra um exemplo do modelo vetorial ao posicionar um documento no espaço vetorial de três dimensões, cada uma representando um índice. Cada documento é visto como termos no espaço n-dimensional através dos eixos (x=4, y=5 e z=3). O ponto de interseção entre estes termos é representado pelo documento DOC = (5, 4, 3).

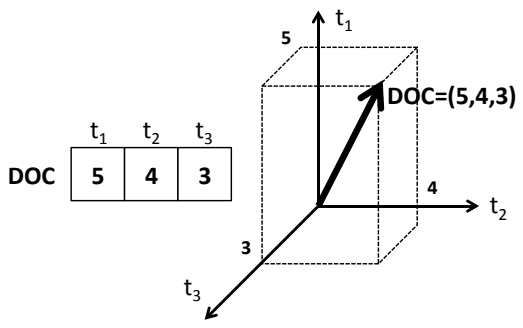

Figura 1. Exemplo de uso do Método Vetorial. Fonte: Adaptado de [Ramiro et al. 2005]

Neste trabalho, um documento é um script Canvas *fingerprinting* qualquer e uma consulta é um dos scripts rotulados como *fingerprinting*.

2.4. Detecção de *Knee Points*

Este trabalho emprega o conceito de detecção de *Knee Points* (em tradução livre, pontos de joelho), definida por Satopää et al. [Satopää et al. 2011], com a finalidade de encontrar um ponto em uma função na qual não haverá tanta variação de resultados e realizar um corte nos resultados de similaridade naqueles em que o algoritmo não considerar relevantes. Satopää et al. [Satopää et al. 2011] destacam que detecção de *Knee Points* é um processo inerentemente heurístico que independe da aplicação, ou seja, em uma definição coerente ele pode ser aplicável a qualquer sistema. Contudo, existe uma grande dificuldade em defini-lo formalmente, pois este pode ser "bom o suficiente" em um sistema, mas pode não ser "bom o suficiente" em outro.

Neste artigo, o emprego de *Knee Points* permite reduzir a quantidade de amostras sem a perda na qualidade dos dados. O exemplo a seguir elucida o uso de *Knee Points* neste trabalho. Ao realizar a comparação entre uma consulta com, por exemplo, 20.000 documentos (scripts), supõem-se que são obtidos mais de 4.000 scripts similares a esta consulta. Neste caso, o algoritmo *Knee Points* verifica um ponto da curvatura da função em que não há tanta representatividade (devido a invariância dos resultados de similaridade) e elimina estas comparações, resultando em, por exemplo, 1.000 scripts similares.

3. Trabalhos Relacionados

Nesta seção são apresentas pesquisas relacionadas ao uso da API Canvas para *fingerprinting* visando a identificação de usuários/dispositivos em páginas Web.

Englehardt e Narayanan [Englehardt and Narayanan 2016] utilizaram a plataforma OpenWP para vasculhar os sites que fazem parte do top 1 milhão da Alexa.com, com o objetivo de analisar quais deles rastreiam seus usuários por *fingerprinting*. Embora tenham testado várias formas de *fingerprinting* (API AudioContext, API Battery, CookieSync e WebRTC), foram detectados 14.371 sites, da base de 1 milhão (cerca de 1,6 %), contendo as propriedades e métodos Canvas como *toDataURL*, *getImageData*, *save* e *restore*.

Laperdrix et al. [Laperdrix et al. 2016] realizaram um estudo sobre os atributos mais relevantes para extrair o *fingerprinting* do dispositivo do usuário e analisar a eficácia da técnica tanto em dispositivos móveis quanto em computadores. Criaram um script de *fingerprinting* que faz uso de 10 atributos utilizados no trabalho de Eckersley [Eckersley 2010] e mais 7 atributos adicionais provenientes da tecnologia Web (Java Script, HTML5 e Canvas), hospedado no site AmIUnique.org. O site registrou 118.934 acessos, o que permitiu aos autores identificar, de maneira única, 89.4% dos dispositivos analisados.

Ximenes et al. [Ximenes et al. 2016] desenvolveram um mecanismo baseado em Canvas para identificar usuários na Web resistente às contramedidas que bloqueiam ou alteram o comportamento do navegador. O TARP *Fingerprinting* implementa três grupos distintos de instruções gráficas Canvas, para permitir diferentes níveis de entropia dos dados coletados. Foram coletadas 64.086 assinaturas (cada assinatura equivale a uma instância do TARP), na qual, por meio do cálculo da entropia, era verificado o uso ou não de contramedidas. Os resultados obtidos provaram a eficácia da solução em burlar as contramedidas e mostraram que quanto maior o número de instruções gráficas empregadas, maior a entropia.

Bursztein et al. [Bursztein et al. 2016] desenvolveram e avaliaram uma técnica capaz de detectar e prevenir ataques automatizados de clientes maliciosos (possivelmente automatizados) em lojas de aplicativos, através de desafios elaborados em Canvas. A intenção foi identificar classes de dispositivos, unindo os atributos do navegador e do sistema operacional. Os desafios baseiam-se em quatro operações de desenhos (primitivas gráficas): *arc()*, *strokeText()*, *bezierCurveTo()*, *quadraticCurveTo()*. Desta forma, os autores realizaram dois experimentos: um em ambiente fechado, contendo 272,198 dispositivos, e outro em ambiente aberto, com a participação anônima de 52 milhões de usuários. Em ambos os casos, a ferramenta conseguiu fazer a distinção das classes dos dispositivos com 100% de acurácia, aplicando-se para todas as combinações de navegadores e sistemas operacionais (sejam estes móveis ou desktops).

Nakibly et al. [Nakibly et al. 2015] apresentaram técnicas de *Device Fingerprinting* baseadas em HTML5 e características de hardware, implementadas no site fingerprintme.herobo.com. O *fingerprinting* proposto realiza três fases de medições (verificar a frequência de relógio, verificar o número de núcleos e verificar outros parâmetros que afetam o desempenho da GPU). Cada fase das medições deu-se por meio da API Canvas *requestAnimationFrame*. Como resultados, os autores foram capazes de identificar 130 dispositivos unicamente, onde 34 destes foram "fingerprintados" mais de uma vez.

Acar et al. [Acar et al. 2014] investigaram os 100.000 sites mais populares do Alexa.com a procura de scripts *fingerprinting*, incluindo os baseados em Canvas. Dentre os métodos utilizados para detectar o Canvas *fingerprinting*, destacam-se *toDataURL*, *fillText* e *strokeText*. O resultado mostrou que dos 100.000 sites analisados, 5.5% utilizam a tecnologia HTML5 Canvas para realizar *Website Fingerprinting*, sendo 95% dos scripts de Canvas pertencentes a um único provedor (*addthis.com*), e os demais 5% restantes pertencem a outros 20 provedores (11 de companhias terceirizadas e 9 desconhecidos).

3.1. Discussão

A Tabela 1 sumariza os trabalhos apresentados, incluindo o método proposto neste artigo.

Tabela 1. Comparação entre Trabalhos com Canvas Fingerprinting

Artigo	Método de Detecção	Propriedades/Métodos Canvas	Fonte de Dados
[Englehardt and Narayanan 2016]	Ferramenta OpenWPM	toDataURL, getImageData, save e restore	OpenWPM
[Laperdrix et al. 2016]		fillText, fillStyle, fillRect, toDataURL, strokeText, globalCompositeOperation, lineTo, arc, canvas.Text, getImageData	AmIUnique
[Ximenes et al. 2016]	Entropia Ponderada e de Shannon	toDataURL	Tarp FP
[Bursztein et al. 2016]	Máxima Entropia	arc, strokeText, bezierCurveTo, quadraticCurveTo, createRadialGradient, shadowBlur, shadowColor	Picasso
[Nakibly et al. 2015]		requestAnimationFrame	fingerprintme
[Acar et al. 2014]		DataURL, fillText, strokeText	alexa.com
Canvas FP no modelo vertorial	Método Vetorial e Knee points	**Propriedades** fillStyle, Canvas.font, textBaseline, Canvas.width, Canvas.height, strokeStyle, globalAlpha, lineWidth, lineCap, lineJoin, miterLimit, shadowOffsetX, shadowOffsetY, shadowBlur, shadowColor, globalCompositeOperation, textAlign e **Métodos** fillRect, fillText, lineTo, arcTo, beginPath, clearRect, createImageData, createPattern, createRadialGradient, measureText, putImageData, quadractiveCurveTo, restore, rotate, scale, setTransform, strokeRect, strokeText, getImageData, toDataURL, getElementById('Canvas'), Canvas.getContext, createElement('Canvas'), getElementsByTagName('Canvas')	Diversas

De modo geral, percebe-se que a maioria das pesquisas apresentadas focam provar a existência do problema, desenvolvendo, em sua maioria, algum site para realizar esta identificação dos usuários/dispositivos. Além disso, a maioria das pesquisas discutidas tem uma abordagem on-line e não faz uso de scripts.

Em relação às características (propriedades e métodos Canvas), percebe-se que algumas são bastante recorrentes, mas outras são mais raras, fato que proporciona uma atenção maior a estas com um número de incidências menor. Por exemplo, as propriedades *toDataURL* e *getImageData*, presentes nos trabalhos de [Englehardt and Narayanan 2016], [Laperdrix et al. 2016], [Ximenes et al. 2016] e [Acar et al. 2014], são elencadas no trabalho de Saraiva [Saraiva et al. 2016] como características de alta periculosidade, por evidenciar a prevalência maior da existência ou não do Canvas *fingerprinting*. Neste trabalho não são utilizadas somente essas 17 propriedades/métodos mencionados, mas sim outras 24 propriedades para uma análise mais abrangente.

Já sobre a classificação e o ranqueamento, é importante salientar que não existem trabalhos que empregam um método para predizer os níveis de similaridade. Tudo leva a crer que este trabalho é o primeiro a fazer isso. Acerca das bases de dados utilizadas, os trabalhos ou utilizam o site Alexa.com ou coletaram seus próprios dados. Já este trabalho emprega 4 bases de dados distintas: Canvas, PhishTank, Alexa.com e DMOZ.

4. Método Proposto

O método proposto objetiva detectar scripts Canvas *fingerprinting* em páginas Web, empregando o modelo vetorial, a partir de um conjunto de características de interesse, para calcular a similaridade dos scripts e ranquea-los. A Figura 2 exemplifica o método proposto, apresentando suas etapas de funcionamento.

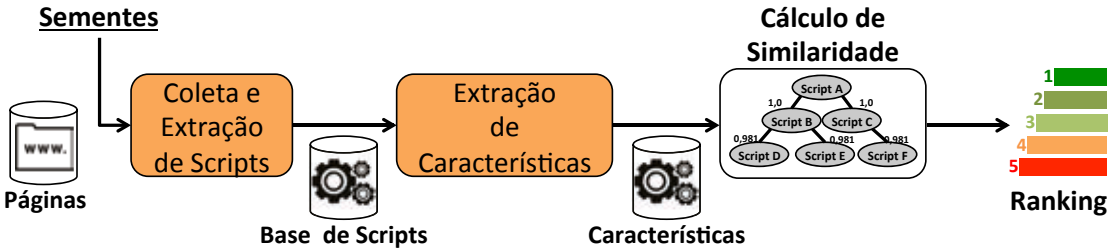

Figura 2. Método Proposto

A etapa de **Coleta e Extração de Scripts** corresponde a coleta e extração dos scripts integrantes do código HTML das páginas Web, tipicamente fazendo uso de um crawler. A etapa seguinte, **Extração das Características**, objetiva identificar os termos (objetos, propriedades e métodos) relacionados à Canvas *fingerprinting* presentes nos scripts coletados na etapa anterior. Para tanto, emprega um dicionário de termos, previamente estabelecido com base na literatura acadêmica, composto por 41 termos.

Na próxima etapa, **Cálculo de Similaridade**, o modelo vetorial calcula o grau de similaridade de uma consulta (propriedades e métodos Canvas *fingerprinting* existentes em cada script) em relação a um documento prévio (da base Canvas, por exemplo) através da similaridade entre cossenos. Para melhorar o entendimento desse processo, a Tabela

2 ilustra o resultado do cálculo de similaridade da consulta `http://2016election.com` em comparação aos 8.000 scripts da base Canvas (documentos).

Tabela 2. Exemplo de Valor de Similaridade na Base Canvas

Consulta	Ranqueamento	Script na Base	Nível de Similaridade
http://2016election.com	1	Pistolsfiringbloy	0.9802
	2	Akdirahost	0.9539
	...	...	...
	324	Fieldgulls	0.4209
	...	...	...
	1724	Creativeplanetnetwork	0.1001
	...	...	...
	7992	Makespace	0.00368
	7993	Profilepic	0.00242

Nesta tabela 2, os scripts da base mais similares a consulta são ordenados de acordo com a similaridade, em ordem decrescente. É necessário mencionar que o método vetorial realiza o cálculo da similaridade até para aqueles scripts que possuem pelo menos um termo igual ao da consulta. Entretanto, só são contabilizados e validados para este método, os valores que estejam entre o intervalo de similaridade de 0.1 até 1.0.

A quarta e última etapa é o **Ranking**, onde é realizado o rankeamento dos scripts cuja similaridade foi calculada na etapa anterior. Em outras palavras, o método vetorial calcula a similaridade e gera um ranqueamento inicial, que o algoritmo *Knee Points* usa para definir um corte e apresentar os mais relevantes. Tomando a consulta usada na Tabela 2 como exemplo (`http://2016election.com`), o método vetorial encontrou 3.724 scripts similares na base de controle e os ranqueou. Em seguida, após a execução do algoritmo *Knee Points* obteve-se uma redução para 324 scripts.

4.1. Detalhes de Implementação

De maneira prática, a implementação do método proposto utiliza:

- Script em linguagem PHP (versão 5.5.15) como Crawler, onde para cada site visitado, são coletados todos os scripts em JavaScript e armazenados em um único arquivo por site;
- Script em linguagem Python 3.4 para extrair as características, por meio de expressões regulares, nos scripts JavaScript. É importante mencionar que ele não é capaz de lidar com scripts ofuscados e mal formados;
- Script em Python 3.4 para realizar tanto o cálculo da similaridade quanto o rankeamento, resultando em um arquivo texto rotulado para cada consulta.

5. Protocolo Experimental

Esta seção descreve o protocolo experimental necessário para avaliação do método proposto.

5.1. Ambiente

Os experimentos realizados foram executados em uma estação de trabalho Intel Core i7 de 2.7 Ghz, com 8 GB de memória RAM e disco SATA de 1 TB de armazenamento sob a plataforma Linux, distribuição Ubuntu 11.10.

5.2. Bases de Dados

Este trabalho empregou quatro (4) bases de dados para realização dos experimentos: Canvas, Phishtank, DMOZ e Alexa. A base **Canvas** é oriunda do trabalho de Englehardt et al. [Englehardt and Narayanan 2016]. Assim, a base foi composta por 8.000 scripts coletados nos meses de agosto a novembro de 2016. A base **Phishtank**[2] é composta por scripts de sites relacionados a *phishing* na Internet. É composta por 2.050 scripts coletados entre novembro e dezembro de 2016.

A base **DMOZ**[3] contém scripts de sites considerados benignos e é composta por 596 scripts. Os scripts da base foram coletados no mês de agosto de 2016. A base **Alexa**[4], da empresa Amazon, é composta por sites ranqueados pela quantidade de usuários que os visitam em determinado período. Foram escolhidos apenas sites do Brasil relacionados a conteúdo adulto, redes sociais e compras, uma vez que Nikiforakis et al. [Nikiforakis et al. 2013] afirma que estes tipos de sites fazem uso de técnicas de *fingerprinting*. A base é composta por 1.478 scripts, coletados no mês de setembro de 2016.

Por fim, foi montada uma base de consultas para realizar a comparação com as bases de dados. Desta forma, foram escolhidos 100 scripts comprovadamente de Canvas, de forma aleatória, do repositório da Universidade de Princeton. Vale ressaltar que embora seja o mesmo local de origem da base Canvas, os 100 scripts não se repetem (não parte da base Canvas).

5.3. Avaliação

Para melhor apresentar os resultados obtidos, foram propostos três (03) cenários de experimentação. O **Cenário 1** mede a similaridade entre os scripts da base de dados e das 100 consultas, quantificando o número de vezes que aparecem no topo do *ranking*. Para exemplificar, considere que o site `http://www.google.com.br` esteja em uma das bases de dados analisadas. O método vetorial, ao fazer a comparação dos scripts desta base de dados com os scripts das 100 consultas, irá retornar que o script do site mencionado apareceu 10 vezes no ranking. A intenção é demonstrar, de maneira generalizada, a quantidade de vezes que os sites (scripts das bases de dados) apareceram no ranqueamento.

O **Cenário 2** reflete o inverso do primeiro. Ele visa demonstrar quais das 100 consultas mais se repetem na comparação com as bases de dados. O **Cenário 3** apresenta o nível de similaridade entre os scripts da base de dados analisada e as 100 consultas. Além do valor da similaridade, calculado pelo modelo vetorial, foi empregada a classificação de risco da ISO 27005 [ISO 2008], onde um nível de risco alto corresponde a uma similaridade entre 80% e 100%, um nível de risco médio corresponde a valores entre 50% e 79% de similaridade e um nível de risco baixo à valores menores à 50%.

6. Resultados

Esta seção explicita os resultados alcançados e subdivide-se de acordo com os cenários.

[2]http://www.phishtank.com/
[3]http://DMOZ.org/
[4]http://www.alexa.com/

6.1. Resultados do Cenário 1

A Tabela 3 apresenta os valores da aplicação dos scripts das bases em relação as 100 consultas, considerando apenas aqueles scripts com mais de 20 repetições no ranqueamento e com similaridade igual à 1.0 (100%).

Tabela 3. Similaridade entre Scripts

Base	Total de Scripts	Scripts	Ranqueamentos
Canvas	8.000	guidingtech, ftlauderdalewebcam, preppyrunner, embassy-pages, nzhuntingandshooting	29
		sandraandwoo, filmhafizasi, dcclothesline, cleverlyinspired	24
		eastcoastcreativeblog	23
Phishtank	2.050	caraudioacapulco	41
		sigarabirakmak	40
		info-setting2016, ricardoeletro2, infobel, maisponto, lagos-statenews	29
		replacementroofingtx, rcacas, davinciresidence	24
DMOZ	596	mypet-memorial	44
		sonicyoga, modern-rocket, mdsafrica	43
		ipanta-rhei	42
		bobthealien, cottonclouds, shesmoke, plum	29
Alexa	1.478	iadulto, caadf, agorams, aiesec, bigshopping	41
		biomedicinapadrao, amofilmeshd, batepapo, cidade-brasil, bussolaescolar	29

É possível notar na Tabela 3 que, para a base Canvas, três grupos de scripts (páginas) apresentaram maior incidência no ranqueamento, obtendo respectivamente 29, 24 e 23 vezes. Em outras palavras, os scripts $guidingtech$, $ftlauderdalewebcam$, $preppyrunner$, $embassypages$ e $nzhuntingandshooting$ apresentaram similaridade de 100% com 29 das 100 consultas. E assim sucessivamente para os outros grupos de script. Na base Phishtank, quatro grupos de scripts foram ranqueados respectivamente, 41, 40, 29 e 23 vezes. Na base DMOZ, quatro grupos de scripts foram ranqueados 44, 43, 42 e 29 vezes, respectivamente. Por fim, na base Alexa, dois grupos se destacaram com 41 e 29 repetições.

Um ponto a ser destacado na Tabela são os valores que as bases Alexa e DMOZ obtiveram. Por se tratar de uma base "benigna", não se esperavam sites ranqueados mais de 10 vezes, mas a DMOZ obteve quatro grupos de scripts com respectivamente, 44, 43, 42 e 29 repetições. Uma explicação para tal fato, um resultado de similaridade tão alto em comparação com as consultas Canvas, tem a ver com o uso das caraterísticas de *fingerprinting* estarem presentes nos mais diversificados tipos de sites, alguns apenas para ajustar conteúdos, mas outros aptos à capturar dados dos usuários que os visitam. Por outro lado, a base Alexa que possui sites cujo conteúdo é mais ligado a *fingerprinting*, apresentou apenas dois grupos de scripts com 41 e 29 repetições, respectivamente.

Por fim, apenas para conhecimentos, os sites listados na Tabela 3 são relacionados a: (i) elaboração de vídeos, notícias, desenhos, filmes, decoração, guia técnico, consulados, entre outros, na base Canvas; (ii) notícias, serviços, design, decoração, e-commerce, entre outros, na base Phishtank; (iii) pet shop, religião/espiritualidade, oferta de serviços audio- visuais e serviço de teste de DNA, na base DMOZ; e (iv) conteúdo adulto, notícias, e-commerce, médico, filmes, bate papo, principais cidades do Brasil e escolar, na base Alexa.

6.2. Resultados do Cenário 2

A Figura 3 ilustra as as 10 consultas mais relevantes para as quatro bases de dados.

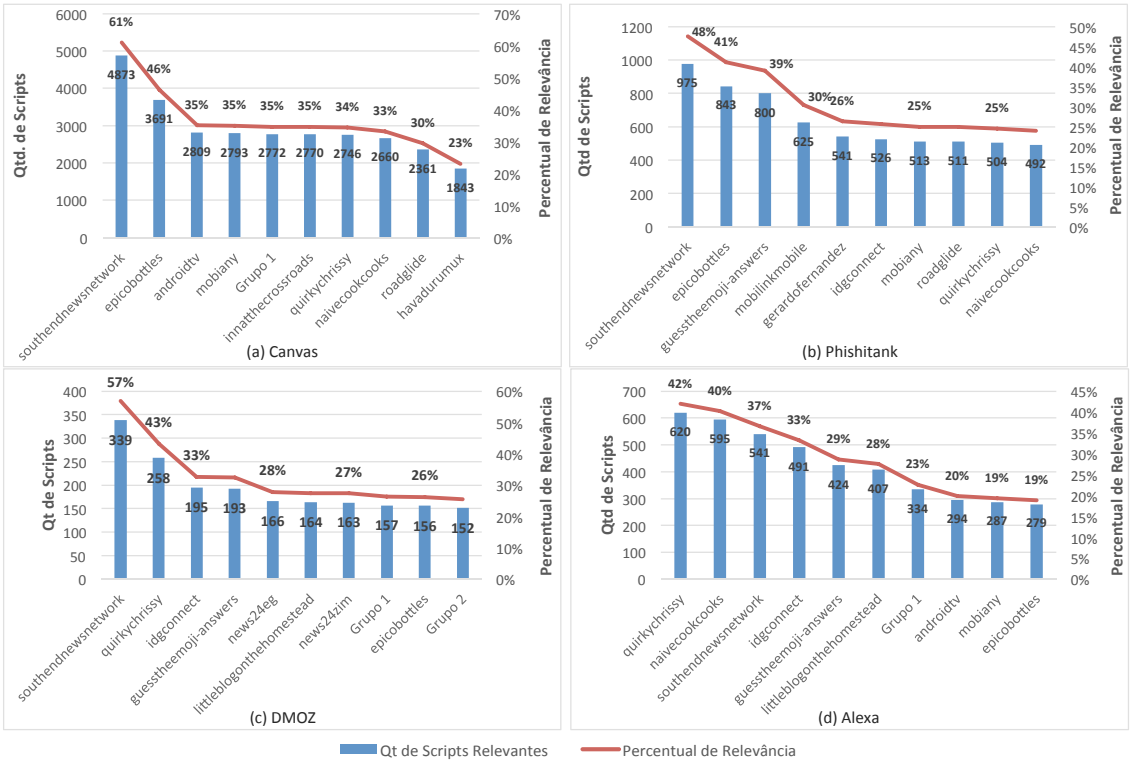

Figura 3. Consultas mais Relevantes

.

Percebe-se na Figura 3 que a base Canvas (a) possui duas consultas (`http://southendnewsnetwork.com` e `http://epicobottles.de`), as quais obtiveram um percentual de relevância de 61% e 46% respectivamente, sendo a primeira relacionada 4.873 scripts da base e a segunda um total de 3.691 scripts. Na base Phishtank, as mesmas consultas são as mais relevantes, tendo percentuais de 48% (para 975 scripts) e 41% (para 843 scripts). Já para a base DMOZ, o script da consulta `http://southendnewsnetwork.com` também foi o mais relevante, obtendo um percentual de 57% com 339 scripts da base, a qual possui um total de 593 scripts. O outro script dessa base é a consulta `http://quirkychrissy.com`, que apresenta um nível percentual de relevância de 43%. Por fim, na base Alexa, o script da consulta `http://quirkychrissy.com` tem resultado bastante similar ao da base de dados DMOZ, com 42% de relevância. E a segunda consulta, script `naivecookcooks`, tem relevância de 40%.

Vale explicar que o grupo 1 presente na Figura 3 (a) representa 40 scripts que obtiveram relevância de 35% em relação as 100 consultas. Os grupos 1 e 2 na Figura 3 (c) representam, respectivamente, 2 e 40 scripts que obtiveram, ambos, relevância de 26% em relação as 100 consultas. Por fim, o grupo 1 na Figura 3 (d) representa 2 scripts que obtiveram relevância de 23% em relação as 100 consultas.

O fato dessas consultas serem tão relevantes para cada uma das bases de dados mencionadas se deve à semelhança entre as características prevalentes entre elas.

Todas possuem em comum características como *fillText*, *textBaseline*, *toDataURL*, *clearRect*, *getImageData*. A consulta `http://southendnewsnetwork.com` possui três características a mais (*scale*, *textAlign*, *restore*). A consulta `http://epicobottles.de` têm apenas *scale* como característica adicional. Para a consulta `http://quirkychrissy.com` existe a mais as características *rotate* e *restore* e, por fim, a consulta `http://naivecookcooks.com` tem somente a característica *restore*. O interessante sobre algumas dessas características é que o trabalho de Saraiva et al. [Saraiva et al. 2016] destaca que duas delas, *getImageData* e *toDataURL*, são classificadas com de alta periculosidade e empregadas em *fingerprinting* maliciosos.

6.3. Resultados do Cenário 3

A função deste experimento é demonstrar os níveis percentuais de similaridade obtidos aplicando-se as 100 consultas no modelo vetorial para cada uma das quatro bases de dados. A Tabela 4 apresenta, de maneira sumarizada, os resultados obtidos nas 4 (quatro) base de dados, elucidando os níveis percentuais de similaridade obtidos em cada uma.

Tabela 4. Nível Percentual de Similaridade

Base	Qt. Sites	Baixo	Médio	Alto
Canvas	8.000	0%	1%	**99%**
Phishtank	2.050	2%	6%	**92%**
DMOZ	596	1%	8%	**91%**
Alexa	1.478	2%	1%	**97%**

Nota-se que todas as bases obtiveram os maiores níveis de scripts de sites com Alta similaridade. Tal fato é explicado pela grande presença de elementos Canvas em páginas *Web* hoje em dia e o fato das 100 consultas serem reconhecidamente ligadas a Canvas *fingerprinting*.

Um resultado não esperado ocorreu com a bases DMOZ. Dita benigna, ela obteve 91% de similaridade no nível alto. Para entender melhor as similaridades obtidas na base DMOZ, um grafo de similaridade foi montado, onde retângulos representam as consultas e os círculos brancos representam os documentos das bases de dados. Foi utilizada uma escala de cores onde vermelho representa similaridade alta (90% à 100%), amarelo representa similaridade média (60% à 89%) e verde similaridade baixa (0% à 59%).

A Figura 4 apresenta parte de um grafo de similaridade, na qual é possível visualizar 4 consultas que se relacionam com os 11 scripts da base de dados DMOZ, todas com similaridade alta variando entre 80% e 100%. Nota-se que a Tabela 5 apresenta os resultados das 4 consultas em relação aos 11 scripts da base DMOZ, destacando o nível de similaridade exposto na figura anterior.

Tabela 5. Similaridade entre 4 Consultas com a Base DMOZ

	Consultas			Consulta		Consulta
Scripts da Base	news24eg	news24zim	Scripts da Base	littleblogonthehomestead	Scripts da Base	idgconnect
pancero	0,928	0,944	kingsford	0,941	cambo	0,918
jnproductions	0,928	0,944	jnproductions	0,939	doublezoot	0,864
smoking-meat	0,923	0,939	kingsford	0,934	manningdigital	0,863
ontariogenomics	0,908	0,924	kamadojim	0,929	kingsford	0,863
atmananda	0,902	0,918	simonsmd	0,927	simonsmd	0,862

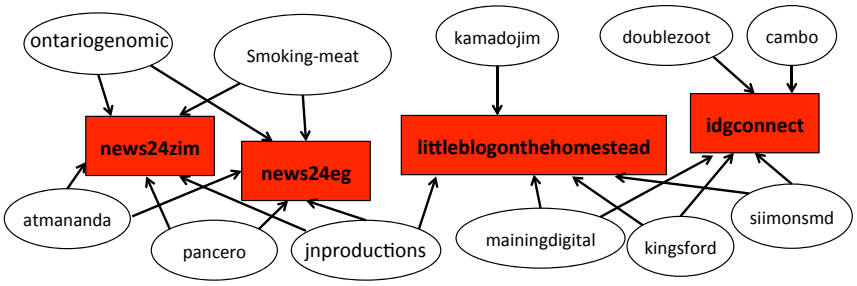

Figura 4. Grafo Parcial do Nível de Similaridade para 4 Consultas.

7. Validação

Para validar o método proposto, utilizou-se o trabalho de Saraiva et al. [Saraiva et al. 2016], que fez uma classificação de severidade de risco de ataques *fingerprinting*. Os autores destacam duas propriedades Canvas como sendo de alta periculosidade: $Canvas.toDataUrl()$ e $Canvas.getImageData()$. O método $Canvas.getImageData()$ retorna um objeto *ImageData* que copia os dados de pixel para o retângulo especificado em área Canvas enquanto o método $Canvas.toDataURL()$ permite obter o conteúdo da tela do navegador. Ambos tem alto risco de obtenção de dados dos usuários *Web*.

A Figura 5 ilustra, para cada base de dados, a ocorrência dessas duas características Canvas de alta periculosidade.

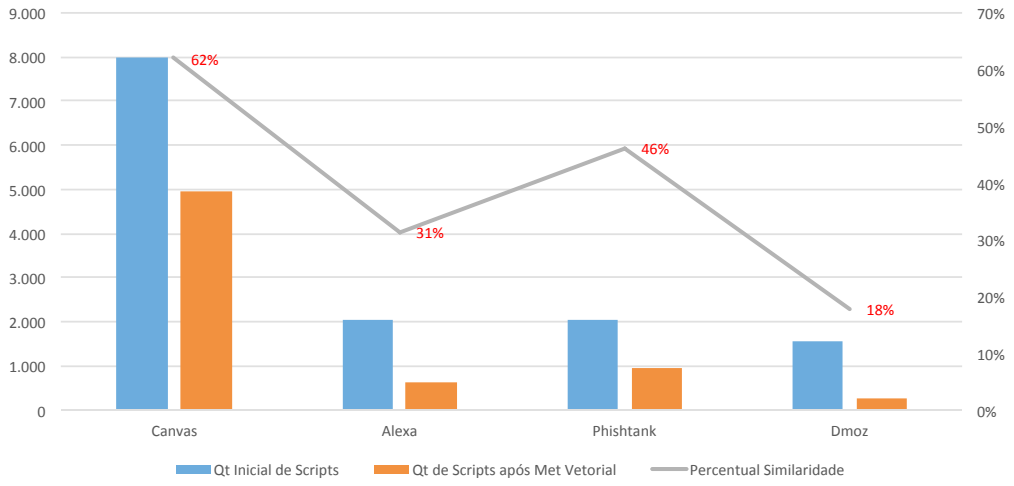

Figura 5. Gráfico do Nível de Similaridade para Duas Características Canvas

.

Os resultados mostram os percentuais de similaridade entre as bases de dados originais (quantidade original de scripts) e a aplicação do modelo vetorial. Após a aplicação do método vetorial, obteve-se 62% de similaridade (utilizando-se apenas as duas características mencionadas no trabalho de [[Saraiva et al. 2016]), com níveis de similaridade variando entre 0.1 e 1.0.

Os valores para as bases Alexa, Phisitank e DMOZ são, respectivamente, 31%, 46% e 18%. Assim, percebe-se que as bases "maliciosas" destacam-se neste critério.

Essa validação serviu para provar que o método proposto é capaz de detectar as características Canvas *fingerprinting* nos scripts das páginas *Web*. Diante do exposto, pode-se afirmar que o método proposto nesta pesquisa, o qual tem a finalidade de ranquear pelo nível de similaridade os scripts das bases de dados, realizando as comparações com consultas Canvas *fingerprinting*, alcançou o objetivo inicial da pesquisa.

8. Considerações Finais

Este artigo propôs um método para analisar *scripts* de Canvas *fingerprinting* em páginas *Web* e informar aos usuários o nível de similaridade entre a página e as consultas que poderão ser prejudiciais a sua privacidade. Deste modo, o conjunto de características (propriedades e métodos) Canvas *fingerprinting* foram avaliados e o nível de similaridade foi calculado.

Para tanto, quatro bases de dados foram averiguadas sendo esta, uma maneira de provar a existência do Canvas *fingerprinting* nos *scripts* das páginas *Web* analisadas. Por meio de testes, em três cenários, os resultados demonstram que há um alto nível de similaridade entre as quatro bases de dados e as consultas Canvas.

8.1. Trabalhos Futuros

A comunidade de segurança no âmbito mundial, frequentemente têm realizado pesquisas sobre *fingerprinting* por meio de novos experimentos, proporcionando não só a sociedade, mas principalmente a comunidade científica inovações tecnológicas. Assim, esta pesquisa elenca alguns trabalhos futuros como forma de retomada do assunto, conforme destaque a seguir:

- Utilizar técnicas de entropia para verificar as características (propriedades e métodos) que possuam uma maior relevância para a detecção de *fingeprinting*;
- Manter e categorizar uma base de scripts para disponibilizar a comunidade acadêmica nos seus estudos referentes ao *fingerprinting*;
- Propor um mecanismo para verificar características de *fingerprinting* ofuscadas em sites *Web*.

Referências

Acar, G., Eubank, C., Englehardt, S., Juarez, M., Narayanan, A., and Diaz, C. (2014). The Web Never Forgets: Persistent Tracking Mechanisms in the Wild. *Proceedings of the 2014 ACM SIGSAC CCS*, pages 674–689.

Baeza-Yates, R. &. R.-N. (2013). *Recuperação de Informação*. Bookman.

Bursztein, E., Malyshev, A., Pietraszek, T., and Thomas, K. (2016). Picasso: Lightweight Device Class Fingerprinting for Web Clients. *Proceedings of the 6th Workshop on Security and Privacy in Smartphones and Mobile Devices*, pages 93–102.

Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and Smith, R. (2013). Privacy Considerations for Internet Protocols. RFC 6973 (Informational). `http://www.ietf.org/rfc/rfc6973.txt`.

Eckersley, P. (2010). How unique is your web browser? In *Proceedings of the 10th International Conference on Privacy Enhancing Technologies*, PETS'10, pages 1–18, Berlin, Heidelberg. Springer-Verlag.

Englehardt, S. and Narayanan, A. (2016). Online Tracking: A 1-million-site Measurement and Analysis. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security - CCS'16*, (1):1388–1401.

ISO (2008). *ISO/IEC 27005: Information technology-Security techniques -Information security risk management*. ISO.

Laperdrix, P., Rudametkin, W., and Baudry, B. (2016). Beauty and the Beast: Diverting Modern Web Browsers to Build Unique Browser Fingerprints. *Proceedings - 2016 IEEE Symposium on Security and Privacy, SP 2016*, pages 878–894.

Mayer, J. R. (2009). Internet Anonymity in the Age of Web 2.0. *Thesis*, pages 1–103.

Mowery, K. and Shacham, H. (2012). Pixel Perfect : Fingerprinting Canvas in HTML5. *Web 2.0 Security & Privacy 20 (W2SP)*, pages 1–12.

Nakibly, G., Shelef, G., and Yudilevich, S. (2015). Hardware Fingerprinting Using HTML5. *Computing Research Repository (CoRR)*, abs/1503.0.

Nikiforakis, N., Kapravelos, a., Joosen, W., Kruegel, C., Piessens, F., and Vigna, G. (2013). Cookieless Monster: Exploring the Ecosystem of Web-Based Device Fingerprinting. *2013 IEEE Symposium on Security and Privacy*, pages 541–555.

Ramiro, T. B., Oliveira, E., Azevedo, L. L., Monteiro, V., and Teixeira, S. (2005). Atribuindo títulos de assuntos na categorização automática de documento. *Congresso Brasileiro de Biblioteconomia, Documentação e Ciência da Informação*.

Saraiva, A. R., de Oliveira, A. M., and Feitosa, E. L. (2016). Determinando o Risco de Fingerprinting em Páginas Web. *Anais do XVI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais - SBSeg 2016*, pages 254–267.

Saraiva, A. R., Elleres, P. A. d. P., Carneiro, G. d. B., and Feitosa, E. L. (2014). Device Fingerprinting: Conceitos e Técnicas, Exemplos e Contramedidas. *Minicursos do XIV Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais - SBSeg 2014*, pages 50–97.

Satopää, V., Albrecht, J., Irwin, D., and Raghavan, B. (2011). Finding a "kneedle"in a haystack: Detecting knee points in system behavior. *Proceedings - International Conference on Distributed Computing Systems*, pages 166–171.

W3C (2015). Html canvas 2d context. `https://www.w3.org/TR/2dcontext/`.

Ximenes, P., Correia, M., Mello, P., Carvalho, F., Franklin, M., and Andrade, R. (2016). TARP Fingerprinting: Um Mecanismo de Browser Fingerprinting Baseado em HTML5 Resistente a Contramedidas. *Anais do XVI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pages 100–113.

