Uma Avaliação das Prevenções de Phishing em Navegadores Web

Carlo M. R. da Silva[1], **Eduardo L. Feitosa**[2], **Vinícius C. Garcia**[1]

[1]CIn – Universidade Federal de Pernambuco (UFPE)

[2]IComp – Universidade Federal do Amazonas (UFAM)

{cmrs,vcg}@cin.ufpe.br, efeitosa@icomp.ufam.edu.br

Abstract. *Web browsers are tools of utmost importance when it comes to data consumption on the Internet, since they enable the interaction and consumption of information provided by various services available over the Web. Several daily companies adapted their services to be used on the Web in order to obtain advantages over competitors. However, it is clear the difficulty of these tools in preventing their users from being victims of phishing attacks. Such attacks if put into effect result in devastating and often irreversible consequences for business. Through this article, we present a study that evidences the lack of effective solutions in analyzing and minimizing such problematic submit to the protection mechanisms designed for the most popular Web browsers available.*

Resumo. *Navegadores Web são ferramentas de extrema importância no que diz respeito ao consumo de dados na internet, pois possibilitam a interação e consumo de informações providas por diversos serviços disponíveis na Web. Diversas empresas cotidianas adequaram seus serviços para serem utilizados na Web no intuito de obter vantagens competitivas entre seus concorrentes. Em contrapartida, é nítida a dificuldade destas ferramentas em evitar que seus usuários sejam vítimas de ataques de* phishing*. Tais ataques quando concretizados resultam em consequências devastadoras, e muitas vezes irreversíveis aos negócios. Através deste artigo, apresentamos um estudo que evidencia a carência de soluções eficazes em analisar e minimizar tal problemática sobre os populares navegadores Web disponíveis para os usuários.*

1. Introdução

A *Web* se consolidou como a plataforma mais eficiente para a distribuição de serviços. Desde seu advento, diversas organizações e segmentos, a exemplo de bancos e lojas virtuais, motivaram-se a migrar suas atividades e funcionalidades para ela, e com o passar dos anos essa tendência ganhou ainda mais força. Contudo, para que essas operações sejam realizadas pela *Web* é necessário, muitas vezes, que sejam fornecidas informações sensíveis de pessoas físicas ou jurídicas, tais como senhas de contas e cartões de crédito, dados cadastrais, entre outros. Com isso, cresce também o número de crimes cibernéticos relacionados à fraudes eletrônicas, em especial *phishing* ou *Phishing Scam*[1], onde atacantes visam interceptar essas informações.

[1]O termo SCAM é uma sigla para *Scheming Crafty Aggressive Malicious*

De acordo com o Cert.BR, *phishing* é o tipo de fraude por meio da qual um golpista tenta obter dados pessoais e financeiros de um usuário, pela utilização combinada de meios técnicos e engenharia social [Cert.BR 2012]. Um ponto importante sobre *phishing* é que sua investida emerge de forma situacional, como, por exemplo, o aumento expressivo de ataques no Brasil devido a liberação de saques do FGTS de contas inativas[2]. Não obstante, o *phishing* também impulsiona o crime cibernético, pois apresenta-se como uma ferramenta para propagação de *malware* na Internet [Khonji et al. 2013]. Um bom exemplo é o *ransonware wannacry*, que gerou problemas em escala global.

Para ilustrar a escala, a Figura 1 registra o número de sites contendo *phishing* confirmados e ainda disponíveis nos últimos 5 anos, no repositório do *PhishTank*[3].

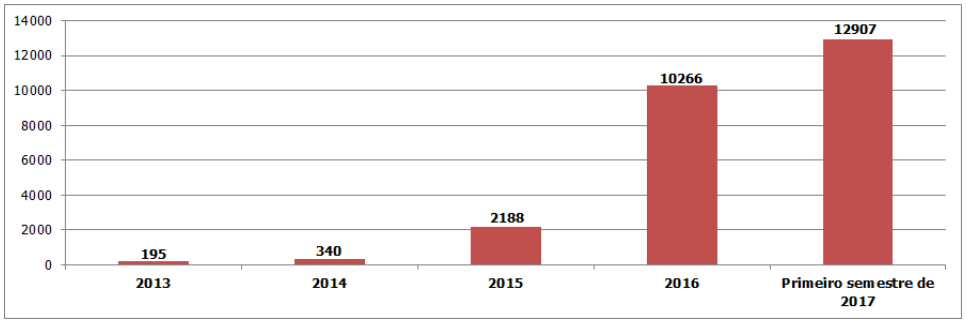

Figura 1. Histórico de registros no *PhishTank* nos ultimos 5 anos

No que tange a detecção de sites com *phishing*, os navegadores *Web* são a primeira linha de defesa e utilizam diferentes mecanismos nativos para identificar e impedir que os usuários caiam em sites maliciosos. A Figura 2 ilustra como esses mecanismos interceptam a ação do usuário, visando impedir o acesso ao *phishing* com base em uma lista negra, sobrepondo a página maliciosa por um página de alerta de perigo.

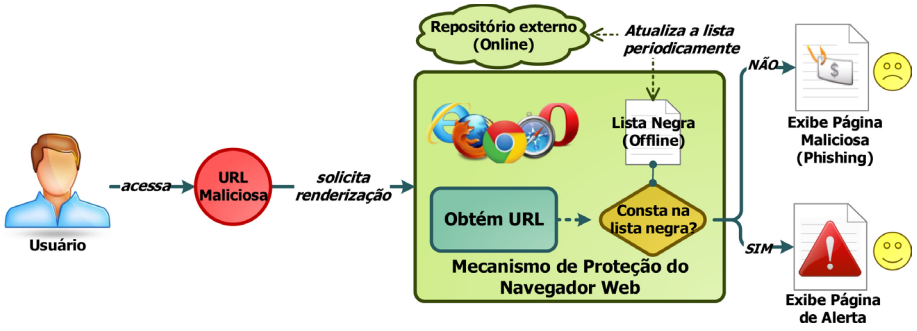

Figura 2. Exemplo do mecanismo de proteção dos navegadores contra *phishing*

Todavia, essas soluções possuem lacunas como, por exemplo, a brevidade na sincronização da lista negra, ou seja, o tempo mínimo de reconhecimento de um *phishing* recém detectado (*zero-day*), além de falhas em recursos do navegador, como *plug-in* e extensões. É neste cenário que este estudo objetiva analisar a eficiência das proteções de *phishing* existentes nos navegadores *Web* mais populares.

[2]Ataques de *phishing* em massa visam roubar informações do FGTS: https://goo.gl/nMS3Wi
[3]PhishTank: http://www.phishtank.com/

Para tanto, uma metodologia, dividida em três perguntas, foi elaborada com o objetivo de **investigar a proteção contra *phishing* dos navegadores *Web***. Para tanto, 16.683 URL maliciosas foram submetidas aos mecanismos de proteção nativos dos navegadores, onde **foi observada a propagação e a semântica das respectivas URL**, considerando também o perfil mais explorado por parte dos criadores de *phishing*, através da análise do padrão de propagação presentes em cada URL.

Este artigo está dividido da seguinte forma: na Seção 2 é descrita a problemática, detalhando o cenário de atuação do *phishing* e trabalhos correlatos ao presente estudo. Na Seção 3 é apresentada a proposta de análise, descrevendo escopo, caminhos e justificativas quanto as decisões levantadas. A Seção 4 apresenta os resultados obtidos com a proposta de análise, incluindo um refinamento dos dados extraídos e interpretações. Por fim, na Seção 5 são apresentadas as considerações finais e perspectivas futuras sobre o desenvolvimento da proposta.

2. Cenário de ataque

De acordo com a *Kaspersky* [Kaspersky 2014], *phishing* é um ataque caracterizado por tentativas fraudulentas contra usuários da Internet. Nesse cerne, o atacante desenvolve uma página falsa que apresenta-se como um ambiente confiável, induzindo suas vítimas à submeterem dados sensíveis, como, por exemplo, o preenchimento de formulários com credencias de acesso a um determinado serviço genuíno [Mohammad et al. 2015].

Conforme ilustrado na Figura 3, tudo se inicia através da engenharia social, que apesar de ter um contexto amplo, é o mecanismo que não se restringe apenas a vulnerabilidades técnicas, uma vez que também explora a confiança de suas vítimas. O atacante faz um estudo clínico do recurso computacional e do fator humano envolvido.

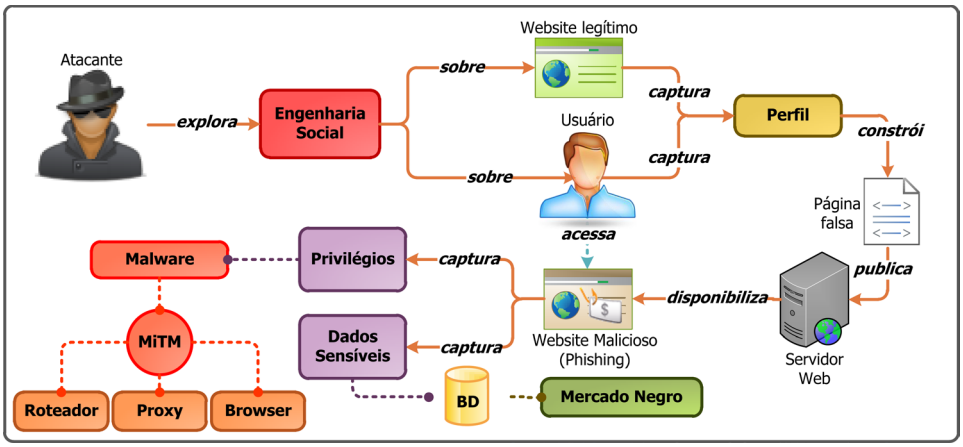

Figura 3. Fluxograma padrão de um *phishing*

Através da engenharia social, o atacante mapeia comportamentos tanto do serviço legítimo (saber que uma determina senha de acesso possui 6 ou 8 dígitos, por exemplo) quanto dos respectivos usuários finais (saber seus interesses e meios de acesso), resultando em uma extração de perfil em cada alvo [Khonji et al. 2013]. Esse perfil se traduz como um conjunto de comportamentos que servem de subsídio para a construção de um site ou aplicativo malicioso. A qualidade desse perfil garante maior similaridade entre o site legítimo e falso, diminuindo as suspeitas de fraude.

Diante disso, o atacante começa a atuar como um "homem do meio", do inglês "*Man in The Middle*" (MiTM), e pode ter objetivos divididos em dois segmentos: uma vez que a vítima caia no golpe, a mesma poderá conceder privilégios de acesso em seu dispositivo, através da execução de um *proxy*, extensão do navegador, redirecionamentos maliciosos no roteador ou demais aplicativos maliciosos [Whittaker et al. 2010]. Também pode usar o computador infectado para propagar o site malicioso entre os contatos da vítima ou obter dados sensíveis, como dados de cartão de crédito, os quais o criminoso possa vender como um produto no mercado negro.

2.1. Novos caminhos e tendências

Ao contrário da exploração de vulnerabilidades em aplicações, a exploração por engenharia social é direcionada ao descuido no fator humano, seja de forma direta, induzindo técnicas que interceptem o fluxo do usuário, ou indireta, em que o atacante analisa vulnerabilidades nos recursos computacionais envolvidos. A seguir são descritas algumas técnicas nesse segmento que exploram tecnologia e o fator humano envolvido.

Exploração por *punycode*: Conforme o RFC 3492 [Costello 2003], o *punycode* é um protocolo que permite a conversão de caracteres com *unicode* específico, como o chines ou russo, em uma versão compatível para nomes de domínios DNS. Essa cadeia de caracteres é sempre precedida do prefixo *"xn–"* e a responsabilidade da conversão é atribuída ao navegador *Web*. Por exemplo, ao utilizar o alfabeto cirílico (alfabeto para linguas eslavas), a palavra *apple* em *unicode* cirílico [Jovanovic 2009] convertida em *punycode* resulta em *xn–80ak6aa92e*, ou seja, uma versão ASCII aceita em nomes DNS. Portanto, ao digitar a URL *xn–80ak6aa92e.com* no navegador *Web* serão exibidos os caracteres *apple.com* na barra de endereço, mas o mesmo será apresentado como um domínio não registrado, ou seja, diferente do domínio oficial *apple.com* em ASCII[4].

Outra maneira de sofisticar o golpe é a possível combinação de *unicode* distintos para resultar em uma cadeia de caracteres idêntica ao site genuíno. O fato é que alguns idiomas, a exemplo do cirílico, não possuem caracteres como *g* ou *l* minúsculos, o que dificulta o infrator forjar um conjunto de caracteres que se assemelhem a um domínio como o *google.com*. O mais aproximado seria algo como *GooGle.com*, porém, suscetível à suspeitas em usuários mais atentos. Contudo, nada impede que o infrator use combinações de *unicode* diferentes, como por exemplo usar o *g* e *l* minúsculos do alfabeto latim básico e os demais caracteres do alfabeto cirílico, resultando na cadeia de caracteres *xn–ggl-tdd6ba.com*, que convertida pelo *punycode*, resultaria em *google.com*[5].

Exploração por Domínio de Topo (*gTLD*): Outro meio de exploração de *phishing* é mascarar URL de domínios enganosos através de domínios genéricos [TrendMicro 2017]. Proposto em 2012 pela ICANN, esse recursos possibilitou que organizações proporcionassem URL amigáveis de acordo com o segmento de atuação, a exemplo de empresas como o Bradesco, que atua no segmento bancário e oferecem endereços *http://bradesco.b.br* ou *http://banco.bradesco*. Esse recurso, apesar dos benefícios, também serve como oportunidade para mal intencionados publicarem URL falsas com modestas variações em comparação ao legítimo, fazendo um jogo de palavras ou trocando letras por números, para que a URL falsa se assemelhe a genuína.

[4]Exemplo extraído da notícia publicada em "The Hackers News": https://goo.gl/USO9Vo

[5]Conversor *online* de *punycode* para ASCII: http://idna-converter.com/

Exploração por *Spear Phishing*: Essa modalidade de ataque usa os mesmos princípios de um ataque de *phishing*, contudo, nesse cenário o golpe ocorre de forma direcionada [TrendMicro 2014]. Diferente do meio convencional, com um escopo mais amplo e disperso, o *spear* phishing direciona em um determinado grupo ou organização. O conteúdo do *phishing* costuma ser mais elaborado, além de fazer uso de táticas inteligentes para torna-se mais convincente a vítima da organização em questão. Em suma, essa modalidade tem um análise bem mais minuciosa sobre a extração de perfis das vítimas consideradas como alvos, trazendo assim maiores possibilidades.

Exploração por política regional e localização geográfica: Outro fator oportuno para os crimes cibernéticos é a diversidade na conduta quanto a punições por crimes digitais que são ocasionadas pelas políticas de um determinado estado ou país [ZDNet 2016]. Diversos servidores que armazenam páginas maliciosas encontram-se geograficamente em países onde não existe quase ou nenhuma legislação que responda por crimes dessa natureza. Isso justifica o fato de existir *phishing* que encontram-se ativos desde 2013.

2.2. Trabalhos Relacionados

Akhawe e Felt [Akhawe and Felt 2013] apresentam uma avaliação empírica sobre a ineficiência dos alertas de seguranças quando os navegadores *Web* notificam seus usuários. A pesquisa é direcionada aos avisos de Malwares e *phishing* que são ou que deveriam ser exibidos durante a navegação, trazendo como conclusão a importância de uma eficiente política de alerta destes incidentes, além de evidenciar um significativo impacto destes alertas ao comportamento do usuário final. Na mesma linha, Mohammad et al. [Mohammad et al. 2015] apresentam uma abordagem sobre os principais métodos de ataques por *phishing*, considerando mecanismos de proteção e detecção de *phishing*.

Mazher et al [Mazher et al. 2013] realizam uma avaliação das barras de ferramentas que podem ser instaladas nos navegadores no intuito de proteger o usuário. Esse tipo de solução vem como uma opção adicional de proteção e, apesar de não serem mecanismos nativos, possuem um certo privilégio sobre a navegação do usuário. Esse viés acaba se tornando uma preocupação, porque ferramentas dessa natureza são desenvolvidas por terceiros, dificultando avaliar com precisão a real intenção dessas ferramentas durante a navegação. Jackson et al [Jackson et al. 2007] apresentam uma avaliação dos mecanismos nativos do navegador sobre a proteção de *phishing* considerando a usabilidade em um estudo de caso que envolveu 27 usuários e 12 sites entre as categorias malicioso ou legítimo. O estudo avalia o certificado https e casos de falsos positivos e foi os primeiros passos para o aperfeiçoamento do Chrome em relação a proteção de *phishing*.

O diferencial do estudo desse artigo em comparação aos demais correlatos é que pouco se aprofundou e discutiu nos trabalhos relacionados sobre o meio de propagação, ciclo de vida e morfologia do *phishing*. A proposta também apresenta um grande número de *phishing* e, por se enveredar em uma análise quantitativa para avaliá-los, investiga os repositórios disponíveis, bem como o mecanismo nativo no navegador, além de levantar padrões sobre esse crime através da análise de propagação.

3. Proposta

Essa Seção descreve a metodologia elaborada para um diagnóstico sobre a eficiência dos principais mecanismos de proteção à *phishing* existentes nos navegadores *Web*. É importante frisar que nesta metodologia não foram considerados mecanismos externos ou

de terceiros, como extensões ou antivírus, apenas os recursos nativos do navegador. A metodologia proposta é baseada em três questionamentos:

- (Q1) Como **obter** um número considerável de *phishing* disponíveis na *Web*?
- (Q2) Como **submeter** esse número de *phishing* aos mecanismos de proteção dos navegadores *Web*?
- (Q3) Como **analisar** o padrão de propagação e semântica de cada *phishing*?

3.1. Q1: Extrair URL maliciosas dos repositórios de *phishing*

Para responder o Q1 foi realizada uma busca por repositórios de *phishing* com as seguintes características: **atualização constante** e **disponível gratuitamente (pública)**. O intuito desses critérios é possibilitar a criação do cenário de avaliação que retrate o mundo real com um número razoável de *phishing*, para ser capaz de responder aos questionamentos com o máximo de precisão. Para o experimento, foram utilizados dois repositórios: *PhishTank*[6] e *OpenPhish*[7]. O *PhishTank* além de atender os requisitos, possui um número expressivo e detalhado de registros em sua base de dados. O repositório *OpenPhish*, embora restrinja o total acervo da sua base através de um acordo comercial, ainda permite baixar uma versão gratuita. Vale destacar que os repositórios do *SafeBrowsing* e *VirusTotal* foram avaliados e descartados por não disponibilizam sua base de registros abertamente.

Conforme ilustrado na Figura 4, é definido o fluxograma **F1**, utilizado pelo estudo para realizar a extração de *phishing* nos repositórios do *PhishTank* e *OpenPhish*. Como primeiro passo é realizar a leitura dos catálogos *offline* disponíveis pelos repositórios, os arquivos *verified_online.json* e *feed.txt*[8], respectivamente do *PhishTank* e do *OpenPhish*. Em seguida, cada URL é lida, observando se a mesma já se encontra cadastrada no banco de dados do cenário da experimentação. Se a mesma já foi cadastrada, observa-se se o *phishing* encontra-se em atividade (ativo e online) e se sofreu algum tipo de modificação através de uma análise pelo *hash* do *response*. Maiores detalhes dessa atividade serão descritos mais adiante nos fluxogramas **F4** e **F5**.

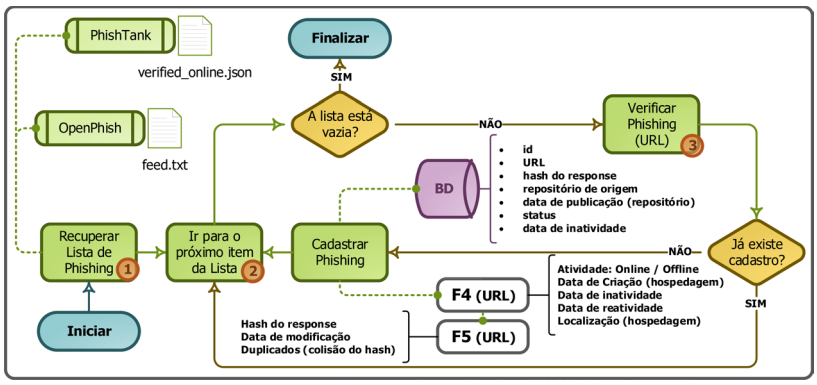

Figura 4. Fluxograma da extração de *phishing* nos repositórios F1

[6]https://www.phishtank.com
[7]https://openphish.com/
[8]*Download* dos arquivos: https://goo.gl/gFCYfJ

3.2. Q2: Avaliar os Mecanismos de proteção dos navegadores *Web*

Para responder o questionamento Q2 foi realizada uma análise sobre as proteções nativas do navegador. Para esse estudo, foi considerado o uso dos seguintes navegadores: Google Chrome, Internet Explorer (IE), Mozilla Firefox, Opera e Safari (**Quais versões**). De acordo com a documentação oficial de cada navegador, foi observado que Chrome, Firefox e Safari compartilham o mesmo mecanismo de proteção, intitulado *SafeBrowsing* API, portanto esses representam um conjunto com mesmo resultado. Já o Opera e o IE terão seus resultados individualizados, pois utilizam, respectivamente, o *PhishTank* API e o *Windows Defender SmartScreen*. Com isso, foi considerado para o testes as APIs do *PhishTank* e *SafeBrowsing*. O fato do IE não ter um mecanismo público foi um problema a ser analisado posteriormente, porém, sanado e descrito na Seção 4.

Conforme ilustrado na Figura 5, o primeiro passo do fluxograma **F2** é garantir a posse e sincronização dos dados extraídos nos repositórios de *phishing*. Ao obter a lista de registros, é verificado se o mesmo encontra-se *online*. Esse passo foi incluído para garantir uma redução de falsos negativos para o caso do *phishing* estar hospedado em um servidor que apresente baixa disponibilidade. Ao identificar como *online*, é observado qual seu repositório de origem, uma vez que ele seja do *OpenPhish*, estará candidato a ser avaliado na API do *PhishTank* e do *SafeBrowsing*, caso contrário, será necessário ser avaliado apenas no *SafeBrowsing*.

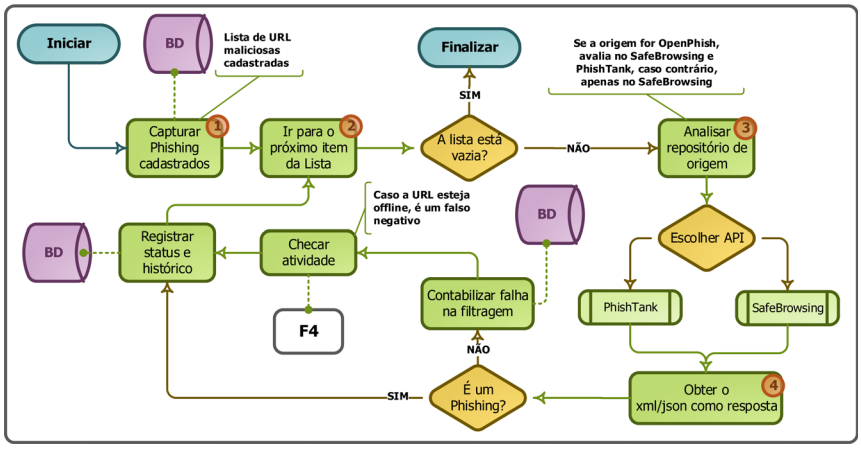

Figura 5. Fluxograma da avaliação da API *PhishTank* e *SafeBrowsing* (F2)

3.3. Q3: Analisar os padrões de propagação e semântica do *phishing*

Por fim, o último passo foi estabelecer uma análise para identificar os padrões de propagação aplicados pelo *phishing*. Como já mencionado na Seção 1, os criadores de *phishing* muitas vezes baseiam-se em determinadas situações eventuais e regionais, como o caso da liberação de FGTS no Brasil. Nesse ponto, o criminoso explora com maior profundidade os perfis mencionados na Figura 3. Para tal objetivo, as URL obtidas no Q1 foram armazenadas em um banco de dados relacional. A justificativa é que posteriormente se fez possível realizar buscas, através de consultas SQL, para a extração de comportamentos com base no conteúdo da URL.

4. Resultados Obtidos

Nessa Seção serão debatidos os resultados obtidos na avaliação proposta pelo estudo.

4.1. Resultados de Q1

Conforme detalhado na Tabela 1, foram coletados 25.896 *phishing* no *PhishTank* e 3.776 no *OpenPhish*, totalizando 29.672 *phishing*. Contudo, para a realização do teste, será considerado o primeiro semestre de 2017 com base na data de confirmação do *phishing* como ameaça, resultando em 12.907 registros de *phishing*. Como o repositório do *OpenPhish*, por ser uma versão gratuita, não informa data de publicação e fornece uma lista limitada de registros, foram considerados todos os 4.122 *phishing* disponibilizados nessa versão. A soma final dos repositórios resultou em 16.683 *phishing* para o experimento. É importante observar que **346** *phishing* estavam presentes em ambos os repositórios, estes foram descartados do *OpenPhish*, reduzindo para 3.776 ocorrências nesse repositório.

Tabela 1. Phishing obtidos nos repositórios

Repositório	**Total de registros**	**Registros de 2017**	**Registros *Online* de 2017**	**Registros *Offline* de 2017**
PhishTank	25.896	12.907	12.390	517
OpenPhish	3.776	3.776	3.748	28
-	**29.672**	**16.683**	**16.138**	**545**

4.2. Resultados de Q2

Não são incomuns os casos em que os repositórios disponibilizam, além de seus registros, uma interface programável (API) para diagnosticar se uma determinada URL trata-se de um *phishing*. Esse tipo de mecanismo favorece cenários de testes da natureza desse estudo, pois representam uma drástica redução de tempo na tarefa de avaliar milhares de URL suspeitas, podendo o trabalho ser realizado em um laço de repetição de um *software*.

A API do *PhishTank* disponibiliza suas consultas sem qualquer tipo de limitação e de forma gratuita. Assim como *PhishTank*, o *SafeBrowsing* também disponibiliza sua API de forma aberta, apenas estabelece uma cota generosa de 12 mil requisições diárias por cada *API key*. Já as APIs do *VirusTotal* e o *OpenPhish* restringem o acesso através de uma licença comercial, o que inviabilizou a utilização das mesmas nesse estudo. O intuito do estudo foi, com base nos registros extraídos em repositórios, avaliar a taxa de acerto de cada API disponível para o teste. O resultado da análise das API com relação a detecção dos *phishing* pode ser observado na Tabela 2. O fato de utilizar *phishing online* (12.390 ao invés de 12.907 no caso do *PhishTank* e 3.748 ao invés de 3.776 no caso do*OpenPhish* é devido aos falsos negativos apresentados pelas API quando checam uma URL *offline*.

Tabela 2. Avaliação das API em confronto aos repositórios

Repositório	**API**	**Total de *phishing***	**Identificados**	**% Acerto**
PhishTank	*SafeBrowsing* API	12.390	7.824	63,15%
OpenPhish	*PhishTank* API	3.748	2.572	68,62%
OpenPhish	*SafeBrowsing* API	3.748	2.265	60,43%

Com esse resultado, teoricamente, era possível concluir que qualquer navegador que utilizasse o *SafeBrowsing*, uma vez submetido aos 12.390 *phishing* (online) do *PhishTank*, também teria uma percentual de acerto de 63,15%, pelo fato da API em questão ser

a mesma adotada pelo seu mecanismo de proteção. Contudo, durante os testes essa teoria não se concretizou devido a uma série de divergências identificadas no experimento.

Primeiramente, foi observado que uma URL na API do *PhishTank* não era reconhecida como *phishing*, mas quando aberta no Opera, era reconhecida como maliciosa. O mesmo ocorreu entre o *SafeBrowsing* e o *Firefox*. O que explica essa diferença de comportamento entre a API e o navegador é a utilização da lista negra adotada pelo navegador, conforme já ilustrado na Figura 1. Por exemplo, o Opera armazena uma lista negra com as URL maliciosas. Periodicamente, o navegador consulta na API do *PhishTank* por novas URL a serem inseridas na lista. Uma vez atualizada, o navegador fará uso da lista para realizar a filtragem, esteja a máquina do usuário *online* ou *offline*.

Já no caso da API, a mesma, aparentemente, confere se o *phishing* está disponível (online), caso não esteja, não é considerado como uma ameaça. Tal comportamento justifica a divergência nos resultados dos testes entre testar diretamente na API e no navegador. Outra questão é a divergência entre o filtro de cada API. Durante o estudo, certas URL foram reconhecidas como *phishing* pela API do *PhishTank*, mas na API do *SafeBrowsing* não eram identificadas [9]. Outro fator é que as API não tratam situações onde URL maliciosas são encapsuladas por URL curtas, já no navegador essas URL seriam detectadas quando o site encurtador redirecionasse o usuário para o destino.

Por fim, também foi observado que algumas URL tinham comportamentos distintos entre o *Chrome*, *Firefox* e *Safari*, mesmo estes compartilhando um único mecanismo de proteção, sugerindo uma possível proteção adicional em cada navegador. O motivo desse comportamento não pode ser identificado por esse experimento, mas diante essas observações, o estudo motivou o desenvolvimento de uma avaliação adicional que considerou a checagem diretamente no navegador. Outro fato que reforça essa decisão é que a proteção do IE, a *SmartScreen*, não disponibiliza uma API externa para consultas nem tampouco um repositório de *phishing* registrados.

Para tanto, foi proposto um novo fluxograma **F3** de coleta de informações, conforme ilustrado na Figura 6. Ele possui um fluxo bem similar ao fluxograma F2, contudo, ao identificar um *phishing* como *online*, é criada uma instância de um determinado navegador e o fluxo captura o corpo da página carregada durante o acesso ao *phishing*. Diferentemente do teste com as API, pelo fato do navegador filtrar um *phishing* independente do mesmo estar *online* ou *offline*, os *phishing offline* também foram considerados no fluxograma F3, checando um total de 12.907 para os testes confrontados com o repositório *PhishTank* e 3.776 quando confrontados com o *OpenPhish*.

Considerando que o cenário irá acessar milhares de *phishing* simultaneamente, foi preciso estabelecer um cenário de testes automatizados. A ferramenta escolhida para proporcionar essa automação foi a *Selenium Server*[10], que oferece automação de testes diretamente no navegador. Também foi utilizada a biblioteca *Selenium WebDriver*, um mecanismo que realiza os testes definidos em uma instância do navegador, dando suporte a praticamente todos os navegadores disponíveis. Para garantir maior precisão nos resultados, após a execução dos testes, foi realizada uma triagem no *LOG* resultante da conclusão da aplicação que invoca as instâncias do navegador, verificando assim o motivo

[9] Por exemplo, a URL http://notariato.info/tmp/mod_cust/ (Testado em 17/07/2017)
[10] http://docs.seleniumhq.org/download/

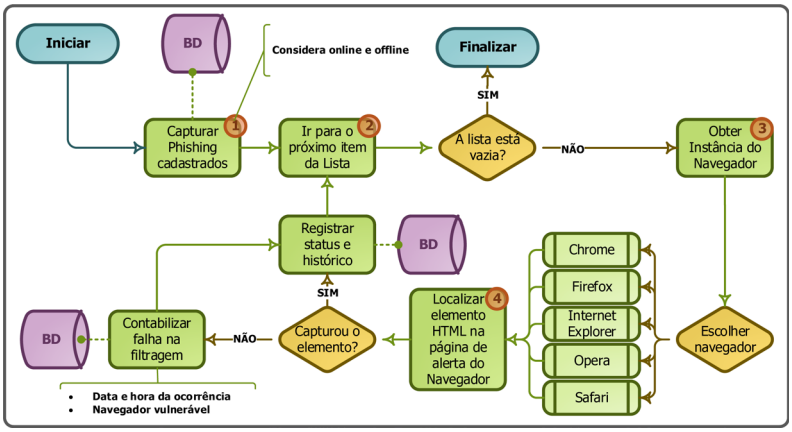

Figura 6. Fluxograma de avaliação nos navegadores F3

das falhas nos métodos, para confirmar se o teste ocorreu conforme o esperado.

O próximo passo é identificar se, ao invés da página do *phishing*, foi exibida a página de alerta do navegador. Esse tratamento precisou ser adaptado de acordo com o navegador utilizado, pois cada um possui uma representação diferente diante o acesso ao *phishing*. Porém, ambos compartilham o mesmo comportamento de sobrepor a página maliciosa com uma página de alerta. É possível obter elementos dessa página de alerta com a biblioteca Selenium WebDriver, tal busca pode ser feita pelo atributo ID de algum elemento HTML existente. Como exemplo, é possível localizar um botão intitulado "Me tire daqui!" que possui *ID "getMeOutButton"* no Firefox ou *"goToHomePage"* no IE.

Já no Opera a checagem era feita pelo texto *"Aviso de Fraude"* no título da página. Da mesma forma, Chrome e Safari apresentam como título o texto *"Erro de Segurança"*[11]. Por fim, se o elemento ou título fosse reconhecido, significaria que a tela de alerta havia sido exibida, indicando que o filtro foi bem sucedido. Caso contrário, seria registrada uma ocorrência de falha na filtragem do navegador. O resultado desse experimento pode ser observado nos gráficos da Figura 7.

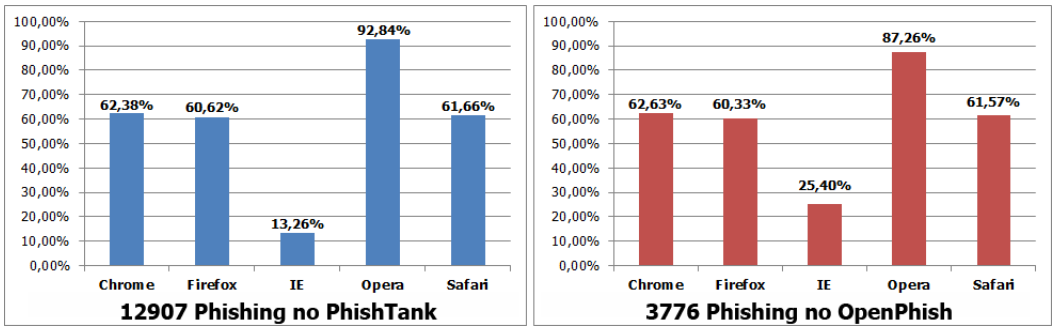

Figura 7. Resultados obtidos na filtragem dos navegadores

4.3. Resultados de Q3

Por fim, foi realizada a análise de padrões nas URL dos *phishing* obtidos na Q1. Foi observado uma grande ocorrência de palavras-chave, revelando o perfil de maior interesse

[11]A busca foi feita pelo título da página porque o navegador impedia o acesso aos detalhes sobre o conteúdo da página de alerta.

pelos mal intencionados, que se envereda em transações financeiras e redes sociais. Esse comportamento pode ser extraído através de uma análise de propagação na URL com base dos repositórios através de consultas SQL na base de dados com os *phishing* extraídos de Q1. O resultado do experimento pode ser observado na Tabela 3.

Tabela 3. Análise dos padrões dos *phishing* (na URL)

Palavra-chave	Ocorrências	Palavra-chave	Ocorrências	Palavra-chave	Ocorrências
login	3.596	domínios .br	1.590	domínios .org	1.345
https	1.266	google	916	paypal	596
card	529	mobile	368	boleto	302
banco	291	facebook	287	cadastr*	269
download	217	payment	204	billing	182
cert*	168	santander	154	banking	136

Observando a tabela 3, é possível perceber a quantidade de domínios que teoricamente possuem um processo de registro mais restrito, a exemplo dos domínios *.br* e *.org.*, mas ainda sim são amplamente explorados e, muitos, até o momento, sem uma postura corretiva sobre o fato. O primeiro caso evidencia o interesse dos mal intencionados em atuar no Brasil. Já o segundo pode representar a vista grossa em determinados países devido a postura política sobre o caso, conforme mencionado a Seção 2. Outra teoria seria os casos de criminosos que conseguem quebrar a segurança dos sites governamentais através de vulnerabilidades existentes, conseguindo realizar *upload* de páginas maliciosas em servidores oficiais, dificultando identificar o *phishing* com base no domínio.

4.4. Ameaças da avaliação

Como em qualquer tipo de experimento, a automação pode ocasionar em riscos na precisão dos resultados obtidos. Contudo, o estudo considera que a confiabilidade da metodologia adquiriu considerável precisão com a decisão de eliminar o caráter subjetivo da avaliação, no caso, a intervenção manual. E diante do contexto desse estudo não há nada que um ser humano possa fazer que a automação não seja capaz de reproduzir.

Quanto à **limitação de escopo**, no estado atual, foram considerados os *phishing* com o status "confirmado" no primeiro semestre de 2017, contudo, foi suficiente para ter um considerável número de URL maliciosas para análise. A justificativa de considerar os *phishing* mais recentes é que a grande relevância do filtro se dar por sua rápida resposta sobre um novo *phishing* que surge, ou seja, o menor tempo da janela de vulnerabilidade.

Em relação aos **falsos positivos** e **falsos negativos**, cada mecanismo avaliado tem suas particularidades. Primeiramente, é preciso considerar que os repositórios dependem da intervenção humana (denúncia) para cada *phishing*. Diante disso, é notória a dificuldade de responder à *phishing* recém-criados ou que não foram denunciados. Com o estado atual do estudo, seria possível fazer uma varredura pela *Web* em busca de *phishing* através da colisão de *hash*, identificando *phishing* não denunciados, porém, replicados em outro servidor. Contudo, o *phishing* poderia apresentar pequenas mudanças em suas variações, que apesar de sutis, seriam suficiente para modificar o *hash*.

Para tanto, como trabalho futuro, é pretendido desenvolver uma **análise morfológica** para acompanhar o comportamento de cada *phishing* durante um determinado

período. Não obstante, em certos casos, reconhecer um *phishing* através de seu *hash* pode ter suas limitações, já que existem diversas formas de atingir um *layout* semelhante a página genuína. Diante disso, um reconhecimento por imagem poderia ser interessante. Em suma, tais técnicas visam minimizar a dependência com os repositórios existentes ou mesmo contribuir com denuncias, melhorando-os.

Outro fato que reforça a preocupação dessa dependência é que existem casos de *phishing* publicados no repositório que ainda não passaram pelo crivo da administração da plataforma, representando uma janela de vulnerabilidade no repositório e, consequentemente, na API. Por exemplo, o *PhishTank*, por ser uma plataforma colaborativa onde qualquer pessoa denuncia de forma irrestrita, o *phishing* uma vez cadastrado, é classificado como "não confirmado", pois será submetido à uma análise manual por parte dos administradores do *PhishTank* para então ser uma ameaça "confirmada". Esse fato constata um atraso significativo da plataforma sobre um *phishing* que representa um perigo real ao usuário final, evidenciando o problema de falsos negativos. Contudo, a análise manual é um preço que se paga para evita falsos positivos.

Diante o estado atual do estudo, seria possível comparar a data do inicio de atividade do *phishing* no servidor de hospedagem (*online*) com a data de publicação do *phishing* na plataforma. Enquanto não lhe for atribuído o *status* "confirmado", nenhuma URL é reconhecida como uma ameaça, seja pela API ou navegador, proporcionado a janela de vulnerabilidade. Durante o experimento, também foi observado que a API não reconhecia uma URL como um *phishing* diante uma das situações: erro 404 (*offline*), conta suspensa ou *timeout* no servidor. Contudo, é sensato considerar que o site malicioso *offline* eventualmente poderá voltar a atividade, seja por correção na execução da página (*timeout*) ou contas reativadas no servidor. Com base nesses aspectos, é pretendido desenvolver uma **análise do ciclo de vida** de cada *phishing* como trabalhos futuros.

O problema também versa no contexto dos navegadores. Por exemplo, na API do *PhishTank*, determinados *phishing* foram de fato confirmados como positivos no repositório, mas ainda não estavam sendo reconhecidos como uma ameaça quando acessados através do navegador. O motivo seria o atraso na sincronização da lista negra do navegador com os registros do respectivo repositório, representando assim uma janela de vulnerabilidade através do uso do navegador. Contudo, para maior precisão sobre o atraso na sincronização da lista negra, seria necessário comparar em cada *phishing*, a data e hora da falha contra a data e hora da confirmação no repositório, obtendo uma diferença.

Outro ponto, é que os testes foram baseados em navegadores para *desktop*, teoricamente, pode ser considerado que os resultados seriam aplicados à dispositivos móveis. De toda forma, talvez fosse interessante comparar entre as versões de cada plataforma. Todavia, existem certos aspectos da computação móvel, como repositório de aplicativos, que merecem análises voltadas para esse cerne. Por fim, uma limitação da análise de propagação é que a mesma inspecionou apenas a composição da URL, o interessante seria considerar elementos textuais e da páginas, uma vez que nem todo *phishing* contem em sua URL palavras-chave que façam referência aos objetivos do site alvo.

5. Conclusão

Dentre os 5 navegadores avaliados, nenhum teve mais que 90% de acerto quando comparado em um repositório diferente de seu mecanismo de proteção. Os casos em que o

navegador foi avaliado em um repositório distinto ao do seu mecanismo, como no teste do Opera com o *OpenPhish*, o navegador teve um desempenho de 87,26%. Já a API do repositório do Opera, *PhishTank*, quando confrontada com o *OpenPhish*, resultou em 68,62% de acerto, evidenciando uma possível proteção adicional no navegador. Contudo, o Opera atingiu 92,84% de acerto enquanto confrontado por seu respectivo repositório, o que indica possíveis falhas de falsos negativos. De fato, essas evidências sugerem uma pesquisa com maior profundidade na investigação sobre essa proteção adicional.

Em termos de baixa ineficiência da filtragem, o que chama atenção é a taxa do IE com o *SmartScreen*, 13,26% no *PhishTank*, bem abaixo dos concorrentes. Apesar das limitações mencionadas sobre o *SmartScreen*, um ponto positivo é que o mesmo pode, adicionalmente, ser integrado aos aplicativos do pacote *Microsoft office*, a exemplo do cliente de *e-mail Outlook*, podendo assim minimizar as explorações por *e-mail* mencionadas na Seção 2. Contudo, essa funcionalidade não reduz os riscos atuantes no navegador. Considerando os resultados dos testes sobre as proteções disponíveis, foi observado que o mecanismo nativo dos navegadores apresentam sérias limitações, sugerindo aos seus usuários a necessidade de considerar soluções *endpoint*, a exemplo de antivírus.

Como trabalhos futuros, o estudo pretende preencher lacunas observadas na Seção 4.4. Conforme ilustrado no fluxograma **F4** da Figura 8, o próximo passo do estudo é realizar uma **análise do ciclo de vida** de cada *phishing*, visando um monitoramento do tempo de criação e atividades posteriores. Na mesma linha, com o fluxograma **F5** da Figura 9 é pretendido realizar uma **análise morfológica** para monitorar as modificações sofridas pelo *phishing* durante um período. Para facilitar a organização e estruturação, diversos dados extraídos nos repositórios de *phishing* (Q1) foram armazenados na base de dados do experimento.

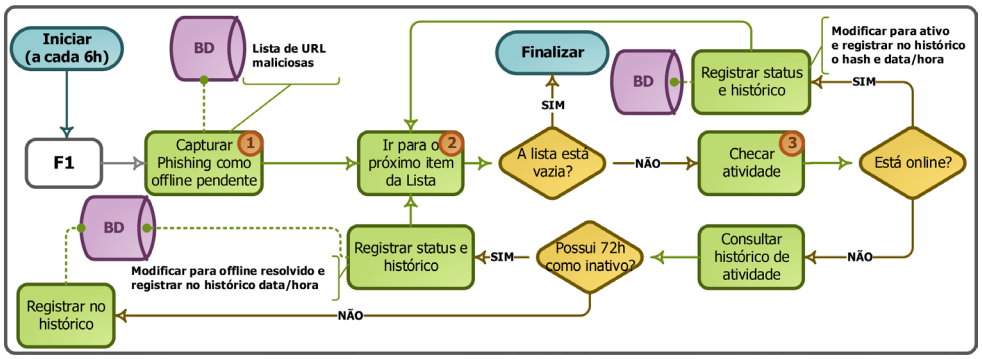

Figura 8. Fluxograma da análise de ciclo de vida do *phishing* F4

Para o ciclo de vida, será registrado datas de eventos do *phishing*. Uma delas é a data de publicação no repositório, obtida no repositório de origem. Porém, a mesma não sugere o inicio da atividade do *phishing*. Para isso, a informação pode ser obtida através do protocolo *WHOIS*[12], que oferece informações sobre o proprietário do domínio, data de publicação, expiração e última atualização do domínio. Por fim, a data de inatividade auxilia na análise do ciclo de vida do *phishing* durante sua atuação na Web.

Quanto a sua morfologia, serão consideradas informações como o conteúdo rece-

[12]WHOIS: https://tools.ietf.org/html/rfc3912

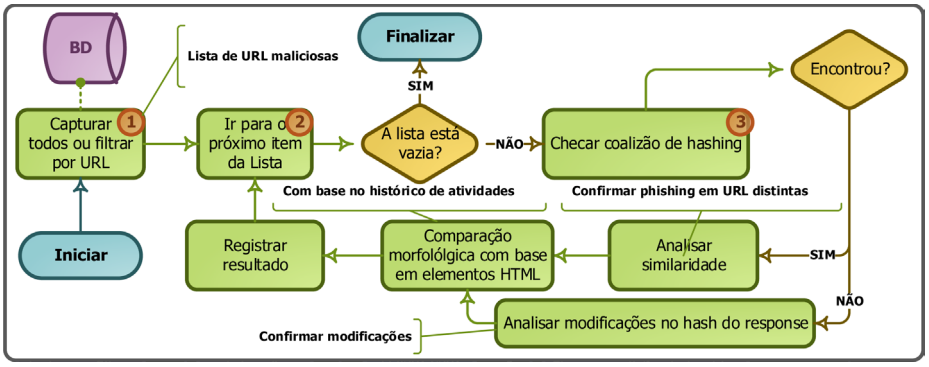

Figura 9. Fluxograma da análise de morfologia do *phishing* F5

bido como resposta ao requisitar a URL maliciosa, que será convertida em um calculo de *hash SHA-256* para facilitar na identificação de URL que apontem para um mesmo *phishing*, através de colisão de *hash*.

Referências

Akhawe, D. and Felt, A. P. (2013). Alice in warningland: A large-scale field study of browser security warning effectiveness. In *22Nd USENIX*, SEC'13, pages 257–272.

Cert.BR (2012). Golpes na internet. *Disponível em: https://cartilha.cert.br/golpes/*.

Costello, A. M. (2003). Punycode: A bootstring encoding of unicode for internationalized domain names in applications (idna). *Disponível em: https://tools.ietf.org/html/rfc3492*.

Jackson, C., Simon, D. R., Tan, D. S., and Barth, A. (2007). An evaluation of extended validation and picture-in-picture phishing attacks. In *Financial Cryptography*, volume 4886 of *Lecture Notes in Computer Science*, pages 281–293. Springer.

Jovanovic, G. (2009). Standardization of the old church slavonic cyrillic script and its registration in unicode. *Disponível em: http://www.unicode.org/charts/PDF/U0400.pdf*.

Kaspersky (2014). O que é phishing? *Disponível em: https://goo.gl/4EEtxk*.

Khonji, M., Iraqi, Y., and Jones, A. (2013). Phishing detection: A literature survey. *IEEE Communications Surveys and Tutorials*, 15(4):2091–2121.

Mazher, N., Ashraf, I., and Altaf, A. (2013). Which web browser work best for detecting phishing. In *2013 5th ICICT*, pages 1–5.

Mohammad, R. M., Thabtah, F., and McCluskey, L. (2015). Tutorial and critical analysis of phishing websites methods. *Comput. Sci. Rev.*, 17(C):1–24.

TrendMicro (2014). O que é spear phishing? *Disponível em: https://goo.gl/GhzUFY*.

TrendMicro (2017). Domínios genéricos e sites fraudulentos: uma conexão perigosa. *Disponível em: https://goo.gl/r9gAZm*.

Whittaker, C., Ryner, B., and Nazif, M. (2010). Large-scale automatic classification of phishing pages. In *NDSS '10*.

ZDNet (2016). Phishing-as-a-service is making it easier than ever for hackers to steal your data. *Disponível em: https://goo.gl/6UST6u*.

