# Uma Metodologia para Agrupamento e Extração de Informações de URLs de Phishing

**Welton Santos**[2] **Elverton Fazzion**[2,1] **Osvaldo Fonseca**[1] **Ítalo Cunha**[1]
**Marcelo H. P. C. Chaves**[3] **Cristine Hoepers**[3] **Klaus Steding-Jessen**[3]
**Dorgival Guedes**[1] **Wagner Meira Jr.**[1]

[1]Departamento de Ciência da Computação
Universidade Federal de Minas Gerais

[2]Departamento de Computação
Universidade Federal de São João del-Rei

[3]CERT.br – Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança
NIC.br – Núcleo de Informação e Coordenação do ponto BR

{weltonsantos,osvaldo.morais,cunha,dorgival,meira}@dcc.ufmg.br

{mhp,cristine,jessen}@cert.br fazzion@ufsj.edu.br

***Abstract.*** *Despite advances in prevention and mitigation mechanisms, phishing remains a threat. One reason for this is that phishers continuously improve their techniques. In this paper we study and characterize one of these improvements: phishers' use of redirection chains to evade identification mechanisms and avoid takedown of the infrastructure hosting the malicious content. We propose a method to group messages and URLs into phishing campaigns, and develop a framework to identify their hosting infrastructure. We apply our method and framework on a dataset of spam and phishing messages collected from low-interactivity honeypots. We explore and characterize phishing campaigns as well as their hosting infrastructure. Our results indicate that phishing campaigns are usually hosted in cloud providers, but some are hosted on devices in access networks, possibly infected end-user devices. This indicates multiple approaches used by phishers, and motivates different fronts to combat this threat.*

***Resumo.*** *Apesar de avanços em mecanismos de prevenção e mitigação, o* phishing *continua uma ameaça. Uma das razões disso é que o agente responsável pelo envio dessas mensagens, o* phisher*, aprimora suas técnicas continuamente. Neste trabalho estudamos e caracterizamos um destes aprimoramentos: o uso pelos* phishers *de cadeias de redirecionamentos para ludibriar mecanismos de identificação e evitar bloqueio da infraestrutura de hospedagem do conteúdo malicioso. Propomos um método para agrupamento de mensagens e URLs em campanhas de* phishing*, e desenvolvemos um arcabouço para identificação da infraestrutura de hospedagem. Aplicamos nosso método e arcabouço em um conjunto de mensagens de* spam *e* phishing *coletado por* honeypots *de baixa interatividade. Caracterizamos campanhas de* phishing *e a infraestrutura de hospedagem. Nossos resultados mostram que a infraestrutura de hospedagem das campanhas identificadas se concentra em provedores*

*de computação em núvem, mas que algumas campanhas são hospedadas por dispositivos em provedores de rede, possivelmente em dispositivos infectados. Isso indica diferentes abordagens de* phishers*, sugerindo esforços em diferentes frentes de combate.*

## 1. Introdução

Apesar dos avanços alcançados no combate ao *phishing*, esses ataques ainda são muito comuns. As técnicas empregadas nos e-mails de *phishing* evoluíram e, atualmente, apresentam um alto nível de personalização alcançado através de engenharia social e dados comprados ilegalmente de ciber-criminosos [Volkamer et al. 2017, Las-Casas et al. 2016]. Durante a última Copa do Mundo, por exemplo, *phishers* enviaram e-mails oferecendo a venda de ingressos para convidados, comprando domínios com nomes nomes similares a "worldcup2018" de forma a convencer as vítimas da legitimidade da mensagem [Vergelis and Kostin 2018]. Devido a esses e outros aprimoramentos, é necessário o acompanhamento constante para capturar estratégias atualizadas, o que permite sofisticar as técnicas de combate a essa atividade criminosa que gera bilhões de dólares de prejuízo para os usuários e organizações [Almomani et al. 2013, Las-Casas et al. 2016].

Para enviar mensagens, o *phisher* utiliza uma complexa infraestrutura composta por máquinas zumbis (pertencentes a *botnet*) e servidores SMTP vulneráveis. Essa infraestrutura permite ao *phisher* ludibriar filtros baseados em *blacklists*, uma vez que máquinas legítimas são responsáveis pela disseminação de mensagens, e dificultar a tomada de ações legais contra eles, pois a identidade dos criminosos na rede é substituída pela infraestrutura legítima utilizada. Além disso, o *phisher* também pode utilizar dessa infraestrutura para hospedar o conteúdo presente nas mensagens de *phishing* como, por exemplo, páginas Web associadas a URLs presentes nas mensagens que simulam serviços legítimos para furtar informações.

Entender a infraestrutura de hospedagem de páginas é um importante passo no combate ao *phishing*. Complementar ao processo de filtragem de mensagens, o rápido bloqueio de páginas de *phishing* torna os e-mails que as contém inócuos, uma vez que a possível vítima não conseguirá acessar o conteúdo malicioso. Ferramentas automatizadas para analisar classificar URLs em URLs legítimas e URLs de *phishing* são fundamentais para permitir combate efetivo ao *phishing* dada a complexa cadeia de redirecionamento utilizada para ofuscar a infraestrutura de hospedagem.

Nesse artigo, propomos uma metodologia de análise para URLs presentes em mensagens de *phishing*, de forma a identificar e agrupar páginas maliciosas similares, facilitando o processo de análise por parte de operadores de rede. Mais especificamente, nossa metodologia calcula a similaridade entre cada par de páginas combinando informações de termos frequentes utilizados por *phishers* (e.g., *please*, *fill*) presentes no código fonte da página. A partir do cálculo de similaridades, a metodologia gera um grafo cujos componentes revelam os grupos de URLs com páginas similares e que, possivelmente, pertencem a uma mesma campanha de *phishing*.

Nós aplicamos nossa metodologia a um conjunto de dados coletados por *honeypots* de baixa interatividade distribuídos na Internet. Combinamos as informações dos componentes gerados com informações de rede, como os endereços IP de servidores acessados na porta 80 e capturados durante o acesso a uma página, e mostramos que

existe (i) uma concentração de URLs similares em poucos provedores de infraestrutura, indicando que alguns *phishers* podem estar fazendo uso de cartões fraudados para a compra de computação em nuvem para hospedagem das páginas e (ii) componentes com uma complexa cadeia de redirecionamento que dificulta a identificação da hospedagem e, consequentemente, do *phisher*. Para ambos os cenários, apresentamos uma caracterização completa das URLs presentes nos componentes e realizamos estudos de casos de forma a entender o comportamento atual de *phishers*.

A metodologia apresentada neste artigo contribui para a extração eficaz de identificação da infraestrutura de hospedagem utilizada pelo *phisher*. Com isso, medidas podem ser tomadas para que máquinas com o conteúdo malicioso de *phishing* sejam bloqueadas mais rapidamente e que medidas sejam tomadas para melhorias nas redes que hospedam essas máquinas.

## 2. Processamento de URLs de phishing

Para este trabalho, realizamos a coleta de e-mails com origem maliciosa usando *honeypots* de baixa interatividade (seção 2). Em seguida, analisamos esses e-mails e identificamos as mensagens de *phishing* utilizando técnicas estado-da-arte (seção 2.2). Para cada mensagem identificada como *phishing*, utilizamos uma sequência de heurísticas estáticas para identificar URLs com conteúdo malicioso (seção 2.3). Por fim, acessamos essas URLs de forma a capturar informações sobre as conexões realizadas (redirecionamentos) e obter o conteúdo da página (seção 2.4). A figura 1 sumariza o processo descrito nesta seção.

### 2.1. Coleta de mensagens

As mensagens foram capturadas por 12 *honeypots* de baixa interatividade [Steding-Jessen et al. 2008] instalados em diferentes países, sendo 2 no Brasil, Estados Unidos e Holanda, e 1 na Argentina, Áustria, Austrália, Equador, Japão, Taiwan e Uruguai. Os *honeypots* são configurados para emular servidores proxy (HTTP e SOCKS) e relay (SMTP) abertos e simulam vulnerabilidades. Quando um agente malicioso se conecta ao servidor SMTP de um honeypot, ele é levado a crer que está interagindo com um servidor SMTP real operando como um relay aberto. O mesmo acontece com os protocolos HTTP/SOCKS, onde o agente é levado a crer que é capaz de estabelecer conexões com outros servidores SMTP na rede. Os *honeypots* utilizados nessa pesquisa não prestam serviço para nenhuma rede e não são anunciados publicamente. Dessa forma, assumimos que todas as mensagens recebidas por eles provém de spammers ou phishers. Toda a interação com os *honeypots* é registrada e as mensagens são armazenadas localmente e enviadas para os servidores centrais do projeto diariamente. Nossos *honeypots* nunca encaminham as mensagens recebidas, com exceção daquelas mensagens cujos conteúdos indicam, de acordo com regras pré-definidas[1], que são mensagens de teste utilizadas pelo agente malicioso para verificar se *proxies* e *relays* abertos estão funcionando. Neste trabalho, analisamos todas as mensagens coletadas entre 29 de Abril e 14 de Maio de 2019.

### 2.2. Identificação de mensagens de phishing

Para separar mensagens de *spam* e *phishing*, utilizamos o classificador Naive Bayes. No nosso contexto, o classificador calcula e atribui a cada mensagem uma probabilidade de

[1]Por exemplo, verificando presença de texto específico no assunto ou corpo da mensagem.

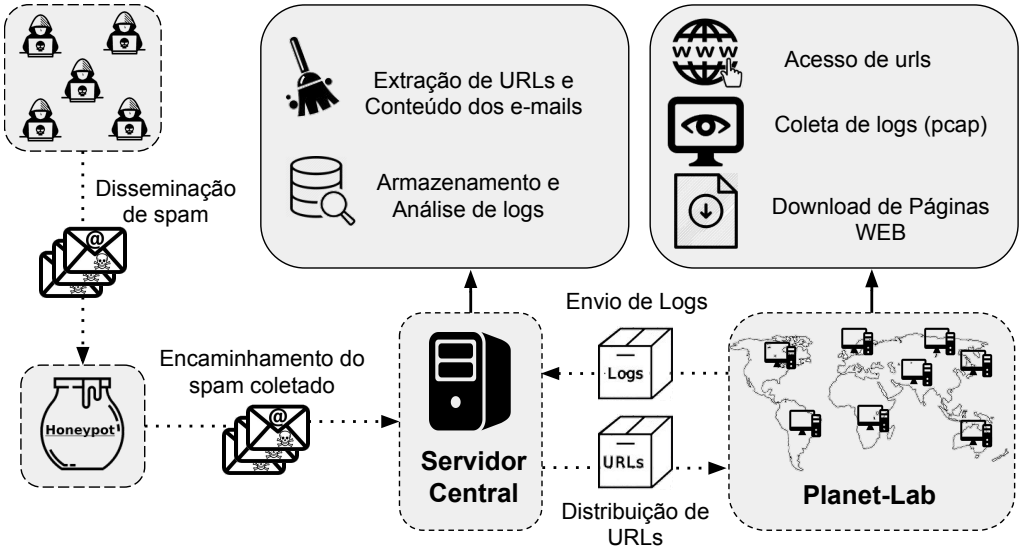


**Figura 1. Metodologia de obtenção e acesso de URLs de *phishing*.**

ser *spam* ou *phishing*, rotulando a mensagem com a classe com maior probabilidade. Para criar o treino do algoritmo, agrupamos as mensagens no idioma em inglês em campanhas [Calais et al. 2008], recuperamos uma mensagem representante de cada campanha, amostramos quinhentas mensagens aleatoriamente e realizamos uma classificação manual sobre a amostra. Ainda, cada uma das mensagens é submetida a um conjunto de regras para reduzir ruídos e alcançar uma maior padronização, evitando que um simples detalhe, como a diferenciação entre letras maiúsculas e minúsculas, afetem o modelo. Mais especificamente, convertemos cada caractere da mensagem para sua representação minúscula e removemos links, números, pontuações e *stopwords*. A composição final do conjunto de treino foram 230 mensagens de *phishing* e 270 mensagens de spam. Verificamos que a taxa de acerto da técnica de classificação foi de 92%.

### 2.3. Extração de URLs de phishing

Em geral, as mensagens de *phishing* contém tanto URLs apontando para páginas falsas quanto URLs apontando para páginas verdadeiras (uma forma de aumentar a "legitimidade" da mensagem). Em algumas, por exemplo, a URL exibida para vítima é diferente daquela contida no HREF; a vítima lê e reconhece o link como referente a uma página legítima mas é redirecionada à página do atacante. Dessa forma, utilizamos heurísticas propostas na literatura para identificar URLs de páginas de *phishing* [Khonji et al. 2012]. Para cada URL verificamos se em seu corpo continha alguma das seguintes características: (i) **urlatchar**, presença de @ utilizado para associação com *usernames* a páginas falsas, (ii) **urlbalink**, termos relacionados a phishing (e.g. click, login, security), (iii) **urlip**, presença de endereço IP no domínio, (iv) **urlnumport**, alteração da porta WEB padrão 80 para portas altas e (v) **urlstwodomain**, presença de múltiplos subdomínios (e.g. http://empresa.com.phish.com). Caso a URL contenha alguma dessas características, ela será mantida no grupo de estudo. De todas as 621.531 URLs (469.763 únicas), apenas 119.122 URLs foram classificadas como *phishing* (89.619 únicas).

[1]https://code.google.com/archive/p/language-detection/

## 2.4. Acessando URLs de phishing

Para mascarar sua identidade, o *phisher* utiliza diversas facetas, dentre elas o acesso às páginas de *phishing* através de cadeias de redirecionamentos. Redirecionamentos são realizados a partir de requisições HTTP ou código JavaScript e uma cadeia de redirecionamento com diversos saltos pode utilizar as duas tecnologias.

O acesso a uma URL é realizado através de duas abordagens. Na primeira abordagem, a URL é resolvida a partir da biblioteca $Requests$, capaz de resolver somente redirecionamentos de requisições HTTP. Após o acesso com a biblioteca $Requests$, o histórico de requisições é salvo assim como o endereço IP e URL obtidos. Essa primeira abordagem consiste em uma estratégia rápida e de baixo custo computacional, porém não é completa, sendo incapaz de tratar URLs com redirecionamentos em JavaScript. A figura 2 exemplifica a cadeia de redirecionamento da URL bankX.com que engloba tanto redirecionamento HTTP quanto redirecionamento via JavaScript. Dessa forma, utilizando apenas a biblioteca $Requests$ não somos capazes de identificar a URL final.

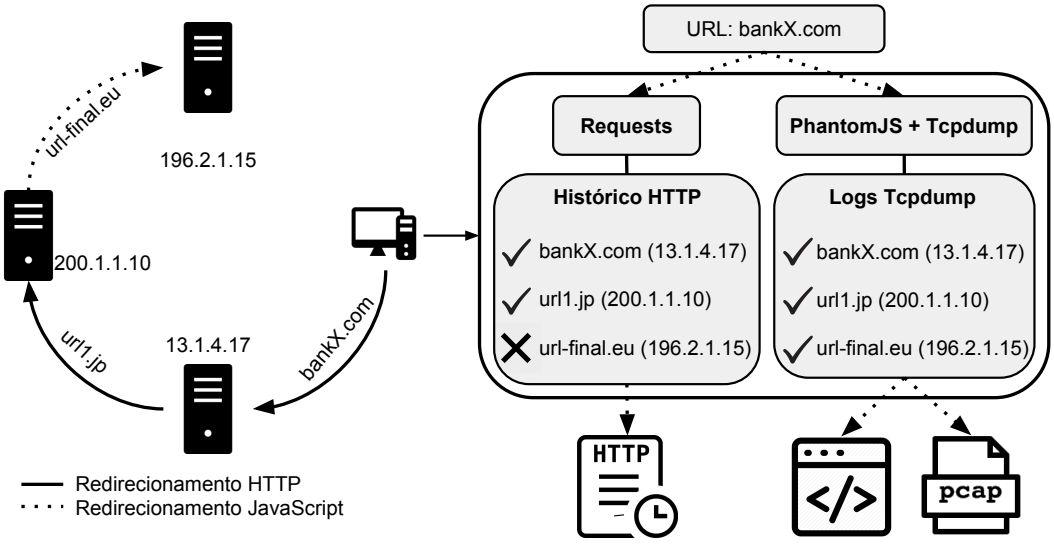


**Figura 2. Página phisher escondida atrás de redirecionamentos**

Como visto na figura 2, a biblioteca Requests retornaria o IP e URL obtidos no segundo salto e não atingiria a página final, do nosso interesse. Para lidar com situações similares, na segunda abordagem utilizamos o motor PHANTOMJS[2] que consiste em um navegador para testes de aplicações web, sem interface gráfica, capaz de interpretar código JAVASCRIPT e que permite resolver URLs com perfil similar ao cenário mostrado na figura 2. Quando o PHANTOMJS consegue acessar a URL, coletamos o endereço IP final, URL final e código fonte da página. Para registrar os endereços intermediários na cadeia de redirecionamento, utilizamos o software TCPDUMP, ferramenta de monitoramento e captura de pacotes. A partir do monitoramento da interface de rede do host, coletamos os endereços IP referentes a cada salto da cadeia de redirecionamento, como mostrado na figura 2.

[2]https://github.com/ariya/phantomjs/

O acesso a URLs em maior escala é realizado utilizando a infraestrutura distribuída pertencente ao PlanetLab [Chun et al. 2003]. Dessa forma, os acessos às URLs são realizados de diversas regiões do mundo, ideal para o nosso trabalho, uma vez que grandes volumes de acessos de uma fonte centralizada poderiam despertar a atenção de *phishers*, comprometendo o sigilo dos monitoramento. É importante mencionar que não acessamos nenhuma URL a partir dos *honeypots* para o anonimato dessas máquinas.

A aplicação é estruturada sobre uma hierarquia mestre-escravo, conforme mostrado na figura 1 onde um servidor central coordena vários outros servidores executando no PLANET-LAB. Diariamente o servidor mestre envia aos servidores escravos um conjunto de arquivos com URLs. Visto que *phishers* mantém suas páginas ativas por curtos períodos de tempo e alteram constantemente os domínios onde residem suas páginas, os conjuntos de URLs enviados aos servidores passam por uma uma etapa de pré processamento. Nesta etapa um conjunto maior de URLs é triado de modo que reste ao final para distribuição somente URLs únicas e com ordem de prioridade precedida pela data de registro mais recente. Dessa forma, tentativas de acesso à URLs repetidas ou expiradas são reduzidas, aumentando a margem de sucesso da aplicação. Em média, cerca de 33.600 URLs são processadas a cada dia, onde 48% são resolvidas com sucesso. Cada URL leva em média 12,75 segundos para ser processada com desvio padrão de 27,75 segundos.

## 3. Agrupamento de Páginas

O *phisher*, para tornar acessível o conteúdo de sua página maliciosa, associa uma ou mais URLs a uma página (ou conjunto de páginas similares). Dessa forma, cada URL associada a uma página pode ser correlacionada a outra URL através da proximidade existente entre os conteúdos de suas páginas, de tal forma que técnicas de agrupamento podem ser empregadas na identificação de campanhas de *phishing*. Na seção 3.4 descrevemos nossa estratégia para identificar essas campanhas.

### 3.1. Detecção de páginas duplicadas

Visto que várias URLs podem estar associadas a uma única página, submeter páginas iguais ao processo de extração de conteúdo e validação de idioma (apresentado na próxima seção), causaria maior consumo de recursos computacionais e tempo. Para evitar o processamento de páginas repetidas e identificar duplicatas foi construído um registro com os valores de hash MD5 de cada página. De forma que, antes de processarmos a página, verificamos se seu valor de hash consta no registro. Caso conste, a URL associada à página é associada ao hash já existente, caso contrário o novo hash é inserido na base associado à URL. Desta maneira conseguimos reduzir o conjunto total de páginas a processar de 119.122 para 43.028, 36% da quantidade coletada.

### 3.2. Extração de termos relevantes

Neste trabalho optamos por lidar apenas com páginas com conteúdo em inglês, porém a técnica pode ser estendida a outros idiomas. Ao iniciar o processamento de uma página, todo o seu texto é extraído e determinamos o seu idioma com a biblioteca $Langdetect$[3], caso a página tenha sido escrita em idioma inglês, removemos pontuação e *stopwords* de seu corpo e avançamos para as próximas etapas do processo. Em seguida, duplicamos o

[3]https://pypi.org/project/langdetect/

conteúdo textual, onde a primeira cópia é submetida a um filtro a partir de um dicionário, como descrito na seção 3.2.1, e a segunda cópia a outro filtro destinado à identificação de nomes próprios, como descrito na seção 3.2.2. O processo completo é ilustrado pela figura 3.

### 3.2.1. Filtro com dicionário

Nesta etapa apresentamos duas alternativas desenvolvidas para extração de termos relevantes das páginas e discutimos a escolha para o trabalho.

Na primeira abordagem foi utilizado um dicionário do idioma inglês para identificar e preservar somente palavras pertencentes ao idioma. Porém, devido à abrangência do dicionário, quantidade significativa de palavras não possuíam relevância (como *you*, *many*, *page*, *social*). Outro problema consiste na frequente má estruturação de código presente nas páginas. Devido ao fato de muitas páginas conterem falhas em seu código, nestes casos, não foi possível distinguir com total precisão palavras reservadas de linguagens como CSS e JavaScript (e.g., hover, opacity, serif e display) e conteúdo textual pertencente ao idioma inglês prejudicando a eficiência do parser utilizado. Dessa forma, optamos por utilizar a segunda abordagem, descrita a seguir.

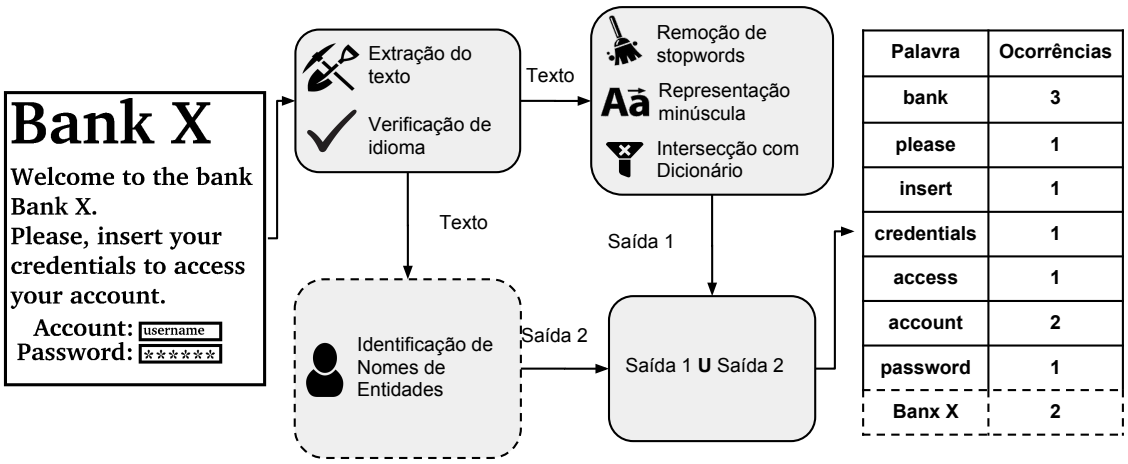


**Figura 3. Etapas do processo de extração de termos das páginas.**

Na segunda abordagem para a extração de termos, utilizamos um dicionário de termos associados a *phishing* baseado no trabalho de [Las-Casas et al. 2016]. Para atualizar as categorias presentes em [Las-Casas et al. 2016], executamos o modelo *word2vec*[4] para a base recente e identificamos uma nova categoria relacionada a conteúdo adulto, resultando nas seguintes categorias de termos relacionados a *phishing*: Tratamento, Menção a Dinheiro, Pedido de Resposta, Urgência, Formulário, Segurança e Conteúdo Adulto. Esta técnica também reduziu a adição de termos irrelevantes como o ocorrido na primeira abordagem, sendo então a técnica adotada.

[4]https://radimrehurek.com/gensim/models/word2vec.html

### 3.2.2. Extração de nomes

Para obter maior acurácia na identificação de páginas com conteúdo similar, empregamos um mecanismo para identificar nomes próprios, uma vez que a extração de nomes de entidades permite identificar com maior precisão ataques contra instituições específicas. Para detectar nomes utilizamos o modelo pré treinado *Spacy*[5]. Dessa forma, submetemos o texto da página no modelo e extraímos os nomes próprios identificados assim como suas respectivas frequências.

## 3.3. Construção do grafo

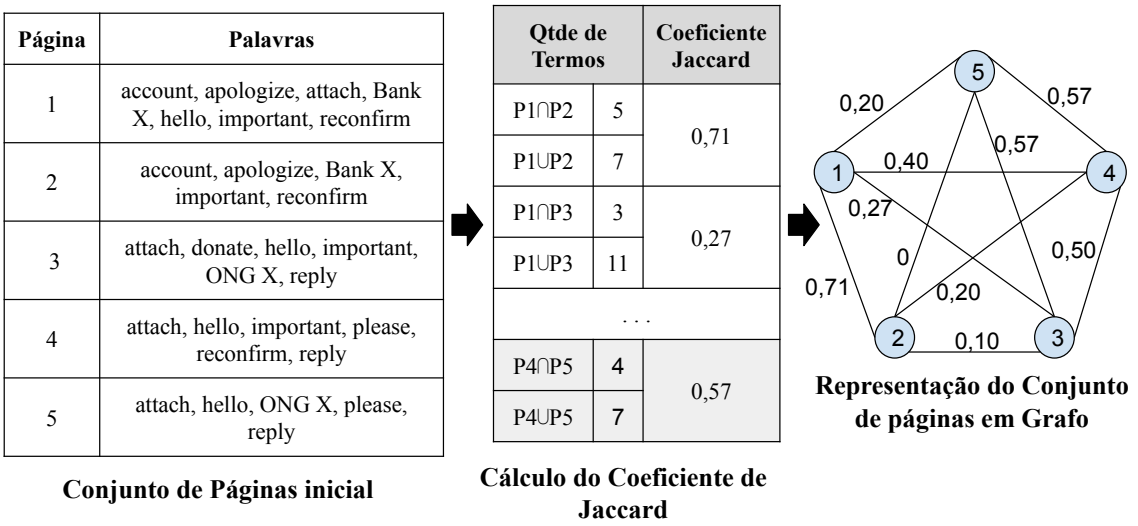


**Figura 4. Representação do conjunto de páginas em grafo a partir da equação 1.**

Nesta seção descrevemos a construção do grafo de similaridade de páginas. Cada vértice nesse grafo representa uma página e os pesos das arestas quantificam a similaridade entre as páginas conectadas. Os pesos são valores reais dentro do intervalo [0, 1], proporcionais à semelhança entre as páginas conectadas pela aresta e determinados a partir do coeficiente de *Jaccard*. O coeficiente de *Jaccard* mensura a proximidade entre dois objetos calculando a razão entre a quantidade de atributos em comum e a quantidade de atributos totais (união entre os atributos contidos nos dois objetos). Para obter o coeficiente de *Jaccard* entre duas páginas, utilizamos como atributos as palavras obtidas no pré processamento das páginas descrito na seção 3.2. Portanto, o valor de uma aresta $A_{ij}$ entre duas páginas $P_i$ e $P_j$ é a razão entre a quantidade de palavras que aparecem em ambas as paáginas, $P_i \cap P_j$, e a quantidade de palavras que aparecem em qualquer uma das páginas, $P_i \cup P_j$: $(P_i \cap P_j)/(P_i \cup P_j)$. Formalmente, o grafo $G$ com páginas de *phishing* é descrito pela equação 1.

$$
\begin{aligned}
&G = (V, A) \\
&V = \{P_1, P_2, ..., P_i, ..., P_n\} \\
&A = \{A_{ij} : \forall(P_i, P_j) | (P_i \in V, P_j \in V) \wedge i \neq j\} \\
&w_{ij} = (P_i \cap P_j)/(P_i \cup P_j)
\end{aligned}
\tag{1}
$$

[5]https://spacy.io/usage/linguistic-features#section-named-entities

Após obter um conjunto de páginas com seus respectivos termos, conforme o apresentado na seção 3.2, o processo de representação do conjunto e suas relações através de um grafo descrito pela equação 1 é ilustrado pelo exemplo na figura 4. Devido ao fato de campanhas de *phishing* permanecerem ativas por curtos intervalos de tempo, aplicamos o processo ilustrado na figura 4 para cada dia de um conjunto de testes de 16 dias, obtidos entre as datas 29 de Abril de 2019 e 14 de maio do mesmo ano. No total foram analisadas 43.028 páginas de *phishing* únicas (2.689,25 páginas por dia, em média).

### 3.4. Identificação de campanhas

Para identificar conjuntos de páginas com objetivo comum, primeiro removemos arestas cujo peso seja inferior a um limiar $\alpha$, tal que $0 < \alpha \leq 1$ e, em seguida, extraímos os componentes conectados do grafo. Reutilizando como base o grafo apresentado na figura 4, o processo de identificação de campanhas é ilustrado na figura 5.

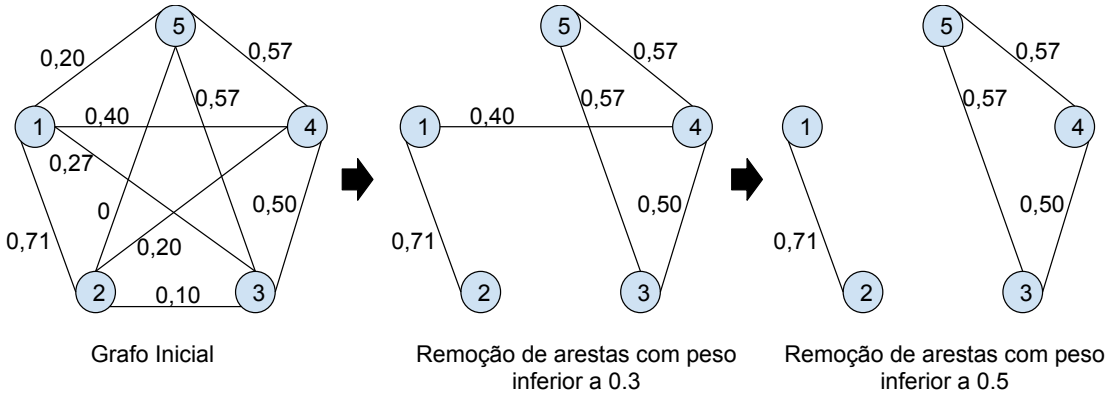


**Figura 5. Identificação de componentes após a remoção de arestas para $\alpha = 0.3$ e $\alpha = 0.5$**

Para diferentes valores de $\alpha$ obtemos diferentes quantidades de componentes, como exibido no gráfico da figura 6, que mostra o número de componentes à medida que variamos o limiar. Da mesma forma também observamos a variação na densidade de páginas por componente, descrita na figura 7, onde é exibida a razão entre a quantidade de páginas por componente para cada limiar.

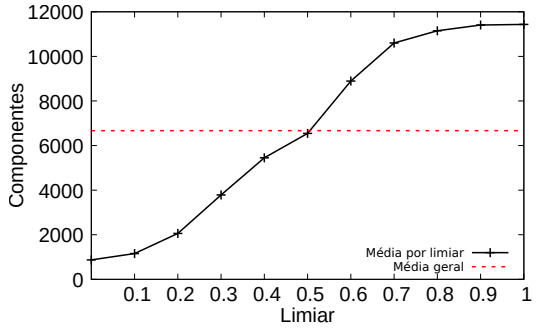


**Figura 6. Média de componentes por limiar.**

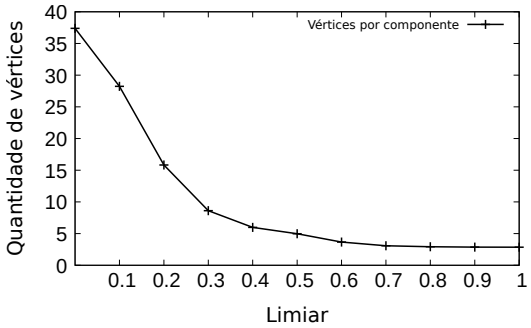


**Figura 7. Média de vértices por componentes.**

A partir dos gráficos apresentados nas figuras 6 e 7, é notória a relação inversamente proporcional entre a quantidade de componentes e a densidade de páginas por

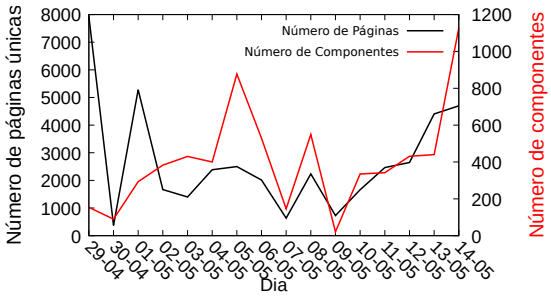


**Figura 8. Distribuição do número de Páginas e Componentes observados por dia.**

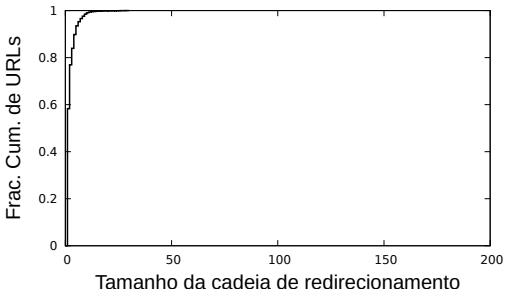


**Figura 9. Tamanho das cadeias de redirecionamento para URLs que aparecem em mais de um dia.**

componentes. Quando $\alpha$ se aproxima de 0.1, componentes maiores contem páginas pouco semelhantes e, opostamente, quando $\alpha$ se aproxima de 1.0 é gerado um número maior de componentes, mas cujas páginas são mais semelhantes entre si. Para escolher o valor de $\alpha$, buscamos pelo ponto de inflexão na curva do gráfico da figura 7, o limiar para o qual aumentar o valor de $\alpha$ têm pouco impacto no tamanho dos componentes. Considerando o gráfico da figura 6, observamos que valores de $\alpha$ maiores que $0.5$ não reduzem significativamente o tamanho dos componentes, mas o seu número é crescente, o que pode dificultar a análise. Para os próximos resultados utilizamos $\alpha = 0.5$.

## 4. Resultados

Nesta seção mostramos como a metodologia proposta auxilia operadores e analistas de rede a extraírem informações relevantes sobre conteúdos de *phishing* a partir de URLs coletadas. Inicialmente, caracterizamos as páginas Web obtidas através da aplicação do processo de coleta da seção 2 e mostramos como a metodologia da seção 3 reduz o número de páginas a serem analisadas (subseção 4.1). Em seguida, caracterizamos os componentes identificados e mostramos como podemos utilizá-los para sumarizar informações importantes sobre o conjunto de páginas como a concentração de páginas em redes de hospedagem de conteúdo (subseção 4.2).

### 4.1. Caracterização de páginas e componentes

O gráfico da figura 8 mostra que o número de páginas únicas por dia (curva preta) varia ao longo do tempo, indicando variação do comportamento do *phisher* na propagação de campanhas de *phishing*. Apesar do número de páginas únicas observadas ser da ordem de milhares de URLs, nossa metodologia consegue agrupar páginas similares, reduzindo o número de páginas a serem avaliadas para algumas centenas, como mostrado no gráfico da figura 8 (curva vermelha). Esse resultado mostra que, apesar de existirem milhares de páginas únicas, muitas delas possuem uma estrutura similar e podem estar associadas a uma única campanha disseminada pelo *phisher* na rede.

Nossa técnica também revela que um aumento no número de páginas únicas num dado dia não está necessariamente correlacionado com o aumento do número de campanhas de *phishing* naquele dia. Esse comportamento pode ser visto contrastando o número de páginas únicas em um dia com o número de componentes. Por exemplo, no dia 29/04 foram observadas cerca de 8.000 páginas únicas agrupadas em cerca de 200 campanhas de

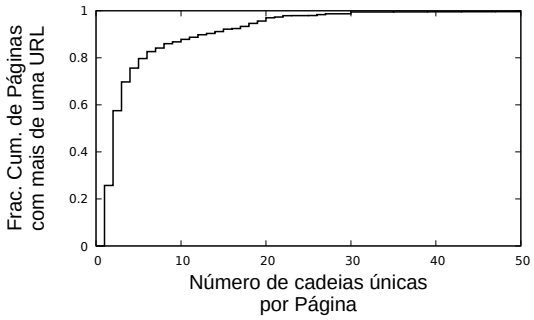


**Figura 10. Número de cadeias de redirecionamento por página.**

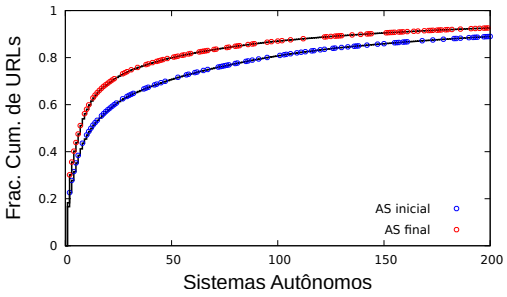


**Figura 11. Distribuição de ASes iniciais e finais presentes nas cadeias de redirecionamento.**

*phishing*. Porém, se observarmos o dia 14/05, verificamos cerca de 4.500 páginas únicas agrupadas em aproximadamente 1.100 campanhas. Além disso, esse resultado corrobora estudos na literatura que mostram que alguns *phishers* automatizam o processo de ataques de *phishing*, utilizando um modelo de página padrão que varia pouco entre diferentes tipos de ataques, conforme identificado pela técnica no dia 29/04.

Também avaliamos a sequência de Sistemas Autônomos (ASes) contactados a partir da URL para alcançar o conteúdo da página de *phishing*. Mais especificamente, para cada URL, extraímos a sequência $S_{ip}$ de endereços IP acessados na porta 80 em ordem cronológica no registro TCPDUMP daquela URL. Em seguida, para cada endereço IP na sequência $S_{ip}$, realizamos o mapeamento para o seu respectivo Sistema Autônomo utilizando a base do Team Cymru[6], produzindo uma nova sequência $S_{as}$ – endereços IP sem mapeamento não são considerados em $S_{as}$. Por fim, substituímos ocorrências consecutivas de um mesmo AS $x$ em $S_{as}$ por uma única ocorrência do AS $x$. Por exemplo, na sequência $\{AS_1, AS_2, AS_2, AS_2, AS_3, AS_2\}$ substituímos as três ocorrências do $AS_2$ na sequência por uma única ocorrência do $AS_2$, resultando em $\{AS_1, AS_2, AS_3, AS_2\}$. Chamamos a sequência final de $S_{as}$ como *cadeia de redirecionamento*.

A figura 9 mostra a distribuição acumulada do tamanho das cadeias de redirecionamento pelas URLs que conseguiram atingir uma página Web. Podemos observar que 41% das URLs analisadas utilizam algum tipo de redirecionamento que verificamos ser JavaScript (75%) e HTTP (25%). Além disso, como diferentes URLs podem apontar para uma mesma página, verificamos a distribuição da quantidade de cadeias de redirecionamento únicas em páginas com mais de uma URL. A figura 10 mostra que apenas 25% das páginas com mais de uma URL possuem uma única cadeia de redirecionamento. Isso sugere que as diversas URLs que apontam para uma página passam por diferentes rotas na Internet e podem estar sendo utilizadas pelo *phisher* para incrementar a robustez de acesso à página Web como, por exemplo, variando as URLs entre os e-mails enviados.

Para melhor entender o tipo de sistema autônomo presente nessas cadeias, extraímos os ASes iniciais e finais de cada cadeia e consideramos o AS inicial igual ao AS final em cadeias com um único AS. Para identificar o tipo de negócio de cada AS, utilizamos a base de classificação de ASes da CAIDA [7]. A figura 11 mostra a distribuição

[6]http://www.team-cymru.org/

[7]https://www.caida.org/data/as-classification/

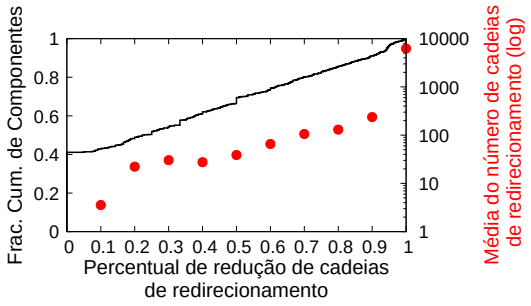


**Figura 12. Agregação de cadeias de redirecionamento nos componentes.**

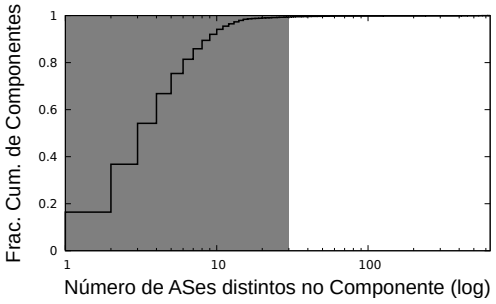


**Figura 13. Distribuição de URLs únicas das cadeias de redirecionamento nos componentes.**

acumulada dos ASes iniciais e finais presentes nas cadeias de redirecionamento observadas. É possível notar que 50 ASes iniciais e finais estão presentes em 70% e 80% das cadeias de redirecionamento das URLs, respectivamente, e são, em sua maioria, ASes de conteúdo como indicado pelos círculos em cada curva da figura 11. Esse resultado corrobora o fato que *phishers* podem estar fazendo uso de computação em nuvem para hospedar páginas de *phishing* utilizando cartões de créditos fraudados. Isso sugere que esforços nessas redes, como a instalação de algoritmos de identificação de páginas de *phishing* na rede e políticas mais rígidas na identificação do usuário que hospeda uma página, podem levar ao rápido bloqueio e a diminuição dos prejuízos causados por ataques de *phishing*.

### 4.2. Análise dos Componentes

Nesta seção avaliamos a infraestrutura de hospedagem dos componentes. A figura 12 mostra o percentual de redução do número de cadeias de redirecionamento ao agrupar páginas similares, i.e., a redução do número de cadeias de redirecionamento a serem avaliadas quando consideramos os componentes ($1 - \frac{Cadeias\ unicas}{Total\ de\ cadeias}$). Podemos observar que, em 40% dos componentes, o agrupamento de páginas não reduziu o número de cadeias observadas, enquanto uma redução superior a 50% foi observada em 30% dos componentes. Para entender a eficácia das reduções observadas, analisamos a média do número de cadeias de redirecionamento nos componentes do intervalo $[x - 0.1, x]$. Representamos essa média com um círculo em $x$ na figura 12. Observamos que os maiores percentuais de redução ($x > 0.6$) ocorrem em componentes com cem ou mais cadeias de redirecionamento. Por exemplo, um componente com quase 10.000 cadeias de redirecionamento obtidas de suas páginas teve redução de quase 98%, levando a poucas dezenas de cadeias de redirecionamento. Isso mostra a eficácia da metodologia proposta neste artigo para agrupar URLs similares.

Na figura 13 verificamos o número de ASes distintos nos componentes provenientes das cadeias de redirecionamento de suas páginas. Verificamos que cerca de 95% dos componentes possuem 10 ASes ou menos relacionados à infraestrutura de redirecionamento e hospedagem das páginas. Além disso, observamos o tipo de negócio dos ASes em cada componente. Verificamos que componentes com poucos ASes possuem, em sua maioria, ASes de conteúdo (área sombreada na figura 13). Isso reforça novamente a hipótese que *phishers* podem estar fazendo uso de computação em nuvem para hospedagem das páginas utilizando cartões fraudados. Para componentes com grandes quan-

tidades de ASes (aŕea não sombreada com $x > 30$), verificamos que o tipo de negócio da maioria dos ASes é provedor de infraestrutura. Neste caso, existe a possibilidade do *phisher* estar utilizando máquinas vulneráveis de usuários para hospedar o conteúdo de páginas de campanhas de *phishing*, o que explica a grande diversidade de ASes relacionados a páginas similares.

## 5. Trabalhos Relacionados

Mensagens de *phishing* atuais carregam um alto nível de personalização que induz o usuário a acessar e revelar informações pessoais para uma página falsa. O alto nível de personalização presente nas mensagens de *phishing* diminuem a eficácia de métodos utilizados na detecção de *spam* [Almomani et al. 2013]. Dessa forma, trabalhos recentes vêm aprimorando técnicas existentes, identificando novos atributos e aplicando novos algoritmos para classificar mensagens de *phishing* [Las-Casas et al. 2016, Smadi et al. 2018], enquanto outros exploram novas frentes de combate, como a criação de alertas de *links* suspeitos em mensagens de e-mails [Volkamer et al. 2017] e a filtragem colaborativa, que utiliza a classificação de usuários sobre uma mensagem para tomar decisões sobre filtragem [Higbee et al. 2016]. Nosso trabalho vai além dos trabalhos existentes ao propor uma metodologia completa para a análise e extração de informações de URLs presentes em mensagens de *phishing*.

Os trabalhos de agrupamento de e-mails maliciosos mais próximos à nossa proposta são (i) Li & Hsieh [Li and Hsieh 2006], que agrupa mensagens utilizando o endereço IP dos *links* presentes nas mensagens, (ii) Fazzion *et al.* [Fazzion et al. 2014], que correlaciona atributos de rede e de conteúdo das mensagens para identificar infraestruturas de envio pertencentes ao mesmo *phisher* e (iii) Shoeb *et al.* [Shoeb et al. 2015], que identifica campanhas de *spam* utilizando o endereço IP final da URL presente nas mensagens. Nosso trabalho se diferencia desses trabalhos pois foca na identificação de campanhas de *phishing* através da análise do conteúdo de URLs dessas campanhas. Além disso, estendemos o entendimento da infraestrutura de hospedagem utilizada pelo *phisher*, possibilitando que analistas e operadores de rede possam identificar a infraestrutura de hospedagem subjacente e reduzir os prejuízos gerados.

## 6. Conclusão e Trabalhos Futuros

Neste trabalho apresentamos uma metodologia de agrupamento e análise de URLs presentes em e-mails de *phishing*. Mostramos como extraímos URLs maliciosas e identificamos o conteúdo das páginas apontadas pelas URLs, mesmo que o acesso esteja ofuscado por cadeias de redirecionamento. Para identificar campanhas de *phishing*, propomos um método baseado em grafos que combina informações do conteúdo com palavras tipicamente utilizadas em mensagens de *phishing* para calcular a similaridade entre as páginas identificadas. Nossas técnicas mostram que a infraestrutura de campanhas de *phishing* se concentra em redes de conteúdo, indicando que políticas mais rígidas de hospedagem de conteúdo podem contribuir no combate ao *phishing*. Além disso, mostramos que, em menor frequência, existem *phishers* que podem estar utilizando máquinas vulneráveis de usuários legítimos para a hospedagem de conteúdo malicioso. Como trabalhos futuros, pretendemos estender a metodologia proposta para outros idiomas. Além disso, realizar um estudo comparativo com e-mails de *spam* para estabelecer diferenças e similaridades na infraestrutura de hospedagem entre esses dois tipos de abusos.

## Agradecimentos

Este trabalho foi parcialmente financiado pelo NIC.br, RNP/CTIC (2955), FAPEMIG, CNPq, CAPES, e EUBra-Atmosphere (H2020-EU.2.1.1 777154).

## Referências

Almomani, A., Gupta, B., Atawneh, S., Meulenberg, A., and Almomani, E. (2013). A survey of phishing email filtering techniques. *IEEE Communications Surveys & Tutorials*, 15(4):2070–2090.

Calais, P., Pires, D. E., Neto, D. O. G., Meira Jr, W., Hoepers, C., and Steding-Jessen, K. (2008). A campaign-based characterization of spamming strategies. In *Proc. in Conference on Email and Anti-Spam (CEAS)*.

Chun, B., Culler, D., Roscoe, T., Bavier, A., Peterson, L., Wawrzoniak, M., and Bowman, M. (2003). Planetlab: An overlay testbed for broad-coverage services. *SIGCOMM Comput. Commun. Rev.*, 33(3):3–12.

Fazzion, E., Las-Casas, P. H., Fonseca, O., Guedes, D., Meira Jr, W., Hoepers, C., Steding-Jessen, K., and Chaves, M. (2014). Spambands: uma metodologia para identificaçao de fontes de spam agindo de forma orquestrada. In *Proc. of Brazilian Symposium on Information and Computational Systems Security (SBSeg)*.

Higbee, A., Belani, R., and Greaux, S. (2016). Collaborative phishing attack detection. US Patent 9,398,038.

Khonji, M., Iraqi, Y., and Jones, A. (2012). Enhancing phishing e-mail classifiers: A lexical url analysis approach. *International Journal for Information Security*.

Las-Casas, P. H., Fonseca, O., Fazzion, E., Hoepers, C., Steding-Jessen, K., Chaves, M., Cunha, Í., Meira Jr, W., and Guedes, D. (2016). Uma metodologia para identificação adaptativa e caracterização de phishing. In *Proc. of Brazilian Symposium on Computer Networks and Distributed Systems (SBRC)*.

Li, F. and Hsieh, M.-H. (2006). An empirical study of clustering behavior of spammers and group-based anti-spam strategies. In *Proc. in Conference on Email and Anti-Spam (CEAS)*.

Shoeb, A. A. M., Mukhopadhyay, D., Al Noor, S., Sprague, A., and Warner, G. (2015). Spam campaign cluster detection using redirected urls and randomized sub-domains. In *Social Informatics (Harvard)*.

Smadi, S., Aslam, N., and Zhang, L. (2018). Detection of online phishing email using dynamic evolving neural network based on reinforcement learning. *Decision Support Systems*, 107:88–102.

Steding-Jessen, K., Vijaykumar, N., and Montes, A. (2008). Using low-interaction honeypots to study the abuse of open proxies to send spam. *INFOCOMP*, 7(1).

Vergelis, M. and Kostin, A. (2018). 2018 fraud world cup. [https://securelist.com/2018-fraud-world-cup/85878/ (Kaspersky Lab)].

Volkamer, M., Renaud, K., Reinheimer, B., and Kunz, A. (2017). User experiences of torpedo: Tooltip-powered phishing email detection. *Computers & Security*, 71:100–113.