Université Grenoble Alpes

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

Spécialité : **Informatique**
Arrêté ministériel : 25 mai 2016

Présentée par

Sourena MAROOFI

Thèse dirigée par **Andrzej DUDA**

et coencadrée par **Maciej KORCZYŃSKI**

Préparée au sein du **Laboratoire d'Informatique de Grenoble (LIG)**, dans **l'École Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique (EDMSTII)**.

Classification, Detection and Prevention of Domain Name Abuse

Thèse soutenue publiquement le **15 Novembre 2021**, devant le jury composé de :

Vivien Quéma
Professeur, Grenoble INP-ENSIMAG, Examinateur

Laurent Toutain
Professeur, IMT Atlantique, Rapporteur

Philippe Owezarski
Directeur de recherche, LAAS, CNRS, Rapporteur

Maciej KORCZYŃSKI
Maître de conférences, Grenoble INP-Ensimag, Invité

Andrzej Duda
Professeur, Grenoble INP-Ensimag, Directeur de thèse

Abstract

Domain name system (DNS) plays an important role in the today's Internet infrastructure. Thousands of domain names are registered every day for different purposes. Among those, some are registered daily by miscreants and abused in different forms, such as spam campaigns, phishing attacks, and malware distribution. This thesis presents three contributions related to DNS abuse, namely i) designing a DNS reputation system to distinguish maliciously registered domain names from compromised ones, ii) evaluating the existing anti-phishing entities and their performance regarding advanced phishing attacks using evasion techniques, and iii) evaluating the adoption of email security extensions to prevent email spoofing.

Keywords

DNS, Domain name abuse, phishing, malware, malicious domain registration, compromised domain names, email spoofing, email security extensions.

Résumé

Le système de noms de domaine (DNS) joue un rôle important dans l'infrastructure Internet actuelle. Des milliers de noms de domaine sont enregistrés chaque jour à des fins différentes. Parmi ceux-ci, certains sont enregistrés quotidiennement par des personnes mal intentionnées qui en abusent de différentes façons, telles que des campagnes de spam, des attaques de phishing et la distribution de malwares. Cette thèse présente trois contributions liées à l'abus de DNS, à savoir : i) la conception d'un système de réputation DNS pour distinguer les noms de domaine enregistrés de manière malveillante des noms de domaine compromis, ii) l'évaluation des systèmes anti-phishing existants et de leurs performances face aux attaques de phishing avancées utilisant des techniques d'évasion, et iii) l'évaluation de la portée d'adoption des extensions de sécurité pour les e-mails afin d'empêcher l'usurpation d'adresses e-mail.

Mots-clefs

abus de DNS, domaine malveillant, domaine compromis, phishing

Contents

Contents 7

List of Figures 9

List of Tables 11

List of Acronyms 13

List of Publications 15

Foreword 17

Introduction 19

1 COMAR: Classification of Compromised versus Maliciously Registered Domains 27
- 1.1 Introduction . . . 30
- 1.2 Related Work . . . 32
- 1.3 Domain Life Cycle . . . 34
- 1.4 Methodology . . . 36
  - 1.4.1 Data Collector Module . . . 36
  - 1.4.2 Features . . . 39
  - 1.4.3 Further Notes on Features . . . 46
  - 1.4.4 Handling Missing Values . . . 46
- 1.5 Experimental Results . . . 48
  - 1.5.1 Ground-Truth Datasets . . . 48
  - 1.5.2 Classifier . . . 49
- 1.6 Evaluation of the Results . . . 51
  - 1.6.1 Comparing COMAR with APWG Method . . . 51
  - 1.6.2 Feature Analysis . . . 53
  - 1.6.3 Case Studies . . . 57
- 1.7 Conclusion . . . 59
- 1.8 Evaluation Metrics . . . 59
- 1.9 Phishing and Malware Datasets . . . 60
- 1.10 Evasion Techniques . . . 61
- 1.11 Captcha Evasion Technique . . . 63

2 Are You Human?: Resilience of Phishing Detection to Evasion Techniques Based on Human Verification 65
- 2.1 Introduction . . . 68
- 2.2 Related Work . . . 69
- 2.3 Phishing Detection and Anti-Analysis Techniques . . . 70

2.3.1 Server-Side vs. Client-Side Detection . . . 70
2.3.2 Alert Box Evasion . . . 71
2.3.3 Session-Based Evasion . . . 73
2.3.4 CAPTCHA-Based Evasion . . . 74
2.4 Experiment Methodology . . . 75
2.5 Experiment Results . . . 78
2.5.1 Preliminary Test . . . 79
2.5.2 Main Experiment . . . 80
2.6 Testing Client-Side Anti-Phishing Extensions . . . 81
2.6.1 Discussion . . . 81
2.7 Conclusion . . . 82

3 Adoption of Email Anti-Spoofing Schemes: A Large Scale Analysis 85
3.1 Introduction . . . 88
3.2 Related Work . . . 90
3.3 Background on Anti-Spoofing Schemes . . . 91
3.3.1 SPF – Sender Policy Framework . . . 92
3.3.2 DKIM structure . . . 94
3.3.3 DMARC . . . 95
3.3.4 Threat Models . . . 96
3.4 Methodology for analyzing Sender Policy Framework (SPF) and DMARC deployment 97
3.4.1 Global Measurements . . . 97
3.4.2 Top 500 Websites of All Countries . . . 98
3.4.3 Defensive Registrations . . . 98
3.4.4 Subdomain Enumeration . . . 99
3.4.5 Banks and Financial Websites . . . 100
3.5 Results on SPF and DMARC Adoption . . . 100
3.5.1 Global Scan of the SPF and DMARC Rules . . . 100
3.5.2 High-Profile Domains and Defensive Registrations . . . 101
3.5.3 Analysis of Spoofing Possibilities for Subdomains . . . 102
3.5.4 SPF Emulation Results . . . 103
3.5.5 End-to-End Spoofing Measurement . . . 104
3.6 Trust-based Authentication Issue . . . 105
3.7 Methodology for Preventing Domain Spoofing . . . 106
3.8 Remediation . . . 108
3.8.1 Results of the First Notification Campaign . . . 108
3.8.2 Results of the Second Notification Campaign . . . 109
3.8.3 Notes on Notification Campaigns . . . 109
3.9 Conclusion . . . 111

4 Conclusion 113

Bibliography 115

List of Figures

1 DNS query/response procedure 20
2 Different parts of URLs 21
3 DNS registration hierarchy 22

1.1 COMAR system structure. 36
1.2 Website structure of a compromised (left) and a malicious domain (right) with the depth level of 3. 37
1.3 Proportion of the domains vs. the difference between real registration dates and the estimated ones using the proposed method, mean, and median approaches. 48
1.4 (a) The homepage of the 'afrikfinancialgroup.com' captured in the first scan showing a database connection error. (b) The homepage of the same domain name re-visited after 10 days. 49
1.5 Evaluation metrics of phishing (top) and malware (bottom) datasets using logistic regression. 50
1.6 Partial cumulative distribution of the compromised domains after registration. . . . 53
1.7 Distribution of the: 56
1.8 Distribution of the *'internal hyperlink$^+$'* (top-left), *'number of technologies'* (top-middle), *'Bing search results$^+$'* (top-right), *'content length$^+$'* (bottom-left), *'number of passive DNS MX$^+$'* (bottom-middle), and *'TLD maliciousness index'* (bottom-right) features in phishing datasets. Y-axis is log-transform for +. 56
1.9 Venn diagram of the collected URLs from four blacklists. 60
1.10 Home page of the compromised domain (top), Google reCAPTCHA with a fake PayPal logo (middle), and a fake PayPal login page for phishing user's credentials (bottom). 64

2.1 Alert box evasion technique. Alert box protected page (top) and phishing payload targeting PayPal (bottom). 73
2.3 Google reCAPTCHA message evasion technique. Google reCAPTCHA-protected page (top) and the phishing payload targeting PayPal (bottom). 74

3.1 Email sending and receiving procedure. 92
3.2 Generating the list of defensively registered domains. 97
3.3 Methodology for preventing domain spoofing. 107
3.4 Content of the email for the first campaign. 110
3.5 Sequence of the emails we have exchanged with one of the CSIRTs. 111

List of Tables

1.1 Features and their characteristics. Feature types are binary (B) or continuous (C). The availability column shows the availability of features as highly available (high), medium, or low. The source column shows the features defined by us (*new*) or appeared in previous work. 38
1.2 Frequency of TLDs on blacklist resources 39
1.3 Lexical features used in maliciously registered domain names. 40
1.4 Evaluation of the Random Forest (RF), Logistic Regression (LR), and the APWG method on phishing and malware datasets. 51
1.5 Logistic regression coefficients of the significant features for the phishing dataset. . 54
1.6 Logistic regression coefficients of the significant features for the malware dataset. . 55

2.1 Preliminary test results after reporting the Gmail (G), Facebook (F), and PayPal (P) phishing URLs. 78
2.2 Results of the main experiment after reporting phishing URLs. X/Y shows the number of detected URLs (X) out of all submitted ones (Y), where Alert box (A), Session-based (S), or Google reCAPTCHA (R) are used to hide phishing sites from server-side anti-phishing bots. 80
2.3 Client-side anti-phishing extensions. Number of installations is the sum of installations for Chrome and Firefox. X/Y shows the number of detected URLs (X) out of all submitted ones (Y). 80

3.1 Possible results of the SPF `check_host` function and their definitions. 93
3.2 Scan results for SPF rules. 98
3.3 Scan results for DMARC rules. 98
3.4 Specified DMARC action for subdomains with no SPF rule in the `TXT` resource record. 100
3.5 Result of the SPF `check_host` emulation. 100
3.6 Selected syntactically wrong rules that lead to the *Permerror* result in SPF. 103
3.7 Measurements of message delivery to inbox (IN), spambox (SP), or no delivery (ND) for five major email service providers. 103

List of Acronyms

AGD algorithmically generated domain names. 30, 33, 40, 41, 46, 54
APWG anti-phishing working group. 23, 32, 68, 75
ASN autonomous system number. 38
ASP active server pages. 37

BEC Business Email Compromise. 91

C&C Command and Control. 30, 32–34, 45
CA certificate authorities. 45, 47
CAPTCHA Completely Automated Public Turing test to tell Computers and Humans Apart. 24, 68, 74, 82
ccTLD Country code Top-level domain. 88, 97, 99
CMS content management system. 41, 42
CPH Contracted Party House. 22
CSIRT Computer Security Incident Response Teams. 90, 108–112
CZDS Centralized Zone Data Service. 89, 97

DDoS Distributed Denial-of-service. 22
DGA Domain generation algorithm. 34
DKIM DomainKeys Identified Mail. 88, 91, 92, 95
DMARC Domain-based Message Authentication, Reporting, and Conformance. 24, 25, 35, 38, 88–92, 95, 96, 100–103, 105–107, 111, 112
DNS Domain Name System. 19–25, 35, 36, 38, 43, 55, 59, 63, 89, 91–94, 96, 101, 104–106, 112, 113
DNSSEC Domain Name System Security Extension. 76
DoS Denial-of-Service. 91

FQDN Fully Qualified Domain Name. 21, 98

GDPR General Data Protection Regulation. 35, 44, 109
GSB Google Safe Browsing. 33, 70, 74, 80
gTLD generic top-level domains. 88, 97

HTTP Hypertext Transfer Protocol. 37, 38

ICANN Internet Corporation for Assigned Names and Numbers. 21, 97
ISP Internet service provider. 23

LD Levenshtein distance. 40

MCC Matthews correlation coefficient. 50, 60

MDA Mail Delivery Agent. 92
MTA Message Transfer Agent. 91, 92

PHP PHP: Hypertext Preprocessor. 37, 74

RFC Request for Comments. 20, 21
RR Resource Record. 19, 21

SEO Search Engine Optimization. 61, 62
SMTP Simple Mail Transfer Protocol. 88, 90, 92, 95, 96, 102, 106
SPF Sender Policy Framework. 8, 11, 24, 25, 35, 38, 43, 85–98, 100–103, 105–107, 109, 111, 112

TLD Top-Level Domain. 20, 31, 45, 47, 53, 89
TLS Transport Layer Security. 35, 38, 45, 46, 58, 59, 91

URL Uniform Resource Locator. 21

YSB Yandex Safe Browsing. 69

List of Publications

- **Maroofi**, **S.**, Korczyński, M., Hesselman, C., Ampeau, B., & Duda, A. (2020, September). Comar: Classification of compromised versus maliciously registered domains. In 2020 IEEE European Symposium on Security and Privacy (EuroS&P) (pp. 607-623). IEEE.
- **Maroofi**, **S.**, Korczyński, M., & Duda, A. (2020). From Defensive Registration to Subdomain Protection: Evaluation of Email Anti-Spoofing Schemes for High-Profile Domains. In Proc. Network Traffic Measurement and Analysis Conference (TMA) - **Best paper award**.
- **Maroofi**, **S.**, Korczyński, M., & Duda, A. (2020, October). Are You Human? Resilience of Phishing Detection to Evasion Techniques Based on Human Verification. In Proceedings of the ACM Internet Measurement Conference (pp. 78-86).
- **Maroofi**, **S.**, Korczyński, M., Hölzel, A., & Duda, A. (2021). Adoption of Email Anti-Spoofing Schemes: A Large Scale Analysis. IEEE Transactions on Network and Service Management.
- Soussi, W., Korczyński, M., **Maroofi**, **S.**, & Duda, A. (2020, September). Feasibility of Large-Scale Vulnerability Notifications after GDPR. In 2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW) (pp. 532-537). IEEE.
- Le Pochat, V., **Maroofi**, **S.**, Van Goethem, T., Preuveneers, D., Duda, A., Joosen, W., & Korczyński, M. (2020). A practical approach for taking down avalanche botnets under real-world constraints. In Proceedings of the 27th Annual Network and Distributed System Security Symposium. Internet Society.

Foreword

Acknowledgments

This thesis could not have been completed without the support, help and encouragement of many people that have been (and have not been) by my side during this experience.

I express my gratitude to the companies AFNIC (registry of .fr domains) and SIDN (registry of .nl domains) for supporting me in this research, especially to Benoît Ampeau, Partnerships and Innovation Director of AFNIC and Cristian Hesselman, Director of the SIDN Labs, for the support they provided during my research.

I am indebted to Maciej Korczyński and Andrzej Duda for the enthusiasm they transmitted, for their competence, as well as for the richness of their guidelines and their suggestions. The time they spent with me in discussing and reviewing my work was extremely precious.

I would like to thank all members of the jury for having accepted to review my thesis and for the interest they demonstrated in respect to my work.

I am grateful to my colleagues and friends (Mohsen and Saeid) who have shared with me the journey of my thesis from the beginning.

Thanks also to all members of LIG laboratory who have (or have not) contributed in many (or no) ways to the fun and comfort of my stay.

Finally, I want to thank me for all the efforts I did and for never giving up.

My family deserves a special gratitude for the patience and the attention that I enjoyed: it was a priceless assistance throughout this period. I am indebted to my dear wife Sanaz and my beloved mother for having supported me in all moments of my life. This Thesis is dedicated to two groups: i) my family for their endless love, support and encouragement and, ii) those who left me alone on the road of life not due to the fact that they left me but because they taught me that I am completely capable of doing it by myself.

Introduction

"Security is a public good. You don't get as much of it if you try to only protect yourself as you do if you work to protect the whole #community and others do the same"

Tarah M. Wheeler

Domain Name System (DNS), is one of the most essential components of the Internet since 1985. The main and primary purpose of DNS is to map names (e.g., a website name) to numbers (i.e., the IP address) acting as a phone book for the Internet. However, today, DNS plays more than a phone book of Internet. By defining several DNS records known as DNS Resource Record (RR), a DNS server can serve various types of information for different purposes such as authentication and general purpose information.

By the expansion of the Internet and consequently growth of DNS applications in 1990s, more concerns were raised about the security of DNS infrastructures. Nowadays, security considerations play an important roles in designs and implementations of DNS-related systems and infrastructures.

Before we dive deeper into the security measures of the DNS, we briefly discuss the fundamentals of domain name systems, definitions and security concerns. Then, we discuss various types of attacks on DNS-related applications, state of the art solutions and current problems of the existing solutions.

Domain Name System Fundamentals

Domain name system is a distributed database work in a form of client-server scheme. The client part of the scheme is known as `resolver` while the server part of the scheme is usually called `DNS servers`. Basically a resolver can be a server directly trying to look up a name from a DNS server (recursive resolver), a server which asks other servers to look up a domain name on behalf

of it (DNS forwarder) or it can be a piece of software (on the operating system) trying to connect to another server (forwarder or recursive resolver), asking for a domain name which is called a stub resolver in this case. The question a resolver ask a DNS server is called *DNS query* which can be different based on the desired resource and the answer she gets from the server is called DNS *response* [1].

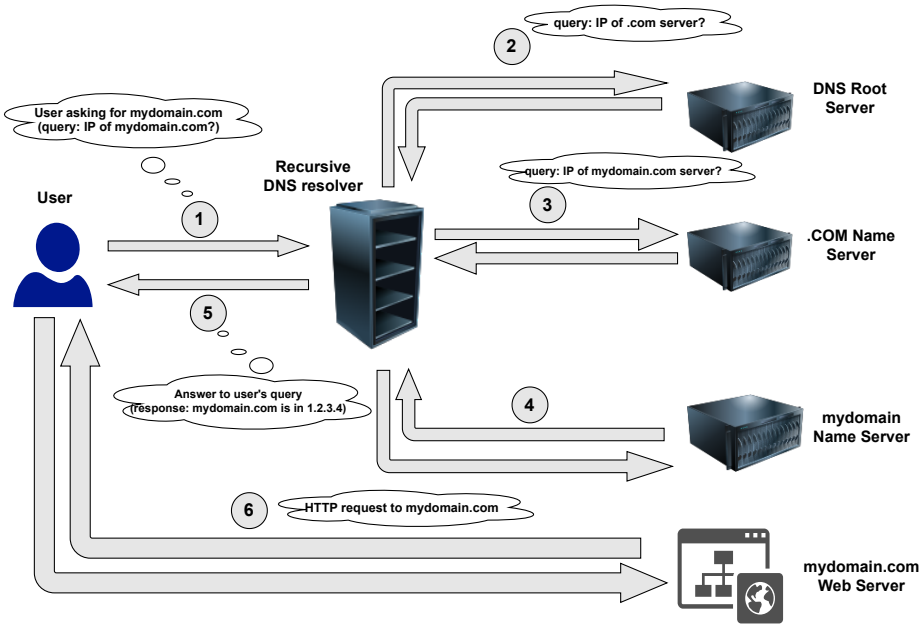


Figure 1 – DNS query/response procedure

Figure 1, shows the DNS resolution procedure. First, the user asks for an IP address of a domain name (e.g., mydomain.com). The *stub resolver* (piece of software installed on the user's operating system) checks the its internal cache system looking for the IP address of the mydomain.com. In case of hit, there is no need to send any request to the *recursive DNS resolver* since the result already exists in the user's system. Otherwise, the stub resolver sends the first request to the *recursive DNS resolver* (usually a server in the local network). The *recursive DNS resolver* repeats the same cache-lookup procedure (as the stub resolver) to answer the user. If there is no cache related to *mydomain.com*, the second request (step 2) is sent to one of the DNS root servers (historically there are 13 DNS root servers). Then, the *DNS root server* returns the IP address of the server which is responsible for all .COM Top-Level Domain (TLD). Again, the *recursive resolver* sends another request to the .COM TLD asking for the name server of the *mydomain.com* domain (step 3). Finally, after retrieving the IP address of the *mydomain.com* name server from step 3, the *recursive resolver* sends the final request to *mydomain.com* name server asking for the IP address of its corresponding web server which is known as the '**A**' record of the domain name.

Apart from the '**A**' record, there are other types of DNS resource records specified in Request for Comments (RFC)1034 and RFC1035 [1, 2] such as **MX** records (for specifying the mail server responsible for accepting email messages on behalf of the domain name), **TXT** record (which can

contain any arbitrary type of text information) and **AAAA** (known as *quad A*) record for specifying IPv6 addresses. While each DNS RR extends the capabilities of domain name systems, it also introduces new security concerns and various attack types which we briefly explain later on this section.

Another aspect of DNS, as part of RFC 1035, is the syntactical definition of a domain name which is important to address to better understand several types of attacks. Figure 2 shows the definition of the Fully Qualified Domain Name (FQDN), registered domain name, subdomain, Uniform Resource Locator (URL) and different parts of the URL specified in RFC 1035 and RFC 3986 [3] and RFC 1738.

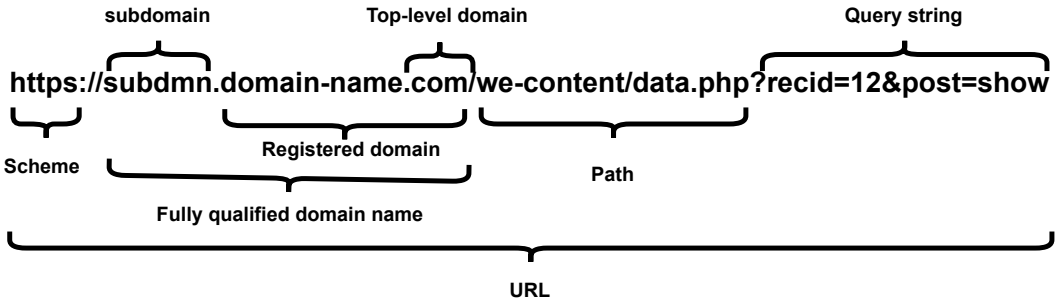


Figure 2 – Different parts of URLs

Domain Name System Entities

Another important aspect of DNS is the entities involved in the domain name registration process. Figure 3, shows the domain name registration hierarchy and all the entities involved. Internet Corporation for Assigned Names and Numbers (ICANN) is at the top of the tree. ICANN is responsible for coordinating the maintenance and procedures of several databases related to the namespaces and numerical spaces of the Internet. The second level of the tree is registries, the organizations which manage top-level domain names (TLDs) such as *.com*, *.net* and *.news*. The third level is registrars which are responsible for selling domain names to public either directly to end-users (known as registrants) or to another smaller companies called resellers (4th level of the tree). Resellers usually provides domain name registrations on behalf of registrars and other facilities such as hosting plans, email management and etc.

It is important to know the role of each entity involved in the domain name registration process since in some specific types of DNS abuse, there is a direct relationship between the mitigation action and the entity which perform the action.

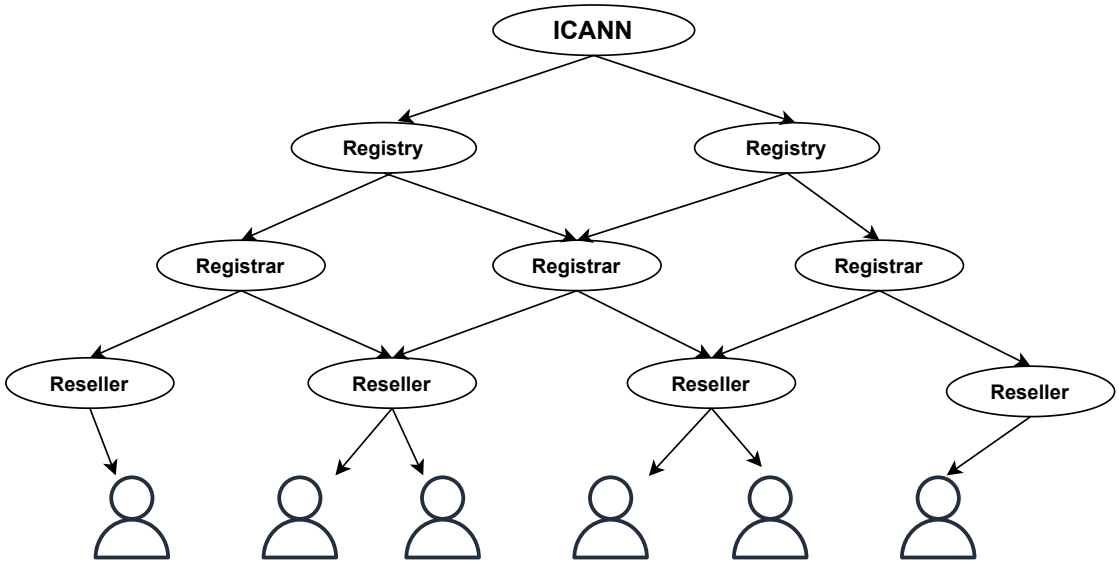


Figure 3 – DNS registration hierarchy

DNS Abuse: Definition and Types

Several organizations and communities defined DNS and DNS abuse in various ways. However, the most common definition which cover almost all of them is the one defined by Contracted Party House (CPH) DNS abuse working group [4] as follows:

> *"DNS Abuse is composed of five broad categories of harmful activity insofar as they intersect with the DNS: malware, botnets, phishing, pharming, and spam when it serves as a delivery mechanism for the other forms of DNS Abuse."*

Based on the above definition, there exist five types of abuse which define as below:

- **Malware**. Any type of malicious software installed on the users device without their permission for various reasons such as gathering sensitive information, gaining access to private networks in different forms of virus, unwanted software, ransomware and spyware.
- **Phishing**. A type of social engineering where the attackers lure the victims to reveal their sensitive information like credentials either by sending spoofed emails or creating fake webpages similar to legitimate web-services.
- **Spam**. Unsolicited bulk email. While spam itself is not a form of DNS abuse, it may lead to other types of DNS abuse (mostly as a delivery method for other types of DNS abuse).
- **botnets**. A collection of connected computers and devices controlled remotely by miscreants to perform activities such as Distributed Denial-of-service (DDoS) attacks.

- **Pharming**. Unwanted redirection of Internet users to malicious and fraudulent sites and services usually through DNS hijacking and poisoning.

Motivations of the Work

The motivation of this work comes from the current, existing solutions for mitigation of the above-mentioned attacks namely: i) Phishing, ii) malware and, iii) spam.

Regarding phishing and malware attacks, current solutions have two steps: i) detection of the attack, ii) blacklisting the malicious content and warning users in order to protect them from the attack. There exists some security organizations and communities such as anti-phishing working group (APWG)[1] who publish the list of known attacks everyday to inform their customers so that they can block the malicious content by blocking the registered domain name. However, based on the characteristics of the phishing and malware attacks, they can be performed either by registering a new domain name or by compromising (hacking) an existing one (mostly legitimate domains with legitimate content) and abusing it. Therefore, in case of compromised domain names, blocking the domain name by e.g., an Internet service provider (ISP), not only block the malicious content (and achieving the final goal) but also block the legitimate content of the domain which may cause probable damages to the underlying business behind the legitimated (but hacked) domain name. Therefore, distinguishing between different types of phishing attacks and their underlying domain registration process is critical to perform an appropriate mitigation action i.e., take down the domain name if we know that the intention of the registration is solely malicious or notifying the hosting provider to remove the malicious content without interupting the usual, legitimate activity of the domain name if we know that the domain has been compromised by attackers and abused to serve malicious content. This gives us the idea of designing a system capable of distinguishing *maliciously registered* and *compromised* domains which we will discuss in more details in Chapter 1.

As we mentioned before, the first phase of the mitigation action is the detection of the attack e.g., detection of a phishing attack. In order to detect a phishing attack, first the detection system should be able to capture the content of the page in order to extract features and finally make the appropriate decision (i.e., phishing or not) based on the extracted features. Therefore, without having access to the page content, it is impossible to make decision about the state of the page. During our investigation about phishing attacks, we noticed that in some cases, attackers using some techniques to make sure that bots and crawlers are unable to visit the phishing content and only humans are able to correctly fetch the malicious content. This is called evasion techniques where attackers reveal the malicious content only to potential victims of the attack. This approach

[1] https://apwg.org

is similar to what is known as Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA). By solving a puzzle that is only solvable by humans, we can make sure that the visitors are not automated bots and crawlers. The idea is to answer the following questions: i) What is the performance of the existing anti-phishing bots regarding pages which are protected with one of the techniques similar to human verification techniques?, ii) How effective these anti-phishing bots are, confronting these phishing pages and, iii) what are the solutions to bypass the evasion techniques. We will discuss this topic in depth in Chapter 2.

Another type of DNS abuse is spam and email spoofing which is very popular among attackers. While email spoofing itself does not consider directly as DNS abuse, it is a very common technique to deliver malware and unwanted software as well as starting point for social engineering and phishing attack. Although various types of solutions proposed and exists to mitigate this type of attack, it is one of the most common techniques to both delivering malware and starting phishing campaigns. Despite the existing solutions for fighting spam, why spamming and email spoofing are used by attackers. This question encouraged us to perform a measurement on email security extensions known as SPF and Domain-based Message Authentication, Reporting, and Conformance (DMARC) which we will discuss in Chapter 3.

Main Contributions

This thesis presents three main contributions:

1. we have studied the existing efforts to detect and blacklist malicious activities and DNS abuse specially phishing and malware distributions. We propose a new method to distinguish maliciously registered and compromised domains to i) more effectivly mitigate the attack based on the intention of the registered domain, ii) create a more effective blacklist feed in order to gain a better insight into attackers behaviour. The proposed method uses only publicly available sources to collect data and assemble a set of features to be used in a machine learning approach to label the abused domain name. We also propose a new method to estimate the domain age when WHOIS data is not publicly available for some registries and registrars. We call the system **COMAR**: Classification of COmpromised versus MAliciously Registered domain. We evaluate COMAR using phishing and malware blacklists and show that it can achieve high accuracy with low false-positive without using any non-publicly available data.

2. We study the resilience of anti-phishing entities to three advanced anti-analysis techniques based on human verification: Google re-CAPTCHA, alert box, and session-based evasion. We design a framework for performing our testing experiments, deploy 105 phishing web-

sites, and provide each of them with one of the three evasion techniques. We show that the performance of 7 major existing anti-phishing engines is very low regarding phishing pages armed with evasion techniques. Finally, we propose solutions for bypassing these anti-analysis techniques which can be implemented in anti-phishing engines.

3. We evaluate the extent of the SPF and DMARC deployment in two large-scale campaigns measuring their global adoption rate with a scan of 236 million domains and high-profile domains of 139 countries. We propose a new algorithm for identifying defensively registered domains and enumerating the domains with misconfigured SPF rules by emulating the SPF check_function. We define for the first time new threat models involving subdomain spoofing and present a methodology for preventing domain spoofing, a combination of good practices for managing SPF and DMARC records and analyzing DNS logs.

Structure of the thesis

This document is split into three chapters. Each chapter is independent of others (can be read in any order), presenting the problem, its importance, related works, methodology, experimental results and finally the conclusion of the chapter.

In particular, the content of the thesis is as follows:

Chapter 1, Design and Implementation of COMAR This chapter provides a background on domain reputation systems and classifications. We present the most recent proposed methods in domain classifications, their caveats and finally we propose our method by explaining the feature collection process, the learning model for the system and presenting the experimental results.

Chapter 2, Phishing and Evasion Techniques. This second chapter zooms in on the resilience of anti-phishing bots against anti-analysis techniques. We introduce three new evasion techniques which are used by attackers in phishing attacks. Then we simulate these attacks by registering more than 100 domain names and creating fake phishing pages to evaluate the performance of anti-phishing bots against these evasion techniques.

Chapter 3, Evaluating the Adoption of Email Security Extensions. This chapter emphasizes on the email security extensions by first introducing them and their mechanism and then evaluating the registered domains to measure the deployment of these extensions. We also propose a method to identify defensively registered domains and finally, we review a scenario in which the domain owner is not only able to mitigate email spoofing but also able to identify the attackers by setting up email security extensions appropriately.

We conclude this work with a conclusion. It summarizes the contributions from the individual chapters and provides future research directions.

Chapter 1

COMAR: Classification of Compromised versus Maliciously Registered Domains

This chapter is adapted from:

- **MAROOFI**, **Sourena**, KORCZYŃSKI, Maciej, HESSELMAN, Cristian, et al. Comar: Classification of compromised versus maliciously registered domains. In : 2020 IEEE European Symposium on Security and Privacy (EuroS&P). IEEE, 2020. p. 607-623.

Contents

1.1 Introduction 30

1.2 Related Work 32

1.3 Domain Life Cycle 34

1.4 Methodology 36

1.4.1 Data Collector Module 36

1.4.2 Features 39

1.4.3 Further Notes on Features 46

1.4.4 Handling Missing Values 46

1.5 Experimental Results 48

1.5.1 Ground-Truth Datasets 48

1.5.2 Classifier 49

1.6 Evaluation of the Results 51

1.6.1 Comparing COMAR with APWG Method 51

1.6.2 Feature Analysis 53

1.6.3 Case Studies 57
1.7 Conclusion 59
1.8 Evaluation Metrics 59
1.9 Phishing and Malware Datasets 60
1.10 Evasion Techniques 61
1.11 Captcha Evasion Technique 63

Abstract

Miscreants abuse thousands of domain names every day by launching large-scale attacks such as phishing or malware campaigns. While some domain names are solely registered for malicious purposes, others are benign domains, which but get compromised and misused to serve malicious content. Existing methods for their detection can either predict malicious domains at the time of registration or identify domains that exhibit malicious behaviour relying on indicators of an ongoing malicious activity conflating maliciously registered and compromised domains into common blacklists to cope with cybersecurity threats. However, the created blacklists also contain compromised domains abused by malicious actors. Since the mitigation actions for these two types are different, there should be a complement method to distinguish between them and perform an appropriate action on each of them. Since the mitigation actions for these two types domains are different, in this thesis, we propose COMAR, an effective classification approach to differentiate between compromised domains from maliciously registered domains, complementary to previously proposed domain reputation systems. We start the chapter with a thorough analysis of the domain life cycle to determine the relationship between each step and define its associated features. In this way, we can capture the characteristics of the benign but compromised and malicious domain registration. COMAR uses a set of 38 features based on publicly available data which we extract from *publicly available* resources and most importantly, robust and costly to evade.

COMAR can be used to create more effective domain blacklists by excluding compromised domains from existing ones. Furthermore, we propose a model to address the problem of missing values in the WHOIS data in real-world scenarios. We evaluate COMAR using phishing and malware blacklists and show that it can achieve high accuracy (97% accuracy with a 2.5% false-positive rate) *without* using any privileged or non-publicly available data, which makes it suitable for the use by any organization. We plan to deploy COMAR at two domain registry operators of the European country-code TLDs and set up an early notification system to facilitate the remediation of blacklisted domains.

1.1 Introduction

Domain names play an important role in almost all types of cybercriminal activities. Miscreants tent to use domains in various attack scenarios such as phishing (e.g., to collect sensitive information) or spam campaigns, or as part of Command and Control (C&C) services with algorithmically generated domain names (AGD). In all these cases, the involved domains are either solely registered for malicious purposes (which we refer to as *malicious* for simplicity) or registered for legitimate reasons but have been compromised at some time to serve malicious content (we refer to these domains as *compromised*).

One common way to fight malicious activities is to build domain blacklists so a security system can check whether a domain exists on the blacklist and decide on how to treat the incoming traffic related to that domain [5]. However, this method is effective when the blacklist only contains the malicious domains because if it includes the compromised ones, the legitimate services associated with the domains may be interrupted and cause financial loss.

At the time of registration, each domain has two possible states: either it is registered for a malicious purpose or a legitimate one. Then, when the domain is active, there are three possible states: (1) *Benign*: the incoming traffic from the domain is benign and can be passed to users safely, (2) *Malicious*: the traffic related to the domain should be considered as malicious and treated differently (e.g., blocked), and (3) *Compromised*: an attacker leverages an arbitrary vulnerability to upload malicious content, e.g., a phishing page. In this way, while the legitimate website is likely to continue serving benign content to its customers, the attacker benefits from the good reputation of the website to conduct her phishing attack. Therefore, the traffic related to the domain can be either malicious or benign.

The existing domain name reputation systems only consider the first two states. They can detect malicious domains either at registration (e.g., PREDATOR [6]) or after they exhibit malicious behavior (e.g., EXPOSURE [7]). However, none of them can detect compromised domains due to two major problems: (1) there is no such state as *compromised* at the registration time, and (2) compromised domains may exhibit the same behavior as malicious domains while they are benign and abused to serve malicious content. In this regard, domain reputation systems may detect a compromised domain as malicious and blacklist it [5]. While this method successfully prohibits malicious traffic, it also blocks the traffic to the legitimate part of the compromised domain. If such a system identifies a compromised domain as benign, it helps attackers achieve their goals. Therefore, in both cases, the decision on the state of the domain may cause collateral damage. For this reason, a complementary system is required to work along with domain reputation systems to differentiate the compromised domains from the malicious ones.

Apart from creating effective domain blacklists, distinguishing compromised from malicious

domains is also important for intermediaries involved in the domain name registration and deployment process. When confronted with a malicious URL, it is critical to assess the registrant's intention for registering the underlying domain since the mitigation action could be different if the registration is for malicious purposes or not. Regarding TLD registries, one appropriate action for malicious domains is domain delisting, i.e., removing the name from the zone file and changing its status to *hold* to completely deactivate it [8]. Another appropriate action is to block access to the domain (*domain blocking*) or redirect the traffic of the domain to another server under the control of authorized entities (also known as domain *sinkholing*), which can be done by registrars. The latter is a popular and widely used technique to identify the victims infected by malware and to reduce its spread [9].

Taking appropriate action against blacklisted domains is also important for hosting providers since hosting malicious content can adversely affect their reputation [10, 11, 12, 13]. Canali et al. studied the reaction of hosting providers when confronted with compromised websites [14]. They showed that in more than 50% of the cases, the reaction of the hosting providers was to suspend (or terminate) users' accounts. For large providers, which may receive hundreds of abuse notifications every day, it is not feasible to manually investigate each case. Therefore, there is a need for a system that can help hosting providers identify the compromised domains and differentiate them from the malicious ones for taking appropriate actions.

Distinguishing between malicious and compromised domains may also lead to revealing the profit-maximizing behavior of attackers. For example, there has been anecdotal evidence indicating that miscreants choose to abuse registrars that offer low domain registration prices [15, 16, 17, 18]. However, no study has systematically proved this conjecture mainly because the existing URL blacklists conflate compromised and malicious domains. One attacker may indeed prefer lower registration prices but, others may choose to abuse a registrar that offers specific payment methods or a free API allowing for domain registration in bulk. On the other hand, registrars might offer cheap domains but, to prevent domain abuse, perform additional checks to confirm the identity of registrants.

In this Chapter, we propose **COMAR** (Classification of COmpromised versus MAliciously Registered Domains), a system capable of differentiating compromised (and misused) domains from the malicious ones to 1) create more effective domain blacklists, 2) help registries, registrars, and hosting providers to take appropriate mitigation actions depending on the abuse type, and 3) gain better insights into the attackers' behavior for choosing candidate domains to hack and intermediaries to abuse.

We thoroughly study the domain life cycle to understand the intentions of both miscreants and benign users and determine the relationship between each step and its associated features.

We use OpenPhish[19], PhishTank[20], APWG[21], and URLhaus[22] as our initial URL blacklist resources, but the system is not only limited to phishing or malware feeds.

Our results illustrate that COMAR achieves high classification accuracy by leveraging only *publicly available* data without relying on any privileged resources like historical WHOIS or passive DNS traffic. We also show how it is possible to compensate for the lack of domain creation time if there is no access to WHOIS information.

In summary, we make the following contributions:

- We develop a system to classify domains exhibiting malicious behavior as either compromised or maliciously registered by *only* using publicly available and readily accessible resources, and achieve 97% accuracy with 2.5% of false positives.
- We leverage 38 features to identify the state of a domain, 14 of which are new and have not been used in previous work.
- We introduce a new method to estimate the domain creation time in cases there is no access to WHOIS information, which outperforms standard statistical methods in filling missing values.
- We show that content-based features are the most important ones in representing the domain status.

1.2 Related Work

Detecting malicious activity from URLs. Several authors proposed techniques in this category, which makes it one of the most prevalent research topic in the field. The main purpose of these methods is to detect phishing pages and malware C&C panels using machine learning techniques.

In case of phishing attacks, Jain and Gupta, for example, proposed a machine learning approach that uses a set of 20 features to identify the input URL as malicious or legitimate [23]. Tian et al. proposed a combination of visual and content-based features to detect phishing attacks [24]. Their assumption is that even if attackers can evade content-based features by using obfuscation techniques, the final appearance of the phishing page should be the same as the target to persuade users to enter their credentials.

Tan et al. proposed a phishing detection technique using lexical, URL-based, and content-based features combined with the Google search engine results to detect phishing URLs [25]. However, in a large scale detection system, it is not feasible to use the Google search engine due to the limitation of the number of requests [26]. COMAR uses some of the features from the above men-

tioned systems but the primary goal of COMAR is not to detect the malicious content of the URL since we create the domain classification system on top of already blacklisted URLs.

Detecting maliciously registered domains. Several effective methods have been proposed in this category. Although the ultimate goal of these methods is not the same as in COMAR, it might still be possible to apply some of the techniques on each domain in the URL blacklists and potentially identify the malicious ones. NOTOS [5] is a reputation system based on passive DNS queries to rank input domains. It extracts 41 features in three categories: 1) network-based, 2) zone-based, and 3) evidence-based features. Except for two features related to the lexical characteristics of the domain name itself, all other ones are derived from the IP address associated with the domain. NOTOS calculates the reputation of IP addresses, networks, and autonomous systems. Therefore, if the domain is behind a reverse proxy system (e.g., CloudFlare[27]), NOTOS is unable to capture the true IP address and instead, it calculates the reputation of the network related to the reverse proxy rather than the reputation of the true network that hosts the domain. Another limitation is that it needs a large passive DNS dataset to perform well. COMAR does not rely on passive DNS queries and even by excluding passive DNS features, it can still obtain high accuracy with low false positive rate. PREDATOR [6] is a proactive recognition method to detect maliciously registered domains at the time of registration. It uses lexical features, IP-based features, and batch registration patterns to identify malicious domains. PREDATOR suffers from the same limitation as NOTOS in confronting reverse proxies. It also heavily uses WHOIS information and historical WHOIS data, which makes it only practical at registries that have access to such data.

Le Pochat et al. proposed an automated method for classifying maliciously registered, algorithmically generated domain names and benign ones that accidentally collide with AGDs, within the constraints of the real-world takedown context of the Avalanche botnets [28]. MENTOR [29] is a system designed to remove benign domains from a blacklist of C&C domains. Both COMAR and MENTOR look for features related to the benign parts of the domains. While the goal of COMAR is to use these features to identify a domain as compromised, the goal of MENTOR is to distinguish benign domains (that are not abused) from malicious ones. One important caveat of MENTOR is the training and testing datasets. The authors used top 500 domains in the Alexa ranking list as the benign dataset. To form the malicious dataset, they used domains from various blacklists double-checked with the Google Safe Browsing (GSB) system. However, top 500 domains in the Alexa list are professionally designed, well-structured websites, which make them inappropriate to be used as fair samples of the benign domains in the wild. Moreover, for the malicious training set, if a domain is labeled by GSB as *'not safe'*, it does not necessarily mean that the domain name is completely malicious, since the goal of the GSB system is to detect malicious content (also hosted on benign but compromised domains) rather than malicious domains.

Detecting malicious activity on compromised domains. The main purpose of these methods is to detect malicious activity on compromised domains. Rao and Pais proposed a technique based on Google search engine queries to detect phishing activity [30]. Apart from the limitation of the number of queries, during the manual labelling of the domains in our dataset, we observed that most of the compromised domains are low ranked websites and many of them had been compromised in the first month of their registration and never got indexed by search engines. Corona et al. [31] proposed 11 content-based features along with image similarity combined using a fusion classifier to detect phishing URLs on compromised websites. We leverage some of their features in our work. However, our ultimate goal is not to detect phishing URLs but to classify domains as maliciously registered or compromised ones. Le Page et al. [32] proposed a method to classify maliciously registered domains and compromised ones. They used 15 features in three categories of lexical (5 features), domain name popularity (3 features), and 7 features related to the Internet Archive. Their results show that features derived from the Internet Archive perform the best among all features. However, relying heavily on the Internet Archive may lead to generate feature vectors with a considerable number of missing values since there is, high likely, no archive history for newly registered domains compromised in a short period after their registration.

1.3 Domain Life Cycle

To understand better the intentions of both malicious actors and benign users for registering and maintaining a domain name, we need to thoroughly inspect the domain life cycle and determine the relationship between each step and its associated features. In this way, we can capture the characteristics of the benign but compromised and malicious domain registration. We divide the domain life cycle into four phases as follows:

L1. Choosing the domain name. In this phase, both miscreants and benign users try to register an appropriate domain name based on their needs. Benign users tend to choose easier to remember, meaningful domain names related to the service provided by the domain. Malicious actors with the purpose of a phishing attack in mind, may try to choose a deceptive name to lure benign users and steal their personal information (e.g., facebook-account.support). In the case of malware C&C panels, miscreants may choose the names that can be generated by the malware family as part of a Domain generation algorithm (DGA). These domain names are likely to be long and meaningless (to increase the chance of availability). We expect spammers to use domains that contain keywords of the targeted service to effectively persuade users to click on the link to increase the click rates and search engine ranks (e.g., earn-bitcoin.biz). The knowledge we gain from this phase can help us to build appropriate lexical features related to the characteristics of

the domain name.

L2. Registration of the domain name. A user (registrant) registers a domain either through a registrar or a reseller by paying the registration fee. The registration period can be between one to ten years depending on the registrars and registries (although shorter registration periods also exist [33]). In this phase, malicious actors tend to choose less expensive (or free) TLDs to maximize their profit [15, 16, 17, 18]. The name of the registrar, domain creation, and expiration dates are stored by registrars and registries as part of WHOIS information. The registrant's information, i.e., the registrant name, address, phone number, are often obscured and not publicly available due to the European General Data Protection Regulation (GDPR) [34]. The COMAR system uses the public part of the WHOIS data as well as TLD-related registration features such as retail domain pricing to discriminate between malicious and compromised domains.

L3. DNS record configuration. After the domain name registration phase, DNS records should be set up to allow the discovery of the services associated with the domain. Each resource record provides information about the service behind the domain name. For example, the DNS 'A' record gives the IP address of the server providing the content for that domain (sometimes, the 'A' record points to a reverse proxy server, responsible for fetching the content from the backend server and delivering it to end-users). The 'MX' records point to mail servers whereas 'DMARC' and SPF 'TXT' records are for giving the email domain owners the ability to protect their domain from unauthorized use. Passive DNS datasets (e.g., Farsight Security [35]) come from monitoring DNS responses and extracting DNS information. For legitimate domains, we expect more stability and *availability* of DNS records while for malicious domains, we expect frequent changes or unavailability of some records (e.g., 'TXT', 'MX', or 'DMARC'). COMAR uses the monitoring approach of passive DNS to construct a feature set, but it *does not* rely on passive DNS datasets since they are not always publicly available. We also use active DNS features DNS records. by querying blacklisted domains.

L4. Service deployment process and its activity period. This step consists of all the activities to set up the necessary infrastructure for the (legitimate or malicious) service offered by the domain. The activities may include setting up a web server, deploying the application to manage the web content, or ordering a Transport Layer Security (TLS)[36] certificate for the domain name to build trust of the service visitors. We expect that legitimate domain owners put the effort in creating content to increase user interest and therefore, the website popularity, i.e., the amount of web traffic the site receives. Miscreants may or may not take the effort of setting up real websites depending on the type of abuse. We also expect to observe more (vulnerable) libraries and technologies to build a legitimate website, which is not required for the correct operation of malicious domains. In this phase, COMAR collects data mainly through a crawl of blacklisted domains and extracts host-based, popularity, and its most important content-based features.

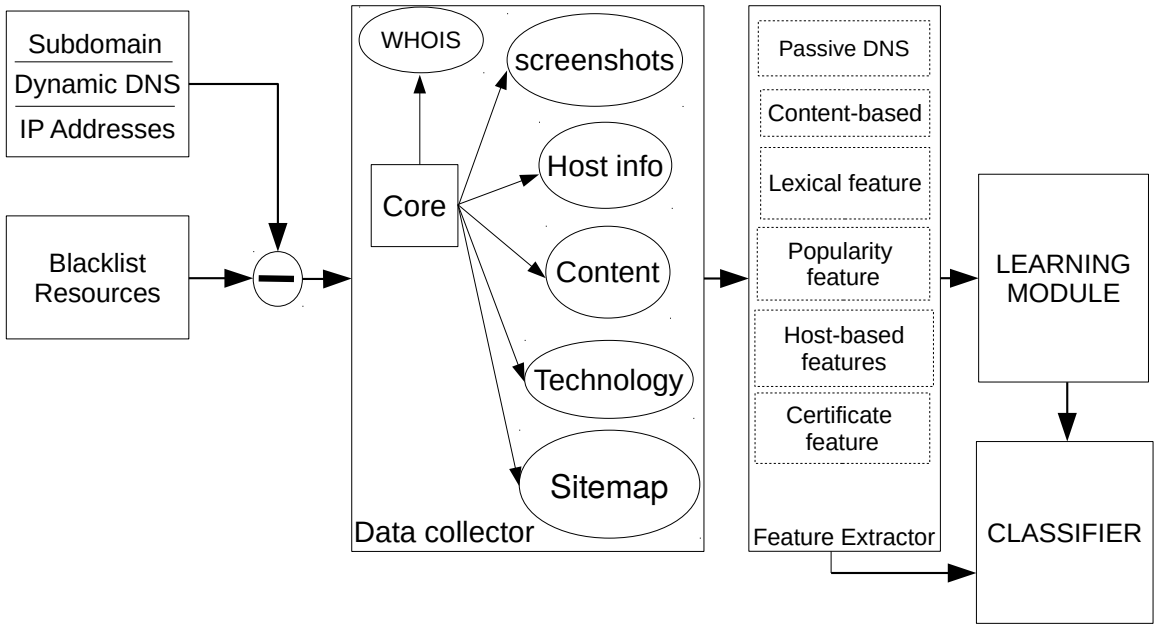


Figure 1.1 – COMAR system structure.

1.4 Methodology

Our system comprises three main modules: 1) data collector, 2) feature extractor, and 3) learning and classification modules. Figure 1.1 presents its structure.

The data collector module gathers data related to the domains derived from URL blacklists. The feature extractor module derives features from the collected data. It can be further used to support efforts of manual labeling domains as maliciously registered or compromised. The learning module takes the labeled data on an input to build a classifier using an appropriate supervised learning technique. Finally, the classification module uses the extracted features and the generated model to classify unlabelled domains derived from URL blacklists in real-time.

1.4.1 Data Collector Module

We use OpenPhish, PhishTank, APWG, and URLhaus as our initial blacklist resources, but the system is not limited to these URL feeds and can use other types of blacklists on input.

The system downloads URL blacklists every 5 minutes to one hour (depending on the blacklist) to get the newly blacklisted URLs. Some URLs are already not operational by the time they are downloaded (domains are taken down or websites are suspended). Some URLs do not contain domain names and use IPv4 addresses instead, whereas some of them use free subdomain services or dynamic DNS services. We use the private part of the public suffix list [37] to exclude dynamic DNS and free subdomain services from further analysis. For each remaining newly appeared URL in the blacklist, we collect the following information:

Technology information. We define technology information as frameworks and libraries used

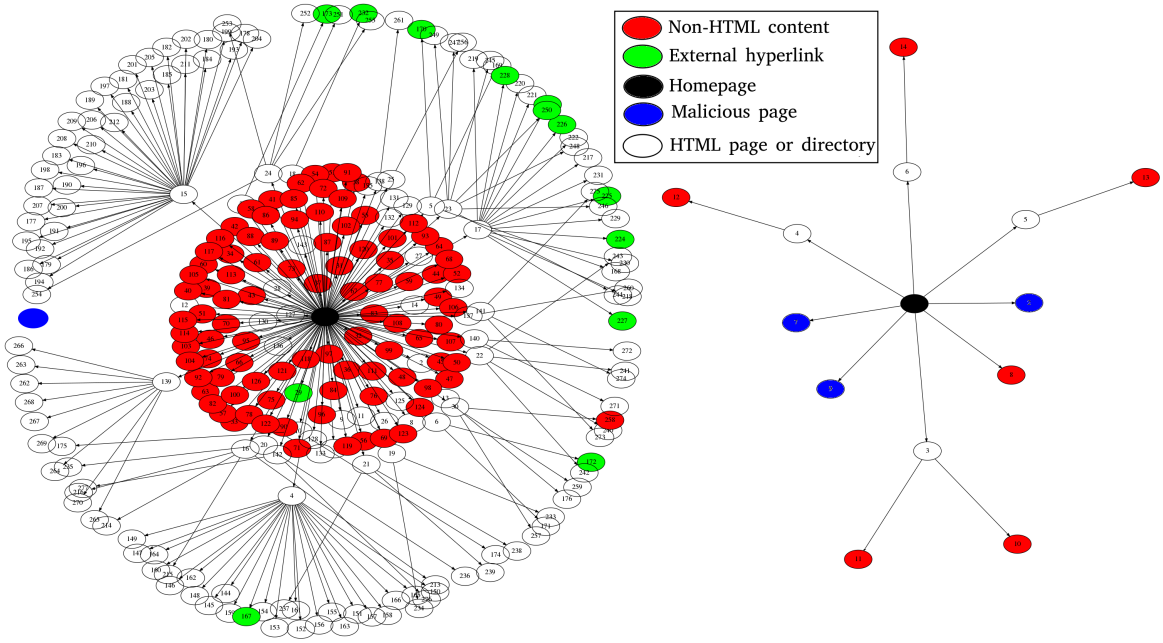


Figure 1.2 – Website structure of a compromised (left) and a malicious domain (right) with the depth level of 3.

to build websites (both client-side libraries like JQuery and WordPress, and server-side technologies like PHP: Hypertext Preprocessor (PHP) or active server pages (ASP) programming languages). To extract such data, we use the Wappalyzer[38] signature list. For each signature in the list, we search in (a) the URL, (b) Hypertext Transfer Protocol (HTTP) headers, and (c) page content to extract all the libraries and tools used to build the website.

Page content. For each domain name, we download the corresponding homepage for further analysis and extracting features. To catch the real content of the domains, which are behind the reverse proxy service (e.g., Cloudflare) with the anti-DDoS feature enabled, we emulate the behavior of a real browser to solve the JavaScript anti-DDoS challenge [39] by using a headless version of the Firefox and Selenium browsers.

Sitemap structure. We further extract all the hyperlinks on the homepage and generate the tree structure of the domain name. For professionally designed websites, the sitemap is often stored in the root directory of the website. However, most of the compromised websites are not well designed, whereas malicious domains rarely have a sitemap file (even if they do, they are not trustworthy). Therefore, we develop our crawler to generate a sitemap for domains. For example, Figure 1.2 shows the website structure of two sample domains with 3 levels of depth for compromised (left) and malicious (right) domains. The black node in the center of the image is the homepage of the domain name. The green nodes are the links to external domain names, the red nodes are the links to non-HTML data types such as PDF or ZIP files, whereas the white nodes are either HTML pages (leaf) or directories (non-leaf). The blue node shows the malicious page. Having this graph, we can extract various information about the website. For example, the num-

Table 1.1 – Features and their characteristics. Feature types are binary (B) or continuous (C). The availability column shows the availability of features as highly available (high), medium, or low. The source column shows the features defined by us (*new*) or appeared in previous work.

Feature#	Type	Availability	Source
(1) - (3)	B	High	[40],[41],[15]
(4)	B	High	new
(5)	C	High	[42]
(6)	C	High	[7]
(7)	C	High	[43]
(8)	B	High	[44]
(9)	B	High	new
(10)	C	Medium	[11]
(11)	B	High	new

Feature#	Type	Availability	Source
(12),(13)	C	High	new
(14)-(16)	B	Medium	new
(17)	B	Medium	[43]
(18)	C	Medium	new
(19)	C	Medium	new
(20)-(24)	C	High	[45]
(25)	B	High	new
(26)	B	Medium	[6]
(27)	C	Medium	[40]

Feature#	Type	Availability	Source
(28)	C	Medium	[46]
(29)-(31)	B	High	[47]
(32)	B	High	[47]
(33)	C	High	[32]
(34)	B	High	new
(35)	C	Low	new
(36)	B	Low	new
(37)	B	High	[43]
(38)	B	Medium	[48]

ber of internal links to pages with different HTML content is higher in the compromised domain compared to the maliciously registered domain due to the fact that compromised domains have legitimate parts for their users. More importantly, most of the time, there is no connection between the phishing page and the homepage in compromised domains since malicious actors do not tend to change the homepage of the compromised domain. Malicious domains have often a connection between the homepage (if there is one) and the malicious page.

DNS resource records. For each domain, we actively collect the 'A', 'AAAA', 'NS', 'TXT', 'SOA', 'DMARC', and 'MX' resource records. Then, using the Maxmind database [49], we convert the 'A' record to the country code and the autonomous system number (ASN) for further use. We also extract the SPF [50] rule from the 'TXT' record if available.

Host information. The host information module is responsible for collecting all the host information related to the input domain (and a possible subdomain) at the time of blacklisting. This information includes the TLS certificates of the domain, the HTTP headers of the web server, the AS number, and its related organization name.

WHOIS data. We collect and parse the WHOIS data, however, we only use the domain creation date in our features. Since this field is not available as part of the WHOIS data for all TLDs, we estimate the missing value based other features (see Section 1.4.4 for more details).

Table 1.2 – Frequency of TLDs on blacklist resources

#	TLD	Frequency	#	TLD	Frequency
0	.com	11846	11	.pl	122
1	.net	805	12	.tk	333
2	.org	702	13	.nl	95
3	.com.br	377	14	.cf	283
4	.info	506	15	.xyz	301
5	.co.uk	311	16	.icu	378
6	.ru	348	17	.top	478
7	.com.au	199	18	.gq	189
8	.ga	239	19	.fr	139
9	.ml	372	20	.de	230
10	.in	260	21	.ca	118

Screenshots. The lifespan of the blacklisted URLs is short [51]. Therefore, for each domain, we save the screenshots of the homepage as well as the malicious URL (and a subdomain if there exists any) for further manual analysis and labeling of domains in case the website has been taken down by registrars, hosting providers, or miscreants.

Table 1.2 shows the most common TLDs used in blacklist resources which have been collected during five months (March to July, 2019). As it can be seen, Freenom[1] TLDs (i.e., .ml, .tk, .gq, .ga, and .cf) are among the most popular TLDs for phishing attacks which indicates that price plays an important role in choosing TLDs by attackers.

1.4.2 Features

The feature extractor module extracts features from the collected data. It operates along with the data collector in a real-time manner to convert plain data into features. In total, we extract 38 features divided into seven main categories (feature set F_1 through F_7) as presented below:

1. Lexical features (F_1)
2. Ranking system and popularity features (F_2)
3. Passive DNS features (F_3)
4. Content-based features (F_4)
5. WHOIS and TLD-based features (F_5)
6. TLS certificate features (F_6)
7. Active DNS features (F_7)

[1] https://www.freenom.com

Table 1.3 – Lexical features used in maliciously registered domain names.

Domain name	Attack type	Lexical features
paypala.com	Phishing	(1) (2) (3)
suportaccount-services.com	Phishing	(4) (5)
3lf4vlxegj1luy6kbs.com	AGD (Rovnix)	(6)
erdoypf-inr.net	AGD (Redyms)	(5)
applid.appsgr.girtrusgirs.com	Phishing	(7)

Table 1.1 shows the characteristics of each feature along with their availability, types (B: binary or C: continuous), and if they appeared in previous work or are defined by us.

1.4.2.1 Lexical features

They are the features extracted from the registered domain (e.g., *example.com*), the subdomain (e.g., *sub.example.com*) as well as the path part of the URL.

Famous brand name in the domain name ($\mathbf{f_1}$). We have identified 231 brand names mostly targeted by attackers in phishing attacks (e.g., PayPal, Amazon, Yahoo, or Gmail). We have created a list of keywords by manually inspecting phishing pages and the corresponding domain names. If the domain name consists of one of these words, it is an indication of maliciousness.

Misspelled target brand name in the domain name ($\mathbf{f_2}$). We use dnstwister[52] to generate possible similar domain names for each of 231 brand names and compare them with the domain name to check the existence of these words. We also consider internationalized domain names and convert the unicode characters to their look alike ASCII equivalent to cover homograph attacks.

Levenshtein distance of the domain name and targets ($\mathbf{f_3}$). We calculate the Levenshtein distance (LD) between the domain name and every 231 targets on our list. We choose $LD = 1$ as the threshold as proposed by Korczyński et al. [15].

Special words but not brand names in the domain name ($\mathbf{f_4}$). Some specific words (e.g., verification, account, support) are not brand names but, based on our word frequency analysis, miscreants tend to use them as part of the domain name to lure victims to enter their credentials. We split the domain name into a word list using the hyphen character. For each word in the domain name, we look for a complete or partial match of that word and our predefined list of 28 keywords. For example, for the domain name 'supportacc-paypal.com', we have one brand name match (i.e., 'paypal') and one special word match (i.e., 'support').

Number of hyphens in the domain name ($\mathbf{f_5}$). The only special character that can be used in a domain name is hyphen ('-'). Both phishing [42] and AGDs (e.g., Redyms malware [53]) tend to use hyphens as part of the domain name.

Digit ratio ($\mathbf{f_6}$). AGDs tend to have more digits than legitimate domain names [7]. This feature is more suitable for domains generated by malware families.

Level of subdomains ($\mathbf{f_7}$). As miscreants control the DNS records of the malicious domains[2], they can create as many subdomains as necessary for a successful attack [43].

Presence of a brand name in the path part of URL ($\mathbf{f_8}$). For compromised domains for which attackers generally do not have access to the domain zone file to create new subdomains, the only way for the malicious actors to use the target brand name is to include it as a part of URL.

Presence of the dot character in the path part of URL ($\mathbf{f_9}$). By manual analysis of blacklisted URLs, we have observed that some malicious actors tend to use the dot character ('.') before file or directory names, for example: *https://masseffect.co.za/.lilman/login.php?cmd=submit*, which may allow the attacker to deceive an unskilled administrator who may not notice the hidden malicious content on the compromised system.

For features $\mathbf{f_1}$-$\mathbf{f_6}$, we only consider the domain name part of the blacklisted URL. Feature $\mathbf{f_7}$ considers the subdomain, whereas $\mathbf{f_8}$ and $\mathbf{f_9}$ are only related to the path part of the URL. Table 1.3 shows the use of selected features in various types of maliciously registered domains.

1.4.2.2 Content-based features

The ultimate goal of domains is to identify a website or a web service that serve content to their customers in various forms. While it is not trivial to examine the content validity, yet it is feasible to extract informative content-based features.

Content length ($\mathbf{f_{10}}$). Malicious domains tend to have less content [11]. For this feature, we only consider the content length of the homepage for each domain part of the blacklisted URL. If there is no index page for that domain (i.e., default directory listing page of the web server), or the web server returns any HTTP code other than the success code (e.g., 404 not found or 403 not authorized), we consider the length to be zero.

Number of used technologies ($\mathbf{f_{11}}$). Using different frameworks and libraries in building a website needs time and effort. The more different technologies used in creating a website, the more time spent on the development. Therefore, we consider the number of used technologies as an indication of the domain being benign. We crawl websites to fingerprint software using unique words and patterns found in the source code. We derive the fingerprints and signatures used in Wappalyzer[56].

Vulnerable technology ($\mathbf{f_{12}}$). It is a binary feature that indicates whether the website uses a technology with at least one known vulnerability. For example, 271 known vulnerabilities have been found in the WordPress content management system (CMS), including themes and plugins

[2] In some types of attacks like zone poisoning [54] or domain shadowing[55], it is still possible for miscreants to change almost any DNS record of a benign domain and generate arbitrary subdomains.

that enable the attackers to upload an arbitrary file to the server [57]. Other familiar technologies with known vulnerabilities are *Joomla* or *Drupal* CMSes, and *Magenta*, *PrestaShop*, or *DotNetNuke* frameworks. The intuition is that if a website uses one of these CMSes, frameworks, modules, or libraries, then there is more chance to get compromised. To obtain the list of technologies with at least one reported vulnerability, we use the exploit [58] and vulnerability databases[59].

Number of internal working hyperlinks ($\mathbf{f_{13}}$). The website with some content is not always benign since miscreants may create fake content on the website so that it looks legitimate. The easiest way for malicious actors is to clone the content of a legitimate website. For each internal hyperlink in the homepage, i.e., a page belonging to the same domain, we fetch the content (only HTML content not files) and calculate the fuzzy hash as proposed by Kornblum [60] to make sure all the pages are not the same and then count the number of unique hashes as the number of working internal hyperlinks.

Content-related domain name ($\mathbf{f_{14}}$). This feature defines the relationship between the content of the homepage and the domain name itself. We extract the meaningful words (based on a dictionary) of the domain name and search for those words in the visible text of the homepage. It is a binary feature with the values 1 (at least one match between a word from the domain name and a related word in the textual content) or 0 otherwise. Another approach would be to use the 'Google trends' service but it is paid and difficult to use on a larger scale.

Presence of the index page ($\mathbf{f_{15}}$). It is common for attackers to upload their files to the web server and just use them without appropriate configuration. In this case, if they forget to upload an appropriate index page (e.g., index.html, index.php, or index.asp depending on the server and server-side programming language), the default behavior of the most web servers (e.g., NGINX or Apache) is to list the directory content. One possibility is that the attacker could remove the index page from the compromised domain but it leads to immediate reaction of the webmaster.

Presence of the default index page ($\mathbf{f_{16}}$). The index pages (homepages) of some domains are the default sites deployed by the registrars, hosting providers, or resellers after the domain name registration process is complete. Resellers often offer free software installation plans like WordPress or Joomla CMSes along with hosting plans. The whole process of installing a CMS on the server takes a few minutes. Sometimes, attackers leverage these free plans to make the domain looks more legitimate. For each domain name, we compare the content of the index page with our pre-defined list of default pages from familiar CMSes and default control panels to check whether the home page is a default page or not.

Using page redirection ($\mathbf{f_{17}}$). Homepage redirection and web cloaking [61] are two common methods among attackers to conceal their malicious intention by displaying benign contents to web crawlers and bots. Regarding homepage redirection, when users try to visit the homepage of

the malicious domain, they will be redirected to a benign website. In case of phishing, the redirection is mostly to the real website of the phishing target (e.g., the real Bank of America website). In case of malware domains, it can be a random website like google.com. To distinguish between page redirection and web cloaking, we crawl each malicious URL with the Selenium browser and with the Python *requests* library. We set this feature to true if the destination URL of the homepage requested using both the browser emulation and the *requests* library shows the same domain name but it is different from the domain name of the blacklisted URL.

Number of external hyperlinks ($\mathbf{f_{18}}$). This feature works the same way as *internal working hyperlinks* but counts the number of hyperlinks that refer to external domains. Sometimes miscreants, especially in phishing attacks, tend to clone the target website to perform a more successful attack. In these cases, the cloned website often has hyperlinks to the real target. Using this feature, we can capture such fake content.

1.4.2.3 Passive DNS features

Passive DNS has become an industry-standard tool for more than a decade. It can give us insights into how the behavior of a domain changes over time (e.g., changing the IP address) and how popular the domain name was in the past. Although the features based on passive DNS data proved to be significant, they can be compensated by other features without lowering accuracy. Therefore, the absence of this feature set does not affect the classification results.

First passive DNS query before the blacklist time ($\mathbf{f_{19}}$). The number of days between the first occurrence of a passive DNS query (for 'A' or 'NS' records) and the blacklist time. This feature provides the estimation of the age of the domain in terms of usage and not only with respect to the registration.

Passive DNS queries ($\mathbf{f_{20}}$-$\mathbf{f_{24}}$). The number of queries for each resource record before appearing in the blacklist resources (i.e., 'A', 'AAAA', 'NS', 'MX', 'TXT' records). For example, the higher the number of observed 'MX' queries, the higher the chance that the domain has an active mail service.

1.4.2.4 Active DNS features

We extract the following features from DNS data queried shortly after the blacklisting time.

Presence of the SPF ($\mathbf{f_{25}}$). 'TXT' records are used (among others) for setting SPF rules [50], domain message authentication reporting and conformance (DMARC) rules [62], and in some cases for domain ownership verification by third-party services (like Google App verification). The presence of SPF for a specific domain can be considered as an indication of legitimacy. For example, a domain owner for whom protection against email spoofing is important would set an appropriate SPF rule in the 'TXT' record [63]. Nevertheless, the malicious actors may also set up SPF rules to

increase domain reputation.

Self-resolving name server ($\mathbf{f_{26}}$). Miscreants may use self-resolving name servers i.e., name servers responsible for resolving their own domain names (e.g., *ns1.domain.com* for resolving *domain.com*) [6], whereas legitimate users tend to use the default DNS resolvers of their DNS service providers.

1.4.2.5 WHOIS features

Due to the introduction of GDPR and our requirement that the proposed method should only depend on publicly available data sources, we only derive the domain creation date from WHOIS and propose the following feature:

Domain age ($\mathbf{f_{27}}$). The older the domain name, the higher the chance to be legitimate. However, according to the 2016 APWG report [40], some miscreants age registered domains waiting weeks or sometimes months before using them. In this way, they can gain reputation for the domain and bypass the detection methods that work based on the registration date. However, according to the report, the number of such domains is low because maintaining a domain name for a long time needs extra effort and money, not always possible for attackers. We use the time lapse between the domain registration and the blacklist dates.

1.4.2.6 Ranking system and popularity features

This feature set consists of 8 features related to search engine results, the Internet Archive [64], and domain name popularity in different ranking systems.

Search engine results ($\mathbf{f_{28}}$). The number of results returned by the Bing search engine for 'site:example.com' queries. The higher the number of results, more popular the domain is. We do not consider Yahoo and Google search engine results because although they are free, with publicly available APIs, the number of requests per day is limited. For example, at the time of writing, the Google custom search engine only allows 100 queries per day. While the Bing search engine is not free (we used the trial version), the price ($3/1000 requests[65]) is low compared to its equivalent alternatives.

Top ranking websites ($\mathbf{f_{29}}$-$\mathbf{f_{32}}$). The presence of the domain name in the Alexa[66], Majestic[67], Quantcast[68], and Umbrella[69] top 1 Million website and domain ranking lists. While we could merge features $\mathbf{f_{29}}$-$\mathbf{f_{32}}$ into a single one based on the Tranco list[47], each of these ranking systems uses its own metrics to calculate domain popularity and therefore, captures different characteristics. We only consider the presence of a domain in such lists as a sign of its popularity.

Wayback Machine ($\mathbf{f_{33}}$). The Internet Archive project started in 1996 by archiving the Internet itself. The sources of the captures come from different plans of the project, e.g., capturing Alexa top

domains, domains that have at least one link from different domains that the Wayback Machine already captured at least one time, and several more plans covering the most part of the Internet [64]. We consider the high number of captures as a sign of benignness for domains.

1.4.2.7 TLD-related features

Chosen TLD is not random among miscreants [15]. They tend to use TLDs based on some factors like the TLD price. We extract two features related to TLD.

TLD maliciousness index ($\mathbf{f_{34}}$). It is a number greater than or equal to zero corresponding to the proportion of abused to all registered domains for each TLD introduced by Spamhaus [70].

TLD price ($\mathbf{f_{35}}$). The price of domains is very important among miscreants since they want to maximize their profit by minimizing the costs. For example, free TLDs (i.e., those provided by Freenom) are among the most common TLDs used in phishing attacks [40, 16].

1.4.2.8 TLS certificate features

COMAR uses three features related to TLS certificates.

TLS certificate price ($\mathbf{f_{36}}$). The purpose of making a TLS certificate available free of cost was to make access to HTTPS available for all websites [71], which means that miscreants can also benefit from it. By using a TLS certificate, attackers can make their attacks look more legitimate (e.g., by showing the green lock in the address bar of the browsers). Free TLS certificates do not either require their owners to provide any personal information. Therefore, the conjecture is that miscreants would prefer to choose free TLS certificates rather than the paid ones.

Presence of TLS certificate ($\mathbf{f_{37}}$). Although the report published by Phishlab [72] shows that almost half of the phishing websites were hosted on domains with an active TLS certificate, we can still leverage this feature since our analysis is not limited to only phishing attacks.

Valid TLS certificate ($\mathbf{f_{38}}$). Trusted but expired TLS certificates or those issued by untrusted certificate authorities (CA)s trigger an error (or a warning) in most standard browsers (e.g., Chrome or Firefox). This behavior may alert victims about an attack. Therefore, most of the phishing URLs are either HTTP or HTTPS with valid certificates. However, for websites that are used in malware spread or domains for C&C panels, the victims are not humans but infected machines. Having a TLS certificate let the infected machines to communicate with their hosts (e.g., bot masters) securely regardless of the validity of the certificates [48]. For each domain, with a TLS certificate, we define a binary feature indicating whether the certificate is valid or not.

1.4.3 Further Notes on Features

So far, we have introduced 38 features in 7 categories. There are some aspects to consider regarding these features.

- Not all features are available for domain names but some features rely on the presence of other ones. For example, all the content-related features are available if there is a homepage available for the domain name. As another example, a TLS certificate price solely relies on the existence of a TLS certificate (i.e., the domain should be HTTPS-enabled). Such dependency enables suitable handling of missing values discussed below in Section 1.4.4.

- For each type of domain abuse, only a specific set of features may be related to that type. For example, lexical features (more specifically, URL-based features) are mostly used in phishing attacks and are not relevant to AGDs. However, we apply all the features in the classifier and let the classifier decide the relevance of each feature. Then, by interpreting the results, we can choose the appropriate feature set for each type of domain abuse.

- Another important aspect of feature engineering is feature evasion, i.e., how robust each feature is against manipulation. In Appendix 1.10, we discuss potential evasion strategies and how difficult they are for attackers to deploy.

1.4.4 Handling Missing Values

Ideally, the classifier operates on a complete ground-truth dataset without missing values. However, in practice, it is not always possible to collect all the features due to several reasons. Two important considerations regarding missing values are their types as explained by Little and Rubin [73], and the reasons for the absence of data. For example, one important feature in our set is the domain age, which depends on the availability of the registration date. However, it is not always feasible to parse the WHOIS data [74]. Some registries do not provide the registration date as part of WHOIS information or WHOIS data at all (e.g., Freenom registries for .ml, .tk, .ga, or the German registry for .de). Therefore, the lack of the registration date means losing important information, which may result in misclassification. The common strategy to fill missing values is to use statistical methods such as the mean (or median) of the feature. However, the mean and median values may lead to biased results since each sample in the dataset (i.e., each registered domain name) is independent of other samples [75]. Another way to fill missing values is to estimate the best value based on the available evidence. In the case of the registration date, although we cannot find the exact date, we can use the earliest day we observed the domain name in the wild. We use the following formula:

$$\min_{date}\{wayback_machine, SSL_certificate, first_pDNS\} \tag{1.1}$$

with respect to the following constraints:

- Regarding passive DNS, we consider the first seen 'A' (or 'NS') record that matches the 'A' (or 'NS') record of the domain name before the time it was submitted to one of the blacklist resources. The justification comes from the possibility that a domain name was registered by someone before, then re-registered by another user (miscreant) and misused.
- Regarding TLS certificates, we use Certificate Transparency logs to retrieve all the previous certificates of the domain (and subdomains, if any) and extract the issuance date of the oldest one that matches the certificate of the domain name before the blacklisting time.

In this way, we obtain the earliest date the domain appeared in the 1) Wayback Machine [3], 2) Certificate Transparency [76, 77], and 3) the passive DNS database [35], which ensures that the real domain registration date is earlier than (or equal to) our estimated value.

Figure 1.3 shows the proportion of the domains for which the difference between the real registration dates and the estimated ones using the proposed method, mean, and median approaches is less than 1 year, between 1 and 2 years, and so on. As the ground-truth data, we use 10,000 domains with different TLDs with known registration dates. For approximately 67% of the domains, the difference is less than one year, while for the mean and median, the result is less than 30%. Furthermore, filling the registration date with the mean for a specific TLD requires to have at least partially the data for that TLD, while for some TLDs (e.g., .ml, .tk), the responsible registries do not provide registration dates at all.

Apart from the registration date, there may be some more missing values in our feature set. For the *'TLD maliciousness index'*, whenever we do not have the data, we fill the value with zero. For *content-related* features, we send requests to domains using the headless version of Selinium and Firefox browsers to mimic user-oriented actions. If we do not get any response from the server, we can assume that the domain has no content to offer to visitors. Therefore, we consider *'content length'*, the *'internal hyperlink'*, and *'external hyperlink'* features as zero. However, in some rare cases, it is possible that the attack type is location-based that either serves the content to specific IP addresses or serves different contents to different IP addresses [78]. In this case, due to our limited resources, we may not be able to fetch the real content. Concerning the TLS certificate price, our approach is to create a binary feature, paid vs. free. However, for some certificate authorities, there is no clear cut boundary between these two options. For example, Comodo CA [79] (also known as Sectigo) offers both free and paid TLS certificates for domains. For these CAs,

[3] https://archive.org

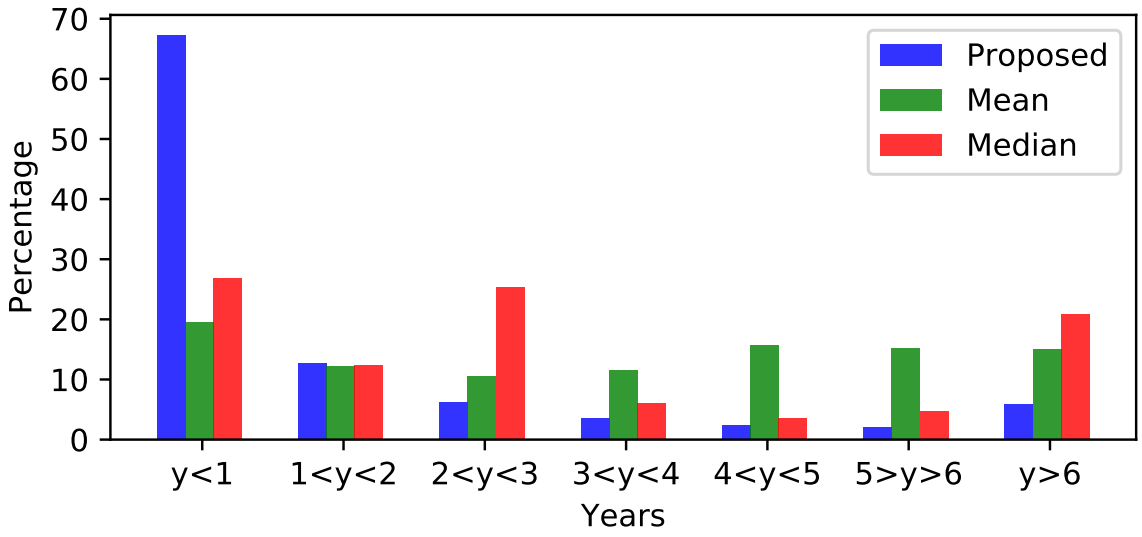


Figure 1.3 – Proportion of the domains vs. the difference between real registration dates and the estimated ones using the proposed method, mean, and median approaches.

we consider the validity period of the certificate. If the validity period is less than three months, then we consider the certificate as free.

In Section 1.2, we compare handling of missing values, data availability, and usage limitations of previously proposed methods with COMAR.

1.5 Experimental Results

In this section, we provide the details of the phishing and malware ground-truth datasets and describe our method to classify compromised and maliciously registered domains.

1.5.1 Ground-Truth Datasets

We have collected 41,002 URLs from four blacklists. Figure 1.9 in Appendix 1.9 shows the number of collected URLs for each blacklist and the overlap between them—it is only the number of working (live) URLs at the time of crawling (March to July, 2019), after removing URL shorteners, free subdomain services, and inactive URLs. Then, we have created two ground-truth datasets from the subset of collected URLs with: 1) URLs from phishing blacklists (APWG, PhishTank, and OpenPhish) and 2) URLs from malware distribution blacklist (URLhaus).

We start with labeling URLs by manually visiting the homepage of the domain and investigating its content and functionality. It is not always trivial even for a human to decide if a domain name is compromised or a malicious one. For instance, it is easy to label 'pyplcompte.fr' (without any homepage and one URL to a fake PayPal login page) as malicious while for 'afrikfinancial-group.com', the domain name does not contain any suspicious word and the registration time is 2017 but looking at the homepage of the domain, there is only a database connection error de-

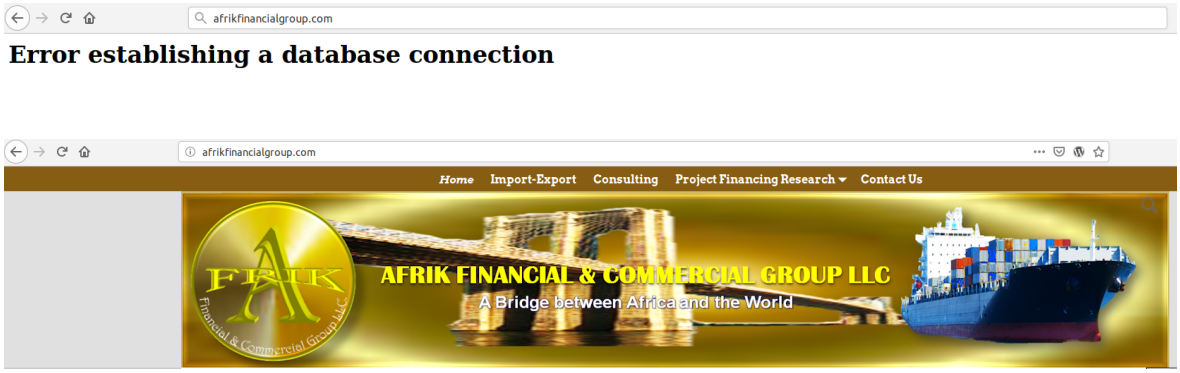


Figure 1.4 – (a) The homepage of the 'afrikfinancialgroup.com' captured in the first scan showing a database connection error. (b) The homepage of the same domain name re-visited after 10 days.

scription (Figure **??**). The error can be the result of an attack or it can be just a simple message to fill the homepage of the maliciously registered domain. To be certain that the chosen label is correct, we re-visit each domain manually after a period of 10 days and check the homepage and the presence of the malicious URL again (the hypothesis is that a 10 day period is long enough for a webmaster to notice that the website is defaced). If the homepage is fixed after 10 days (see Figure **??**), we consider the domain as compromised.

We have manually labeled domains as either 1) maliciously registered, 2) compromised, 3) subdomain/free service, or 4) false positive. Although the data collector module automatically excludes free subdomain services, still some of them, which were not in our predefined subdomains list, appeared in the labeling process.

After removing subdomain services and false positives (i.e., URLs mistakenly blacklisted) the final datasets consist of 1,321 domains from phishing blacklists and 1,008 malware domains from URLhaus. The proportion of the phishing dataset is 58% malicious - 42% compromised and for the malware dataset 57% compromised - 43% malicious.

1.5.2 Classifier

We use two classification methods: 1) Logistic Regression and 2) Random Forest. We apply each method separately on the malware and phishing datasets. We choose the methods because of their characteristics. Logistic regression is a machine learning algorithm that works on linearly separable data and uses the combination of the weighted input features to predict the output class. It is a parametric method known for its efficiency, low computational resources, and interpretability. However, feature engineering plays an important role with respect to its performance.

On the other hand, the random forest is a non-parametric machine learning algorithm capable of training a non-linear model based on the input samples. Generally, it does not need any feature transformation or any assumption about the underlying mapping function. With a sufficient

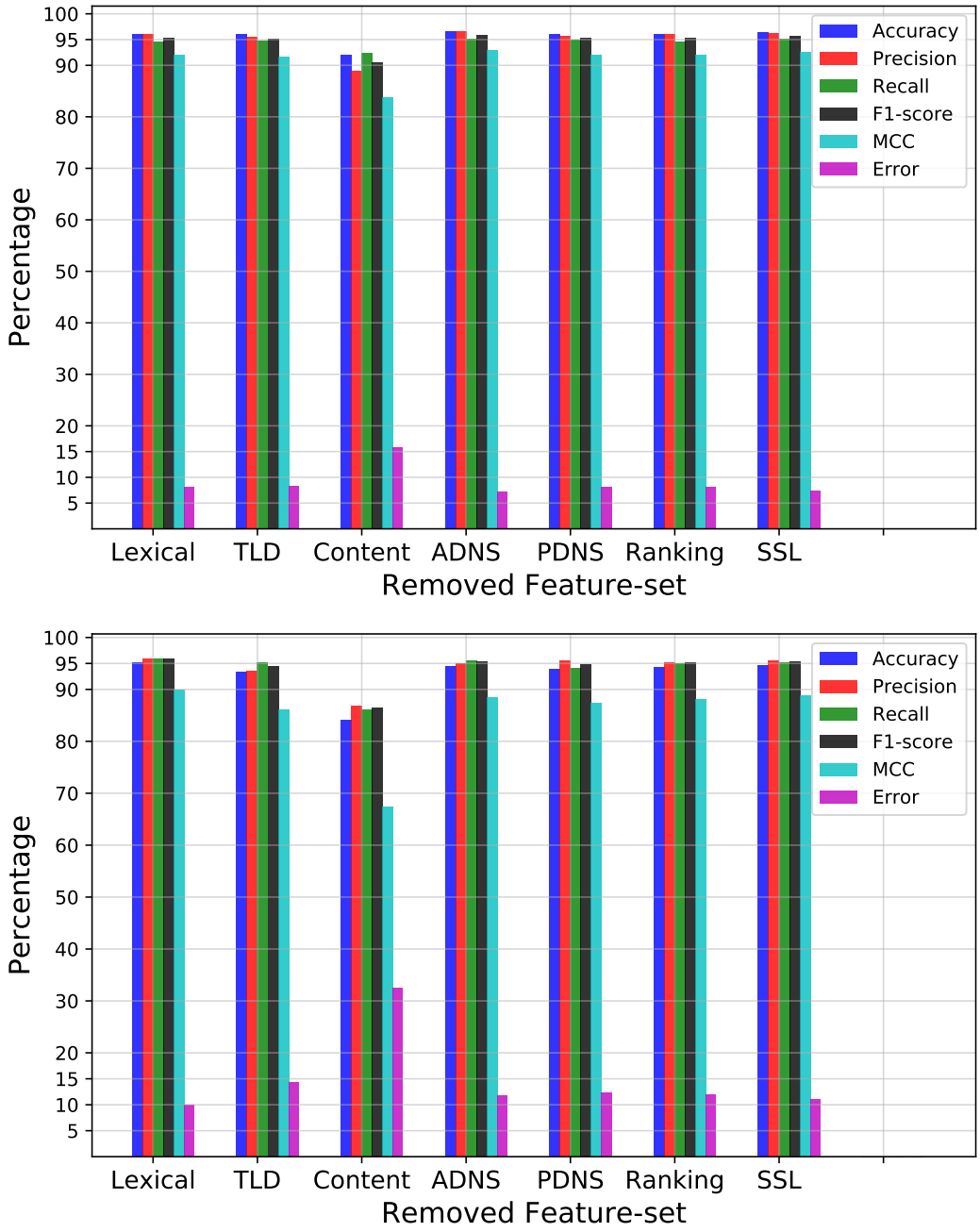


Figure 1.5 – Evaluation metrics of phishing (top) and malware (bottom) datasets using logistic regression.

number of training samples, it may result in a better performance model compared to logistic regression [80]. As for evaluation metrics, we use accuracy, precision, recall, F1-score, and Matthews correlation coefficient (MCC) defined in Appendix 1.8. We use the MCC metric since our datasets are not completely balanced and we also need to consider false positives and false negatives in the final results of the classifier.

Table 1.4 shows the results of the random forest (RF) and logistic regression (LR) classifiers for phishing and malware datasets. We can notice that the classification results of the random forest are slightly better than logistic regression for both datasets. However, we use logistic regression to describe the data and explain the relationship between input features and output classes since it

Table 1.4 – Evaluation of the Random Forest (RF), Logistic Regression (LR), and the APWG method on phishing and malware datasets.

Method	DB	Acc	Precision	Recall	F1	MCC
RF	Phish	97%	95%	97%	96%	0.93
LR	Phish	96.5%	96.59%	95%	95.7%	0.92
APWG	Phish	85%	82%	93%	88%	0.69
RF	Mal	96%	97%	96%	97%	0.92
LR	Mal	94.5%	95.6%	95.2%	95.4%	0.89

can produce interpretable coefficients.

Figure 1.5 shows the classification results by applying logistic regression on the phishing and malware datasets, and eliminating one feature set at a time. We set the maximum number of iterations to 10,000, using 10-fold cross-validation to evaluate the algorithm and ridge regularization to create a less complex model and avoid overfitting. The classification error is the sum of false positives and false negatives. A false positive refers to the malicious domains misclassified as compromised and a false negative refers to the compromised domains misclassified as malicious ones. We can observe that removing content-based features can severely affect the results of the classifier both in phishing and malware datasets, and increase the classification error up to 16% and 30%, respectively. On the other hand, removing the *passive DNS* feature set has almost no effect on the final results (Acc: 96.14%, Precision: 95.78%, Recall: 94.91%, F1: 95.34%, MCC: 0.92 for phishing datasets). Moreover, content-based features are more important for malware samples than phishing. The reason is that most of the maliciously registered domains related to malware spreading or C&C panels have no content in their homepages.

1.6 Evaluation of the Results

In this section, we first compare our results with the simple approach used in the 2016 APWG phishing survey [40] to distinguish between malicious and compromised domains. Then, we study the features extracted and used in the classification process. We analyze the 'strength' of each feature (i.e., how it is related to each output class) and select those with the highest impact on the classification results. This section 1.4.3, where we discussed the robustness of the features, provide a better insight into how we can select the features to create a more effective classifier. We also present three case studies that may influence the classification results.

1.6.1 Comparing COMAR with APWG Method

In the 2016 global APWG phishing survey [40], Aaron and Rasmussen used a simple set of heuristics to distinguish maliciously registered from compromised domains in phishing attacks. They

considered a domain to be malicious if it was reported within a very short time after registration and/or contained a brand name and/or was registered in a batch or there existed a pattern indicating common ownership or intent. Since we do not have access to the registrant's information in the WHOIS data to detect batch registration or any pattern of common intent, we use only the first two conditions to evaluate the APWG method on our ground-truth data. The report did not specify the exact meaning of the 'very short time of being registered', so we chose three months as it is used in the previous study [15]. If the domain has appeared in a blacklist in less than three months of its registration time, or if it has a famous brand name/string in its name, we consider it as a malicious one otherwise it is categorized as compromised.

Table 1.4 shows the classification results of the APWG method. Although the accuracy of the result is relatively high (85%), the false-positive rate is also very high (27%), which results in low MCC (69%). The reason for the high false-positive rate is that the method is unable to detect malicious domains that were registered more than three months before blacklisting and that have no famous brand name or a misleading string as part of the domain name.

In general, there are three limitations of this and other methods that use the registration date as the main feature for classification. As discussed in Section 1.4.4, the registration date is not always available for all TLDs. Therefore, the evaluation is limited to TLDs with the registration date available as part of the WHOIS data. The second drawback is that identifying patterns or evidence of bulk registrations need registration information such as the registrant's name and the address no longer publicly available [34]. Finally, the third caveat of using this heuristic is the fact that it does not consider legitimate domains compromised in the first few months or even days after registration.

Figure 1.6 shows the partial cumulative distribution of the compromised domains after their registration date. We collect the data of hacked websites for 18,810 domains from various resources like accounts of the hacker groups on Facebook, Twitter, and hacking forums for 2 months. The results show that 12% of the domains get compromised in the first three months of their registration, and approximately 32% get hacked in the first year after registration probably because of the lack of appropriate configurations or because the website is still under active development. These types of domains may lead to false-negative results (classifying newly registered benign but compromised domains as maliciously registered). However, COMAR does not suffer from these limitations since it does not heavily rely on the registration dates (only one feature out of the other 38 proposed ones), and we estimate the missing values of the domain registration dates for TLDs that do not provide the WHOIS data.

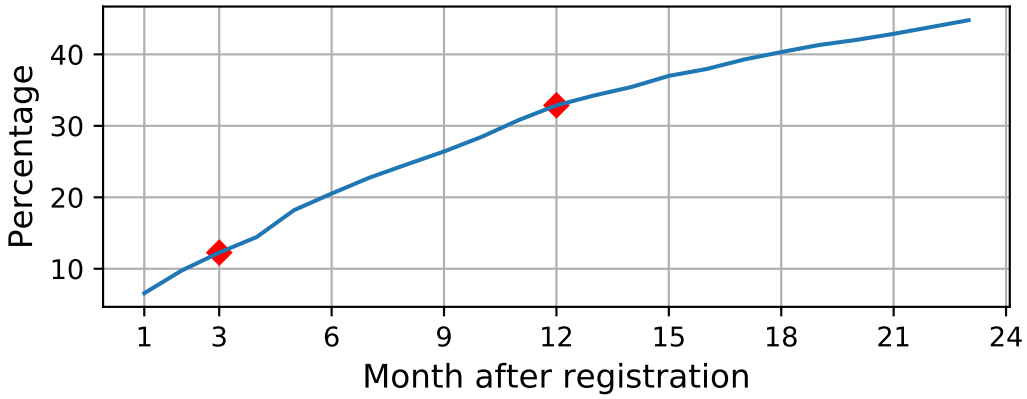


Figure 1.6 – Partial cumulative distribution of the compromised domains after registration.

1.6.2 Feature Analysis

By using logistic regression, we can measure how important individual features are to the overall performance. Table 1.5 and 1.6 show the logistic regression weights for 24 most significant features. We use the L2 norm regularization to keep the weights small to avoid overfitting and reduce model complexity. Moreover, small weights help us making sure that one feature with a large value cannot heavily affect the final classifier result. We also use log transformation for some features (e.g., *'number of Bing result'*) to increase the linearity between the input features and the output class. The sign of each coefficient shows the relationship between the feature and the *compromised* output class. For example, the *'TLD_maliciousness_index'* feature has a negative relationship with the *compromised* (and a positive relationship with the *malicious*) output class. Therefore, a higher maliciousness index of TLD indicates a higher probability of a domain being maliciously registered rather than compromised.

We can observe that the *'number of internal hyperlinks'*, *'number of Bing search results'*, *'number of technologies'*, and *'content length'* features are in the top five strongest features indicating compromised domains for both malware and phishing datasets. The '*number of internal hyperlinks*', '*number of technologies*', and '*content length*' features are content-based the effort the owner (legitimate or malicious) put into creating a fully-featured website. The results support the conjecture that attackers spend less time to deploy a fully-functional website with rich content since it is time consuming.

Content-based features play an important role in the classification: 5 out of 10 most significant features are content-based. The '*number of Bing search results*' is related to domain popularity, which reflects the conjecture that malicious domains are less popular than compromised domains since they have legitimate traffic generated by benign users.

Another interesting feature is *'number of external hyperlinks'* with different signs for phishing and malware datasets probably because phishers tend to copy the entire HTML code of the target

Table 1.5 – Logistic regression coefficients of the significant features for the phishing dataset.

#	Feature	Category	Weights
1	$f_{\text{number of internal hyperlink}}$	Content-based	1.88
2	$f_{\text{number of technology used}}$	Content-based	1.28
3	$f_{\text{Bing search result}}$	Ranking	1.26
4	$f_{\text{content length}}$	Content-based	0.98
5	$f_{\text{first PDNS before blacklist}}$	Passive DNS	0.78
6	$f_{\text{number of PDNS MX}}$	Passive DNS	0.56
7	$f_{\text{TLD maliciousness index}}$	TLD-based	-0.56
8	$f_{\text{domain aging}}$	WHOIS-based	0.49
9	$f_{\text{using redirection}}$	Content-based	-0.46
10	$f_{\text{has vulnerable tech}}$	Content-based	0.41
11	$f_{\text{presence of index page}}$	Content-based	0.39
12	$f_{\text{wayback machine captured}}$	Ranking	0.30
13	$f_{\text{URL has famous brand name}}$	Lexical	0.28
14	$f_{\text{is content related}}$	Content-based	0.21
15	$f_{\text{special word in domain name}}$	Lexical	-0.18
16	$f_{\text{number of external hyperlink}}$	Content-based	-0.17
17	$f_{\text{using HTTPS}}$	SSL-based	0.15
18	$f_{\text{using brand name in domain name}}$	Lexical	-0.12
19	$f_{\text{presence of default homepage}}$	Content-based	-0.10
20	$f_{\text{has SPF}}$	Active DNS	-0.07
21	$f_{\text{self-resolve NS}}$	Active DNS	-0.05
22	$f_{\text{presence in quantcast}}$	Ranking	0.03
23	$f_{\text{using misspelled brand name}}$	Lexical	0.03
24	$f_{\text{presence in umbrella}}$	Ranking	0.02

website to create the exact look and feel experience, and most of the time, the cloned HTML code contains hyperlinks related to different pages of the target website. On the other hand, malware domains (e.g., AGDs) often have less (or no) content, which leads to less (or no) external hyperlinks. Therefore, in this case, having an external hyperlink is the indication of a compromised domain.

For URL-based features (e.g., *'URL has famous brand name'*), we observe a significant decrease from phishing dataset to malware dataset because URL-based features are mostly related to phishing attacks. For example, the weight of *'URL has famous brand name'* is 0.28 for phishing while it is 0.02 for the malware dataset.

Considering ranking and popularity features, the presence of the domain name in four ranking websites (i.e., Alexa, Quantcast, Majestic, and Umbrella) has a weak association with the output class in favor of compromised domains in both datasets. Although these features are less significant and it is not difficult to manipulate these ranking lists [47], still using these features combined with others can provide more accurate results.

The *'presence of HTTPS'* feature has a small weight in the phishing dataset (0.15) and near

Table 1.6 – Logistic regression coefficients of the significant features for the malware dataset.

#	Feature	Category	Weights
1	$f_{\text{number of technology used}}$	Content-based	0.87
2	$f_{\text{number of internal hyperlink}}$	Content-based	0.84
3	$f_{\text{content length}}$	Content-based	0.82
4	$f_{\text{Bing search result}}$	Ranking	0.74
5	$f_{\text{TLD maliciousness index}}$	TLD-based	-0.72
6	$f_{\text{number of PDNS MX}}$	Passive DNS	0.50
7	$f_{\text{wayback machine captured}}$	Ranking	0.50
8	$f_{\text{presence of index page}}$	Content-based	0.19
9	$f_{\text{number of external hyperlink}}$	Content-based	0.18
10	$f_{\text{domain aging}}$	WHOIS-based	0.16
11	$f_{\text{has vulnerable tech}}$	Content-based	0.14
12	$f_{\text{presence of default homepage}}$	Content-based	-0.14
13	$f_{\text{self-resolve NS}}$	Active DNS	-0.13
14	$f_{\text{is content related}}$	Content-based	0.11
15	$f_{\text{presence in umbrella}}$	Ranking	0.05
16	$f_{\text{using HTTPS}}$	SSL-based	0.05
17	$f_{\text{first PDNS before blacklist}}$	Passive DNS	0.04
18	$f_{\text{using redirection}}$	Content-based	0.04
19	$f_{\text{URL has famous brand name}}$	Lexical	0.02
20	$f_{\text{using brand name in domain name}}$	Lexical	0.01
21	$f_{\text{presence in quantcast}}$	Ranking	0.01
22	$f_{\text{using misspelled brand name}}$	Lexical	0.01
23	$f_{\text{special word in domain name}}$	Lexical	0.0
24	$f_{\text{has SPF}}$	Active DNS	0.0

zero (0.05) for the malware dataset, which is not surprising since more than 58% of the phishing attacks used TLS certificates in the first quarter of 2019 according to the phishing activity report [81]. Therefore, the presence of a TLS certificate cannot be considered as a strong feature to distinguish malicious and compromised domains due to the popularity of using TLS certificates among both attackers and legitimate users.

Figure 1.8 shows the distribution of six selected features for each output class for the phishing dataset. For better representation of the distribution, we use logarithmic scales for the *'Bing search result'*, *'number of passive DNS for MX'*, *'content length'*, and *'number of internal hyperlink'* features. For example, in Figure 1.8 (d) the average length of the homepage content for compromised domains is greater than for maliciously registered domains. Looking at Table 1.5, the weight of the *'content length'* feature is 0.98 in favor of compromised domains, which means that more content on the homepage is an important characteristic of the benign but compromised domains.

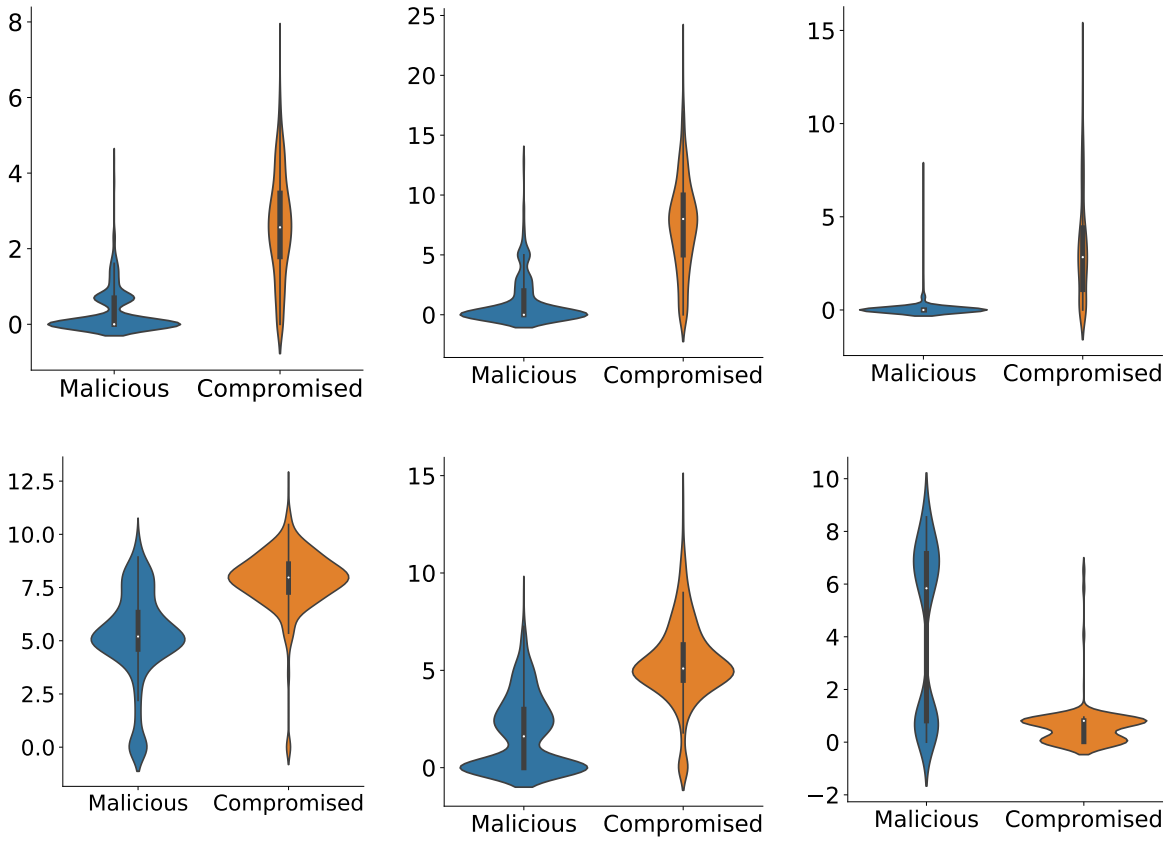


Figure 1.7 – Distribution of the:

Figure 1.8 – Distribution of the *'internal hyperlink^{+}'* (top-left), *'number of technologies'* (top-middle), *'Bing search results^{+}'* (top-right), *'content length^{+}'* (bottom-left), *'number of passive DNS MX^{+}'* (bottom-middle), and *'TLD maliciousness index'* (bottom-right) features in phishing datasets. Y-axis is log-transform for +.

1.6.3 Case Studies

In this section, we present three case studies that may influence the classification results. The first one is related to website defacement when an attacker changes the visual appearance of a website by replacing the index page of the domain. To the best of our knowledge, the second case presents a new technique observed in phishing attacks for the first time. Finally, the third case is related to domain dropcatching in which attackers register expired benign domains to take advantage of their residual trust.

1.6.3.1 Case 1: Homepage defacement

It concerns a compromised domain name registered back in 2017 but detected by COMAR as malicious. We manually investigated the results, visited the homepage of the domain, and compared it with the data and screenshots from the data collection process. When we found the domain name in the OpenPhish blacklist, the homepage of the domain name was defaced and the content replaced by following HTML code:

```
<html><head></head>
    <body>dddddddd</body>
</html>
```

The COMAR classifier uses 9 content-based features (as explained in Section 1.4.2.2). With the replaced homepage, COMAR was not able to extract features effectively, therefore, misclassified the domain as malicious (with the probability of 67.1% in favor of the malicious class).

As a matter of fact, this result is one of the drawbacks of the content-based features. If we do not fetch the real content of the domain for any reason, the classification results are uncertain. However, homepage defacement is very rare since attackers tend to keep the homepage of the compromised domains as intact as possible to avoid early detection by the website owners.

1.6.3.2 Case 2: New anti-phishing evasion technique

Phishers always look for new techniques to extend the lifetime of the phishing pages by evading anti-phishing bots and detection systems. One of the best ways to do so is to use defense techniques like page redirection, web-cloaking or server-side techniques like filtering famous user agents like *googlebots* or known scanners IP addresses [82]. As mentioned in Section 1.5.1, we scan each URL and domain twice, within ten days in between, to make sure that the state of both URL and the domain in the labeling process is correct. During our scan, we noticed an URL labeled as safe by Google safe browsing in both scans. Since the URL was in the Phishtank blacklist, which is a community based URL blacklist based on user reports, we had to manually check it to avoid

a false positive. By visiting the URL, we noticed that the attacker used Google CAPTCHA to hide the real content of the malicious page. Therefore, even the browser emulation technique was not able to fetch the real phishing content unless a human solves the CAPTCHA manually. Figure 1.10 in Appendix 1.11 shows the website homepage, the phishing page protected by Google CAPTCHA, and a fake PayPal login page for phishing the user's credentials. Although COMAR classified correctly the domain as compromised (since we do not use any feature related to the content of the phishing URL), any phishing (fraud) detection system based on the content of the phishing URL cannot probably automatically fetch the page content. This is the first time we observe an evasion strategy using one of the strongest counter-attack techniques (CAPTCHA). Using techniques like CAPTCHA by phishing attackers may raise a serious challenge to security vendors in detection of malicious pages.

1.6.3.3 Case 3: Domain dropcatching

Domain dropcatching is the practice of registering a domain name once it is expired and released for new registration [83]. In this process, it is possible for miscreants to register an already expired benign domain name and inherit its residual trust. Miramirkhani et al. [83] showed that approximately 10% of the dropped domains are picked and registered by attackers for malicious purposes. The problem with these domains is that while they should be treated as newly registered domains (as they are), some of the features will match the original registration leading to misclassification of the domains as compromised. The feature sets concerned by drop-caught domains are TLS certificate, passive DNS, and ranking and popularity features. To show the effect of domain dropcatching, we compared the domain registration date with the date related to the first observed DNS query in DNSDB and the first captured page in the Wayback machine only for domains manually labeled as malicious. Whenever the date of the first captured page in the Wayback machine or first seen DNS query is older than the real registration date, we consider the domain as a drop-caught one. In this way, we found 7 samples in our dataset, 6 of them correctly classified as malicious.

Then, we applied the classifier two times on the samples: first, by removing passive DNS features (since they are affected by dropcaching and COMAR does not heavily rely on them) and then, by removing content-based features (since they can be relatively easily evaded and may affect classification). COMAR misclassified 1 and 2 samples (out of 7 samples) in the first and second experiment, respectively. While the number of samples is not enough to evaluate the generalizability of the method in the context of domain dropcaching, we assume that the benign history of the domain may mislead the classifier. We believe that this situation can be worse when attackers clone the content of the original website using the Wayback machine (we have not observed such

a case in our dataset).

To reduce the negative impact of the drop-caught domains on the classifier, we could improve the Bing search engine result feature ($\mathbf{f_{28}}$) by only retrieving the results for a specific time slots i.e., after the registration date. Regarding the Wayback machine ($\mathbf{f_{33}}$), we already count only the number of captured pages after the domain registration date. However, passive DNS features and the TLS feature set are still heavily affected by the benign history of the domain and in the worst case scenario, attackers could also consider bypassing content-based features by cloning the content of the original website.

1.7 Conclusion

In this Chapter, we present COMAR, a system capable of distinguishing maliciously registered from compromised domains. COMAR leverages publicly available data and makes classification decisions based on the extracted features. Registries, registrars, and hosting providers can use it to decide on appropriate mitigation actions for each domain with malicious content. It can also serve as an effective tool for creating domain blacklists from the existing URL ones.

We show that the content-based features are the most effective in capturing the 'amount of benignness' of domains during their life cycles. We examine features regarding their robustness and the possible ways attackers can bypass them. High cost and effort for attackers complicates the evasion from COMAR and may therefore discourage malicious actors.

We introduce a new technique to compensate missing values in the 'domain registration date' field of the WHOIS data that outperforms the existing methods. We also show that approximately 12% of the domains get compromised in the first three months of their registration, which suggests that domain reputation systems based on the domain age cannot distinguish maliciously registered from compromised domains with high accuracy.

1.8 Evaluation Metrics

We use the following metrics to evaluate our machine learning algorithms.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1.2}$$

$$Precission = \frac{TP}{TP + FP} \tag{1.3}$$

$$Recall = \frac{TP}{TP + FN} \tag{1.4}$$

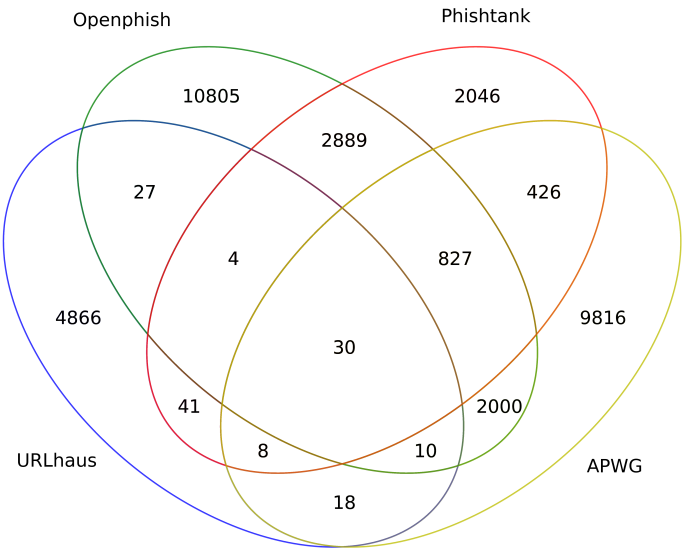


Figure 1.9 – Venn diagram of the collected URLs from four blacklists.

$$F_1 - score = \frac{2TP}{2TP + FP + FN} \tag{1.5}$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}, \tag{1.6}$$

where *TP, TN, FP, FN* are the number of true positives, true negatives, false positives and false negatives, respectively. Compromised domains are considered positive and maliciously registered domains negative. Accuracy is the ratio of the number of correct predictions to the total number of input samples. Precision means the percentage of relevant results. Recall refers to the percentage of total relevant results correctly classified by the algorithm. The F1 score is the harmonic mean of precision and recall.

The MCC [84] is a measure of the quality of binary classification. The return value of MCC is between -1 and +1 which +1 represents a perfect prediction, 0 means random prediction and -1 means total disagreement between the predictions and true labels. The advantages of MCC over accuracy and F1-score is that it considers the size as well as the imbalance of dataset. Most importantly, MCC takes into account true and false positives and negatives (all the entries of the confusion matrix not only true-positives and true-negatives).

1.9 Phishing and Malware Datasets

Figure 1.9 shows the Venn diagram of the collected URLs from a) URLhaus, b) APWG, c) OpenPhish, d) Phishtank, and the overlap between them.

1.10 Evasion Techniques

In Section 1.1, we discussed the appropriate mitigation actions for compromised and maliciously registered domains by different intermediaries. For malicious domains, one recommended action is to take down the domain or suspend the hosting service related to that domain. This action may generate extra costs for malicious actors (losing the domain name or the hosting service), which makes it a good reason to avoid their domain being classified as maliciously registered. However, manipulating COMAR features also requires extra effort. In this section, we examine the possibility of feature evasion and associated costs. We take into account i) the amount of money the attackers should pay to bypass a specific feature, ii) the amount of time the attacker should spend to evade each feature and, iii) the necessary skills the attacker should have to bypass a feature.

Generally, it is safe to consider external features as more difficult to evade compared to the features under the control of the attacker. For example, manipulating search engine results, the Wayback Machine as well as passive DNS data are more difficult compared to content-based or lexical features in case of maliciously registered domains. However, it does not necessarily make external features completely bulletproof against manipulation. Furthermore, any feature with a one-time cost (either in terms of time or money) for the attacker cannot be considered as robust.

1.10.0.1 Content-based features

. In Section 1.6.2, we show that content-based features are among the best ones, yet most available in our set. Through this feature set, we exploit the *benignness* of the domain by analyzing 1) the length of the generated content on the homepage of the domain, 2) the relationship between the homepage and other (possible) pages related to that domain (i.e., the number of internal and external hyperlinks), 3) the amount of effort required by the domain owner to design a professional websites (i.e., the number of technologies that are used to create the website), and 4) the number of technologies prone to attacks. We now consider possible evasion techniques the attacker can use to bypass content-related features.

1. **Content length and hyperlinks**. To bypass the content length feature ($\mathbf{f_{10}}$), the attacker needs to generate lengthy content either manually (which is not feasible in large-scale attacks) or automatically through third-party applications. The same methodology can be applied to features related to internal and external hyperlinks (i.e., $\mathbf{f_{13}}$ and $\mathbf{f_{18}}$). Wang et al. [85], studied the effectiveness of black hat Search Engine Optimization (SEO) campaigns to evaluate the possibility of manipulating search engine results for specific keywords by generating fake contents and leveraging various linking strategies. This method can be used to evade features related to the content length and hyperlinks but it requires a fair amount of

effort and costs not always available for the attacker.

2. **Technology-related features**. As mentioned in Section 1.4, we use Wappalyzer to enumerate the technologies used by the domain owner to design the website. Wappalyzer is a fast, free, easy to use, signature-based tool able to extract the used technologies by partial string and regular expression matching. Unfortunately, it is also easy to evade. For example, using PHP as a server-side programming language, the default name for the session ID stored as a cookie in the client machine is '*PHPSESSID*'. Wappalyzer uses this name to decide if the server-side code is PHP or not. Therefore, it is possible to mislead Wappalyzer and force it to make a wrong decision on the server-side language just by changing one keyword in cookies. However, decisions can be made using more advanced techniques e.g., hash-based fingerprinting [86].

Overall, to evade content-based features, the miscreant must establish a fully-functional website with different content and hyperlinks related to the domain name itself either manually, which takes **time**, or automatically, which impose additional **costs** on them.

1.10.0.2 Ranking and popularity features.

Manipulating features in this category is not completely under control of the attacker as it represents an external feature. However, it is feasible for a sufficient amount of time and effort.

1. Regarding the Wayback Machine, the attacker can manually submit URLs related to their domains to the Internet Archive project [87].

2. Regarding the Bing search engine results, using SEO techniques (e.g., black hat SEO as explained earlier [88]), it is possible to increase the number of indexed pages for each domain name in search engines.

3. Regarding top ranking websites (e.g., Alexa ranking system), previous research shown that it feasible to manipulate them [47].

However, the cost of evading *'ranking and popularity'* features is related to the **expertise** and **amount of time** the attacker should spend to make her domains as popular as it is necessary to evade the COMAR classifier.

1.10.0.3 TLD and WHOIS features.

COMAR uses one WHOIS-based feature (i.e., *'domain age'*) and two TLD-based features (i.e., *'TLD maliciousness index'* and *'TLD price'*).

1. To evade the *'domain age'* feature, the attacker should register domains long time before using them since we use the number of years before blacklisting, which imposes costs in terms of **money** on the attacker as she needs to register or re-new the domain for a period of a few years to evade this feature.

2. *'TLD maliciousness index'* is another strong feature of COMAR to decide on the state of a domain. One of the factors affecting the value of this feature is pricing. Cheap TLDs (or the free ones) have a higher 'maliciousness' value compared to the expensive TLDs [70, 15]. A higher value of the maliciousness index increases the chance that the domain name is maliciously registered. Therefore, to avoid being detected by the COMAR classifier, the attackers should register domains with TLD suffixes with low maliciousness values, which means they should pay more **money**.

1.10.0.4 Lexical, passive, and active DNS features

1. Lexical features are relatively easy to evade. For malware distributors, the domain name is not important since the victims that download the malicious content from the website are not humans but infected machines. However, for phishers, the choice of the domain name is relatively important to conduct a successful attack. For example, `insta-support.com` is more appealing to lure Instagram users compared to the name that has no indication of Instagram.

2. Regarding active DNS features, it is feasible to setup a mail server and/or define, for example, SPF rules in 'TXT' records. However, for attacks performed at a larger scale, the process needs automation.

3. Passive DNS features are the most difficult to evade as the sensors are distributed all around the world. Attackers are not aware of their locations and even if they were, it is not trivial to inject a large number of DNS packets as the monitoring sensors are placed above the local recursive resolvers.

1.11 Captcha Evasion Technique

Figure 1.10 shows a compromised website hosting a phishing page protected by Google CAPTCHA to prevent anti-phishing bots from accessing the malicious page content (details in Section 1.6.3.2).

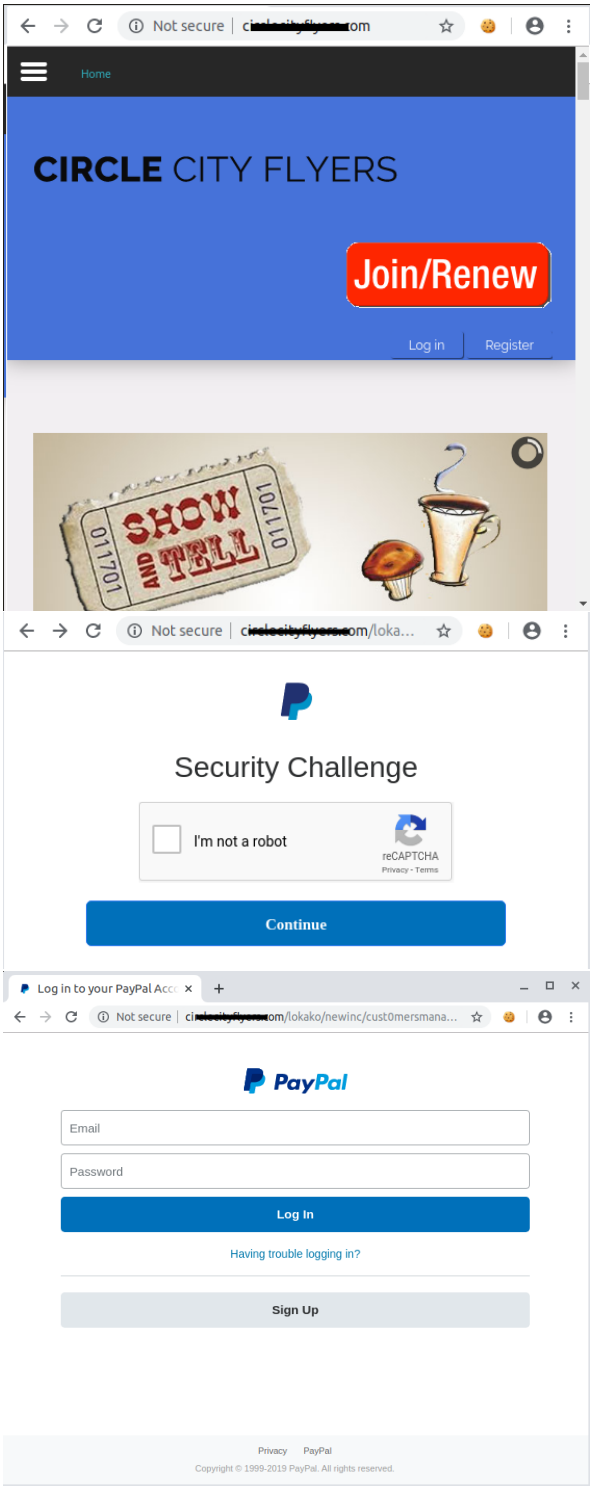


Figure 1.10 – Home page of the compromised domain (top), Google reCAPTCHA with a fake PayPal logo (middle), and a fake PayPal login page for phishing user's credentials (bottom).

Chapter 2

Are You Human?: Resilience of Phishing Detection to Evasion Techniques Based on Human Verification

---

This chapter is adapted from:

- **MAROOFI**, **Sourena**, KORCZYŃSKI, Maciej, et DUDA, Andrzej. Are You Human? Resilience of Phishing Detection to Evasion Techniques Based on Human Verification. In : Proceedings of the ACM Internet Measurement Conference. 2020. p. 78-86.

Contents

2.1 Introduction 68

2.2 Related Work 69

2.3 Phishing Detection and Anti-Analysis Techniques 70

2.3.1 Server-Side vs. Client-Side Detection 70

2.3.2 Alert Box Evasion 71

2.3.3 Session-Based Evasion 73

2.3.4 CAPTCHA-Based Evasion 74

2.4 Experiment Methodology 75

2.5 Experiment Results 78

2.5.1 Preliminary Test 79

2.5.2 Main Experiment 80

2.6 Testing Client-Side Anti-Phishing Extensions 81

2.6.1 Discussion 81

2.7 Conclusion 82

Abstract

Phishing is one of the most common cyberattacks these days. Attackers constantly look for new techniques to make their campaigns more lucrative by extending the lifespan of phishing pages. To achieve this goal, they leverage different anti-analysis (i.e., evasion) techniques to conceal the malicious content from anti-phishing bots and only reveal the payload to potential victims.

In this Chapter, we study the resilience of anti-phishing entities to three advanced anti-analysis techniques based on human verification: Google re-CAPTCHA, alert box, and session-based evasion. We have designed a framework for performing our testing experiments, deployed 105 phishing websites, and provided each of them with one of the three evasion techniques. In the experiments, we report phishing URLs to major server-side anti-phishing entities (e.g., Google Safe Browsing, NetCraft, APWG) and monitor their occurrence in the blacklists. Our results show that Google Safe Browsing was the only engine that detected all the reported URLs protected by alert boxes. However, none of the anti-phishing engines could detect phishing URLs armed with Google re-CAPTCHA, making it so far the most effective protection solution of phishing content available to malicious actors. Our experiments show that all the major server-side anti-phishing bots only detected 8 out of 105 phishing websites protected by human verification systems. As a mitigation plan, we intend to disclose our findings to the impacted anti-phishing entities before phishers exploit human verification techniques on a massive scale.

2.1 Introduction

Phishing is a form of social engineering with the goal of collecting credentials of end-users usually achieved by either email spoofing [89] or directly luring victims to enter their sensitive information into a fake website that matches the look and feel of the legitimate one [90].

There have been much research [91, 92, 93, 94, 95] and industry efforts to combat phishing, just to mention the APWG [96], PhishTank [20], or recently formed the COVID-19 Cyber Threat Coalition [97]. Nevertheless, according to the latest report from IBM X-force, phishing is the number one initial infection vector among attackers [98].

As with any other cybercriminal activity, phishers and security organizations are constantly in battle. While phishers try to develop new or misuse the existing techniques such as URL shorteners [99] to make their attacks more effective, anti-phishing organizations try to adapt their methods to detect phishing attacks swiftly. One of the most effective techniques used by miscreants is the *anti-analysis* also known as *evasion* [100]. The term refers to a wide range of techniques used by attackers to prevent automatic threat analysis [101]. While these techniques are very popular among malware developers [102], phishers also begin to use them [82, 103]. Malicious actors leverage evasion techniques to tell anti-phishing bots and humans apart. If the end-user is human, they reveal the malicious payload while for anti-phishing bots, they deliver a benign page to evade detection.

Previous work analyzed the existing and well known evasion techniques such as web-cloaking [61], URL redirection [104], using URL shorteners [99], and code obfuscation [105]. These techniques can affect the detection time, yet all major anti-phishing systems can cope with them [106]. For example, to detect web-cloaking, we can send two requests to the server, one with a user-agent related to a known anti-phishing bot (e.g., *googlebot*) and the other one with a typical browser user-agent (e.g., Mozilla Firefox). If the destination of the two requests are different (e.g., different domains), then we can infer the existence of web-cloaking [107].

However, we have recently observed more advanced anti-analysis techniques that can severely impede the performance of phishing detection systems or heavily extend the lifespan of phishing attacks. Human verification is one such technique that not only hampers the verification of phishing content but may also induce users to visit a phishing site due to the use of CAPTCHA on benign websites [108]. CAPTCHA is considered an effective anti-bot protection solution (e.g., against page scraping) but it is also misused by cybercriminals to protect their malicious pages from security organizations. A real-world example of such a usage is a recently detected phishing campaign with more than 128,000 emails sent to victims with a link to a fake Microsoft login page protected by Google reCAPTCHA [109].

In this Chapter, we study three advanced anti-analysis techniques observed in real-world phish-

ing attacks: Google reCAPTCHA [110], alert box, and session-based evasion techniques. To evaluate the effectiveness of the three techniques on anti-phishing bots, we have designed an experiment in which we emulate the operation of phishing websites—we register 105 domains, deploy harmless phishing websites, and protect each of them with one of the three evasion techniques. Then, we evaluate the detection performance of seven major anti-phishing engines: the Google Safe Browsing (GSB), Anti-Phishing Working Group (APWG), NetCraft, OpenPhish, PhishTank, Microsoft Defender SmartScreen, and Yandex Safe Browsing (YSB), as well as six most popular client-side anti-phishing extensions for Mozilla Firefox including the Avast Online Security and Avira Browser Safety. We show that almost all anti-phishing engines do not detect the phishing websites and mark the URLs as benign.

Our main contributions are as follows:

1. we qualitatively analyze a new category of anti-analysis techniques observed on real-world phishing websites,
2. we design a semi-automated and scalable framework for experimentally testing the evasion techniques (it is available on request for researchers to support reproducibility),
3. we empirically study the effects of three human verification evasion techniques on major server-side and client-side anti-phishing engines,
4. we show that all the major server-side anti-phishing bots only detected 8 out of 105 phishing websites protected by human verification systems.

2.2 Related Work

Several research have been done related to various aspects of phishing attacks. While researchers constantly propose new methods to effectively capture phishing pages, attackers devise new techniques to overcome the detection systems. We briefly explain attack and defense techniques in previous work:

URL obfuscation. Garera et al. [111], proposed an effective classifier (logistic regression) based on the extracted features only from URLs. They achieved 97% accuracy in detecting phishing pages with 1.2% false positive. However, Han et al. [112] showed that some attackers generate phishing URLs randomly and uniquely for each attack to avoid being blacklisted by anti-phishing engines.

Code Obfuscation. Cova et al. [105] studying different code obfuscation techniques used in phishing kits by collecting kits from underground markets and crawling live phishing websites. Fraiwan et al. [113], studied several code obfuscation techniques attackers leverage to make the detection process more difficult. In order to fight code obfuscation, the proposed methods work

based on the visual similarity of the pages instead of extracting features from page content [114, 91, 24].

Web cloaking. Oest et al. [106], investigated various techniques utilized by attackers to perform web cloaking in order to evade detection systems. They showed that although web-cloaking is a very well known technique [61], it is still used by attackers and can increase the detection time by 112 minutes.

Content-based detection. In the area of phishing detection, CANTINA [46] with the detection accuracy of 97%, uses TF-IDF approach to generate lexical signature from the candidate page and then use Google search engine to match the generated signatures with the top N search results to detect phishing pages. Whittaker et al. [44], marchal2016know et al. [115] proposed classifiers both with the accuracy of 99% which uses URL and content features to classify phishing attacks. This is where evasion techniques based on human verification can seriously hamper the detection results. While all the mentioned methods are able to efficiently detect phishing attacks, they all rely on the page content. If they can not capture the content correctly, they are not able to detect the attack. To the best of our knowledge, this is the first systematic approach to study human verification techniques and evaluate the performance of anti-phishing engines against such attacks.

2.3 Phishing Detection and Anti-Analysis Techniques

To better understand the three evasion techniques, we explain them in detail and discuss the possible approaches for phishing detection. Note that we have observed examples of phishing pages armed with all three techniques in real-world attacks.

The main source of our data is the *unverified* phishing section of the PhishTank [20] where the submitted URLs are not directly published as phishing but instead are pending for 'voters' to manually verify them as phishing URLs or false positives [116].

2.3.1 Server-Side vs. Client-Side Detection

Phishing page detection involves either client-side or server-side systems. In the server-side approach (e.g., GSB), the candidate URL is sent to the server (either by direct human report or by automatic URL collection by crawling). With the URL, the server starts to collect data (e.g., page content, domain content, lexical features, etc.) and decides whether it is malicious or benign. The client-side detection systems (e.g., browser add-ons and extensions) have direct access to the URLs and page content that end-users visit. Therefore, it is possible to collect features from the visited pages and decide on the phishing character of the pages either directly on the client-side

or later on by sending data to server-side systems.

Each detection type has its advantages and drawbacks. The server-side approach can be used globally by every browser that can interact with the system using an application programming interface (API). The internal detection algorithm appears as a black-box to attackers. Users' privacy is preserved by sending the hashed version of the URLs to the server. The drawback is that there is no guarantee that, having the same URL, the server and the client fetch the same content. If the attacker can detect the anti-phishing bot (e.g., IP-based or user-agent detection [106]), it will serve benign content to the bot.

Client-side phishing detection extensions do not suffer from this problem. They have access to the very same content as users visit because they are installed on browsers. However, the lack of sufficient support for extensions in different browsers [117], the inconsistency between desktop and mobile phone browsers [106], and privacy concerns related to third-party extensions [118] are the factors that affect the popularity of client-side extensions. In this chapter, we study seven server-side and six client-side phishing detection systems.

Below, we describe the three evasion techniques used by phishing websites to avoid detection.

2.3.2 Alert Box Evasion

Alert boxes in JavaScript are *modal* i.e., they pause script execution and do not allow the visitor to interact with the rest of the page until the window has been dismissed. Figure 2.1 shows an example of such a protection. The code to create a complete PayPal phishing page with alert box protection can be written in one PHP file and, for example, injected into one of the legitimate pages on a compromised machine. After a random number of seconds, the alert box appears on the screen by blurring the whole page and displaying the message: 'Please Sign In To Continue'. Depending on the content of the compromised website, this alert box not only can lure the victim to continue but also protects the real phishing payload by only revealing it to potential victims who click on the confirm button. When the user clicks on the 'OK' button, an AJAX request is sent to the server to retrieve the malicious payload and replace the current page content with it. Code Listing 2.1 presents example PHP code of this technique.

To bypass the alert box, anti-phishing engines need to detect it first. Most browser emulation libraries, e.g., the Selenium[1] project, can distinguish the alert box window if it is present. They can also confirm or cancel the alert box.

[1] https://www.selenium.dev

```php
<?php
$log_file="name of log file";
$a = $_POST['get_data'];
if (isset($a) && $a == 'getData'){
    /*Anti-phishing engine or user managed
    to confirm the alert box*/
    $d = array('ip' => getip(),'page'=>'payload');
    log_data($d,$log_file);
    echo "SERVE PHISHING HTML";
}else{
    $d = array('ip' => get_ip(),'page'=>'benign');
    log_data($d,$log_file);
    echo "SERVE BENIGN CONTENT WITH ALERT BOX";
}
?>
<script>
    /*Creating JS check variable
    for the second page load*/
    <?php
        if (isset($a) && $a == 'getData'){
            echo 'first_visit = false;';
        }else{
            echo 'first_visit = true;';
        }
        $u = $_POST['login_email'];
        $p = $_POST['login_pass'];
        if (isset($u) && isset($p)){
             echo 'already_served = true;';
        }else{
             echo 'already_served = false;';
        }
    ?>
    window.onload = function(){
         /*execute after the window
            is loaded completely*/
        if (first_visit && already_served){
            setTimeout(get_real_data,2000);
        }
    }
    function get_real_data(){
        var msg='Please sing in to continue...'
        var result = confirm(msg);
        if (result){
            /*
            dynamically generate and
            submit a form with hidden
            value 'getData'
            */
        }else{
            /*
            submit an empty form
            */
        }
    }
</script>
```

Listing 2.1 – PHP phishing code with alert box protection.

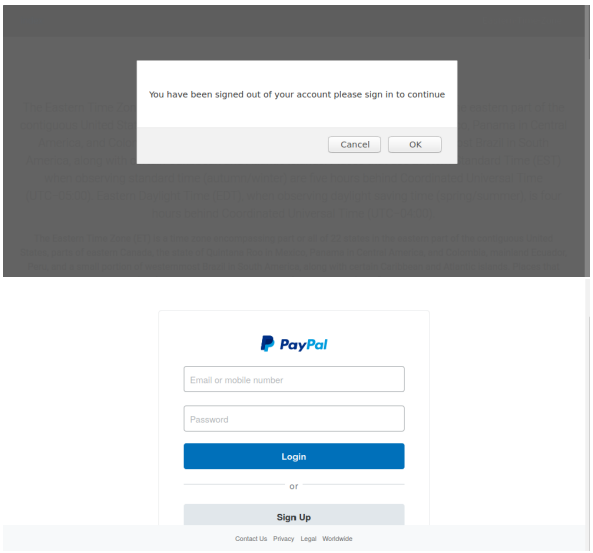


Figure 2.1 – Alert box evasion technique. Alert box protected page (top) and phishing payload targeting PayPal (bottom).

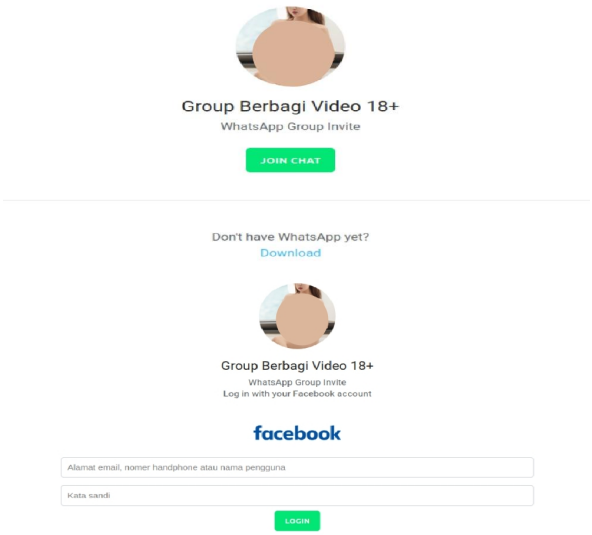


Figure 2.2 – Session-based evasion technique. Cover page on the first visit (top) and the phishing payload targeting Facebook (bottom).

2.3.3 Session-Based Evasion

Major companies like Google or Facebook have used the multi-page sign-in method for several years [119]. In this method, a user provides her username on the first page, submits the page, and enters her password on the second page. Note that the user can only be redirected to the second page if a session was generated on the first one. Phishers use this technique to reveal the second (malicious) page only to the users who have already visited the first page and clicked on the submit button. Figure 2.2 shows a real-world example of a session-based phishing attack. The first page (top figure) shows the 'Join Chat' button to persuade the victim to join a WhatsApp chat group and after pressing the button, the second page shows a fake Facebook login page.

One way to detect this type of evasion techniques is to automatically press the buttons and

submit the forms on the suspicious page. In Section 2.5.1, we show that some anti-phishing engines submit forms on the suspicious pages if they can find the HTML *form* tag.

2.3.4 CAPTCHA-Based Evasion

Google reCAPTCHA is considered as an effective anti-bot protection solution against page scraping [110]. Figure 2.3 in shows the phishing page protected by the Google reCAPTCHA v2 checkbox. By solving the CAPTCHA challenge, the form is submitted to the server and the phishing payload (e.g., PayPal login page) is served to the end-user without any URL redirection (Figure 2.3. To better understand the technique, Code Listing 2.2 shows the PHP code to create the phishing page. The first page is completely benign without an HTML *form* tag. When the user solves the CAPTCHA challenge, the HTML *form* tag is dynamically generated (using JavaScript) and then submitted to the server to reload the page using the same URL but with malicious content. Since the URL has not changed, the built-in browser anti-phishing system (e.g., GSB in Chrome) or the installed third-party extension (e.g., NetCraft toolbar) does not resend it to the server and serves instead the cached result usually valid for 5 to 60 minutes [120]. This behavior makes the detection process difficult (or impossible) and extends the lifespan of the phishing page.

There is no universal solution to bypass this type of evasion techniques. However, using client-side extensions, there is no need to automatically bypass CAPTCHA. If the end-user solves CAPTCHA and visits the second page (potentially malicious), the anti-phishing extension has also access to the content of the second page. So, it can detect phishing using the revealed page content.

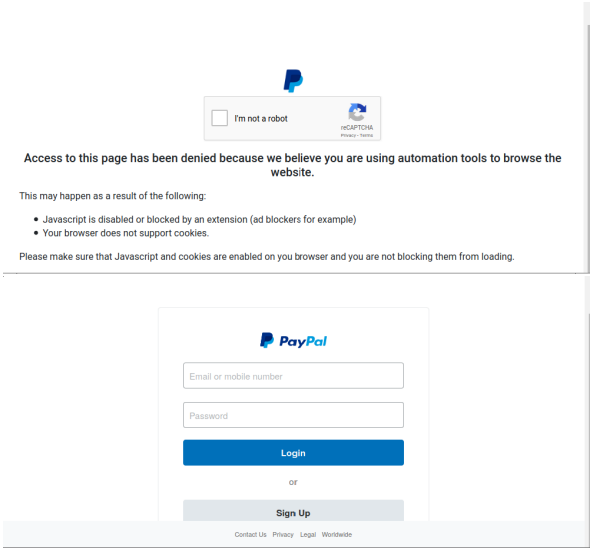


Figure 2.3 – Google reCAPTCHA message evasion technique. Google reCAPTCHA-protected page (top) and the phishing payload targeting PayPal (bottom).

```php
<?php
    $isvalid = false;
    if (isset($_POST['gresponse'])){
        $secret = 'Google CAPTCHA secret key';
        $captcha = $_POST['gresponse'];
        /* Check CAPTCHA result */
        $ans = chk_captcha($secret,$captcha);
        if ($ans->success)
            $isvalid = true;
        else
            $isvalid = false;
    }
    if ($isvalid){
        echo "Serve phishing payload HTML";
    }else{
        echo "Serve CAPTCHA page HTML";
    }
?>
<script>
    function capback(g_response){
        $form = $("<form>").attr({method:'post'});
        $input = $("<input>");
        $input.attr({name:"gresponse"});
        $input.attr({value:g_response});
        $form.append($input);
        $('body').append($form);
        $form.submit();
    }
</script>
```

Listing 2.2 – Single-page PHP phishing code with Google reCAPTCHA protection.

2.4 Experiment Methodology

To investigate the effectiveness of human verification techniques to protect phishing websites, we have designed a testing experiment and evaluated the performance of seven server-side anti-phishing entities. The experiment has two phases: the initial test to check if anti-phishing engines can detect the payload itself and the main test in which we protect the same payload using one of the evasion techniques and report it to anti-phishing entities. Both phases follow the same process as explained below.

For security considerations, we do not publish the source code of the proposed framework. However, it is available to researchers upon request to support reproducibility.

Tested Server-Side Anti-Phishing Entities We choose seven major anti-phishing entities: the Google Safe Browsing (GSB), NetCraft, Anti-Phishing Working Group (APWG), OpenPhish, PhishTank, Microsoft Defender SmartScreen, and Yandex Safe Browsing (YSB). GSB is used by Google Chrome, Mozilla Firefox, and Apple Safari that cover 87% of the end-user browsers both on desktop and mobile devices in 2020 [121]. Internet Explorer (IE) and Microsoft Edge use Microsoft Defender SmartScreen. The Opera browser uses two blacklist services: NetCraft and PhishTank [122]. To the best of our knowledge, OpenPhish and APWG are not directly used by any browser.

However, they are two important blacklist feeds that might be used by anti-phishing engines, so we also consider them in our experiments. Finally, YSB is used by the Yandex browser, the second most popular browser in Russia as of May 2020[2].

Registering Domains We first register 7 domains for the initial experiment and 105 for the main experiment, 112 domains in total. Oest et al. [106] performed a similar experiment to evaluate the effectiveness of web-cloaking in phishing attacks and registered fresh domains in bulk. In our study, we make sure that our domains are reputed enough to emulate the conditions of compromised domains rather than maliciously registered ones as observed in real-world cases. Therefore, we first register 50 drop-catch domains [83, 123] with the following new method:

1. First, we scan the top 1M domains in the Alexa list [66] for 'SOA' and 'NS' DNS records and only keep the domains with the NXDOMAIN answer (770 domains).

2. We use Godaddy and Porkbun APIs (two major registrars) to check the availability of the domains for registration from step 1 (251 domains).

3. We collect WHOIS data for 251 domains from the previous step and only select those with 'NOT FOUND' answer to make sure they are not registered (244 domains).

4. We submit these domains to VirusTotal[3] and Google Safe Browsing to make sure they have not been recently used in malicious activity (244 domains).

5. We select the domains archived at least once by the Internet Archive[4] to make sure they have their web history (50 domains).

6. We select the domains indexed at least once by the Google search engine based on the *site:domain* query (50 domains).

For the set of the remaining domains, we randomly generate keywords from the Unix dictionary and register 21 domains from new generic top-level domain (gTLDs) and the rest from legacy gTLDs (.com, .net, and .org). To reduce the impact of bulk registration patterns, we register our domains manually during two weeks in April 2020 with the OVH[5] registrar and deploy Domain Name System Security Extension (DNSSEC) for all domains. All steps are taken to reduce the chances of being blacklisted due to the low reputation of the domain.

[2] https://gs.statcounter.com/browser-market-share/all/russian-federation
[3] https://www.virustotal.com
[4] https://archive.org
[5] https://www.ovh.com

Website Content and Web Servers Compromised domains are intrinsically legitimate but hacked to host and serve malicious content in addition to legitimate content. Therefore, we have to generate a full-fledged website for each domain. To achieve this, we propose and develop a fake website generator using the following algorithm:

1. We extract meaningful keywords from the registered domain name.
2. For each keyword, we find synonyms using the Datamuse[6] API.
3. For each related keyword, we download the related page from the English version of Wikipedia along with their corresponding images.
4. For each domain, we randomly generate 30 pages (with .php extensions) with different names and different directories. Then, we generate hyperlinks from one page to another to create a fully functional website.

Having the fake website generator, it takes 2 minutes to generate a fully functional website with 30 different pages in an automated manner. The output is a .zip package ready to upload and install on the server. We upload all the generated websites to our hosting infrastructures in one of the European countries with 22 different IP addresses and the Nginx web server. Finally, we issue TLS certificates for all the domains and keep all websites online without uploading phishing kits for one week.

Phishing Kits The next step is to automatically generate phishing kits and upload them to our servers. We keep in mind the following considerations:

1. The aim is to assess the effectiveness of phishing detection bots in bypassing human verification evasion techniques. Therefore, we have to make sure first that the 'naked' phishing payload, i.e., not armed with any evasion technique, can be easily detected.
2. So, we have designed our own phishing pages instead of using existing phishing kits to prevent possible detection based on the *previously observed* attacks.
3. The design of the phishing kits targets three major services: Gmail, PayPal, and Facebook login pages.
4. We download all the external resources (e.g., pictures) from each target and generate a .zip package. This step is necessary since external resources like web favicons (favorite icons), logos as well as web beacons [124] play an important role for anti-phishing companies to track and detect phishing attacks for their websites [108].

[6] http://www.datamuse.com/api

5. For Facebook and PayPal login pages, we just clone their HTML source code, remove JavaScript code, and external requests. We design a Gmail login page from scratch since the original page uses a heavily obfuscated JavaScript code to generate HTML tags at run-time. All the three generated phishing pages look exactly like their true versions.

Reporting and Monitoring Process We submit phishing reports by either using an online form (GSB, SmartScreen, Net-Craft, and YSB) or sending an email (OpenPhish, PhishTank, and APWG). In the main experiment, we only generate one phishing URL for each domain. We never submit a domain to more than one anti-phishing bot. To monitor the results of the reports for each URL, we call GSB API to check if any of the URLs appeared in the GSB blacklist. Regarding OpenPhish, PhishTank, and APWG, we download the corresponding blacklist feeds every half an hour to check if they blacklisted our URLs and when. NetCraft notifies the reporter through emails by sending the results of the reports. SmartScreen has no publicly available API endpoint to check the results of the reports. Therefore, we develop a Python script to open the reported URLs using the Microsoft Edge browser, taking screenshots every 10 minutes for the first 72 hours and every 5 hours for the rest of the experiment, for several days.

Previous work showed some inconsistency among different browsers that use the same anti-phishing entities like Mozilla Firefox and Google Chrome (both use GSB) because of the different caching implementations of the GSB Update API (v4) in each browser [106]. In this experiment, our goal is not to check which browser faster blacklists phishing pages but instead, we test which anti-phishing engines can bypass the human verification evasion techniques.

2.5 Experiment Results

Table 2.1 – Preliminary test results after reporting the Gmail (G), Facebook (F), and PayPal (P) phishing URLs.

Reported to	# of requests	Unique IPs	Reported pages	Also blacklisted by	Blacklisted targets
GSB	8,396	69	G, F, P	-	G, F, P
NetCraft	6,057	63	G, F, P	GSB	G, F, P
APWG	2,381	86	G, F, P	GSB	F, P
OpenPhish	81,967	852	G, F, P	PhishTank, GSB, APWG, SmartScreen	F, P
PhishTank	4,929	275	G, F, P	OpenPhish, GSB	F, P
SmartScreen	1,590	81	G, F, P	GSB	F, P
YSB	82	34	G, F, P	-	-

Oest et al. [106], showed that the average blacklist time (i.e., the time between the URL submission and its appearance in one of the blacklists) was 126 minutes without using the web-cloaking technique and 238 minutes with web-cloaking. They also showed that anti-phishing engines could

only detect 23% of the phishing URLs armed with web-cloaking. We expect a lower detection rate for the phishing sites instrumented with the three human verification techniques because they are more advanced and less observed in real-world phishing attacks.

2.5.1 Preliminary Test

In our initial test, we have submitted three URLs to seven anti-phishing entities targeting PayPal, Facebook, and Gmail without using any evasion technique. The test lasted for 24 hours. Based on the results from the previous study [106], 24 hours is enough to check if a reported URL is classified as malicious. For all seven anti-phishing bots, we received traffic to our webserver within the first 30 minutes after we reported the URLs. Table 2.1 shows the results of the preliminary tests. Since YSB was unable to detect even one of the phishing sites, we had to exclude it from the main experiment.

We also observed that except for GSB and NetCraft, none of the anti-phishing bots could identify the fake Gmail login pages as phishing attacks. Therefore, we also excluded Gmail from our target list in the final experiment. We suspect that the Gmail login page was more difficult to detect compared to PayPal and Facebook pages because of the different design approaches we used. As mentioned earlier, we have implemented the Gmail login page from scratch, whereas for PayPal and Facebook, we cloned the original HTML code. Moreover, log inspection shows that NetCraft, OpenPhish, and PhishTank submit the HTML *form* tags automatically by filling the 'username' field with different values (we do not log the password filed on our servers).

The results presented in Table 2.1 show that:

1. There exist a relationship between different vendors. For example, the URLs we reported to OpenPhish also appeared in other blacklist feeds.

   The results also suggest that GSB uses other major blacklist feeds.

2. For the URLs submitted to OpenPhish and PhishTank, we received abuse notification emails from PhishLabs [125] sent to the registered abuse notification email address related to our IP addresses.

3. Within the first two hours after reporting the URLs to OpenPhish, we received a high amount of requests sent to our servers (81,967 requests).

   Their analysis reveals that anti-phishing bots looked for files related to: *i)* famous web-shells, *ii)* possible phishing kits (.zip files), and *iii)* possibly stolen credentials stored on the server (.log and .txt files).

Table 2.2 – Results of the main experiment after reporting phishing URLs. X/Y shows the number of detected URLs (X) out of all submitted ones (Y), where Alert box (A), Session-based (S), or Google reCAPTCHA (R) are used to hide phishing sites from server-side anti-phishing bots.

Anti-phishing bots	Facebook			PayPal		
	A	S	R	A	S	R
GSB	3/3	0/3	0/3	3/3	0/3	0/3
NetCraft	0/3	2/3	0/3	0/3	0/3	0/3
APWG	0/3	0/3	0/3	0/3	0/3	0/3
OpenPhish	0/3	0/3	0/3	0/3	0/3	0/3
PhishTank	0/3	0/3	0/3	0/3	0/3	0/3
SmartScreen	0/2	0/2	0/2	0/3	0/3	0/3

Table 2.3 – Client-side anti-phishing extensions. Number of installations is the sum of installations for Chrome and Firefox. X/Y shows the number of detected URLs (X) out of all submitted ones (Y).

Extension	Company	# of installations	Sending URLs	Sending Params	X/Y
Avast Online Security	Avast	10,800,000+	✓(plain)	✓	0/9
Avira Browser safety	Avira	7,350,000+	✓(plain)	✓	0/9
TrafficLight	BitDefender	665,000+	✓(plain)	✓	0/9
Emsisoft Browser security	Emsisoft	80,000+	✓(hashed)	✗	0/9
NetCraft Anti-phishing	NetCraft	58,000+	✓(hashed)	✗	0/9
Online Security Pro	Comodo	14,000+	✓(plain)	✓	0/9

2.5.2 Main Experiment

Table 2.2 shows the results of the main experiment to evaluate the effectiveness of six major anti-phishing entities in detecting phishing websites protected by evasion techniques. The experiment lasted for two weeks in May 2020. Nevertheless, we received about 90% of the traffic during the first 2 hours after reporting the URLs.

Regarding the alert box protection, GSB was the only engine that detected all 6 reported URLs, on average 132 minutes after submission, which means that GSB can detect phishing websites protected by alert boxes by handling them in a browser simulation. The log analysis on our server reveals that GSB bots clicked on the 'confirm' button in the alert box and successfully retrieved phishing content, while other anti-phishing engines never reached the phishing content because they failed to click on the alert box.

When it comes to session-based evasion, NetCraft was the only engine that detected 2 out of 6 reported URLs (6 and 9 minutes after the submission). Interestingly, the log analysis indicates that NetCraft bypassed all six session-based pages and reached the phishing sites, but only 2 of them were detected as such. No other anti-phishing engine bypassed the session-based protection.

Regarding Google reCAPTCHA, as expected, none of the anti-phishing engines could detect even one out of 35 reported URLs, which means that it is so far the most effective protection solution of phishing websites available to malicious actors.

The results shown in Table 2.2 indicate that, in general, human verification techniques raise se-

rious challenges to anti-phishing bots. In total, all the major server-side anti-phishing bots could only detect 8 out of 105 phishing URLs used in the main experiment. Fetching the phishing content depends on bypassing the evasion technique, which, as our experiment shows, is not a trivial task for anti-phishing bots. One possible solution to this problem is to let the end-users solve the challenge and reveal the final page content to client-side anti-phishing engines.

2.6 Testing Client-Side Anti-Phishing Extensions

Client-side extensions have access to the same content as users. If the user bypasses CAPTCHA (e.g., confirming alert box, solving Google reCAPTCHA, or pressing 'proceed' button in session-based pages) and visits the second page, the extensions have also access to the new possibly malicious content.

Therefore, in a separate experiment, we install the six most popular anti-phishing extensions on Mozilla Firefox (see Table 2.3).

To make sure there is no conflict between the extensions, we use different Firefox profiles for each extension and disable GSB. For each extension, we submit 9 phishing URLs (3 URLs per evasion technique). We also use the Burp Suite[7] tool to capture the exchanged traffic related to each extension. In this way, we can read all TLS-encrypted requests between extensions and their servers. Then, we visit each URL three times with a 5-hour window between them.

Table 2.3 shows that none of the extensions could detect our phishing pages. Monitoring the traffic generated by the extensions reveals that they only collect the URLs visited by the user, send them to their servers, and check the URLs against their own blacklists. As shown in Table 2.3, four out of six extensions send 'naked' URLs (without hashing) along with all the query parameters to their servers. Since they do not create any feature vector on the client machine and only send the URL to the server for further analysis, they operate like their server-side counterparts, and thus, they are unable to detect CAPTCHA-protected phishing attacks.

2.6.1 Discussion

In Section 2.5, we have presented three quite effective evasion techniques based on human verification to avoid detection by anti-phishing engines. The important question is how popular these techniques are among attackers and what is the solution for effective handling of the attacks that use these techniques.

With respect to popularity, it is unlikely to find these attacks in public blacklist feeds because they mainly rely on anti-phishing engines that cannot bypass advanced types of evasion tech-

[7] https://portswigger.net/burp

niques. In a parallel and independent research study by Oest et al. [126], the authors analyzed 4,393 URLs and only found five of them that used the CAPTCHA challenge and two using the alert box technique to avoid detection. In recent work, Maroofi et al. [103] studied a reCAPTCHA-protected phishing URL that prevents crawlers from fetching the real malicious content. Although the URL was submitted to Phishtank, a community-based URL blacklist based on user reports, it was not confirmed by any other user and thus, it did not appear on the official blacklist. Therefore, due to the difficulties related to detecting such attacks, it is not straightforward to evaluate their popularity.

Another concern is to develop a suitable solution so that anti-phishing engines can detect these attacks and blacklist the URLs. Regarding the alert box technique using server-side detection, the solution is trivial since most automation frameworks (e.g., Selenium library) can interact with alert boxes and modal windows. For session-based attacks (server-side detection), one possible solution is to simulate form submissions. As shown in Section 2.5.1, NetCraft submitted the HTML form for each report but was only able to detect 2 malicious URLs. Finally, bypassing CAPTCHA by a server-side anti-phishing engine is not easy in general since there is no prior information about the characteristics of the CAPTCHA challenge used by attackers.

For client-side detection systems (in the form of extensions installed on the browsers), there is no need to implement any extra mechanism. If the user solves the challenge and visits a malicious page, it is also visible to extensions for the detection process.

2.7 Conclusion

In this Chapter, we evaluate the resilience of anti-phishing detection systems to evasion techniques based on human verification in a controlled experiment. We registered 105 previously-unseen as well as reputed drop-catching domains. For each domain, we automatically generated a full-fledged website and a phishing page protected by either Google reCAPTCHA, alert box, or session-based techniques, and reported them to major anti-phishing entities. An alarming picture emerges from our experiments: the overwhelming majority of phishing sites were not detected. Google Safe Browsing outperforms other entities by only bypassing the alert box anti-analysis technique. NetCraft was able to detect 2 URLs protected with the session-based technique. Other server-side and client-side detection systems were unable to detect even one of the phishing URLs as malicious.

The prevalence of human verification techniques in phishing kits can severely hinder the detection process and raises serious challenges for existing anti-phishing engines. As a mitigation plan, we intend to disclose our findings to the impacted anti-phishing entities before phishers

exploit human verification techniques on a massive scale.

Chapter 3

Adoption of Email Anti-Spoofing Schemes: A Large Scale Analysis

---

This chapter is adapted from:

- **MAROOFI**, **Sourena**, KORCZYNSKI, Maciej, et DUDA, Andrzej. From Defensive Registration to Subdomain Protection: Evaluation of Email Anti-Spoofing Schemes for High-Profile Domains. In : Proc. Network Traffic Measurement and Analysis Conference (TMA). 2020.
- **MAROOFI**, **Sourena**, KORCZYŃSKI, Maciej, HÖLZEL, Arnold, et al. Adoption of Email Anti-Spoofing Schemes: A Large Scale Analysis. IEEE Transactions on Network and Service Management, 2021.

Contents

3.1 Introduction 88

3.2 Related Work 90

3.3 Background on Anti-Spoofing Schemes 91

3.3.1 SPF – Sender Policy Framework 92

3.3.2 DKIM structure 94

3.3.3 DMARC 95

3.3.4 Threat Models 96

3.4 Methodology for analyzing SPF and DMARC deployment 97

3.4.1 Global Measurements 97

3.4.2 Top 500 Websites of All Countries 98

3.4.3 Defensive Registrations 98
3.4.4 Subdomain Enumeration 99
3.4.5 Banks and Financial Websites 100
3.5 Results on SPF and DMARC Adoption 100
3.5.1 Global Scan of the SPF and DMARC Rules 100
3.5.2 High-Profile Domains and Defensive Registrations 101
3.5.3 Analysis of Spoofing Possibilities for Subdomains 102
3.5.4 SPF Emulation Results 103
3.5.5 End-to-End Spoofing Measurement 104
3.6 Trust-based Authentication Issue 105
3.7 Methodology for Preventing Domain Spoofing 106
3.8 Remediation 108
3.8.1 Results of the First Notification Campaign 108
3.8.2 Results of the Second Notification Campaign 109
3.8.3 Notes on Notification Campaigns 109
3.9 Conclusion 111

Abstract

Sending forged emails by taking advantage of domain spoofing is a common technique used by attackers. The lack of appropriate email anti-spoofing schemes or their misconfiguration lead to successful phishing attacks or spam dissemination. In this chapter, we evaluate the extent of the SPF and DMARC deployment in two large-scale campaigns measuring their global adoption rate with a scan of 236 million domains and high-profile domains of 139 countries. We propose a new algorithm for identifying defensively registered domains and enumerating the domains with misconfigured SPF rules by emulating the SPF check_function. We define for the first time new threat models involving subdomain spoofing and present a methodology for preventing domain spoofing, a combination of good practices for managing SPF and DMARC records and analyzing DNS logs. Our measurement results show that a large part of the domains do not correctly configure the SPF and DMARC rules, which enables attackers to successfully deliver forged emails to user inboxes. Finally, we report on remediation and its effects by presenting the results of notifications sent to CSIRTs responsible for affected domains in two separate campaigns.

3.1 Introduction

Email spoofing consists of sending a message with a forged sender address and other parts of the email header so that it appears as sent from a legitimate source. Attackers commonly use this method to mislead the receivers, gain their trust, and eventually, achieve some malicious goals. Phishing and spam campaigns are examples of attacks that rely on email spoofing. Despite tremendous efforts deployed to mitigate this technique, it is still one of the most successful attacks responsible for significant damage. According to the Internet crime report [127], email spoofing costed US victims more than 1.2 billion dollars in 2018.

Email spoofing comes in two types. The first one consists of *compromising legitimate servers* and using their mail transfer agent to send spoofed emails to victims either by specifying a different `Reply-to:` address or providing a phishing URL in the body of the message. The second type is *domain spoofing* in which attackers send emails on behalf of legitimate domains, e.g., a forged email from *account-security-noreply@accountprotection.microsoft.com* impersonating the Microsoft support team with a fake landing page looking alike a real Microsoft login page to steal user credentials [128]. In this chapter, we investigate the second type of email spoofing.

The Simple Mail Transfer Protocol (SMTP) for email distribution does not provide support for preventing spoofing [129]. The system needs to rely on *security extensions* such as the Sender Policy Framework (SPF) [50], the DomainKeys Identified Mail (DKIM) [130], and Domain-based Message Authentication, Reporting, and Conformance (DMARC) [62] to authenticate the sender and decide what to do with suspicious emails. The extensions define a set of rules that specify who is allowed to send emails on behalf of a given domain name and how to deal with suspicious messages. A careful deployment of the extensions can completely mitigate the problem of domain spoofing. However, to be effective, both the domain owner and the mail transfer agent of the recipient should implement the extensions: the domain owner needs to correctly set SPF, DKIM, and DMARC rules, and the recipient has to authenticate incoming messages and correctly implement the verification of the SPF and DMARC rules.

In this Chapter, we evaluate the extent of the SPF and DMARC deployment and analyze spoofing possibilities enabled by the absence or misconfiguration of their rules. We do not analyze DKIM as it requires access to the email header selector tag, not publicly available (see RFC 6376 for more details [130]).

While previous work already investigated the adoption of SPF and DMARC by the Alexa top-ranked one million domains [131, 89], we consider different datasets as well as threat models. We scan approximately **236 million domain names** including generic top-level domains (gTLD), Country code Top-level domain (ccTLD) and new gTLDs collected from different sources such as

the Centralized Zone Data Service (CZDS)[1] made available by the Internet Corporation for Assigned Names and Numbers (ICANN), OpenData project from Rapid7[2] as well as zone files that are public and available for download (e.g, .se). The second dataset includes **32,042 high-profile domains** of 139 countries and their **defensive domain registrations**. The high-profile domains correspond to most popular targets of email spoofing: well-known companies, governmental websites, or financial institutions. To the best of our knowledge, this is the first study reporting on the global-scale measurement of the adoption of email authentication extensions.

We investigate the global adoption of SPF and DMARC protocols by scanning each domain in our datasets. Then, we define a threat model in which attackers use subdomains (both existent and non-existent) for email spoofing. We also identify *defensively registered domains* and evaluate their adoption of email anti-spoofing schemes. We show that even if defensive registrations can mitigate some types of attacks like *cybersquatting* and *brand name abuse*, these domains need to be protected against domain spoofing as well.

We extend our previous work [63] and make the following main contributions:

1. we investigate the global adoption of SPF and DMARC for 236 million domain names of different TLDs,

2. in a separate measurement campaign, we evaluate the adoption of SPF and DMARC by top 500 most popular domains of 139 countries including local businesses, national websites, local governments, and financial sectors,

3. we propose a method to find defensively registered domains for top-ranked websites and assess the extent of their adoption of email security extensions,

4. we are the first to measure the extent of SPF and DMARC deployment by the subdomains of the top-ranked websites to gain better insight into how attackers can abuse subdomains to send spoofed emails,

5. we show that it is possible to send forged emails from non-existent subdomains when a DMARC rule is not strict enough regarding subdomains,

6. we demonstrate how syntactically wrong SPF rules may break the trust-based authentication system of selected email service providers by allowing forged emails to land in the user inbox,

7. we present a methodology for preventing domain spoofing based on good practices for managing SPF and DMARC records and analyzing DNS logs,

[1] https://czds.icann.org
[2] https://opendata.rapid7.com

8. finally, as a proof of concept, we perform an end-to-end email spoofing for subdomains of high profile domains with misconfigured SPF and/or DMARC.

To remediate misconfigured SPF rules, we have contacted relevant Computer Security Incident Response Teams (CSIRT)s responsible for misconfigured domains and measured the effectiveness of our notifications. To encourage reproducibility, we make our measurement data available upon request.

The rest of the chapter is organized as follows. Section 3.2 reviews related work and section 3.3 provides background on SPF and DMARC. Section 3.4 specifies possible threat models and introduces our approach to generate the datasets and find defensively registered domains. Sections 3.5 presents the analysis of the results for scanned domains and subdomains as well as for emulation of SPF rules. In Section 3.6, we study the trust-based authentication issue and Section 3.7 presents a methodology for preventing domain spoofing. Section 3.8 describes our remediation actions. Section 3.9 concludes the chapter.

3.2 Related Work

In this section, we briefly review previous work on measuring and analyzing email security extensions.

Durumeric et al. [132] measured the adoption of SMTP security extensions and their impact on end users. They studied SMTP server configurations for the Alexa top one million[3] domains and SMTP connections to and from Gmail gathered over a year. They reported the existence of a long tail of over 700,000 SMTP servers, of which only 35% successfully configure encryption, and only 1.1% specify a DMARC authentication policy.

In 2017, Durumeric [131] measured the extent of SPF and DMARC adoption for one million top domains in the Alexa list. His results showed that 40.1% of the domains have published SPF records while only 1.1% of them have valid DMARC records. Hu and Wang [89] reported similar statistics in 2018 with the results of 44.9% published SPF records and 5.1% published DMARC records showing approximately 5% of increase in one year. In their end-to-end experiment, they spoofed 30 high-profile domains and reported the ratio of emails that reached inboxes of selected email providers. We perform a similar analysis for both SPF and DMARC records but in two different phases. First, we analyze the global adoption of SPF and DMARC rules for different TLDs and then, we focus on more prominent domains (with transactional emails) including banking websites, government portals, national and international businesses as well as defensively registered domains and their subdomains. We also consider end-to-end spoofing but just as a proof of concept for our defined

[3] https://www.alexa.com/topsites

threat models and only for 10 high-profile domains.

Foster et al. [133] evaluated the security extensions using a combination of measurement techniques to determine whether major providers support the TLS protocol[36] at each point in their email message path, and whether they support SPF and DKIM on incoming and outgoing mail. They reported that while the use of SPF is common, enforcement was limited. Scheffler et al. [134] investigated the consequence of a wrong implementation of the `check_host` function at the receiver, which lets attackers perform Denial-of-Service (DoS) attacks on a local DNS resolver. While our goal is not to evaluate the SPF abuse, we show that 4,349,463 domains in the global scan, 1,131 high-profile, and defensively registered domains have published SPF records that require more than 10 DNS lookups. Therefore, such misconfigured records may lead to abuse of local DNS resolvers.

Finally, Hu et al. [135] investigated the reasons behind the low adoption rates of anti-spoofing protocols. They conducted a user study involving email administrators and showed that they believe the current protocol adoption lacks the crucial mass due to the protocol defects, weak incentives, and practical deployment challenges.

The ultimate goal of all spam email detection systems is to detect unsolicited emails, which consists of various types of malicious emails (e.g., phishing emails) and advertisement emails. These systems use several features such as the content of the email (e.g., [136], [137]), header and attachment features (e.g., [138]), history of the emails (e.g., [139]) and network based features (e.g., [140]). Cidon et al. [141], proposed a method to detect Business Email Compromise (BEC) attack where the attacker impersonate an employee of an organization and often ask for money either through wire transfer or by sending a phishing URL to the victim. The difference of BEC with spear phishing is that the later is often volumetric, i.e., emails send in batch to hundreds of recipients while BEC emails are sent to a small number of recipients.

All the above mentioned work are based on the assumption that the email authentication protocols (SPF, DKIM, and DMARC) cannot be applied either because they are not published by the domain owners or not implemented (or applied to the emails) by the recipient Message Transfer Agent (MTA) to detect and mark unsolicited emails. Hence, there is a need for a post-detection system to distinguish such a fraud.

3.3 Background on Anti-Spoofing Schemes

To understand the issue of email authentication, we briefly explain the process of mail delivery. Figure 3.1 shows Bob (sender) who sends legitimate mails to Alice (receiver). Mallory (attacker) wants to send an email that impersonates Bob to Alice. Mallory and Bob use their respective

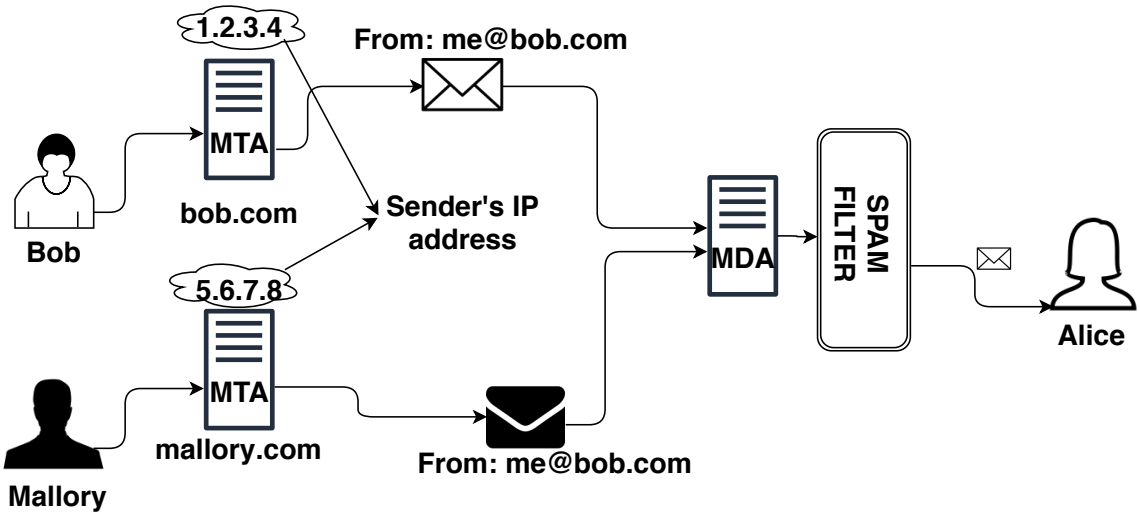


Figure 3.1 – Email sending and receiving procedure.

servers (`mallory.com` and `bob.com`) to send mails. The Mail Delivery Agent (MDA) on the Alice server delivers two emails with the same sender address (`me@bob.com`) but coming from different IP addresses (assuming there is no spam filtering involved). One mail is from Bob (originated from the `1.2.3.4` IP address) and the other from Mallory (originated from `5.6.7.8`).

An effective anti-spoofing mechanism needs to differentiate the Mallory message from the legitimate Bob's mail. The current first lines of defence to protect end-users from spoofed emails include SPF [50], DKIM [130], and DMARC [62].

3.3.1 SPF – Sender Policy Framework

SPF is a set of text-form rules in `TXT` resource records of the DNS. SPF specifies a list of servers allowed to send emails on behalf of a specific domain. During mail delivery over the SMTP protocol, the recipient server authenticates the sender MTA using a given `HELO` or `MAIL FROM` identity based on the published SPF record and the IP address of the sender—SPF needs to contain the domain portion of the `MAIL FROM` identity. In our example, the Alice server gets the `TXT` records of the `bob.com` domain from DNS. Then, it checks whether the sender IP address is on the list of IP addresses allowed to send emails from the `bob.com` domain and decides whether the message should be rejected or delivered to Alice.

The decision is made by the `check_host` function described in RFC 7208 [50] that takes three arguments on input (IP address of the sender, the domain, the `MAIL FROM` or `HELO` identity) and returns one of the seven possible results shown in Table 3.1. The third column of the table presents the actions recommended by RFC 7208.

Below, we review the most common SPF rules useful for understanding the threat models presented in the next section (see RFC 7208 for more details). A valid SPF version 1 record must begin

Table 3.1 – Possible results of the SPF `check_host` function and their definitions.

Result	Definition	Recommended action
None	1. No valid domain name was extracted from the SMTP session. 2. No SPF record was retrieved from the domain name.	1. The action must be the same as the *Neutral* output.
Neutral	1. There is no definite assertion (authorized or not) about the sender.	1. Depends on the receiver system.
Pass	1. Client is authorized to send emails with the given identity.	1. Whitelist the domain in terms of SPF.
Fail	1. Client is not authorized to send emails with the given identity.	1. Depends on the receiver system. 2. Make decision based on the DMARC policy.
Softfail	1. Client is not authorized to send emails with the given identity. 2. No strong policy specified by the domain owner.	1. Receiver should not reject the message. 2. May mark the message as suspicious.
Temperror	1. A temporary error occurred during retrieving the SPF policy.	1. May defer the message. 2. May deliver the message and mark it.
Permerror	1. Parsing problem in published SPF.	1. May deliver the message and mark it.

with string `v=spf1` followed by other SPF *mechanisms*, *qualifiers*, and *modifiers*. Mechanisms describe the set of mail servers for a domain and can be prefixed with one of four qualifiers: `+` (*Pass*), `-` (*Fail*), `~` (*SoftFail*), `?` (*Neutral*). If a mechanism results in a match, its qualifier value is used. *Pass* (i.e., `+`) is the default qualifier.

The most common SPF mechanisms are the following:

- `ip4` and `ip6` – they specify an address or a set of IPv4 (or IPv6) addresses to match by the `check_host` function with respect to the sender IP address.
- `a` and `mx` – they tell the `check_host` function to perform first a DNS lookup for `A` (or `MX`) records of a given domain and then compare the returned IP addresses with the IP address of the sender.

- `exists` – it indicates a DNS domain name used for a DNS `A` query. If the query returns any `A` record, this mechanism matches.

- `include` – it tells the `check_host` function to include the SPF rule of another domain in the evaluation, which may result in calling the `check_host` function recursively to fetch and analyze the SPF records of the included domains.

- `all` – it always matches, so its corresponding qualifier results in the final decision. For example, `v=spf1 mx -all` means: allow `MX` servers of the domain to send mail and prohibit all others.

The final result of the mechanisms could be *Match*, *No match*, or *Exception*. Qualifiers combined with mechanisms, generate the final input for the `check_host` function that evaluates the SPF rule.

Modifiers provide additional information about the SPF records, for instance:

- `redirect=another-domain` – the SPF record for `another-domain` replaces the current record. The redirected domain becomes the target of all DNS queries and evaluations instead of the original domain.

If there is no qualifier, the default qualifier (which is +) will be used for that mechanism. Qualifiers combined with mechanisms, generate the final input for the `check_host` function evaluates the SPF rule. Possible qualifiers are: 1) pass (+), 2) fail (-), 3) softfail (~), and, 4) neutral (?).

For example, if a mechanism matches and the qualifier is '-', then a 'fail' result will be returned. As a consequence, qualifiers play an important role in SPF evaluations. For instance, specifying *+all* instead of *-all* in the SPF record of a domain will lead to *pass* results for all the emails no matter they are spoofed or not.

Let us consider the following example:

```
v=spf1 a ip4:1.2.3.0/24 -all
```

when the `A` record `example.com A 6.7.8.9` is stored in DNS. The SPF rule states that only machines with the IP address of `6.7.8.9` (the `a` mechanism) or with the IP address in the range of `1.2.3.0`...`255` (the `ip4` mechanism) are permitted senders (all others are forbidden). However, by only changing `-all` to `+all`, any machine is permitted to send emails on behalf of the domain `example.com` with the successful SPF *Pass* result.

3.3.2 DKIM structure

DomainKeys Identified Mail (DKIM), is a form of email authentication that allows an organization to claim the responsibility of a message which can be validated by the recipient. DKIM works based on digital signature verification scheme. The public key to verify the sender's signature is

stored in the *<selector>._domainkey.domain.tld*. Analyzing DKIM needs the selector tag, which can only be found in the header of the emails and is not publicly available. Therefore, it is out of the scope of this research (cf. RFC 6376 for more details).

3.3.3 DMARC

DMARC [62] builds on top of SPF and DKIM by explicitly stating the policies to apply to the results of SPF and DKIM. In particular, DMARC binds names checked by SPF with what is listed in the `FROM:` field of the mail header by means of *alignment*, which expresses the fact that these domain names should match (or partially match when using a relaxed setup). For instance, DMARC checks whether the name in the `MAIL FROM` SMTP command and the `FROM:` field of the mail header match or not. In the case of the alignment test failure, a DMARC policy can specify what to do with the message (accept, reject, or quarantine) and where to send reports in case of a mismatch. For a given domain name `domain.tld`, the DMARC policy is stored in the `TXT` record of `_dmarc.domain.tld`. Below, we present selected tags of DMARC that, when misconfigured, can be exploited by an adversary.

- `aspf` (Alignment mode for SPF) – it specifies whether the strict (`s` value) or relaxed (`r` value) alignment mode is required by the domain owner. The default value is the relaxed mode. In the strict mode, the domain name used in SPF must be the same as the domain used in the `FROM:` field of the header. In the relaxed mode, any subdomain of the domain can be used in the `FROM:` field of the header and will result in *Pass*.

- `p` (Policy) – it specifies the action to be taken by the receiver if the alignment test results in *Fail*. Possible values for this tag are: 1) `none` – no specific action, 2) `quarantine` – the message is suspicious and depending on the mail system of the recipient, it could be delivered as spam, 3) `reject` – the domain owner wishes to reject emails during the SMTP transaction that fail the alignment test.

- `ruf` (Reporting URI for failure) – it specifies the email addresses to which message-specific failure information is to be reported. This tag is important since this is the only bridge between the receivers and the true domain owners to fight spam emails [142].

- `sp` (Subdomain policy) – it has the same syntax as `p` but applies to subdomains of the domain name. In the absence of this tag, the policy of the `p` tag must be applied to all subdomains [62]. If subdomains are not used to send emails, the owner can set this tag to the `reject` value to prevent subdomain email spoofing.

Let us assume that the DMARC rule of the domain `example.com` is `v=DMARC1; p=none; aspf=r;`.

If we have the previously mentioned SPF rule for this domain, an illegal sender with the IP address of `9.10.11.12` can forge emails on behalf of `example.com` or any (existent or non-existent) subdomain of `example.com`, and the delivery decision is up to the receiver since no strict rule has been specified in DMARC. However, changing the DMARC rule to `v=DMARC1; p=quarantine; sp=reject; aspf=s;` tells the receiver to label all the emails that did not pass the SPF evaluation as spam and reject all the emails from the subdomains of `example.com` at the SMTP level.

3.3.4 Threat Models

We now consider threats regarding SPF and DMARC in detail. To mitigate mail spoofing, domain owners set up SPF and DMARC rules then used by inbound mail servers. Therefore, if the recipient MTA does not support the SPF or DMARC check, no matter how strict the rules are, they will not be effective. A misconfigured SPF or DMARC (either syntactically or semantically) rule is as dangerous as the absence of the rules since the output of the evaluation does not lead to a correct decision.

We consider three possible types of threats:

- **Related to domain names**. If a domain uses a misconfigured SPF rule, then it is possible to send forged emails from any IP address with the SPF *Pass* result. For example, we have discovered that `microsoft.com.tr` used the `+all` mechanism in its SPF rule, which made it easy for attackers to send forged emails on behalf of Microsoft from any IP address. Note that after notifying Microsoft, the issue was fixed.

- **Related to subdomains**. Each subdomain should have its own SPF and DMARC rules. Another possibility is to use the `sp` tag in DMARC of the domain name (lower-level domain) to explicitly specify the action to take when receiving messages from subdomains. A possible abuse of subdomains is the following:

  - If a subdomain has no SPF rule (and there is no specified wildcard rule) and no explicit DMARC action, then it is possible to misuse the subdomain for sending forged emails. For example, while `icann.org` has a strict SPF rule, there is no rule specified in `account.icann.org` and no DMARC policy regarding subdomains (also the default action for domains is `none`, which in this case applies to subdomains). Hence, it is possible to send emails with forged sender addresses (e.g., `support@account.icann.org`) with the SPF *Neutral* result.

  - If a subdomain does not exist, the result of the DNS query for the `TXT` record returns a name error (NXDOMAIN). Thus, the `check_host` function returns the *None* result (see Table 3.5). If there is no wildcard `TXT` record that covers non-existing subdomains and

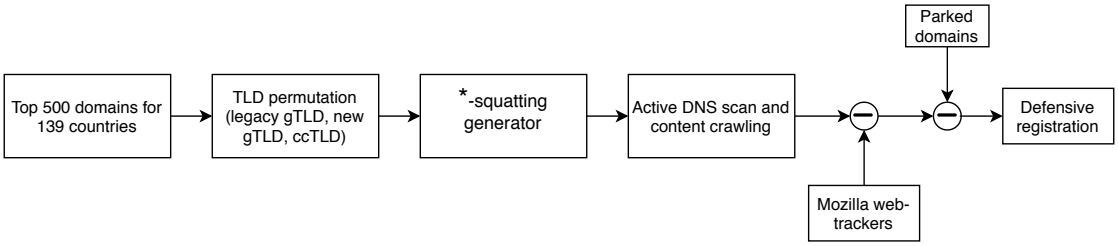


Figure 3.2 – Generating the list of defensively registered domains.

there is no DMARC policy specified for subdomains and the domain itself, then again, it is possible to send spoofed emails.

- **Wrong SPF rules**. If the `check_host` function cannot evaluate the existing SPF record of a domain name because of a syntax error, then the result is either *Temperror* or *Permerror*, and a legitimate email will likely arrive in the spam box. However, when the user marks this email as safe, the mail service may also accept spoofed emails from other IP addresses. We show in Section 3.6 how syntactically wrong SPF rules may break the trust-based authentication system of email service providers by allowing forged emails to land in the user inbox.

3.4 Methodology for analyzing SPF and DMARC deployment

In this section, we describe the methodology for analyzing the deployment of SPF and DMARC. We start with three datasets to perform two different measurements: in one campaign, we use a dataset of approximately 236 million domains from various resources to measure the global adoption of SPF and DMARC. In the second campaign, we use top 500 domains of 139 countries from the Alexa list [143] and online banking systems for all countries provided by FONDY.[4] In the second campaign, our focus is on high-profile domains (well-known companies, governmental websites, and financial institutions) and their defensive domain registrations.

3.4.1 Global Measurements

Regarding the global scan of domains for SPF and DMARC, we collected approximately 333 million domains from open zone files, OpenData project of Rapid7, and all the available zone files in CZDS offered by ICANN. Our data consist of all domains with `.com`, `.net`, `.org`, `.biz` gTLDs, approximately 1,100 new gTLDs, `.se` and `.nu` ccTLDs, operated by the Internet Foundation in Sweden, and samples of other domains obtained from Rapid7. Then, we scanned all the domains for `A` resource record using the ZDNS[5] scanner from the ZMap project [144] to keep only alive ones.

[4] https://fondy.eu
[5] https://github.com/zmap/zdns

Table 3.2 – Scan results for SPF rules.

dataset	total	norecord (%)	noerror (%)	servfail (%)	nxdomain (%)	timeout (%)
TOP500 domains	32,017	29.88	65.92	0.23	0.18	3.78
TOP500 subdomains	212,361	76.15	5.77	0.1	16.31	1.68
Bank domains	7,022	22.39	64.95	1.28	2.75	8.63
Bank subdomains	39,310	70.34	3.53	0.09	22.96	3.09
Defensive domains	55,095	1.2	95.37	0.43	1.03	1.97

Table 3.3 – Scan results for DMARC rules.

dataset	total	noerror (%)	servfail (%)	nxdomain (%)	timeout (%)
TOP500 domains	32,017	34.32	0.24	63.44	2.0
TOP500 subdomains	212,361	12.61	0.36	82.95	4.09
Bank domains	7,022	35.86	1.21	52.32	10.61
Bank subdomains	39,310	7.95	0.55	87.92	3.58
Defensive domains	55,095	40.08	0.36	57.86	1.7

Finally, our dataset consists of 235,960,991 active domain names in total. We performed the measurement in September 2020.

3.4.2 Top 500 Websites of All Countries

The Alexa website ranging system provides top 500 lists of most visited websites for 139 countries, which we collect for the purpose of this study as high-profile domains. Previous work [133, 89] used the Alexa top 1 million domains. However, we are interested in specific domains that may not be in the top 1M global popularity list but in the top list of each country, e.g., government websites or national businesses. In total, we collect 69,500 FQDNs, which lead to 32,042 unique domains. Domain names are defined as 2nd–level, or lower-level if a given TLD operator provides such registrations, e.g., `example.br` or `example.com.br` [16]. We use a modified version of the public suffix list maintained by Mozilla[6] to get domains from FQDNs. For the purpose of this study, we exclude all private TLDs such as `s3.amazonaws.com` or `blogspot.com`. The dataset consists of 14,084 domains with legacy gTLDs, 1,070 domains with new gTLDs, and 14,084 domains with country-code TLDs. We refer to this list as the TOP500 list.

3.4.3 Defensive Registrations

Defensive registration refers to the process of registering domain names (often across multiple TLDs) with different grammatical formats to protect brands from attacks like *typo-squatting* [145]. For example, the `brand.com` company may register `brand.net` and `brand.org`, then redirect them

[6] https://publicsuffix.org

to the original website. Figure 3.2 shows the algorithm to generate their list. We use the following steps to generate defensively registered names using the names in the TOP500 list:

- For each domain name in the TOP500 list, we generate the domain names over all the possible TLDs including new gTLDs, legacy gTLDs, and ccTLDs. For example, for `paypal.com`, we generate `paypal.tld` where `tld` refers to all the ccTLDs (e.g, `paypal.in`), legacy gTLDs (e.g., `paypal.net`), and new gTLDs (e.g., `paypal.support`).

- For each domain in the TOP500 list that uses country code TLD or legacy gTLD, we generate *-squatting domains (for *-squatting, we use insertion, deletion, substitution, and internationalized domain names using DNSTwist package[7]). We generate 145,250,849 unique domain names.

- We scan all generated domains for `TXT` records with ZDNS. By excluding all DNS error results (e.g., NXDOMAIN, TIMEOUT, and SERVFAIL), we end up with 1,185,167 unique domains. Then, we extract the defensively registered domains based on the following three conditions:

  1. IP address in the requested `A` record of the domain is the same as for the `A` record of at least one corresponding domain in the TOP500 list,
  2. authoritative name server in the `NS` record of the domain is the same as in the `NS` record of at least one corresponding domain in the TOP500 list,
  3. domain part of the automatically visited domain homepage URL is the same as one domain in the TOP500 list, and the list reduces to 235,508 domains.

- Some of the domains in the list are related to web trackers [146] and parked domains. For parked domains, we exclude them using the method proposed by Vissers et al. [147], whereas for web trackers and advertising domains, we exclude them by using the Mozilla blacklist for trackers [148].

Our final list contains 55,059 defensively registered domains. For example, we find 226 domain names either registered by Google Inc. for `google.com` or by MarkMonitor[8] on behalf of Google, and 201 domain names related to PayPal Inc.

3.4.4 Subdomain Enumeration

We have generated the list of known subdomains for each entry of the TOP500 list using the Spyse[9] API. We only consider 'first-level' subdomains and exclude `www` and name servers since it is more

[7] https://pypi.org/project/dnstwist/
[8] https://markmonitor.com
[9] https://spyse.com

Table 3.4 – Specified DMARC action for subdomains with no SPF rule in the TXT resource record.

data	total	no-DMARC	none	reject	quarantine	invalid rule
TOP500-sub-no-SPF	161,720	108,535 (67.1%)	32,008 (19.7%)	13,286 (8.21%)	7,803 (4.82%)	88 (0.05%)
Bank-sub-no-SPF	27,650	19,070 (68.9%)	4,849 (17.5%)	2,682 (9.6%)	1,023 (3.69%)	26 (0.09%)

Table 3.5 – Result of the SPF check_host emulation.

Result	TOP500	bank	defensive	bank subdomains	TOP500 subdomains
None	10,106	1,956	1,441	37,149	198,615
Neutral	1,497	236	6,220	56	683
Pass	50	10	114	2	37
Fail	7,083	2,268	22,255	860	4,511
Softfail	10,617	1,591	21,804	354	6,019
Temperror	135	155	523	778	1,485
Permerror	2,529	806	2,738	111	1,011
Total	32,017	7,022	55,095	39,310	212,361

likely that attackers use a first-level subdomain for sending spoofed email since it looks more legitimate. In total, we generate 212,361 subdomains for domains in the TOP500 list.

3.4.5 Banks and Financial Websites

For banking and financial websites, we leverage a list of 7,022 domains from the FONDY Github repository[10] and generate 39,310 subdomains using the same method as described in the previous section.

3.5 Results on SPF and DMARC Adoption

After collecting all the datasets, we perform three types of scans for all domains and subdomains: 1) find TXT records to extract SPF rules, 2) find TXT records by prepending _dmarc to the domains and subdomains (i.e., _dmarc.domain.tld) to retrieve DMARC rules, and 3) analyze SPF and DMARC rules by emulating the check_host function [149] using our server IP address as the IP address of the sender (without actually sending emails).

In this section, we present the results of the first two scans for SPF and DMARC rules at each domain and its subdomains.

3.5.1 Global Scan of the SPF and DMARC Rules

As the result of scanning 236 million domain names, we find that only 73,833,342 domains have SPF records set, which is approximately **31%** of all domains. The comparison of the obtained

[10] https://github.com/cloudipsp/all_banks_ips

results with the scanning results of the top 1M domains in the Alexa list performed by Hu et al. [89] with 44.9% SPF adoption rate, shows that the global adoption of SPF is approximately **13.9%** lower than in the Alexa top 1M domains. We expected this result because Alexa top 1M domain names are more valuable and well-established in terms of DNS resource records, and therefore, they do not give a representative overall picture of the global SPF deployment.

Regarding DMARC, only 310,185 out of 236 million domains have DMARC corresponding to approximately **0.13%** of the population. For the domains with a DMARC rule, 41% of them have `p=reject`, 9.3% have `p=quarantine`, and 39.6% have `p=none` rule. These figures are also far different from the 5.1% of the domain names in the Alexa top 1M domains with DMARC rules [89], which again confirms that more popular domain names deploy email anti-spoofing schemes on a wider scale.

3.5.2 High-Profile Domains and Defensive Registrations

Tables 3.2 and 3.3 present the results of the scans using ZDNS to retrieve SPF and DMARC rules. Columns contain the following information: 'norecord' – domains exist but there is no SPF rule in the `TXT` record of the domains, 'noerror' – the record exists and can be retrieved successfully, 'servfail' – DNS lookup failure, 'nxdomain' – the domain name does not exist in the zone file, 'timeout' – the DNS timeout error. For DMARC, the 'nxdomain' column is the same as 'norecord' column for SPF (if we get 'NXDOMAIN' answer to the DNS query for `_dmarc.domain.tld`, it means that `_dmarc` subdomain does not exist so there is no DMARC rule).

We can notice in Table 3.2 that **29.9%** of the domains in the TOP500 list and **22.4%** of the online banking domains do not have SPF rules at all. As the `check_host` function for the domains without SPF rules returns *None* (see Table 3.1), it is up to the receiver of the email to decide on whether to deliver a message and/or mark it as suspicious or not. While this behavior can be acceptable for regular domains, it is insecure for transactional domains (e.g., banking domains) as well as for high-profile domains (e.g., domains in the TOP500 list).

For defensively registered domains, Table 3.2 shows that only **1.2%** of them have no SPF rules, which is significantly lower than the results for TOP500 and banking domains. However, evaluating SPF alone is not sufficient since it is up to DMARC policies to make the final decisions about the delivery of messages.

As shown in Table 3.3, as many as **63.4%** and **52.3%** of TOP500 and banking domains have no DMARC rule, which means that even with correctly configured SPF rules, it is still possible to spoof emails. Furthermore, for the domains with a DMARC rule in place (34.3% and 35.9% for TOP500 and banking domains, respectively), we have observed that a large part of them have the tag `p` equal to `none` (**60%** and **53.8%**, respectively, not shown in the table), which make them prone to

email spoofing as well.

For defensively registered domains (see Table 3.3), **57.9%** of them do not have a DMARC rule, which means that it is possible to send spoofed emails. Among 40.1% of the domains with a DMARC rule, **26.7%** have the `p` tag equal to `none` and 65% have the `p` tag set to `reject`, which makes them bulletproof from domain spoofing at the SMTP level.

Overall, we expect much wider deployment of SPF and stricter DMARC rules for defensively registered domains in comparison to high-profile domains—if organizations decide to register domains defensively to avoid domain name abuse, they are also more likely to configure the appropriate SPF and DMARC rules.

3.5.3 Analysis of Spoofing Possibilities for Subdomains

Regarding subdomains, the results are worse since **76.1%** of the subdomains related to the domains in the TOP500 list and **70%** of the subdomains related to banking websites do not have SPF records at all (see Table 3.2). While it is not dangerous in itself, the absence of strict DMARC rules for subdomains makes them prone to subdomain spoofing. To mitigate this vulnerability, domains need to provide appropriate DMARC rules. The `sp` tag (or `p` tag in the absence of `sp`) in a DMARC rule specifies the default action to be taken upon receiving messages from subdomains with no SPF rule [62].

Table 3.4 shows the DMARC results for subdomains without SPF rules in both TOP500 and banking website lists. To obtain this result, we first scan `_dmarc.sub.domain.tld` to extract a `p` tag from each subdomain and in case of no DMARC rule in the subdomain, we scan `_dmarc.domain.tld` for `sp` or (in the case of its absence) `p` tags and apply the rule to subdomains (cf. RFC 7489 for more details [62]). In Table 3.4, `none`, `reject`, and `quarantine` columns correspond to the extracted rules as explained in Section 3.3.3. The 'invalid rule' column refers to the rules that do not follow the syntax specified in RFC 7489 and 'no-DMARC' column corresponds to the domains without DMARC rules in subdomains nor in the domain name. Note that sending emails from a subdomain of any domain with 'no-DMARC' (**67.1%** for TOP500 and **68.9%** for banking websites), with `none` rule (**19.7%** for TOP500 and **17.5%** for banking websites), and 'invalid-rule' (less than 0.1% in both cases), regardless of the fact if the subdomain exists or not (non-existing subdomains), does not result in a strict reject decision. This behavior is potentially dangerous for transactional domains as it is possible to send emails with forged sender address using subdomains with no SPF record for as many as approximately **87%** of TOP500 and banking domains.

Table 3.6 – Selected syntactically wrong rules that lead to the *Permerror* result in SPF.

Error type	Example	Correct rule	Frequency
Too many DNS lookups	-	SPF rule must generate less than 10 DNS query	4,349,463 (70%)
Two or more SPF records	-	must set one SPF record for each domain	733,750 (12%)
No valid SPF record for included domain	-	must set one SPF record for included domains	556,811 (9%)
Unknown mechanism found: all.	v=spf1 a mx -all.	v=spf1 a mx -all	153,455 (2.5%)
Invalid IP4 address: ip4:	ip4:xxx.xxx.xxx.xx?all	ip4:xxx.xxx.xxx.xx ?all	72,011 (1.1%)
Empty domain:: a:	v=spf1 mx a: -all	v=spf1 mx a:example.com -all	18,190 (0.2%)

Table 3.7 – Measurements of message delivery to inbox (IN), spambox (SP), or no delivery (ND) for five major email service providers.

Threat model	Gmail			Yahoo			Outlook			Yandex			Laposte		
	IN	SP	ND	IN	SP	ND	IN	SP	ND	IN	SP	ND	IN	SP	ND
+all in SPF of domain	10	0	0	8	1	1	6	0	4	10	0	0	6	0	4
Defensive registration	9	1	0	9	1	0	3	7	0	9	1	0	9	0	1
Non-existent subdomain	8	2	0	3	0	7	2	8	0	10	0	0	10	0	0
Existent subdomain	7	3	0	7	2	1	4	6	0	10	0	0	10	0	0
Trust-based authentication issue		✗			✓			✓			N/A			✗	

3.5.4 SPF Emulation Results

To analyze the validity of SPF rules using the `check_host` function further, we take advantage of `pyspf` [149] with our server IP address as the IP address of the mail sender. `pyspf` evaluates the SPF rule for a given domain and returns the SPF result. For the global scan of SPF and DMARC, 213,112 out of 73,833,342 domains result in SPF *Pass*, approximately 0.28% of the domains with SPF records. We also found that 6,199,210 domains (8.3% of the domains with SPF records) result in SPF *Permerror*.

Regarding the second database, Table 3.5 shows the results of the SPF emulation (see also Table 3.1 for the definition of each result and the corresponding recommended action). The reason for the SPF *Pass* result is either the `+all` mechanism in the SPF rule or the possible `redirect` modifier. Among the defensively registered domain names with the *Pass* result (114 domains), we have observed some well-known names like `microsoft.com.tr`[11] registered by MarkMonitor Inc.[8] on behalf of the Microsoft Corporation, as well as some major IT companies, local government, and TV channels websites for which we cannot provide the names for security considerations. However, the emulation results are available upon request.

We have noticed 12 different banking websites (1 in Spain and 11 in the United States) with the SPF *Pass* result. Although the number is fairly low, it is still enough for attackers to conduct a successful attack if they obtain the list of customer emails. In the TOP500 list for domains and subdomains, we have found 87 records with the SPF *Pass* result (50 for domains and 37 for subdomains)

[11] The issue was fixed after sending notifications.

including several local government websites (mostly in the US), national financial websites, and national mobile operators with thousands of customers.

Table 3.5 shows 7,195 *Permerror* as the result of the `check_host` function. The majority of these domains and subdomains have at least one of the following three problems: i) syntax problem in the published SPF rule (approximately 5,400 records), ii) excessive number of DNS lookups because of too many recursive `include` mechanisms [50] (1,131 records), and iii) published more than one valid SPF records (640 samples). Table 3.6 shows selected syntactically and semantically wrong published SPF records. We can observe that not only the syntax is important to parse an SPF record correctly, but also the number of DNS lookups must be limited to 10 queries based on RFC 7208 (cf. Section 4.6.4). Approximately 91% of the SPF *Permerror* results are related to only three types of misconfigurations with a 70% violation in the number of DNS queries, followed by 12% of domains with more than one SPF record.

The domains and subdomains with *Permerror* are important because they may cause serious problems. Since the domains have SPF records, *Permerror* indicates that they are used by their owners to send legitimate messages to users. However, emails may never get delivered or delivered but labeled as spam (based on the action recommended for *Permerror* as described in Table 3.1). Importantly, we find that any attempt by the end user to detach the spam label from the legitimate email may whitelist all the emails from that domain name with the SPF *Permerror* result including forged emails (see Section 3.6).

Moreover, a wrong implementation of the `check_host` function on the receiver without strict limitation of the number of DNS queries, may allow the attacker to put extra burden on the local recursive DNS resolver, which may lead to a Denial of Service (DoS) attack against the DNS server, as explained by Scheffler et al. [134]. Among the domains with syntactically wrong SPF rules, we observe some major IT companies e.g., `eset.lu`, the defensively registered domain for `eset.com` related to the ESET Internet Security.[11]

The SPF emulation results show that for several major IT companies, government websites, and one of the topmost banking website in the world, it is possible to send spoofed emails from both existent and non-existent subdomains as well as from some of their defensively registered domains due to weak or misconfigured SPF or DMARC rules.

3.5.5 End-to-End Spoofing Measurement

To show the possibility of email spoofing based on the different threat models presented in Section 3.3.4, we have tested end-to-end email spoofing from well-known brands to our own registered email addresses at i) Gmail, ii) Yahoo, iii) Outlook, iv) Yandex, and v) Laposte email services. We follow the same steps as Hu et al. [89] to ensure research ethics. Table 3.7 shows the test results. We

have considered four different possibilities, namely, a) the SPF record of the domain has `+all` in its rule set, b) the defensively registered domain has neither an SPF nor DMARC rule to reject our emails, c) non-existent subdomains (e.g., `accounts`.icann.org), and finally, d) an existent subdomain without proper SPF configuration or a restrictive DMARC rule (e.g., `account`.icann.org). For ethical reasons, we do not provide the brand names of high-profile domains on behalf of which we sent emails, because for some of them, the problem is still unsolved.

We can observe in Table 3.7 that in the first case (for which there is a `+all` in the SPF record), almost all the emails were delivered into the inbox by Gmail, Yahoo, and Yandex. Outlook and Laposte perform slightly better with 60% inbox delivery and 40% rejected emails. For the defensively registered domains, except for Outlook (with 70% delivered into the spam-box), all other email service providers successfully delivered almost all the sent mails into their inbox. Regarding non-existent subdomains, Outlook labeled 80% of the emails as spam while Yahoo rejected 70% of them. Other three services delivered almost all the emails. For the existent subdomain, Outlook performed the best by labeling 60% of the emails as spam. Surprisingly, Yandex delivered 97.5% of all sent emails into inbox, the worst performance in terms of the SPF and DMARC evaluation. The results show that attackers can successfully spoof all the tested email services by sending emails from non-existent subdomains, if domains do not have a strict reject DMARC policy.

3.6 Trust-based Authentication Issue

In this section, we show how a syntactically wrong SPF rule in a legitimate domain can push users to break the trust-based authentication system by labeling a legitimate email as safe and letting forged emails land in the user inbox. We examine five popular email providers: Outlook, Yahoo, Gmail, Laposte, and Yandex. We explain the issue using the Outlook service as an example, but the process is the same for other email service providers. Table 3.7 presents the summary of results.

First, we register a domain (`dnsabuse.xyz`), set up a mail server, and the DNS `A` record of the domain. We use `v=spf1 a aaaa -all` as the SPF rule in the `TXT` record for our registered domain (i.e., syntactically wrong SPF rule because of a nonexistent `aaaa` mechanism to generate the *Permerror* result). Then, we send a legitimate email with our server to our `outlook.com` email address. Since the SPF record is syntactically wrong and the reputation of our domain is low, the legitimate email lands in the spam box (as we expect) with the SPF *Permerror* result. If the user marks the email as *'safe sender'* (in case of Yahoo, the button label is *'add sender to contacts'*), then the Outlook service considers this email as safe (a correct assumption as it is a legitimate email). However, from now on, Outlook (as well as Yahoo) also accepts spoofed emails from other IP addresses that spoof the domain name.

We suspect that Yahoo and Outlook services whitelist the sender domain name instead of their IP addresses. On the other hand, the Laposte service rejects the sender with SPF *Permerror* at the SMTP level and sends a bounce message informing the sender about the reason for rejecting the mail (i.e., syntax error of SPF). We were not able to evaluate the trust-based authentication for Yandex since both emails (from the legitimate and illegitimate servers) land in the user inbox. Finally, Gmail does not suffer from the issue. We assume that when users detach the spam label from a legitimate email, Gmail only whitelists the IP address instead of the domain name.

3.7 Methodology for Preventing Domain Spoofing

In this section, we present a methodology for preventing domain spoofing elaborated based on the experience gained in a study of a real-world scenario related to attacks performed on one of the government financial sectors in a European country. Due to security and ethical considerations, we do not give the name of the organization nor the name of the country.

The organization had one official registered domain (with ccTLD) and more than 500 defensively registered domains to protect the official one. In 2019, the domain administrators realized that a quite considerable number of attacks targeted their organization using different attack vectors: i) sending a forged email on behalf of the main domain, ii) sending emails with the `MAIL FROM` address of the defensively registered domains, and most importantly iii) sending emails from non-existent subdomains of either the main domain or defensively registered domains. The main problem was that the targeted organization had no control over any part of the attack scenarios. They did not know anything about the sender, which could be the attacker or a compromised machine sending spoofed emails on behalf of the attackers, nor anything about the receivers of the emails. Thus, to solve the problem, not only they had to identify the sender but also inform possible recipients so that they do not accept incoming messages and potentially send a report related to these emails. Figure 3.3 illustrates the resulting methodology for preventing domain spoofing, a combination of good practices for managing SPF and DMARC records and analyzing DNS logs.

Assume that the IP address of the attacker is `1.2.3.4`, the host name used in the SMTP `HELO/EHLO` command is `helloserver.tld`, and the `MAIL FROM` field used in the spoofed email is `organization.com`, the same as the targeted brand. In this scenario, the SPF rule of the main domain, as well as all the defensively registered domain names, point to a single subdomain (`_spf.organization.com`) under the control of the organization using `redirect` modifier. When the receiver receives a spoofed email on behalf of `organization.com` (or of any defensively registered domain), it asks for the `TXT` record, retrieves the SPF rule of the domain (step ①), and gets the following answer: `v=spf1 redirect=_spf.organization.com` (step ②). In step ③, the receiver again asks for the SPF record of

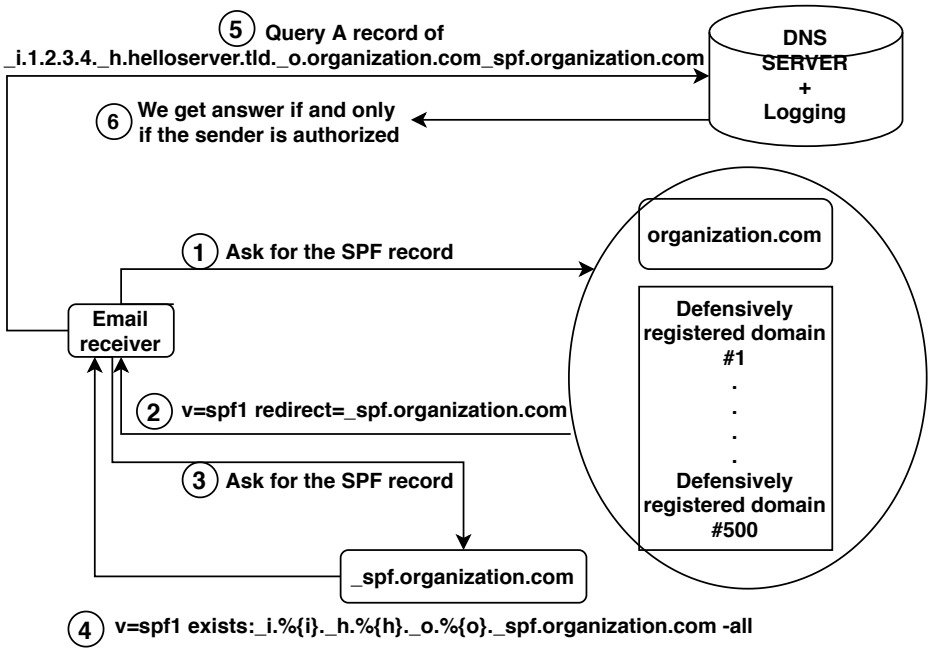


Figure 3.3 – Methodology for preventing domain spoofing.

the specified domain name in the `redirect` modifier and receives a macro specified by the `exists` mechanism (step ④). The `exists` mechanism tells the receiver to create the domain name based on the specified rules and query the generated domain for the `A` resource record. The receiver can make the final decision based on step ⑥. If the domain name in step ⑤ resolves (no matter to which IP address), it means that the email is legitimate in terms of SPF. However, if the query returns no result (e.g., NXDOMAIN) not only the SPF will fail but also the DNS server logs the IP address of the attacker (or the compromised machine used by attacker) as well as the targeted domain (the main domain of the organization or one of the defensively registered domains). In addition, the receiver sends an extra `TXT` query for the DMARC policy to make the final decision about the received email. By specifying the `ruf` field in the DMARC rule, the domain administrators will receive a copy of the rejected email (e.g., phishing email) for further forensic analysis both to identify bugs in their mail software and gain better insight into the possible phishing/spam attacks on their domains.

After one year of using the methodology to protect the targeted organization, the results show that this technique can effectively reduce the number of phishing attempts on the organization, which we can consider as a good practice not only to protect the brands from phishing/spam attacks that use domain spoofing but also to identify the malicious email senders.

3.8 Remediation

Notifying the owners of the affected domains with misconfigured or missing SPF and DMARC rules is highly problematic since there is no straight way to retrieve the contact information of the domain owners [150, 151]. Public availability of the domain WHOIS data is affected by the introduction of the General Data Protection Regulation (GDPR) and "Temporary Specification for gTLD Registration Data" adopted by ICANN [34]. It obliges generic TLD registries and registrars to redact the Registrant and Administrative Contact in the public WHOIS.

Therefore, we decided to perform notifications through the CSIRTs. We use the following bottom-up approach to send notifications—we send email notifications if there is a CSIRT responsible for: 1) the domain name, 2) the TLD of the domain (mostly in case of private TLDs), 3) the IP range to which the IP address of the domain belongs to, 4) the autonomous system of the IP address for that domain, or 5) the national CERT responsible for the TLD (in case of country-code TLD) or the entire IP address space.

We used this approach to perform two notification campaigns: the first one for high-profile domains, which are more critical to be fixed as soon as possible, in December 2019, and the second campaign related to the global scan in September 2020.

3.8.1 Results of the First Notification Campaign

Regarding high-profile domains, we have sent 128 emails to notify CSIRTs responsible for 7,653 domains with SPF *Pass* or *Permerror* results.

We were not able to find any abuse contact address of responsible CSIRTs for 573 domains. For some high-profile domains prone to phishing attacks, e.g., `microsoft.com.tr`, we manually visited their websites and contacted them directly. In the first 5 days after sending notifications, we repeated our scans and found that 160 domain owners re-configured their SPF rules. The quickest clean-up action was initiated by the US government CERT (50 domains), national CERT of Austria (7 domains), Spain (7 domains) followed by CERT Polska, French CERT (ANSSI) and Danish CERT (CFCS-DK): 5 domains each.

Re-scanning the same set of domain names in October 2020 shows that 1,734 domain names changed their status from *Permerror* to *Softfail* (663 domains), *Fail* (569), *Neutral* (83), *None* (361), *Pass* (2), and *Temperror* (56). Moreover, 43 out of 152 high-profile domains changed their status from *Pass* to another status. Note that it is challenging to assess the effectiveness of our notification campaign because administrators may replace, for example, one misconfiguration by another (e.g., *Permerror* by *Pass*), however overall, after notifying CSIRTs responsible for misconfigured domains, as many as **23.2% (1,777 out of 7,653) were re-configured**.

3.8.2 Results of the Second Notification Campaign

Regarding the global scan, we found the total number of 6,412,322 misconfigured domains, 213,112 with SPF *Pass* results, and 6,199,210 with SPF *Permerror* results. For 23,116 domain names, we were not able to find any contacts to responsible CSIRT. Using the same above-mentioned notification approach, we sent emails to 110 CSIRTs. For some CSIRTs, due to the large size of the attachment files, we had to send two separate emails, one related to domains with SPF *Pass* and the other one related to SPF *Permerror*. Then, we re-scanned the domains every week to see how CSIRTs react to our notifications. After one week, we observed changes in the SPF results of 11,552 domains and another 917 domains after the second scan (we did not observe any major change after the third scan). For those domains that changed their SPF results, 567 changed from *Pass* to *Fail*, 56 domains to *Neutral*, 8,792 domains to *None*, 2,344 to *Softfail*, and others to *Temperror* and *Permerror*. We also did not observe any major changes in domains with *Permerror*. Overall, after notifying CSIRTs responsible for affected domains, **0.2% (12,469 out of 6,412,322) were re-configured.**

The differences between the remediation rates of the first and second campaign are to be expected and likely caused by: 1) the importance of vulnerable domains (high-profile domains are more likely to be fixed), 2) the magnitude of vulnerable domains (the number of vulnerable domains in the second campaign was three orders of magnitude larger). The magnitude of vulnerable resources is important since obtaining contact information at scale is highly problematic (for researchers, security companies, or CSIRTs), especially after the introduction of GDPR, and there is no alternative method suitable for large-scale notifications [151].

3.8.3 Notes on Notification Campaigns

We present below more insight into our notification campaigns and summarize major problems we encountered.

Figure 3.4 (see Appendix) shows the email template of the first notification campaign we sent to CSIRTs about vulnerable/misconfigured SPF records. Although we did not explain the problem in detail, we received many replies from the CSIRTs in the first 24 hours after sending notifications either thanking us for notifying them (we only consider manually typed emails rather than automatic replies) or with followup questions about the problem, e.g., whether we can prove it by sending a spoofed email. Figure 3.5 (see Appendix) shows one of the replies we received from one of the CSIRTs stating that they do not understand the problem and they think that the receiving MTA should be "smart enough" to handle *Permerror* responses. After providing the proof of concept, they notified the domain owners and fixed all the SPF records. On the other hand, in the second campaign, we used a more detailed email template and explained more about the problem

(for each domain, we specified the reason for misconfiguration, i.e., *Permerror* or *Pass*).

Hello,

We are writing to inform you of a misconfiguration in the Sender Policy Framework (SPF) of the domain names under your jurisdiction. This means that attackers are able to send spoofed emails on behalf of these domains.

Please find the list of vulnerable/misconfigured domains along with the corresponding SPF error in the attached file.

This vulnerability/misconfiguration has been rated as 5.4 out of 10.0, according to the scale published on the Common Vulnerability Scoring System (CVSS).
More information about the score of the vulnerability/misconfiguration can be found here: https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator?vector=AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

The vulnerability/misconfiguration was brought to our notice on ${date_of_scan}.

If you have any question regarding this matter, please feel free to write us at ${our_email} referencing ${subject_of_notification}.

Sincerely,
${your_name}
${affliation}

Figure 3.4 – Content of the email for the first campaign.

Note that re-configuring domain names does not necessarily mean that the domain owners permanently solved the problem. As mentioned earlier, for 8,792 domains, the SPF result changed from *Pass* to *None*, which means that either the administrators removed the SPF record (possibly thinking that removing the record is better than setting a wrong one) or the domain just expired and was not registered anymore.

Sending large scale notifications present its own difficulties already discussed in previous work [151, 150, 152]. In addition to them, we encountered three major problems with sending emails to CSIRTs:

- Some countries do not have an official CSIRT to notify.
- Some CSIRTs do not have an officially published email address. Therefore, to notify them,

----------CSIRT reply----------
Hello,
Thank you for your notification.

We were looking into the reported domains and tried to reproduce your observations/thoughts.

However, we didn't come to the same conclusion.

In case of (partially) invalid SPF records like most of the reported domains are, the system is smart enough to accept the specifically mentioned IPs, but refuse everything else. I'm not sure how you think an attacker could abuse those.

Could you perhaps pick an example of the supplied list and explain what you have in mind?
----------OUR reply----------
We provided the POC by spoofing one of the domains

----------CSIRT reply----------
I got it. Despite the fact that the RFC in paragraph G.3. encourages to take special care of Permerror on checking site, it looks like most companies are just openly letting mails through. Definitely not what I expected.

We informed all the domain responsible, also for the ticket you opened through ******* (they all end up at the same place, so please be invited to use ******** in the future).

Figure 3.5 – Sequence of the emails we have exchanged with one of the CSIRTs.

one needs to fill an online form on their websites making it impractical for large scale notifications.

- Finally, some CSIRTs changed their email addresses so that we received bounced emails.

Overall, our experience from the two notification campaigns shows that reporting vulnerabilities through CSIRTs can be effective but depends on its possible impact and magnitude of affected resources.

3.9 Conclusion

It is paramount for high-profile domains and defensively registered domains to establish appropriate SPF and DMARC policies to reduce the chance of successful spear phishing attacks. In this chapter, we evaluate the adoption of the SPF and DMARC security extensions by domain names

in two phases and analyze spoofing possibilities enabled by the absence of their rules or their misconfigurations. The results show that a large part of the domains do not correctly configure the SPF and DMARC rules, which enables attackers to successfully deliver forged emails to user inboxes. In particular, we show that for top 500 domains of 139 countries, the adoption rate of SPF and DMARC records are 65.9% and 34.3%, respectively. For banking websites, we obtain almost the same results (64.9% and 35.9%) as for the TOP500 list. However, for defensively registered domains, the results are significantly higher especially in terms of published SPF records with 95.37% adoption and 40.1% for DMARC. We also, for the first time, investigate the problem of subdomains in the anti-spoofing techniques and their possible abuse to send forged emails.

We also emulate the SPF `check_host` function not only to evaluate *Pass* and *Fail* results but also obtain all the possible results such as *Permerror*, *None*, and *Neutral* for both domains and subdomains. The investigation shows that syntactically wrong SPF rules may break the trust-based authentication system of email service providers (e.g., Outlook and Yahoo) by allowing forged emails to land in the user inbox. To improve deployment of SPF and DMARC, we have presented a methodology for managing SPF and DMARC records and analyzing DNS logs that may prevent domain spoofing.

For remediation, we have sent the total of 238 emails to notify the CSIRTs responsible for 6,419,975 domains. Within the first two weeks after the notification campaigns, they managed to inform domain owners and re-configure DNS records of 12,629 vulnerable/misconfigured domains. More importantly, as many as 23.2% of high-profile domains were re-configured at the end. Our experience shows that disclosing vulnerabilities through CSIRTs can be effective, especially for valuable domain names. Finally, while we do not publish the scan data because of ethical concerns, we make the data available upon request to encourage reproducibility.

Chapter 4

Conclusion

The work presented in this thesis is split into three independent research regarding DNS abuse.

In the first part of this thesis (Chapter 1), we studied the state of the art of domain reputation systems. We evaluate the advantages and disadvantages of existing systems and finally we propose a new method consists of three steps: i) data collection, ii) feature extraction and, iii) machine learning to distinguish maliciously registered domain names from the compromised ones. We also proposed a new method to estimate the domain age based on publicly available resources such as archive machine and certificate transparency logs which outperforms the existing methods such as mathematical mean and median.

Then, in the second part of the thesis (Chapter 2), we studied the resilience of anti-phishing bots against advanced phishing attacks armed with anti-analysis techniques. First, we introduced three advanced evasion techniques that are used by attackers. Then, we emulated those phishing attacks each with one anti-analysis technique at the time and report them to 7 major anti-phishing engines. further, by evaluating the blacklists results, we found out that most of the anti-phishing bots are not able to detect evasion techniques.

In the third part of this thesis (Chapter 3), we measured the adoption of email security extensions for 3 different categories of domain name: i) high profile domain names such as banks and government, ii) defensively registered domains and, iii) a global measurement for 236 million registered domains. We defined different threat models for email spoofing where a subdomain is used to send spoofed emails. We also proposed a method to discover defensively registered domain names in bulk.

Bibliography

[1] P. Mockapetris, "Domain names - concepts and facilities," STD 13, RFC Editor, November 1987. http://www.rfc-editor.org/rfc/rfc1034.txt. 20

[2] P. Mockapetris, "Domain names - implementation and specification," STD 13, RFC Editor, November 1987. http://www.rfc-editor.org/rfc/rfc1035.txt. 20

[3] T. Berners-Lee, R. T. Fielding, and L. Masinter, "Uniform resource identifier (uri): Generic syntax," STD 66, RFC Editor, January 2005. http://www.rfc-editor.org/rfc/rfc3986.txt. 21

[4] CPH DNS Abuse Work Group Community, "CPH DNS Abuse Work Group Community Outreach - ICANN70," 2021. 22

[5] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, "Building a Dynamic Reputation System for DNS.," in *USENIX security symposium*, pp. 273–290, 2010. 30, 33

[6] S. Hao, A. Kantchelian, B. Miller, V. Paxson, and N. Feamster, "Predator: proactive recognition and elimination of domain abuse at time-of-registration," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1568–1579, ACM, 2016. 30, 33, 38, 44

[7] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi, "Exposure: Finding malicious domains using passive dns analysis.," in *Ndss*, pp. 1–17, 2011. 30, 38, 41

[8] "Epp status codes | what do they mean, and why should i know?," 2019. 31

[9] "Avalanche network dismantled in international cyber operation," 2016. 31

[10] A. Noroozian, M. Korczynski, S. Tajalizadehkhoob, and M. van Eeten, "Developing security reputation metrics for hosting providers," in *8th Workshop on Cyber Security Experimentation and Test ({CSET} 15)*, 2015. 31

[11] M. Kührer, C. Rossow, and T. Holz, "Paint it black: Evaluating the effectiveness of malware blacklists," in *International Workshop on Recent Advances in Intrusion Detection*, pp. 1–21, Springer, 2014. 31, 38, 41

[12] B. Stone-Gross, C. Kruegel, K. Almeroth, A. Moser, and E. Kirda, "Fire: Finding rogue networks," in *2009 Annual Computer Security Applications Conference*, pp. 231–240, IEEE, 2009. 31

[13] A. Noroozian, M. Ciere, M. Korczynski, S. Tajalizadehkhoob, and M. Van Eeten, "Inferring the security performance of providers from noisy and heterogenous abuse datasets," in *16th Workshop on the Economics of Information Security. http://weis2017. econinfosec. org/wp-content/uploads/sites/3/2017/05/WEIS_2017_paper_60. pdf*, 2017. 31

[14] D. Canali, D. Balzarotti, and A. Francillon, "The role of web hosting providers in detecting compromised websites," in *Proceedings of the 22nd international conference on World Wide Web*, pp. 177–188, 2013. 31

[15] M. Korczynski, M. Wullink, S. Tajalizadehkhoob, G. C. Moura, A. Noroozian, D. Bagley, and C. Hesselman, "Cybercrime after the sunrise: A statistical analysis of dns abuse in new gtlds," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, pp. 609–623, 2018. 31, 35, 38, 40, 45, 52, 63

[16] M. Korczynski, S. Tajalizadehkhoob, A. Noroozian, M. Wullink, C. Hesselman, and M. Van Eeten, "Reputation metrics design to improve intermediary incentives for security of tlds," in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 579–594, IEEE, 2017. 31, 35, 45, 98

[17] P. Agten, W. Joosen, F. Piessens, and N. Nikiforakis, "Seven months' worth of mistakes: A longitudinal study of typosquatting abuse," in *Proceedings of the 22nd Network and Distributed System Security Symposium (NDSS 2015)*, Internet Society, 2015. 31, 35

[18] H. Liu, K. Levchenko, M. Félegyházi, C. Kreibich, G. Maier, G. M. Voelker, and S. Savage, "On the effects of registrar-level intervention.," in *LEET*, 2011. 31, 35

[19] "Openphish." 32

[20] "Pishtank: Join the fight against phishing." 32, 68, 70

[21] "Apwg: Anti-phishing working group." 32

[22] "Urlhaus: sharing malicious urls that are being used for malware distribution." 32

[23] A. K. Jain and B. B. Gupta, "Towards detection of phishing websites on client-side using machine learning based approach," *Telecommunication Systems*, vol. 68, no. 4, pp. 687–700, 2018. 32

[24] K. Tian, S. T. Jan, H. Hu, D. Yao, and G. Wang, "Needle in a haystack: Tracking down elite phishing domains in the wild," in *Proceedings of the Internet Measurement Conference 2018*, pp. 429–442, 2018. 32, 70

[25] C. L. Tan, K. L. Chiew, K. Wong, *et al.*, "Phishwho: Phishing webpage detection via identity keywords extraction and target domain name finder," *Decision Support Systems*, vol. 88, pp. 18–27, 2016. 32

[26] "Google custom search api," 2019. 32

[27] "How we made our dns stack 3x faster," August 2019. 33

[28] V. Le Pochat, S. Maroofi, T. Van Goethem, D. Preuveneers, A. Duda, W. Joosen, M. Korczyński, *et al.*, "A practical approach for taking down avalanche botnets under real-world constraints," in *Proceedings of the 27th Annual Network and Distributed System Security Symposium*, Internet Society, 2020. 33

[29] N. Kheir, F. Tran, P. Caron, and N. Deschamps, "Mentor: positive dns reputation to skim-off benign domains in botnet c&c blacklists," in *IFIP International Information Security Conference*, pp. 1–14, Springer, 2014. 33

[30] R. S. Rao and A. R. Pais, "Jail-phish: An improved search engine based phishing detection system," *Computers & Security*, vol. 83, pp. 246–267, 2019. 34

[31] I. Corona, B. Biggio, M. Contini, L. Piras, R. Corda, M. Mereu, G. Mureddu, D. Ariu, and F. Roli, "Deltaphish: Detecting phishing webpages in compromised websites," in *European Symposium on Research in Computer Security*, pp. 370–388, Springer, 2017. 34

[32] S. Le Page, G.-V. Jourdan, G. V. Bochmann, I.-V. Onut, and J. Flood, "Domain classifier: Compromised machines versus malicious registrations," in *International Conference on Web Engineering*, pp. 265–279, Springer, 2019. 34, 38

[33] Freenom, "Free and paid domains," 2017. 35

[34] ICANN, "Temporary Specification for gTLD Registration Data," May 2018. 35, 52, 108

[35] Farsight Security, "Passive dns historical internet database: Farsight dnsdb." 35, 47

[36] E. Rescorla, "The transport layer security (tls) protocol version 1.3," RFC 8446, RFC Editor, August 2018. 35, 91

[37] "Public suffix list." 36

[38] "Wappalyzer: Identify technology on websites." 37

[39] T. Miu, A. Hui, W. Lee, D. Luo, A. Chung, and J. Wong, "Universal ddos mitigation bypass," *Black Hat USA*, 2013. 37

[40] "Global phishing survey: Trends and domain name use in 2016," 2016. 38, 44, 45, 51

[41] V. Le Pochat, T. Van Goethem, and W. Joosen, "A smörgåsbord of typos: Exploring international keyboard layout typosquatting," in *2019 IEEE Security and Privacy Workshops (SPW)*, pp. 187–192, IEEE, 2019. 38

[42] W. Wang and K. Shirley, "Breaking bad: Detecting malicious domains using word segmentation," *arXiv preprint arXiv:1506.04111*, 2015. 38, 40

[43] S. Marchal, G. Armano, T. Gröndahl, K. Saari, N. Singh, and N. Asokan, "Off-the-hook: An efficient and usable client-side phishing prevention application," *IEEE Transactions on Computers*, vol. 66, no. 10, pp. 1717–1733, 2017. 38, 41

[44] C. Whittaker, B. Ryner, and M. Nazif, "Large-scale automatic classification of phishing pages," 2010. 38, 70

[45] P. Lison and V. Mavroeidis, "Neural reputation models learned from passive dns data," in *2017 IEEE International Conference on Big Data (Big Data)*, pp. 3662–3671, IEEE, 2017. 38

[46] Y. Zhang, J. I. Hong, and L. F. Cranor, "Cantina: a content-based approach to detecting phishing web sites," in *Proceedings of the 16th international conference on World Wide Web*, pp. 639–648, 2007. 38, 70

[47] V. L. Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczyński, and W. Joosen, "Tranco: A research-oriented top sites ranking hardened against manipulation," *arXiv preprint arXiv:1806.01156*, 2018. 38, 44, 54, 62

[48] B. Anderson, S. Paul, and D. McGrew, "Deciphering malware's use of tls (without decryption)," *Journal of Computer Virology and Hacking Techniques*, vol. 14, no. 3, pp. 195–211, 2018. 38, 45

[49] "Maxmind: Detect online fraud and locate online visitors." 38

[50] S. Kitterman, "Sender policy framework (spf) for authorizing use of domains in email, version 1," RFC 7208, RFC Editor, April 2014. http://www.rfc-editor.org/rfc/rfc7208.txt. 38, 43, 88, 92, 104

[51] L.-H. Lee, K.-C. Lee, H.-H. Chen, and Y.-H. Tseng, "Poster: Proactive blacklist update for anti-phishing," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1448–1450, 2014. 39

[52] "dnstwister: The simple and fast domain name permutation engine." 40

[53] D. Plohmann, "Dgarchive," 2018. 40

[54] M. Korczyński, M. Król, and M. Van Eeten, "Zone poisoning: The how and where of non-secure dns dynamic updates," in *Proceedings of the 2016 Internet Measurement Conference*, pp. 271–278, 2016. 41

[55] NormShield Blog, "Domain shadowing," Retrieved: August 2019. 41

[56] "Wappalyzer signature list." 41

[57] "Wordpress vulnerability statistics," May 2019. 42

[58] "Exploit database." 42

[59] "Vuldb: the community-driven vulnerability database." 42

[60] J. Kornblum, "Identifying almost identical files using context triggered piecewise hashing," *Digital investigation*, vol. 3, pp. 91–97, 2006. 42

[61] L. Invernizzi, K. Thomas, A. Kapravelos, O. Comanescu, J.-M. Picod, and E. Bursztein, "Cloak of visibility: Detecting when machines browse a different web," in *2016 IEEE Symposium on Security and Privacy (SP)*, pp. 743–758, IEEE, 2016. 42, 68, 70

[62] M. Kucherawy and E. Zwicky, "Domain-based message authentication, reporting, and conformance (dmarc)," RFC 7489, RFC Editor, March 2015. 43, 88, 92, 95, 102

[63] S. Maroofi, M. Korczynski, and A. Duda, "From defensive registration to subdomain protection: Evaluation of email anti-spoofing schemes for high-profile domains," in *Proc. Network Traffic Measurement and Analysis Conference (TMA)*, 2020. 43, 89

[64] "Wayback machine general information," 2019. 44, 45

[65] "Cognitive services pricing - bing search api," Retrieved: March 2020. 44

[66] "Alexa: Seo and competitive analysis software." 44, 76

[67] "Majestic: Find out who links to your website." 44

[68] "Quantcast ranking." 44

[69] "Umbrella: top 1 million websites." 44

[70] "The spamhaus project." 45, 63

[71] M. Aertsen, M. Korczyński, G. C. Moura, S. Tajalizadehkhoob, and J. Van Den Berg, "No domain left behind: is let's encrypt democratizing encryption?," in *Proceedings of the applied networking research workshop*, pp. 48–54, 2017. 45

[72] "Phishing trends and intelligence report: The growing social engineering threat," 2019. 45

[73] R. J. Little and D. B. Rubin, *Statistical analysis with missing data*, vol. 793. John Wiley & Sons, 2019. 46

[74] S. Liu, I. Foster, S. Savage, G. M. Voelker, and L. K. Saul, "Who is. com?: Learning to parse whois records," in *Proceedings of the 2015 Internet Measurement Conference*, pp. 369–380, ACM, 2015. 46

[75] A. R. T. Donders, G. J. Van Der Heijden, T. Stijnen, and K. G. Moons, "A gentle introduction to imputation of missing values," *Journal of clinical epidemiology*, vol. 59, no. 10, pp. 1087–1091, 2006. 46

[76] "Google transparency report," 2018. 47

[77] O. Gasser, B. Hof, M. Helm, M. Korczynski, R. Holz, and G. Carle, "In log we trust: Revealing poor security practices with certificate transparency logs and internet measurements," in *International Conference on Passive and Active Network Measurement*, pp. 173–185, Springer, 2018. 47

[78] "Location-based threats: How cybercriminals target you based on where you live," 2016. 47

[79] "Comodo certification authority." 47

[80] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016. 50

[81] "Phishing activity trends report, 1stquarter 2019," 2019. 55

[82] A. Oest, Y. Safei, A. Doupé, G.-J. Ahn, B. Wardman, and G. Warner, "Inside a phisher's mind: Understanding the anti-phishing ecosystem through phishing kit analysis," in *2018 APWG Symposium on Electronic Crime Research (eCrime)*, pp. 1–12, IEEE, 2018. 57, 68

[83] N. Miramirkhani, T. Barron, M. Ferdman, and N. Nikiforakis, "Panning for gold.com: Understanding the Dynamics of Domain Dropcatching," in *In 27th International Conference on World Wide Web*, pp. 257–266, 2018. 58, 76

[84] B. W. Matthews, "Comparison of the predicted and observed secondary structure of t4 phage lysozyme," *Biochimica et Biophysica Acta (BBA)-Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975. 60

[85] D. Y. Wang, S. Savage, and G. M. Voelker, "Juice: A longitudinal study of an seo botnet.," in *NDSS*, 2013. 61

[86] P. Thomas, "Web application fingerprinting and vulnerability inferencing," 2010. 62

[87] "Save pages in the wayback machine," 2018. 62

[88] T. Van Goethem, N. Miramirkhani, W. Joosen, and N. Nikiforakis, "Purchased fame: Exploring the ecosystem of private blog networks," in *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, pp. 366–378, 2019. 62

[89] H. Hu and G. Wang, "End-to-end measurements of email spoofing attacks," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pp. 1095–1112, 2018. 68, 88, 90, 98, 101, 104

[90] T. N. Jagatic, N. A. Johnson, M. Jakobsson, and F. Menczer, "Social phishing," *Communications of the ACM*, vol. 50, no. 10, pp. 94–100, 2007. 68

[91] E. Medvet, E. Kirda, and C. Kruegel, "Visual-Similarity-Based Phishing Detection," in *4th International Conference on Security and Privacy in Communication Netoworks*, pp. 1–6, 2008. 68, 70

[92] M. Moghimi and A. Y. Varjani, "New Rule-Based Phishing Detection Method," *Expert Systems with Applications*, vol. 53, pp. 231–242, 2016. 68

[93] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine Learning Based Phishing Detection from URLs," *Expert Systems with Applications*, vol. 117, pp. 345–357, 2019. 68

[94] J. Mao, W. Tian, P. Li, T. Wei, and Z. Liang, "Phishing-Alarm: Robust and Efficient Phishing Detection via Page Component Similarity," *IEEE Access*, vol. 5, pp. 17020–17030, 2017. 68

[95] S. Marchal, J. François, R. State, and T. Engel, "Phishstorm: Detecting Phishing with Streaming Analytics," *IEEE Transactions on Network and Service Management*, vol. 11, no. 4, pp. 458–471, 2014. 68

[96] "Anti-Phishing Working Group (APWG): Cross-Industry Global Group Supporting Tackling the Phishing Menace." http://www.antiphishing.org, 2020. 68

[97] "COVID-19 Cyber Threat Coalition." https://www.cyberthreatcoalition.org, 2020. 68

[98] "X-Force Threat Intelligence Index," 2020. 68

[99] S. Chhabra, A. Aggarwal, F. Benevenuto, and P. Kumaraguru, "Phi. sh/$ ocial: the phishing landscape through short urls," in *Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference*, pp. 92–101, 2011. 68

[100] "Fortinet: Threat Landscape Report." https://www.fortinet.com/content/dam/fortinet/assets/threat-reports/threat-report-q2-2019.pdf, 2019. 68

[101] M. Egele, T. Scholte, E. Kirda, and C. Kruegel, "A survey on automated dynamic malware-analysis techniques and tools," *ACM computing surveys (CSUR)*, vol. 44, no. 2, pp. 1–42, 2008. 68

[102] A. Bulazel and B. Yener, "A survey on automated dynamic malware analysis evasion and counter-evasion: Pc, mobile, and web," in *Proceedings of the 1st Reversing and Offensive-oriented Trends Symposium*, pp. 1–21, 2017. 68

[103] S. Maroofi, M. Korczyński, C. Hesselman, B. Ampeau, and A. Duda, "Comar: Classification of compromised versus maliciously registered domains," in *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 607–623, IEEE, 2020. 68, 82

[104] I. Fette, N. Sadeh, and A. Tomasic, "Learning to detect phishing emails," in *Proceedings of the 16th international conference on World Wide Web*, pp. 649–656, 2007. 68

[105] M. Cova, C. Kruegel, and G. Vigna, "There is no free phish: An analysis of" free" and live phishing kits.," *WOOT*, vol. 8, pp. 1–8, 2008. 68, 69

[106] A. Oest, Y. Safaei, A. Doupé, G.-J. Ahn, B. Wardman, and K. Tyers, "Phishfarm: A scalable framework for measuring the effectiveness of evasion techniques against browser phishing blacklists," in *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 1344–1361, IEEE, 2019. 68, 70, 71, 76, 78, 79

[107] Y.-M. Wang and M. Ma, "Detecting stealth web pages that use click-through cloaking," in *Microsoft Research Technical Report, MSR-TR*, 2006. 68

[108] A. Oest, P. Zhang, B. Wardman, E. Nunes, J. Burgis, A. Zand, K. Thomas, A. Doupé, and G.-J. Ahn, "Sunrise to sunset: Analyzing the end-to-end life cycle and effectiveness of phishing

attacks at scale," in *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pp. 361–377, 2020. 68, 77

[109] "Threat Spotlight: Malicious use of reCaptcha," 2020. 68

[110] "Google reCAPTCHA v2," 2020. 69, 74

[111] S. Garera, N. Provos, M. Chew, and A. D. Rubin, "A framework for detection and measurement of phishing attacks," in *Proceedings of the 2007 ACM workshop on Recurring malcode*, pp. 1–8, 2007. 69

[112] X. Han, N. Kheir, and D. Balzarotti, "Phisheye: Live monitoring of sandboxed phishing kits," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1402–1413, 2016. 69

[113] M. Fraiwan, R. Al-Salman, N. Khasawneh, and S. Conrad, "Analysis and identification of malicious javascript code," *Information Security Journal: A Global Perspective*, vol. 21, no. 1, pp. 1–11, 2012. 69

[114] T.-C. Chen, S. Dick, and J. Miller, "Detecting visually similar web pages: Application to phishing detection," *ACM Transactions on Internet Technology (TOIT)*, vol. 10, no. 2, pp. 1–38, 2010. 70

[115] S. Marchal, K. Saari, N. Singh, and N. Asokan, "Know your phish: Novel techniques for detecting phishing sites and their targets," in *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*, pp. 323–333, IEEE, 2016. 70

[116] "PhishTank FAQ," 2020. 70

[117] "Comparison of web browsers," 2020. 71

[118] O. Starov and N. Nikiforakis, "Extended tracking powers: Measuring the privacy diffusion enabled by browser extensions," in *Proceedings of the 26th International Conference on World Wide Web*, pp. 1481–1490, 2017. 71

[119] "Multiple Sign-In Pages," 2015. 73

[120] " Safe Browsing APIs (v4) - Caching," 2020. 74

[121] "Browsers Market Share," 2020. 75

[122] "Opera Browser FAQ," 2020. 75

[123] T. Lauinger, A. Chaabane, A. S. Buyukkayhan, K. Onarlioglu, and W. Robertson, "Game of Registrars: An Empirical Analysis of Post-Expiration Domain Name Takeovers," in *26th USENIX Security Symposium*, pp. 865–880, 2017. 76

[124] J. C. Sipior, B. T. Ward, and R. A. Mendoza, "Online privacy concerns associated with cookies, flash cookies, and web beacons," *Journal of internet commerce*, vol. 10, no. 1, pp. 1–16, 2011. 77

[125] "Phishlabs." https://www.phishlabs.com. 79

[126] A. Oest, Y. Safaei, P. Zhang, B. Wardman, K. Tyers, Y. Shoshitaishvili, and A. Doupé, "Phishtime: Continuous longitudinal measurement of the effectiveness of anti-phishing blacklists," in *29th USENIX Security Symposium*, pp. 379–396, 2020. 82

[127] "Internet Crime Report," 2019. 88

[128] "A Phishing Campaign Reports Unusual Activity on Your Microsoft Account," 2019. 88

[129] J. Klensin, "Simple mail transfer protocol," RFC 5321, RFC Editor, October 2008. http://www.rfc-editor.org/rfc/rfc5321.txt. 88

[130] D. Crocker, T. Hansen, and M. Kucherawy, "Domainkeys identified mail (dkim) signatures," STD 76, RFC Editor, September 2011. http://www.rfc-editor.org/rfc/rfc6376.txt. 88, 92

[131] Z. Durumeric, *Fast internet-wide scanning: A new security perspective*. PhD thesis, 2017. 88, 90

[132] Z. Durumeric, D. Adrian, A. Mirian, J. Kasten, E. Bursztein, N. Lidzborski, K. Thomas, V. Eranti, M. Bailey, and J. A. Halderman, "Neither snow nor rain nor mitm... an empirical analysis of email delivery security," in *Proceedings of the 2015 Internet Measurement Conference*, pp. 27–39, 2015. 90

[133] I. D. Foster, J. Larson, M. Masich, A. C. Snoeren, S. Savage, and K. Levchenko, "Security by any other name: On the effectiveness of provider based email security," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 450–464, 2015. 91, 98

[134] S. Scheffler, S. Smith, Y. Gilad, and S. Goldberg, "The unintended consequences of email spam prevention," in *International Conference on Passive and Active Network Measurement*, pp. 158–169, Springer, 2018. 91, 104

[135] H. Hu, P. Peng, and G. Wang, "Towards understanding the adoption of anti-spoofing protocols in email systems," in *2018 IEEE Cybersecurity Development (SecDev)*, pp. 94–101, IEEE, 2018. 91

[136] S. B. Rathod and T. M. Pattewar, "Content based spam detection in email using bayesian classifier," in *2015 International Conference on Communications and Signal Processing (ICCSP)*, pp. 1257–1261, IEEE, 2015. 91

[137] U. K. Sah and N. Parmar, "An approach for malicious spam detection in email with comparison of different classifiers," *International Research Journal of Engineering and Technology (IRJET)*, vol. 4, no. 8, pp. 2238–2242, 2017. 91

[138] H. Faris, A.-Z. Ala'M, A. A. Heidari, I. Aljarah, M. Mafarja, M. A. Hassonah, and H. Fujita, "An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks," *Information Fusion*, vol. 48, pp. 67–83, 2019. 91

[139] R. Kumar, M. Punetha, H. Soni, and M. Bhattacharya, "User history based mail filtering process," in *2013 International Symposium on Computational and Business Intelligence*, pp. 143–145, IEEE, 2013. 91

[140] S. B. Abd Razak and A. F. B. Mohamad, "Identification of spam email based on information from email header," in *2013 13th International Conference on Intellient Systems Design and Applications*, pp. 347–353, IEEE, 2013. 91

[141] A. Cidon, L. Gavish, I. Bleier, N. Korshun, M. Schweighauser, and A. Tsitkin, "High precision detection of business email compromise," in *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pp. 1291–1307, 2019. 91

[142] "Dmarc overview," 2018. 95

[143] "The Top 500 Sites on the Web," 2019. 97

[144] Z. Durumeric, E. Wustrow, and J. A. Halderman, "Zmap: Fast internet-wide scanning and its security applications," in *22nd {USENIX} Security Symposium ({USENIX} Security 13)*, pp. 605–620, 2013. 97

[145] J. Szurdi, B. Kocso, G. Cseh, J. Spring, M. Felegyhazi, and C. Kanich, "The long "taile" of typosquatting domain names," in *23rd {USENIX} Security Symposium ({USENIX} Security 14)*, pp. 191–206, 2014. 98

[146] S. Schelter and J. Kunegis, "Tracking the trackers: A large-scale analysis of embedded web trackers," in *Tenth International AAAI Conference on Web and Social Media*, 2016. 99

[147] T. Vissers, W. Joosen, and N. Nikiforakis, "Parking sensors: Analyzing and detecting parked domains," in *Proceedings of the 22nd Network and Distributed System Security Symposium (NDSS 2015)*, pp. 53–53, Internet Society, 2015. 99

[148] "Shavar Tracking Protection Lists," 2019. 99

[149] "Python SPF Package," 2019. 100, 103

[150] O. Cetin, C. Ganan, M. Korczynski, and M. van Eeten, "Make notifications great again: learning how to notify in the age of large-scale vulnerability scanning," in *Workshop on the Economics of Information Security (WEIS)*, 2017. 108, 110

[151] W. Soussi, M. Korczynski, S. Maroofi, and A. Duda, "Feasibility of large-scale vulnerability notifications after gdpr," in *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pp. 532–537, IEEE, 2020. 108, 109, 110

[152] F. Li, Z. Durumeric, J. Czyz, M. Karami, M. Bailey, D. McCoy, S. Savage, and V. Paxson, "You've got vulnerability: Exploring effective vulnerability notifications," in *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pp. 1033–1050, 2016. 110