UNIVERSIDAD POLITÉCNICA DE MADRID


ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INFORMÁTICOS

# A Systematic Empirical Analysis of Unwanted Software Abuse, Prevalence, Distribution, and Economics

PH.D THESIS

**Platon Pantelis Kotzias**

DEPARTAMENTAMENTO DE LENGUAJES Y SISTEMAS INFORMÁTICOS E INGENIERIA DE SOFTWARE

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INFORMÁTICOS

# A Systematic Empirical Analysis of Unwanted Software Abuse, Prevalence, Distribution, and Economics

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF:
*Doctor of Philosophy in Software, Systems and Computing*

Author: **Platon Pantelis Kotzias**

Advisor: **Dr. Juan Caballero**

April 2019

**Chair/Presidente:** Marc Dasier, Professor and Department Head, EURECOM, France
**Secretary/Secretario:** Dario Fiore, Assistant Research Professor, IMDEA Software Institute, Spain
**Member/Vocal:** Narseo Vallina-Rodriguez, Assistant Research Professor, IMDEA Networks Institute, Spain
**Member/Vocal:** Juan Tapiador, Associate Professor, Universidad Carlos III, Spain
**Member/Vocal:** Igor Santos, Associate Research Professor, Universidad de Deusto, Spain

# Abstract of the Dissertation

Potentially unwanted programs (PUP) are a category of undesirable software that, while not outright malicious, can pose significant risks to users' security and privacy. There exist indications that PUP prominence has quickly increased over the last years, but the prevalence of PUP on both consumer and enterprise hosts remains unknown. Moreover, many important aspects of PUP such as distribution vectors, code signing abuse, and economics also remain unknown. In this thesis, we empirically and systematically analyze in both breadth and depth PUP abuse, prevalence, distribution, and economics. We make the following four contributions.

First, we perform a systematic study on the abuse of Windows Authenticode code signing by PUP and malware. We build an infrastructure that classifies potentially malicious samples as PUP or malware and use this infrastructure to evaluate 356K samples. We show that most signed samples are PUP and that malware is not commonly signed. We also evaluate the efficacy of Certification Authority (CA) defenses such as identity checks and revocation. Our results suggest that CA identity checks pose some barrier to malware, but do not affect PUP. CA revocations are equally low for both malware and PUP. We conclude that current CA defenses are largely ineffective for PUP.

Second, we measure the prevalence of unwanted software on real consumer hosts using telemetry from 3.9 million hosts. We find PUP installed in 54% of the hosts in our dataset. We also analyze the commercial pay-per-install (PPI) service ecosystem showing that commercial PPI services play a major role in the distribution of PUP.

Third, we perform an analysis of enterprise security and measure the prevalence of both malware and PUP on real enterprise hosts. We use AV telemetry collected from 28K enterprises and 67 industry sectors with over 82M client hosts. Almost all enterprises, despite their different security postures, encounter some malware or PUP in a three year period. We also observe that some industries, especially those related to finance, secure their systems far better than other industries.

Fourth, we perform an analysis of PUP economics. For that, we first propose a

novel technique for performing PUP attribution. Then, we use our technique to identify the entities behind three large Spanish-based PUP operations and measure the profitability of the companies they operate. Our analysis shows that in each operation a small number of people manages a large number of companies, and that the majority of them are shell companies. In the period 2013–2015, the three operations have a total revenue of 202.5M € and net income of 23M €. Finally, we observe a sharp decrease on both revenue and income for all three operations starting mid-2014. We conclude that improved PUP defenses deployed by various software and security vendors significantly impacted the PPI market.

# Resumen de la Tesis Doctoral

Los programas potencialmente no deseados (PUP) son una categoría de software que, aunque no totalmente malignos, pueden presentar considerables riesgos a la privacidad y seguridad de los usuarios. Existen indicios que la relevancia del PUP ha aumentado rápidamente durante los últimos años, pero la prevalencia del PUP en equipos informaticos de consumidores y empresas es desconocida. Además, hay varios aspectos importantes del PUP tales como sus vectores de distribución, su abuso de la tecnología Windows Authenticode, y sus beneficios económicos que siguen siendo desconocidos. En esta tesis se analiza empíricamente y sistemáticamente, en amplitud y profundidad, el abuso, la prevalencia, la distribución y los beneficios económicos del PUP. Esta tesis engloba las siguentes cuatro contribuciones.

Primero, presentamos un estudio sistemático sobre el abuso del PUP y malware en Windows Authenticode una tecnología para firmar digitalmente código ejecutable. Construimos una infraestructura que clasifica programas como PUP o malware y la usamos para evaluar 356K muestras. Concluimos que la mayoría de las muestras firmadas son PUP y que el malware normalmente no está firmado. Por otra parte, evaluamos la eficacia de las defensas usadas por las Autoridades de Certificación (CA) tales como la verificación de identidad y la revocación. Nuestros resultados indican que la verificación de identidad constituye una barrera al malware, pero no afecta al PUP. Las revocaciones de los certificados son mínimas tanto en malware como en PUP. Concluimos que las defensas de los CAs no son eficaces para el PUP.

Segundo, medimos la prevalencia del PUP en equipos informaticos reales de usuarios usando telemetría de 3.9 millones de sistemas. Detectamos PUP en 54% de los sistemas en nuestros datos. Adicionalmente, analizamos el ecosistema de servicios comerciales de pago por instalación (PPI) y mostramos que los servicios comerciales PPI desempeñan una función importante en la distribución del PUP.

Tercero, presentamos un análisis de la seguridad informática en las empresas y medimos la prevalencia del PUP y malware en equipos informáticos de empresas. Usamos la telemetría de 28K empresas en 67 sectores industriales, con más de 82 millones

de usuarios en total. Casi todas las empresas, independientemente de sus propiedades, han sido afectadas por algún malware o PUP durante el periodo de tres años. Además, observamos que algunos sectores industriales, particularmente sectores relacionados con las finanzas, protegen sus sistemas mucho mejor que otros sectores.

Cuarto, realizamos un análisis económico del PUP. Para ello, proponemos una novedosa técnica para realizar atribución del PUP. Usamos nuestra técnica para identificar las entidades detrás de tres grandes operaciones PUP españolas y medimos la rentabilidad de sus empresas. Nuestro análisis determina que en cada operación hay un pequeño número de personas que controla un gran número de empresas, de las cuales la mayoría son empresas pantallas. En el periodo 2013–2015, las tres operaciónes tienen ingresos por un total de 202.5M € y beneficios de 23M €. Por último, observamos una disminución drástica tanto de los ingresos como de los beneficios de las tres operaciones desde mediados de 2014. Concluimos que las nuevas defensas desplegadas por grandes empresas han impactado significativamente a los servicios comerciales PPI.

# Contents


**1 Introduction** **1**
- 1.1 Code Signing Abuse . . . 3
- 1.2 PUP Prevalence & Distribution in Consumer Hosts . . . 4
- 1.3 PUP Prevalence in Enterprise Hosts . . . 4
- 1.4 PUP Economics . . . 5
- 1.5 Thesis Contributions . . . 6
- 1.6 Thesis Organization . . . 7

**2 Related Work** **8**

**3 Code Signing Abuse** **13**
- 3.1 Introduction . . . 13
- 3.2 Overview . . . 15
  - 3.2.1 Microsoft Authenticode . . . 16
  - 3.2.2 Authenticode Market . . . 19
- 3.3 Revoking Timestamped Code . . . 21
- 3.4 Approach . . . 22
  - 3.4.1 Sample Processing . . . 22
  - 3.4.2 Clustering . . . 23
  - 3.4.3 PUP classification . . . 25
- 3.5 Evaluation . . . 26
  - 3.5.1 Datasets . . . 26
  - 3.5.2 Clustering and PUP Classification . . . 28
  - 3.5.3 Evolution over Time . . . 29
  - 3.5.4 Authenticode Validation . . . 30
  - 3.5.5 Revocation . . . 33
  - 3.5.6 Timestamping . . . 36
  - 3.5.7 Largest Operations . . . 38

3.5.8 Blacklist Coverage . . . 41
3.6 Discussion . . . 41

**4 PUP Prevalence & Distribution in Consumer Hosts 43**
4.1 Introduction . . . 43
4.2 Overview and Problem Statement . . . 45
4.2.1 Pay-Per-Install Overview . . . 45
4.2.2 Datasets . . . 48
4.2.3 Problem Statement . . . 50
4.3 Identifying PUP Publishers . . . 51
4.4 Clustering Publishers . . . 52
4.5 PUP Prevalence . . . 54
4.6 Classifying Publishers . . . 56
4.7 PUP Distribution Methods. . . . 61
4.8 PUP–Malware Relationships . . . 62
4.9 Domain Analysis . . . 63
4.10 Discussion . . . 65

**5 PUP Prevalence in Enterprise Hosts 67**
5.1 Introduction . . . 67
5.2 Datasets . . . 69
5.2.1 Selection Bias . . . 73
5.2.2 Ethical and Privacy Considerations . . . 74
5.3 Threat Landscape . . . 74
5.3.1 Family Classification . . . 74
5.3.2 Malware and PUP Prevalence . . . 76
5.3.3 Malware and PUP Specificity Analysis . . . 78
5.3.4 Longitudinal Analysis . . . 79
5.3.5 Case Study: Ransomware . . . 81
5.3.6 Outside-in Perspective . . . 82
5.4 Vulnerability Patching Behavior . . . 84
5.4.1 Analysis of client-side vulnerabilities. . . . 84
5.4.2 Analysis of server-side vulnerabilities . . . 87
5.4.3 Operating System Upgrade Behavior . . . 90

**6 PUP Economics 92**
6.1 Introduction . . . 92
6.2 Overview . . . 95
6.2.1 Privacy & Legal Considerations . . . 95
6.2.2 PUP Operations Analyzed . . . 96
6.2.3 Entity Graph . . . 97
6.2.4 Input Company List . . . 98
6.3 Datasets . . . 99

6.4 Building Entity Graphs . . . . . 102
6.5 PUP Entity Graphs . . . . . 105
6.5.1 OP1 Analysis . . . . . 107
6.5.2 OP2 Analysis . . . . . 109
6.5.3 OP3 Analysis . . . . . 111
6.6 PUP Economics . . . . . 112
6.6.1 OP1 Economics . . . . . 117
6.6.2 OP2 Economics . . . . . 118
6.6.3 OP3 Economics . . . . . 119
6.7 Discussion . . . . . 120

**7 Conclusions & Future Work** **122**

**Bibliography** **126**

# List of Figures


3.1 Code signing process: ❶,❷ publisher acquires a code signing certificate providing its personal information; ❸,❹ publisher signs code; ❺,❻ (optional) publisher submits the signed code to be timestamped; ❼ publisher distributes the signed (and timestamped) code. . . . . 16

3.2 Format of a signed PE file. The red text box fields are not included in the calculation of the digest. . . . . 17

3.3 Approach overview. . . . . 21

3.4 Number of collected, signed, timestamped, signed PUP, and signed malware samples over time. The cluster classification is used to label signed PUP and malware samples. . . . . 29

3.5 Number of collected, PUP, and malware samples over time including both signed and unsigned samples. The sample classification is used to label PUP and malware samples. . . . . 30

3.6 Time difference in days between a sample was timestamped and it was first observed in VirusTotal. There are 44 samples with a negative time difference of at most -10 minutes that are not shown in the figure. . . . 37

3.7 CA-issued certificates used by the InstallRex operation over time. Each line corresponds to a different certificate and its length marks the period between the certificate issuing date and its expiration or revocation (denoted by a cross) date. A single cross in one line indicates a hard revocation, i.e., a revocation on the certificate issuing date. . . . . 38

4.1 Typical transactions in the PPI market. (❶) Advertisers provide software they want to have installed, and pay a PPI service to distribute it. (❷) Affiliate publishers register with the PPI service, provide their program, and receive a bundle of their program with the PPI installer. (❸) Affiliate publishers distribute their bundle to target users. (❹) The PPI service pays affiliate publishers a bounty for any successful installations they facilitated. . . . . 46
4.2 Cluster in-degree distribution. . . . . 57
4.3 Cluster out-degree distribution. . . . . 58

5.1 Number of families per host . . . . . 78
5.2 Monthly malware and PUP prevalence by number of hosts and enterprises with at least one encounter. . . . . 80
5.3 Monthly number of hosts and enterprises with ransomware appearances. 82
5.4 Percentage of monthly enterprise hosts per Windows OS version. . . . 90

6.1 Mock example of an entity graph. . . . . 104
6.2 Anonymized OP1 entity graph. Green nodes represent persons, orange nodes companies without code signing certificates, and purple nodes companies with certificates. . . . . 108
6.3 Each line represents the lifetime of an OP1 company. Circles mark the date of the first issued certificate of a company (if any) and stars mark the date of the company name change (if any). . . . . 109
6.4 Anonymized OP2 entity graph. Nodes are colored similar to Figure 6.2. 110
6.5 Each line represents the lifetime of an OP2 company. Circles mark the date of the first issued certificate of a company (if any) and stars mark the date of the company name change (if any). . . . . 111
6.6 Anonymized OP3 entity graph. Nodes are colored similar to Figure 6.2. 112
6.7 Each line represents the lifetime of an OP3 company. Circles mark the date of the first issued certificate of a company (if any) and stars mark the date of the company name change (if any). . . . . 113
6.8 Economic data for the three PUP operations for the period 2013-15. . 115
6.9 Percentage of samples in VirusShare that belong to each of the three operations for the period 2013–2015, over the total number of samples collected by VirusShare in that month. The two largest operations show growth until Summer 2014 where the number of samples sharply declines and does not later recover. . . . . 116

# List of Tables


3.1 CAs offering code signing certificates and timestamping. Prices are for 1-year certificates in US Dollars. Revocation shows if a malware clause is present in the CPS, an abuse contact is mentioned, and the delay to publish a revocation. A dash indicates that we were not able to find related information. . . . . 20
3.2 Clustering accuracy on labeled (signed) malware from Malicia dataset. 25
3.3 Datasets used. . . . . 26
3.4 Clustering results on signed samples. . . . . 28
3.5 Summary of PUP classification results. . . . . 29
3.6 Validation results using the default Windows policy. . . . . 31
3.7 Leaf certificates issued and revoked by CAs and used to sign PUP and malware. . . . . 31
3.8 Summary of revocation reasons. . . . . 35
3.9 Timestamping authorities used by malware and PUP: number of samples and timestamping chains for each TSA. . . . . 35
3.10 Top 10 operations. The validity period is in years and the cost in US Dollars. . . . . 36

4.1 Top 15 countries with the highest average price per install collected from 3 PPI services [1–3] on June 2016. . . . . 47
4.2 Summary of datasets used. . . . . 48
4.3 Top 20 publishers in the feed of 11M samples by number of samples and percentage over all samples signed and flagged by at least 4 AV engines. . . . . 52
4.4 Publisher clustering results. . . . . 53
4.5 Top 20 PUP publishers by installation base. . . . . 55
4.6 Top publishers by install base (benign and PUP). . . . . 56
4.7 PPI services services identified sorted by installation base. . . . . 59

4.8 PPI services found through manual analysis on PPI forums and other Internet resources that are not present in our dataset. The reseller data comes from [4]. . . . . 60
4.9 Top 30 advertiser clusters by installation base. For each publisher cluster it shows: whether we found an affiliate program (Aff), the in-degree (IN), out-degree (OD), detection ratio (DR), installation base (Hosts), number of parent PPI services (PPPI), number of child PPI services (CPPI), the main product advertised, and whether that product is a browser add-on (BAO) including toolbars, extensions, sidebars, and browser helper objects. . . . . 61
4.10 Analysis of PUP download events. . . . . 62
4.11 Top 20 ESLDs by number of distinct publishers of downloaded executables. FL means file locker, DP download portal, PPI pay-per-install service, and Oth other. For brevity, d3d6wi7c7pa6m0.cf stands for d3d6wi7c7pa6m0.cloudfront.net. . . . . 65
4.12 Top ESLDs by number of downloads from them. The two rightmost columns are the number of publishers and files of the downloads. . . . 66

5.1 Summary of datasets used. . . . . 70
5.2 Number of enterprises, hosts, IPv4 addresses, employees, and country codes for the top 20 industries sorted by number of hosts. The high number of IPs for IT Services is due to that industry including ISPs and hosting providers. . . . . 72
5.3 Breakdown of low reputation files. . . . . 74
5.4 Top 20 families by number of hosts. . . . . 76
5.5 Lax and conservative PUP and malware prevalence estimates. . . . . 76
5.6 Top 10 (most affected) and bottom 10 (least affected) industries by malware and PUP prevalence. . . . . 79
5.7 Top 10 ransomware families by number of hosts. . . . . 81
5.8 Breakdown of malicious activity exhibited by industries (top 10 and bottom 10). . . . . 83
5.9 Client application patching summary. It shows the number of application versions, the number of hosts and enterprises where the application was installed, the number of vulnerabilities analyzed, the number of hosts unpatched at the end of the analysis, the 50% and 90% enterprise patch time in days measured in this work, and the 50% and 90% consumer patch time in days measured in previous work [5]. . . . . 84
5.10 Industry ranking of vulnerability patching time (in days). . . . . 86
5.11 Summary of the server-side applications vulnerability assessment. These results are computed for the 28 K enterprises and the 112 server-side applications. . . . . 88

5.12 Summary of the server-side applications and patching behavior of the enterprise servers. Results per application are given for the top 10 vulnerable applications in number of affected servers. The average, 50%, 90% patch time and the average vulnerability window are also provided for the total 112 server-side applications. . . . . 88
5.13 Industry ranking of server-side applications vulnerability patching time (in days). Blank fields indicate industries in which a server-side application was not found. . . . . 89

6.1 Whether each operation runs a PPI service, download portals, and PUP software. . . . . 96
6.2 Attributes used in the entity graph, the objects holding the attribute, and the datasets used to obtain their information. The datasets are described in Section 6.3 and correspond to BORME (BE), HerdProtect (HP), and Infocif (IF). . . . . 98
6.3 Summary of entity graphs for the three operations. . . . . 105
6.4 Summary of financial data for all operations. Revenue, net income, and EBITDA are provided in Euros. . . . . 112
6.5 Expenses of the 3 audited companies for 2013–15. Percentages are calculated over the company's yearly revenue. . . . . 114
6.6 OP1 financial data. . . . . 117
6.7 OP2.C08 revenue split. Percentages are calculated over the company's yearly revenue. . . . . 118
6.8 OP2 financial data. . . . . 118
6.9 OP3.C18 revenue split. Percentages are calculated over the company's yearly revenue. . . . . 119
6.10 OP3 financial data. . . . . 120

# 1 Introduction

Potentially unwanted programs (PUP) are a category of undesirable software that while not outright malicious exhibit intrusive and undesirable behaviors that generate user complaints and have led security vendors to flag PUP in ways similar to malware. Salient examples of unwanted software include adware, rogue software (i.e., rogueware), and risktools. Adware performs intrusive advertising such as ad-injection, ad-replacement, pop-ups, and pop-unders. Rogueware pushes the user to buy licenses of software of dubious value, e.g., registry optimizers. Risktools contain functionality that may be used with malicious intent (e.g., concealing files, hiding windows running applications, terminating active processes.)

Unwanted software can pose significant risks to users' security and privacy. It may change users' browser settings, track users' Internet activity [6] (even when the communication is encrypted [7]), take screenshots of the users' activity without their consent [8], and even steal users' data and credentials [9]. Unwanted software can also act as an entry point of malware by serving malicious advertisements [10], pushing malware that performs clickfraud [11] and cryptocurrency mining [12], For these reasons, security vendors and large software vendors actively block unwanted software [13] and have prepared guidelines and policies for developers to follow to avoid behaviours that they consider undesirable for the users [14–16].

There exist indications that PUP prominence has quickly increased over the last years. Already in Q2 2014, AV vendors started alerting of a substantial increase in collected PUP samples [17]. Thomas et al. [18] showed that ad-injectors, a popular type of PUP that injects advertisements into user's Web surfing, affects 5% of unique daily IP addresses accessing Google [18]. Prior work [19, 20] has also measured PUP steadily increasing since 2010 in (so-called) malware feeds, to the point where in 2015 PUP samples outnumber malware samples in those feeds. Still, the prevalence of PUP on consumer hosts remains unknown.

Enterprises are also affected by PUP and malware. This is problematic because enterprises own a significant fraction of the hosts connected to the Internet. Enterprises may differ from consumers in important ways, such as using the same software across

hosts, establishing security policies, installing multiple security products, educating their employees, and having departments dedicated to securing their assets. However, the security posture of enterprises against malware and PUP is understudied and it is unclear whether enterprises are indeed more secure than consumers.

The intrusive behaviours of PUP often makes the boundary between PUP and malware blurry. However, important differences exist between the two classes such as user's consent. Malware distribution is dominated by *silent* installation vectors such as drive-by downloads [21,22], where malware is dropped through vulnerability exploitation and the user is unaware of the malware installation. In contrast, PUP is typically distributed through software bundles and is installed with the consent of the user, who (consciously or not) approves the installation on its host.

Software bundles are typically distributed through download portals, and commercial pay-per-install (PPI) services. Download portals are websites that index, categorize, and host programs. Prior work from security vendors [23–25] and academic researchers [26, 27] have studied the abuse of download portals by PUP and malware. A commercial PPI service acts as an intermediary between advertisers, who want to distribute their programs, and affiliates, who own programs (typically freeware) that users want to install. To monetize installations of freeware, an affiliate bundles the freeware with a downloader from a PPI service, which it distributes to users looking for the freeware. During the installation process of the freeware, users are prompted with offers to also install programs from the PPI advertisers. PPI services also exist for distributing malware [28], but we call those *underground PPI services* to differentiate them from the *commercial PPI services* that PUP uses. Prior work has extensively analyzed underground PPI services, however, the commercial PPI services ecosystem remains unexplored and their role in the distribution of unwanted software unclear.

Since PUP installation requires user's consent, PUP publishers employ different techniques to convince users to install unwanted software. One way is *code signing*, where the software is distributed with a digital signature which, if valid, certifies the integrity of the software and the identity of the publisher. Signed code looks benign, avoids scary warnings by Windows when executed, and maybe assigned higher reputation by security products. To sign Windows programs, publishers need to obtain a valid *code signing certificate* from a Certification Authority (CA). Acquiring a code signing certificate poses barriers to publishers of malicious software since CAs perform identity checks. Also, CAs can always revoke a certificate when it is used for malicious purposes. However, it is unclear to which extent these defenses work against malware and unwanted software, as well as, the extent of the abuse of the Windows code signing mechanism by malicious publishers.

A crucial part in fighting cyber threats is the understanding of the economic dependencies among malicious actors and their services. This may reveal cost-sensitive dependencies that can be leveraged for building and evaluating defenses [29]. Prior work has shed light on the economics of various cyber threats such as spam [30], fake antiviruses [31], and ransomware [32]. Despite anecdotal evidence that shows the profitability of PUP [33–35], many PUP operational and economics details remain un-

known. Also, prior work focuses on revenues but the net income (i.e., revenues minus the expenses) from those malicious activities remains unknown.

Performing an analysis of PUP economics requires PUP *attribution*, i.e., identify the entities behind them. One fundamental difference between PUP and malware, that facilitates PUP attribution, is that PUP is often published by companies, and companies also run the commercial PPI services used to distribute PUP. In contrast, malware publishers are cybercriminals with hidden identities and use underground PPI services also run by cybercriminals. Although PUP attribution is arguably easier than malware attribution, it is still challenging because behind PUP and commercial PPI services there are often networks of companies and those companies are created, dissolved, and renamed over time.

In this dissertation, we empirically and systematically analyze in both breadth and depth the unwanted software abuse, prevalence, distribution, and economics. First, we perform a systematic study on abuse of Windows Authenticode code signing by PUP and malware, and we evaluate the efficacy of CA defenses such as identity checks and revocation. Second, we measure the prevalence of unwanted software on real consumer hosts and expose a large number of commercial PPI services that play a major role in the distribution of unwanted software. Third, we perform an analysis of enterprise security and measure the prevalence of both malware and PUP on real enterprise hosts. Fourth, we perform an analysis of PUP economics. For that, we first propose a novel technique for performing PUP attribution. Then, we use our technique to identify the entities behind three PUP operations and measure the profitability of the companies they operate.

## 1.1 Code Signing Abuse

We begin with a systematic study on the abuse of Windows Authenticode [36] code signing by PUP and malware, and we evaluate the efficacy of CA defenses such as identity checks and revocation. In the process, we identify a problematic interaction between revocation and time stamping in Authenticode, where timestamped signed executables still validate even if their code signing certificate is revoked. To address this issue we propose that CAs perform *hard revocations* that invalidate all executables signed by a certificate.

We build an infrastructure that takes as input a large number of potentially malicious samples, thoroughly analyzes signed samples, clusters signed samples into operations, and classifies them as PUP or malware. We use our infrastructure to analyze 356 K malware samples collected from various malware feeds and distributed between 2006 and February 2015. Our analysis uncovers that most signed samples are PUP and that malware is not commonly signed. We observe PUP rapidly increasing over time in our corpus, reaching 88% of the samples in 2014. We measure the effectiveness of CA defenses such as identity checks and revocation. We find that CA identity checks pose some barrier to malware, but do not affect PUP. CA revocations are equally low for

malware and PUP certificates. In fact, the best CA revokes only 43% of the certificates it issues to malware and PUP publishers. At last, most CAs do not provide abuse email addresses and do not accurately report the revocation reason.

## 1.2 PUP Prevalence & Distribution in Consumer Hosts

We continue with the first systematic study of PUP prevalence and its distribution through commercial pay-per-install (PPI) services. For that, we use AV telemetry from a large security vendor comprising 8 billion events on 3.9 million real consumer hosts during a 19 month time period. This telemetry contains events where parent programs installed child programs and we focus on events where the publishers of either parent or child programs are PUP publishers. This data enables us to measure the prevalence of PUP on real consumer hosts and to map the who-installs-who relationships between PUP publishers, providing us with a broad view of the PUP ecosystem.

Our analysis shows that programs from PUP publishers are installed in 54% of the 3.9M hosts examined. That is, more than half the examined hosts had PUP during our study. We also show that PUP publishers are ranked among the top software publishers (benign or not). For example, the top PUP publisher is more popular than NVIDIA, a leading graphics hardware manufacturer. We estimate that the affected Internet-connected hosts are two orders of magnitude higher than our measurements and that each of the top 20 PUP publisher is installed on 10M–100M hosts.

We also identify 24 commercial PPI services in our dataset that are used by advertisers to distribute PUP. The top PUP advertisers predominantly distribute browser add-ons that perform different types of advertising and by selling software licenses for rogueware. We measure PUP distribution, finding that 65% of PUP downloads are performed by other PUP and that PPI services play an important role in PUP distribution. We also examine the malware-PUP relationships and observe that PUP distribution is largely disjoint from malware distribution.

## 1.3 PUP Prevalence in Enterprise Hosts

After our analysis of consumer hosts we proceed with, to the best of our knowledge, the largest and longest measurement study of enterprise security. For that, we use a wealth of datasets collected from a large cyber security company and public sources. Our data covers nearly 3 years and is collected from 28K enterprises with over 82M real enterprise client hosts and 73M public facing servers. At the core of our study are file reputation logs that capture the installation of files in real enterprise client hosts. These logs enable us an *internal view* of enterprise security which allows us to measure the prevalence of both malware and PUP on real enterprise hosts.

Our analysis shows that enterprises encounter malware much more often than PUP This is in contrast to our prior work on consumer hosts that have shown that 54% had

some PUP installed [37]. We also observe that almost all enterprises, despite their different security postures, will encounter some malware or PUP in a three-year period. We compare enterprises from different industries and discover that some industries secure their systems far better than others. The most secure industry (Banking) has five times less malware and PUP encounters than the worst industry (Electrical Equipment). This matches reports that banking is the industry that invests the most in cyber security products [38]. Overall, the most secure industries are dominated by finance, IT, and biotechnology, while the least secure are manufacturing-heavy including electrical equipment, automobiles, and construction equipment.

We complement the security posture analysis by examining the patching behavior of enterprises. For that, we measure the patching time of 12 popular client-side and 112 server-side applications. We discover that enterprise hosts are faster to patch vulnerabilities compared to consumer hosts and that the patching of server applications is worse than the patching of client applications.

## 1.4 PUP Economics

We conclude this thesis by proposing a generic approach for performing PUP attribution and then leveraging our approach to perform an analysis on the economics of three large PUP operations. Our PUP attribution approach uses *entity graphs*, where nodes represent companies and persons, and an edge from a person to a company indicates that the person is part of the company's management. An entity graph enables structured attribution by tracking the business relationships among persons and companies in an operation. Our approach takes as input an initial list of companies, possibly only one, known to belong to an operation. It uses company registers for obtaining company information, identifying the persons managing the company, finding other companies also managed by those persons, and generating an entity graph.

We evaluate our approach using three large Spain-based PUP operations. All three operations run a PPI service for Windows programs, but are also involved in other parts of the PUP ecosystem such as publishing their own PUP (e.g., system cleaning utilities) and managing download portals. Our analysis shows that the three operations for the period 2013–2015 have a total revenue of 202.5M €, net income of 23M €, and EBITDA of 24.7M €. The largest source of revenue for all three operations is the PPI service, which provides up to 90% of an operation's revenue. At last, we observe a sharp decrease on both revenue and income for all three operations starting mid-2014, leading to all three operations to have losses in 2015. We conclude that improved PUP defenses deployed by different vendors in mid-2014 [39–41] significantly impacted the PPI market, which did not recover afterwards.

We also observe that a small number of people manages a large number of companies. Most of these companies are *shell companies* that have no employees, no revenue, share address with other companies, are often created in batches, and have no website. The same shell companies are being used to obtain code signing certificates from CAs,

later used to sign the distributed executables.

## 1.5 Thesis Contributions

The main contributions of this thesis are:

- We perform a systematic analysis of Windows Authenticode abuse and the effectiveness of existing code signing defenses. We find that CA identity checks pose some barrier to malware but do not affect PUP. Also, CA revocations are equally low for malware and PUP.
- We identify a problematic scenario in Authenticode where timestamped signed malware successfully validates even after their code signing certificate has been revoked. To address this issue we propose that CAs perform a hard revocation, which invalidates any code signed with a certificate after this has been revoked.
- We build an infrastructure that given large amounts of potentially malicious software automatically analyzes signed samples, clusters them into operations, classifies them as PUP or malware, and produces a blacklist of malicious certificates.
- We perform, to the best of our knowledge, the first systematic study of PUP prevalence on real consumer hosts and its distribution through PPI services. We find that 54% of hosts had PUP installed during our study and that PUP publishers are ranked among the top software publishers (benign or not).
- We build a publisher graph that captures the who-installs-who relationships between PUP publishers. Using the publisher graphs we identify 24 commercial PPI services and show that PPI services play an important role in the distribution of PUP.
- We perform, to the best of our knowledge, the largest and longest measurement study of enterprise security. We discover that almost all companies encounter malware or PUP in a three-year period. We also show that enterprises encounter malware much more often than PUP. Also, we observe that specific industries (mostly finance related) secure their systems far better than others.
- We perform, to the best of our knowledge, the first economic analysis of PUP operations and specifically of commercial PPI services used to distribute PUP. For this, we propose a novel approach to perform PUP attribution using entity graphs. Nodes in an entity graph are companies or persons and edge from a person to a company indicates the person holds a management position in the company.
- We generate the entity graphs for three Spain-based operations, each running a commercial PPI service and being involved in other PUP-related activities. We

measure that the three operations for the period 2013–2015 have a total revenue of 202.5M €, net income of 23M €, and EBITDA of 24.7M €. We find that a small number of people manages a large number of companies in each operation. We observe that most of these companies are shell companies being used to obtain code signing certificates.

## 1.6 Thesis Organization

The remainder of this thesis is organized as follows. In Chapter 2, we present the related work on potentially unwanted programs, code signing abuse, enterprise security, and malware economics. The thesis contributions are presented in Chapter 3 to 6. In Chapter 3 we present our systematic study of Windows Authenticode abuse and our evaluation of the effectiveness of existing code signing defenses. Chapter 4 describes our systematic study of PUP prevalence and its distribution through PPI services on consumer hosts. Chapter 5 presents our measurement study on the security posture of enterprises. Chapter 6 details our economic analysis on three PUP operations. Finally, in Chapter 7 we conclude and discuss areas of future work.

# 2 Related Work

**PUP.** Early work on PUP focused on its deceptive methods. In 2005–2007, Edelman studied deceptive installations by spyware and other unwanted software [42]. Good et al. [43] studied the influence of the form and content of End User Licence Agreements (EULAs) on user's software installation decisions. They discovered that users have limited understanding of EULAs and often regret their installation decisions once informed of the contents of those. Good et al. [44] analyzed user behavior during the installation process of spyware and showed that a short notice before the installation, significantly reduced the number of spyware installations. In 2012, Pickard and Miladinov [45] studied a rogue anti-malware program concluding that while not malicious, it only detected 0.3% of the malware and its main purpose was convincing the user to pay the license.

Research on PUP has recently revived with a number of papers examining various aspects of PUP like prevalence, distribution, privacy implications, and defenses. Thomas et al. [18] measured that ad-injectors, a type of PUP that modifies browser sessions to inject advertisements, affect 5% of unique daily IP addresses accessing Google. Urban et al. [46] have analyzed the network communication of adware and PUP and discovered that roughly 37% of the requests issued contain user's private information that can be used for tracking user's activity.

Recent works have focused on PUP distribution through download portals, commercial PPI services, malicious advertisements, and survey scams. Security vendors have analyzed the top downloads from download portals and concluded that are bloated with PUP [23–25]. Geniola et al. [27] collected 800 installers of promoted applications from 8 download portals, executed them in a sandbox, and found that 1.3% of those installers drop well-known PUP to the system. Rivera et al. [26] measured the amount of abuse in download portals. They analyzed all Windows programs offered by 20 download portals and reported an overall ratio of PUP and malware between 8% and 26%. They also discovered two download portals, part of a commercial PPI service, that serve 100% PUP. Thomas et al. [4] analyzed the advertiser software distributed to US hosts by 4 commercial PPI services (OutBrowse, Amonetize, OpenCandy, Install-

Monetizer). They used SafeBrowsing data to measure that PPI services drive over 60 million download events every week in the second half of 2015, nearly three times that of malware. Nelms et al. [47] analyzed web-based advertisements that use social engineering to deceive users to download PUP. They found that most programs distributed this way are bundles of free software with PUP. Kharraz et al. [48] analyzed the online survey scam ecosystem and discovered that survey scams are predominatly used for PUP distribution.

Another line of work focused on defense mechanisms. Jagpal et al. [49] proposed WebEval, a system to identify malicious ad-injecting browser extensions. Banescu et al. [50] proposed a solution to counter changeware, a type of software that surreptitiously modifies resources of other software applications (e.g., the user's web browser settings). Other recent works propose graph-based approaches to detect PUP and malware distribution through downloader-downloadee relationships [51,52]. Also related, are recent works that built defenses for detecting several types of unwanted software on Android [53,54].

**Malware distribution.** Prior work has studied malware distribution through different vectors, which differs from our focus on PUP distribution. Moschuk et al. [55] crawl 18M URLs finding that 5.9% were drive-by downloads and 13.4% lead to spyware. Provos et al. [21] study the prevalence of distribution through drive-by downloads. Grier et al. [22] analyze the commoditization of drive-by downloads and compare malware distribution through different vectors, concluding that drive-by downloads dominate. Caballero et al. [56] study malware distribution through PPI services. The PPI services we study differ in that installations are not silent and are mostly used by PUP and benign software. Kwon et al. [57] recently use WINE data to investigate malware distribution through downloaders. Their work differs in that they do not distinguish malware from PUP and in that they analyze file download graphs for individual machines. Instead, we analyze download relationships between publishers on aggregate over 3.9M machines over a 19 month time period, focusing on PUP distribution through PPI services and affiliate programs.

**Code signing abuse.** Code signing is a key component of binary integrity solutions. DigSig [58] presents a Linux kernel module that validates digital signatures of programs before execution. Wurster and van Oorschot [59] protect executables from malicious modifications using self-signed certificates, where the OS kernel allows modifications only if the current and the new version of the file are signed with the same private key. Wu and Yap [60] leverage code signing in their binary integrity model. Application whitelisting relies on code signing to obtain publisher identity [61]. Recent work has examined the challenges of transparent key updates and certificate renewals in Android applications [62].

Most similar to our work on code signing abuse are measurements of signed malware by two AV vendors in 2010 [63,64]. Those works focus on 2008-2010, while our

analysis covers an 8-year span (2006-2015). Our span covers the significant increase in malware code signing after 2010. Our analysis covers many aspects not addressed in those studies such as timestamping. Our infrastructure clusters samples into operations and classifies each of them as PUP or malware, enabling the analysis of specific operations. We also analyze the revocation of timestamped executables and show the need for hard revocations. Kim et al. [65] have studied the abuse of code signing by malware and identified benign code signing certificates that were likely compromised by malicious actors as well as certificates that were issued to malicious actors that were impersonating legitimate companies.

Other work focused on attacking Windows Authenticode. They shown the possibility of injecting code and data into Authenticode signed executables without invalidating the signature [66, 67] and that executables signed using MD5 are vulnerable to collisions [68].

**Enterprise security.** To our knowledge, there are not many scientific works that performed systematic investigations on the cyber threat landscape of enterprises. This was mainly due to the absence of data that is representative and accurate enough for malware encountered by enterprises. The only way researchers could estimate the maliciousness of enterprises was to use public blacklists that provide information about known infected IP addresses [69, 70]. While the enterprise landscape is greatly understudied by the scientific community, there are many industrial annual threat reports [71–73]. These threat reports mainly focus on general statistics about malware seen on the Internet without making the distinction of the industry and consumer data. After targeted attacks towards specific industries hit the news in 2010, these reports started to provide industry-based statistics, however only on companies that encountered spear-phishing attacks [72].

Depending on which threats were popular on the particular year, these reports provide special details and create new sections that did not exist in previous years. For example, in 2018, we see extensive details about ransomware due to wannacry and petya events in 2017. While the content slightly changes, some of the sections are consistent over the years. It is typically to list the top malware families for each year, the top zero-day and normal vulnerabilities. In our work, we also provide similar statistics however focusing on the threat landscape of enterprises and their vulnerability patching behavior which is shown to be significantly correlated with future malware infections [74]. By conducting this study, our goal was to understand whether particular enterprise profiles have weaker security practices and therefore, more attention should be payed to them to fix these issues before they become the next target. One important finding we found in the course of this study was that the industries that operate with critical infrastructures are very slow to patch their applications, making them vulnerable against possible future cyber attacks.

Another line of research that relates to our work conducts studies to identify enterprise specific threat detection techniques [75–77] and protection mechanisms that rely on hardening the networks [78]. Levin et al. proposed to deploy Honeynets inside en-

terprise networks that typically have higher bandwidth usage and it is harder to detect malicious traffic [75]. The core idea here was that Honeynets are not suppose to send or receive any traffic, and anything that is observed in these are indicators of malicious traffic and can be used to identify infected machines. Yen et al. aimed at improving the incident detection rate by mining security logs that are produced by various security products in an enterprise [76]. Similarly, Oprea et al. mined large-scale log data to identify enterprise infections at earlier stage. The key insight is that the detection of early-stage infections can be modeled with belief propagation algorithms. The data is used for the experimentation was anonymized DNS logs and web proxy logs collected from a large enterprise. McDaniel et al. approaches the problem of securing enterprises differently by proposing to apply hardening policies [78]. The ambitious goal of the paper was to define the normal behavior such that anything else could be blocked. In this direction, the authors present techniques to automatically generate host profiles based on their historical interactions.

**Malware economics.** Prior work has measured the revenue of different malicious activities. McCoy et al. [30] analyzed the leaked databases of three pharmaceutical affiliate programs, finding that they had a total revenue of $170M during a 4-year period between 2005–2010. Stone-Gross et al. [79] analyzed three fake antivirus programs with a combined revenue of $130M in 2008–2010. Thomas et al. [80] measured that the 10-month revenue of 27 merchants of fraudulent Twitter accounts reach $127-459K. Pearce et al. [81] measured that the ZeroAccess botnet had earnings of $2.7M per month in 2013. Liao et al. [82] performed a one year study in 2013–2014 on the Bitcoin addresses used by CryptoLocker ransomware and made a lower-bound measurement of revenue of 1.128 BTC (i.e., $310K) per year. In our work, we measure a combined revenue across the three operations of 195M€ in the three-year period of 2013–2015 (202.5M€ throughout all the analysis period). A key difference with these works is that we have access not only to the revenue, but also to the net income of the operations. Since high revenue does not imply high profit, our data enables to truly examine how profitable commercial PPI services are.

Prior work has also studied other aspects of malware economics. Zhen et al. [83] proposed an economic model for understanding the effective rental size and the optimal botnet size that can maximize the profits of botnet masters. Cormac and Dinei [84] analyzed IRC underground markets finding that these markets are a very low-value channel for exchanging goods. Anderson et al. [85] performed a systematic study of the losses caused by various types of cybercrime. To the best of our knowledge these works have not studied the economics of PUP and commercial PPI services.

Also related are analysis of different malicious ecosystems. Caballero et al. [28] showed that miscreants can distribute their malware through underground PPI services by paying $100-$180 for a thousand unique installs in the most demanded regions. Motoyama et al. [86] analyzed CAPTCHA solving services with a cost of $1 per thousand CAPTCHAs. Thomas et al. [87] found that Google phone verified accounts are sold for $85-$500 per thousand. Twitter accounts are also offered from merchants at $1-$20

per thousand [80]. Stringhini et al. [88] showed that Twitter followers are offered for $20-$100 per thousand and promoted tweets for $10 per thousand. De Cristofaro et al. [89] showed that Facebook likes can be bought for $15–$70 for worldwide users and $60–$190 for US users. Kozak et al. [90] analyzed the underground market of code signing certificates and revealed that regular code signing certificates prices range from $350–$500 and Extended Validation (EV) certificates from $1.600–$3.000. Recently, Thomas et al. [29] developed a taxonomy of profit centers and support centers for reasoning about the flow of capital and their dependencies within the black market.

# 3 Code Signing Abuse

## 3.1 Introduction

Publishers of malicious software (malware) and potentially unwanted programs (PUP) are always looking for ways to make their code look benign in order to convince the user to install it and avoid detection. One such way is *code signing*, where the software is distributed with a digital signature which, if valid, certifies the integrity of the software and the identity of the publisher. Signed code looks more benign and may be assigned higher reputation by security products. In Windows, properly signed application code avoids scary warnings when a user executes it and is assigned higher reputation when downloaded through Internet Explorer [91]. Furthermore, kernel-mode code is required to be signed. Aware of these benefits attackers are increasingly leveraging signed code for their goals, e.g., for launching notorious targeted attacks [92–94].

To sign Windows programs, publishers need to obtain a valid *code signing certificate* from a Certification Authority (CA). This should pose a barrier for malicious software, since it requires providing the publisher's identity to the CA and paying a fee ($60–$500 for 1-year certificates). Furthermore, when malicious software is observed in the wild signed with a valid certificate, the CA that issued the certificate should swiftly revoke it. However, it is not clear how well defenses such as identity checks and revocation work. Prior work in 2010 by two AV vendors [63, 64] showed that signed samples were not uncommon in malware datasets. But, there has been no systematic study analyzing the extent to which malware (e.g., bots, trojans) and PUP (e.g., adware, bundles) are abusing code signing and how well defenses such as identity validation and revocation work.

In this work we perform a systematic study on abuse of Windows *Authenticode* [36] code signing. We identify a problematic interaction between revocation and timestamping in Authenticode, where timestamped signed executables still validate even if their code signing certificate is revoked. To address this issue we propose that CAs perform *hard revocations* that invalidate all executables signed by a certificate.

We build an infrastructure that takes as input a large number of potentially mali-

cious samples, filters out benign samples and those that are not signed, and thoroughly analyzes signed samples including their digital signatures, certificate chains, certificate revocation, and file timestamping (i.e., third-party certification of the time they saw some signed code). It also clusters signed samples into operations and classifies them as PUP or malware. Our infrastructure automatically builds a blacklist of malicious certificates, which can be used by CAs to perform revocation, or users can embed it into the Windows untrusted certificate store to block malicious code. Using our infrastructure we analyze 356 K malware samples distributed between 2006 and February 2015, of which 142 K (42%) are signed. This process outputs a blacklist of over 2,170 code signing certificates, 9x larger than existing blacklists [95].

Our analysis uncovers that most signed samples are PUP (88%–95%) and that malware is not commonly signed (5%–12%). We observe PUP rapidly increasing over time in our corpus, reaching 88% of the samples in 2014. We measure the effectiveness of CA defenses such as identity checks and revocation. We find that 99.8% of signed PUP and 37% of signed malware use CA-issued certificates indicating that CA identity checks pose some barrier to malware, but do not affect PUP. Only 17% of malware certificates and 15% of PUP certificates have been revoked, and the best CA revokes 43% of the certificates it issues to malware and PUP publishers. Most CAs do not provide abuse email addresses and do not accurately report the revocation reason. Only 53% of the revocations include a revocation reason and those with one often report key compromise even if it is a malware-abused certificate.

Our clustering of signed samples into operations and the classification into PUP and malware shows that the largest operations correspond to PUP, e.g., adware and gray pay-per-install programs that offer users to install third-party programs. We analyze the 10 largest PUP operations observing that they heavily use polymorphism in files and certificates, possibly to bypass AV and CA checks. Seven of them have multiple certificates revoked, so CAs seem to consider them malicious. To achieve certificate polymorphism, PUP publishers buy certificates from multiple CAs, modify the Subject information, and use multiple companies and individuals. For example, OutBrowse uses 40 different companies across 6 countries to obtain 97 code signing certificates from 5 CAs.

We also leverage the fact that timestamped malware contains a trusted timestamp close to its creation to evaluate how fast VirusTotal [96], a large malware repository, collects malware.

**Contributions:**

- We perform a systematic analysis of Authenticode abuse and the effectiveness of existing defenses. We identify a problematic scenario in Authenticode where timestamped signed malware successfully validates even after their code signing certificate has been revoked. To address this issue we propose that CAs perform a hard revocation, which invalidates any code signed with a certificate after this has been revoked.

- We propose a novel clustering of signed samples into operations using static features extracted from the Authenticode data. We also propose two novel techniques to classify samples as PUP or malware based on the AV detection labels.
- We build an infrastructure that given large amounts of potentially malicious software automatically analyzes signed samples, clusters them into operations, classifies them as PUP or malware, and produces a blacklist of malicious certificates.
- We use our infrastructure to analyze 356 K samples. We observe that PUP is rapidly increasing, most signed samples are PUP, and malware is not commonly signed. We measure that 99.8% of signed PUP and 37% of signed malware use CA-issued certificates and only 17% of malware certificates and 15% of PUP certificates have been revoked. Most revocations lack an accurate revocation reason. We analyze the largest PUP operations exposing that they heavily use file and certificate polymorphism. In addition, most of the largest operations have multiple certificates revoked that indicates that CAs consider them malicious.
- We leverage timestamped malware to evaluate the speed with which the VirusTotal online service collects malware.
- We setup a website for our blacklist and analysis results [19].

## 3.2 Overview

Code signing is the process of digitally signing executable code and scripts. It authenticates the code's publisher and guarantees the integrity of the code. Code signing is used with different types of code in a variety of platforms including Windows executables and kernel drivers, Java JAR files, Android applications, active code in Microsoft Office documents, Firefox extensions, Adobe Air applications, and iOS applications.

The code signing process first computes a hash of the code and then digitally signs this hash using the publisher's private key. The public key of the code's publisher is authenticated using a X509 *code signing certificate* that a certification authority (CA) issues to the publisher after verifying its identity. This code signing certificate is attached to the signed code. The CA also provides its certificate chain, anchored at a trusted root CA. This chain is attached to the signed code or made available online.

In code signing, certificates are distributed with the signed code (e.g., embedded in the executable file) to geographically distributed users. When a certificate expires, it is difficult to update all code installations with a new certificate. In contrast, Web servers can simply update their HTTPS certificate between sessions. To address this issue, some code signing solutions (e.g., Windows Authenticode, Java) introduce an optional timestamping process, that sends the signed code to a Time Stamping Authority (TSA), which certifies that it observed the signed code at a specific time.

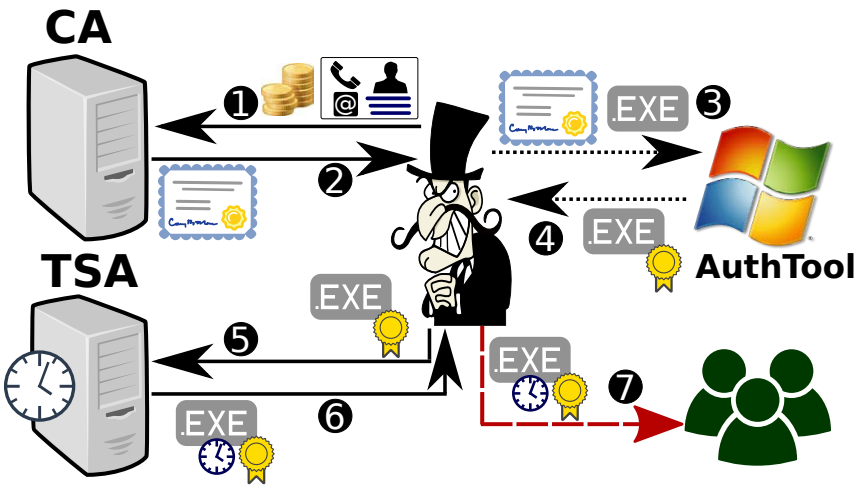



Figure 3.1: Code signing process: ❶,❷ publisher acquires a code signing certificate providing its personal information; ❸,❹ publisher signs code; ❺,❻ (optional) publisher submits the signed code to be timestamped; ❼ publisher distributes the signed (and timestamped) code.

Usually, when the code signing certificate expires, validation fails. But, if the signed code is also properly timestamped within the validity period of the code signing certificate, validation succeeds despite the code signing certificate having expired.

To timestamp signed code, the TSA embeds a timestamp, digitally (counter)signs both the timestamp and the existing code signature using its private key, and authenticates its public key by including its certificate chain anchored at a trusted root CA. Thus, code that is timestamped contains two certificate chains: the *signing chain* and the *timestamping chain*.

Figure 3.1 summarizes the code signing process. A (potentially malicious) publisher buys a code signing certificate from a CA that verifies the publisher's identity before issuing the certificate (❶,❷). The publisher signs its code using the code signing certificate and a signing tool like Microsoft's AuthTool (❸,❹). Optionally, the publisher sends the signed code to the TSA to be timestamped (❺,❻). Finally, the publisher distributes the code to the users (❼).

### 3.2.1 Microsoft Authenticode

Authenticode is a code signing solution by Microsoft [36]. It was introduced with Windows 2000, but its specification was not publicly released until March 2008. It uses a Public-Key Cryptography Standards (PKCS) #7 SignedData structure [97] and X.509 v3 certificates [98] to bind an Authenticode-signed file to a publisher's identity. Authenticode is used to digitally sign portable executable (PE) files including executa-

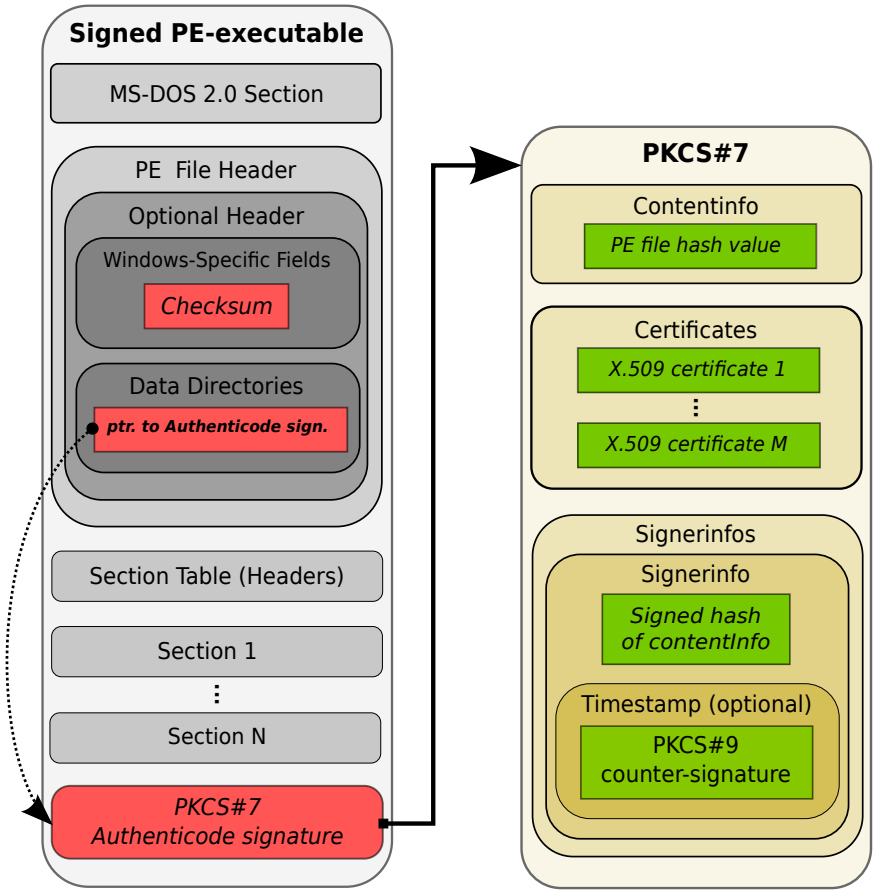



Figure 3.2: Format of a signed PE file. The red text box fields are not included in the calculation of the digest.

bles (.exe), dynamically loaded libraries (.dll), and device drivers (.sys). It can also be used for signing Active X controls (.ocx), installation (.msi), or cabinet (.cab) files.

**File format.** Figure 3.2 presents the basic format of an Authenticode-signed PE file. It contains a PKCS #7 SignedData structure (also called Authenticode signature) at the end of the file, whose starting offset and size are captured in the `Certificate Table` field in the `Optional Header`. The PKCS #7 structure contains the PE file's hash, the digital signature of the hash generated with the publisher's private key, and the certificates comprising the signing chain (the root certificate does not need to be included). It can also optionally include a description of the software publisher, a URL, and a timestamp. Authenticode only supports MD5 and SHA1 hashes and prior work has shown how to produce Authenticode collisions with MD5 [68].

When calculating the hash of the PE file 3 fields are skipped (marked in red in Figure 3.2): the Authenticode signature itself, the file's checksum, and the pointer to the Authenticode signature. In addition, the PE sections are sorted before adding

them to the hash. We call the result *Authentihash* to distinguish it from the file hash that includes all bytes in a file and is often used to uniquely identify a file (e.g., by security vendors). In the past, vulnerabilities have been disclosed where attackers could embed data in unspecified PE fields [66] and the Authenticode signature [67] without invalidating the file's signature.

**Timestamping.** Timestamping is optional in Authenticode. In order to timestamp an Authenticode-signed file, a TSA first needs to obtained the current UTC timestamp. Then, it builds a PKCS #9 counter-signature by digitally signing with its private key the timestamp and the hash of the file's signature in the PKCS #7. Next, it embeds into the PKCS #7 `SignerInfo` structure the timestamp and the counter-signature. If the optional timestamp field already existed, it is overwritten. Finally, it appends the certificates of the timestamping chain to the `certificates` part (the root certificate of the timestamping chain does not need to be included).

**Revocation.** Certificates can be revoked, e.g., if the private key corresponding to the public key in the certificate is compromised, using certificate revocation lists (CRLs) [98] and the online certificate status protocol (OCSP) [99].

**Validation.** Authenticode validation is performed using the WinVerifyTrust function, which supports multiple validation policies. The policy we are interested in is the default one for Windows (WINTRUST_ACTION_GENERIC_VERIFY_V2). This policy is documented in the Authenticode specification [36] as follows:

- The signing chain must be built to a trusted root certificate (in the Windows Certificate Store) following RFC 5280 [98].
- The signing certificate must contain either the extended key usage (EKU) CODE_SIGNING value, or the entire certificate chain must contain no EKUs.
- The certificates in the signing chain must not be in the untrusted certificates store[1].
- Each certificate in the signing chain must be within its validity period, or the signature must be timestamped.
- Revocation checking is optional, but often used.
- The timestamping chain validation differs in that the TSA certificate must include a TIMESTAMP_SIGNING EKU and revocation is turned off by default for this chain.

[1] In Windows XP and 2003 only the signing certificate is checked.

- By default, timestamping extends the lifetime of the signature indefinitely, as long as it happened during the validity period of the signing certificate and before the certificate revocation date (if applicable).
- Timestamped signatures can be prevented from verifying for an indefinite period of time by setting the LIFETIME_SIGNING OID in the code signing certificate or passing a particular flag to the WinVerifyTrust function.
- The Authenticode signature must verify.
- The Authentihash computed on the executable must equal the Authentihash value stored in the PKCS #7 structure.

A failure in any of these steps should cause validation to fail. Unfortunately, the Authenticode validation code is proprietary and thus it is not clear if it follows all steps in the validation, in which order those steps are executed, and how it handles cases where the specification is unclear. Its exact functionality can only be reverse-engineering through testing or code analysis. We discuss validation issues in Section 3.3.

**Code signing in Windows.** A signed executable can embed an Authenticode signature (Figure 3.2) or its hash can be included in a *catalog file*, i.e., a collection of file hashes digitally signed by their publisher [100]. Most non-Microsoft signed executables embed Authenticode signatures. By default, user-level signed applications are validated by Windows before they run if the application was downloaded from the network (including network shares) or requires administrator privileges, which triggers User Account Control (UAC). In addition, Internet Explorer validates the signature of downloaded files [101]. User interaction varies across situations and Windows versions, but generally if the Authenticode signature validates, the window presented to the user to confirm execution contains the verified publisher information and a warning icon. If it fails or is unsigned it states the publisher is untrusted and uses a more threatening icon and textual description. Since Windows 7, AppLocker allows specifying rules for which users or groups can run particular applications, which allows to create a rule for running only signed applications [102].

Device drivers are handled differently depending on the Windows version, whether 32-bit or 64-bit, and if the driver is user-mode or kernel-mode [103]. For 64-bit Windows since Vista, it's mandatory to have both user-mode and kernel-mode drivers signed in order to load. In addition, for kernel-mode code a special process is required where the publisher's code signing certificate must have a chain leading to the Microsoft Code Verification Root [104].

### 3.2.2 Authenticode Market

We analyzed the CAs that are members of the CA Security Council [105] and the CA/Browser [106] forum and that publicly sell Authenticode code signing certificates.

| | | Certificate Price | | Revocation | | |
|---|---|---|---|---|---|---|
| **CA** | **TSA** | **CS** | **HTTPS** | **Mal.** | **Abuse** | **Delay** |
| Certum | ✓ | $199 | $34 | - | - | ≤ 1d |
| Comodo | ✓ | $172 | $109 | ✓ | ✓ | ≤ 1d |
| DigiCert | ✓ | $223 | $175 | ✓ | ✓ | ≤ 1d |
| Disig | - | $109 | $51 | ✓ | ✓ | ≤ 1d |
| Entrust | ✓ | $299 | $199 | ✓ | - | ≤ 1d |
| GlobalSign | ✓ | $299 | $249 | - | - | ≤ 3h |
| GoDaddy/StarField | ✓ | $170 | $63 | - | - | ≤ 7d |
| StartCom/StartSSL | ✓ | $60 | $60 | - | - | ≤ 12h |
| SwissSign | ✓ | $449 | $399 | - | - | ≤ 1d |
| Symantec/GeoTrust | ✓ | $499 | $149 | - | - | ≤ 1d |
| Symantec/Thawte | - | $299 | $149 | - | - | ≤ 1d |
| Symantec/Verisign | ✓ | $499 | $399 | - | - | ≤ 1d |
| TrustWave | - | $329 | $119 | ✓ | - | ≤ 1d |
| TurkTrust | - | $138 | $112 | ✓ | - | ≤ 1d |
| Verizon | - | $349 | $349 | - | - | ≤ 12h |
| WoSign | ✓ | $466 | $949 | - | ✓ | ≤ 1d |
| yessign | - | $153 | - | - | - | ≤ 1d |

Table 3.1: CAs offering code signing certificates and timestamping. Prices are for 1-year certificates in US Dollars. Revocation shows if a malware clause is present in the CPS, an abuse contact is mentioned, and the delay to publish a revocation. A dash indicates that we were not able to find related information.

Table 3.1 summarizes if they offer timestamping services, their certificate prices, and their revocation policies.

Few CAs offer Authenticode code signing certificates compared to HTTPS certificates, possibly reflecting a smaller market. There has been significant consolidation, e.g., Symantec acquired Verisign, GeoTrust, Thwate, and smaller CAs. Unfortunately, we did not find any public market size and CA market share figures. Only 11 CAs publicly advertise themselves as TSAs. In all cases timestamping is offered through HTTP, free of charge, and does not require authenticating to the service. We evaluate these services in Section 3.5.6.

Code signing certificates are pricier than HTTPS certificates ranging $60–$499 for a 1-year validity period. The exception is StartSSL which charges per identity verification rather than per certificate. Code signing certificates can be bought with a 1, 2, or 3-year validity period, with the latter being offered by only 25% of the CAs.

**Revocation.** We examine the revocation sections of the Certification Practice Statement (CPS) documents of the CAs in Table 3.1. The only entity that can perform revocation is the same CA that issued the code signing certificate. For all CAs, the customer can request revocation of its own certificate and the delay to publish the revocation through CRL or OCSP ranges between 3 hours and a week (Delay column). Only 6 CAs have a specific clause on their CPS about revocation being possible if the code signing certificate is abused to sign malicious code (Mal. column), although there is typically another clause that reminds how revocation can happen if the certificate is used in a way "harmful for the image or the business" of the CA. We were able to find an abuse contact email address or Web form for only 4 CAs. In other cases third

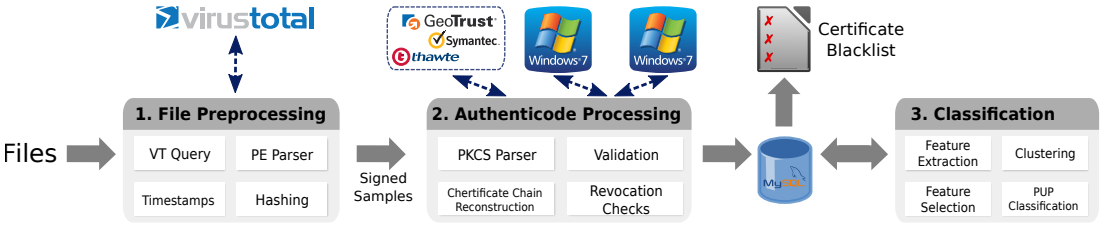



Figure 3.3: Approach overview.

parties reporting abuse would have to go through generic contact forms. Researchers that in the past requested CAs to revoke malicious certificates reported none or little response [63, 64]. Overall, third-party reporting of certificate abuse does not seem a concern by most CAs. In Section 3.5.5 we show that only 15% of the malicious certificates we observe are revoked, and the revocation practices outlined in this paragraph are likely a contributing factor for this low number.

## 3.3 Revoking Timestamped Code

When a code signing certificate is revoked, any executable signed (but not timestamped) with this certificate no longer validates in Windows, regardless if the executable was signed before or after revocation. But, our testing of Windows Authenticode validation reveals that if an executable is both properly signed and timestamped at time $t_{ts}$ (i.e., validates at $t_{ts}$) and then its code signing certificate is revoked at $t_{rev} > t_{ts}$, the executable still validates at any $t > t_{rev}$ despite the code signing certificate having been revoked. This is true as long as the revocation date is larger than the timestamping date ($t_{rev} > t_{ts}$). Executables timestamped after the revocation ($t_{rev} \leq t_{ts}$) will not validate.

Such handling seems to assume that revocation happens because the private key corresponding to the certificate's public key was compromised. In that case, executables signed before the key compromise should be OK and there is no need for them to fail validation, as long as a timestamp certifies they existed before revocation. We call this a *soft revocation* because it only invalidates executables signed with the certificate after revocation.

However, revocation is also needed when an attacker convinces a CA to issue him a code signing certificate, which it uses only to sign malicious code. In this case, if the attacker signs and timestamps a large number of malware before starting to distribute them, revocation happens after the timestamping date of those samples and thus they still validate after the revocation. What is needed in this case is a *hard revocation* that invalidates all executables signed with that certificate regardless when they were timestamped. With a hard revocation the CA sends the signal that it believes the certificate's owner is using it for malicious purposes and none of his executables should be trusted, rather than the owner was compromised and earlier signed executables are OK. The

CA is responsible for distinguishing these two cases.

The easiest way to perform a hard revocation is for the CA to set the revocation date to the certificate's issue date ($t_{rev} = t_i$), even if the CA discovers the improper use of the certificate at a later time. This way, any sample signed with that certificate will fail validation, regardless the timestamping date, because the revocation date will always be smaller than the timestamping date. Note that the attacker cannot forge an old timestamp and also that a valid timestamp needs to be within the certificate's validity period.

Setting the revocation date to the certificate's issue date enables hard revocations without modifying OCSP and CRLs. A caveat is that it hides the real date in which the CA realized the certificate was malicious. Adding this information may require modifications to revocation protocols and CAs may see a benefit on hiding how long it takes them to realize they issued a malicious certificate. We believe that it would be good to use the OCSP/CRL revocation reason to explicitly state that it is a hard revocation. In Section 3.5 we show that the information in this field is currently not useful.

One issue is that an attacker could revoke its own certificate after claiming a key compromise, in order for the CA to perform a soft revocation that does not invalidate previous code. One way to address this is to assign reputation to subjects based on prior revocations. In Section 6.7 we discuss that CAs should share revocation information.

We have reported to Microsoft this issue and the suggested solution using hard revocations.

## 3.4 Approach

Figure 3.3 summarizes our approach. It takes as input a large number of unlabeled files from malware datasets (described in Section 6.3). It preprocesses the files to discard benign files, parses the PE files to identify those signed, and processes the Authenticode signature (Section 3.4.1). All information is stored in a central database. Then, the clustering (Section 4.4) groups the samples into operations. Finally, the certificate blacklist is output. For each blacklisted certificate we provide information about the certificate (i.e., Subject CN, Issuer, Serial Number), a link to VirusTotal with a sample signed with this certificate, and the certificate itself exported in DER format that can be directly installed on Windows untrusted certificates store by following the Windows certificate installation wizard.

### 3.4.1 Sample Processing

Our infrastructure is implemented on Linux using 4,411 lines of Python and C code. Files are first preprocessed to remove non-PE files. Then, the PE files are parsed to extract a variety of information from the PE header including the file type (EXE, DLL, SYS), multiples hashes (MD5, SHA1, SHA256, PEHash [107]), publisher and product

information in the optional PE structures, icon, PDB path, and a number of timestamps. Next, it queries the file hash to VirusTotal (VT) [96], an online service that examines malware with a large number of security tools, to retrieve file metadata such as the number of AV engines that detect the file and the timestamp of the first time the file was submitted. We keep any sample flagged by more than 3 AV engines. Samples that contain an Authenticode signature move on to the next processing phase.

The Authenticode processing parses the PKCS #7 structure to retrieve the code signature, timestamp, and PKCS #9 timestamping counter-signature (if present). Then, it extracts the X.509 certificates from the `certificates` structure of the PKCS #7 structure, which contains certificates from both the signing and timestamping chains. The certificate chains need to be reconstructed because oftentimes the certificates are not properly ordered and certificates from both chains may be mixed. In addition, certificates can include a URL to the next certificate in the chains (if not included). If so, the certificate is downloaded. Next, the certificates are parsed to obtain a wealth of information including among others, the Subject, Issuer, validity period, PEM and DER hashes, Extended Key Usage flags, and OCSP and CRL URLs. The validation component verifies both chains using OpenSSL and queues the files to be distributed across four Windows VMs for Authenticode validation. We use the OpenSSL validation to better understand the error codes returned by Authenticode validation. Next, the revocation component retrieves and processes the CRL and OCSP information from each certificate in the chain. All information is stored in a central database.

### 3.4.2 Clustering

We cluster the signed samples into operations by grouping executables from the same publisher. For computing the sample similarity we focus on features that can be extracted statically from the samples, which enables efficient processing. Since all 142 K samples to be clustered are signed, most of our features focus on properties of the publisher's code signing certificate, with a focus on identifying different certificates from the same publisher. As far as we know certificate features have not been previously used for clustering malware. We also use a previously proposed static feature to identify polymorphic variants of the same code [107].

We first identify a large set of candidate features and perform feature selection on the signed samples of the publicly available Malicia malware dataset [108], for which the majority of files have family labels. We select the following 6 top boolean features based on information gain:

- **Leaf certificate.** Properly signed samples using the same CA-issued code signing certificate (same certificate hash) are distributed by the same publisher, i.e., the one owning the certificate. Publishers typically amortize the cost of a certificate by signing a large number of samples.

- **Leaf certificate public key.** Public keys are left unchanged in many certificate replacements [109]. Thus, two certificates authenticating the same public key

likely belong to the same publisher.

- **Authentihash.** Files with the same Authentihash contain the same code and data. Thus, they correspond to the same program even if they have a different file hash, e.g., due to different certificate chains.

- **Subject common name.** Publishers may try to obtain multiple certificates using the same identity by slightly modifying the company or individual name (e.g., "Company SLU" and "Company S.L."). Given two certificates with a non-empty Subject CN field, this feature computes the normalized edit distance between their Subject CNs. If the distance is less than 0.11 their publishers are considered the same. The threshold is chosen using a small subset of manually labeled certificates.

- **Subject location.** Publishers may reuse the same address in multiple certificates with small changes to fool the CA (e.g., "Rockscheld Blvd. 83 Dublin" and "Rockchilde 83 Dublin"). Given two certificates whose subject location contains a street attribute, this feature computes the normalized edit distance between those fields. If less than 0.27 the publisher is the same, otherwise different. The threshold is chosen using a small subset of manually labeled certificates. If the street attribute is not available, then the location only has the city and is not specific enough, thus they are considered different.

- **File metadata.** PE executables have an optional data structure with file metadata. This feature concatenates the following file metadata fields: publisher, description, internal name, original name, product name, copyright, and trademarks. Two files with the same concatenated metadata string larger than 14 characters are considered to be in the same family. Shorter metadata strings are not specific enough, thus they are considered different.

- **PEhash.** We also use the previously proposed PEhash [107], which transforms structural information about a PE executable into a hash value. If two files have an unknown packer and the same PEhash they are considered polymorphic variants of the same code.

**Clustering.** We use the following algorithm to cluster files into operations. The clustering starts with zero clusters and iterates on the list of samples. For each sample, it checks if it is similar to any other sample using the 6 features above. Two samples are similar if any of the above similarity features returns one. If the sample being examined is similar only to samples in the same cluster, it is added to that cluster. If similar to samples in multiple clusters, those clusters are merged and the sample is added to the merged cluster. If not similar to any other sample, a new singleton cluster is created for it.

| Samples | Families | Precision | Recall | F-Measure |
|---|---|---|---|---|
| 2,046 | 7 | 98.6% | 33.2% | 49.7% |

Table 3.2: Clustering accuracy on labeled (signed) malware from Malicia dataset.

**Clustering accuracy.** To evaluate the accuracy of our clustering we use the publicly available Malicia malware dataset [108], which contains labeled samples. In particular we use the 2,046 samples in the Malicia dataset that are both signed and have a label. Those samples belong to 7 families, but the majority (97%) are Zbot. Table 3.2 summarizes the results. The precision is high (98.6%) but the recall is low (33.2%). The reason for the low recall is that the Malicia labels capture samples with the same code. However, Zbot code can be bought or downloaded online, so it is used by many operations. Our clustering is oriented towards different operations so Zbot is broken into multiple clusters.

**Labeling.** Our clustering automatically generates a cluster label based on the most common feature value in the cluster. For the largest clusters we manually update the label with any popular tag used by security vendors for that operation.

### 3.4.3 PUP classification

We are interested in differentiating how malware and PUP abuse Authenticode, but are not aware of any prior techniques to automatically differentiate both classes. The main challenge is that the behaviors used to determine if a family is potentially unwanted or malicious may differ across security vendors [110, 111]. To address this issue we design two techniques that examine the AV detection labels obtained from VirusTotal, taking into account how multiple AV engines classify samples as PUP or not. One technique classifies a whole cluster as PUP or malware, while the other classifies each sample separately. We find the cluster classification to be more accurate, but it requires the clustering in Section 4.4, which is only available for signed samples and cannot be applied to unsigned samples as most features come from the certificates. We use the sample classification to compare the PUP prevalence among signed and unsigned samples.

Prior work has shown that AV labels are not a good ground truth for classifying malware into families due to inconsistent naming [112, 113]. However, our classification is at a coarser granularity. We only use the AV labels to determine if a cluster or a sample corresponds to PUP or not rather than to a specific family, which is captured by the malware clustering in Section 4.4.

As preparation for both classification techniques we first select 13 case-insensitive keywords that if present in a label indicate a potentially unwanted program: PUP, PUA, adware, grayware, riskware, not-a-virus, unwanted, unwnt, toolbar, adload, adknowledge, casino, and casonline. Then, we select the top 11 AV engines sorted by number of samples in all our datasets whose detection label includes at least one of the 13

| | | PE | | CS chain | | | | TS chain | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Dataset** | **Date** | **All** | **Malware+PUP** | **PE Signed** | **PUP** | **Chains** | **Leaf** | **PE Timestamped** | **PUP** | **Chains** | **Leaf** |
| CCSS | 05/2015 | 197 | 191 (97.0%) | 172 (90.0%) | 6.4% | 172 | 171 | 92 (53.5%) | 6.5% | 18 | 16 |
| VirusShare_149 | 02/2015 | 32,184 | 30,402 (94.5%) | 19,082 (62.8%) | 97.4% | 855 | 815 | 7,419 (38.8%) | 96.0% | 32 | 24 |
| VirusShare_148 | 02/2015 | 64,629 | 59,684 (92.3%) | 45,668 (76.5%) | 97.6% | 1,077 | 1,015 | 15,059 (32.9%) | 96.6% | 34 | 24 |
| VirusShare_138 | 08/2014 | 53,064 | 51,500 (97.0%) | 46,174 (89.7%) | 99.2% | 684 | 656 | 29,491 (63.9%) | 99.1% | 28 | 21 |
| NetCrypt | 08/2014 | 1,052 | 1,051 (99.9%) | 892 (84.9%) | 99.5% | 28 | 26 | 8 (0.9%) | 62.5% | 3 | 3 |
| VirusShare_99 | 09/2013 | 99,616 | 96,355 (96.7%) | 26,424 (27.4%) | 92.5% | 1,057 | 990 | 7,002 (26.5%) | 90.8% | 46 | 33 |
| Malicia | 05/2013 | 11,337 | 11,333 (99.9%) | 2,059 (18.2%) | 0% | 87 | 87 | 2 (0.10%) | 0% | 1 | 1 |
| VirusShare_0 | 06/2012 | 87,126 | 86,112 (98.8%) | 1,906 (2.2%) | 56.2% | 466 | 447 | 847 (44.43%) | 39.5% | 23 | 19 |
| Italian | 11/2008 | 7,726 | 5,175 (67.0%) | 136 (2.6%) | 78.0% | 42 | 42 | 112 (82.3%) | 91.0% | 7 | 6 |
| Total | | 356,931 | 341,803 (96%) | 142,513 (42%) | 95% | 3,186 | 2,969 | 60,032 (42%) | 96.1% | 76 | 49 |

Table 3.3: Datasets used.

above keywords. Those engines are: Malwarebytes, K7AntiVirus, Avast, AhnLab-V3, Kaspersky, K7GW, Ikarus, Fortinet, Antiy-AVL, Agnitum, and ESET-NOD32.

Using the selected keywords and AV engines, the classification module automatically classifies each cluster or sample as PUP or malware. Both classifications perform a majority voting on whether the selected engines consider the cluster or sample as PUP or not. We detail them next.

**Cluster classification.** The cluster classification first obtains for every engine the most common label the engine outputs on samples in the cluster (engines often output multiple labels for samples in the same cluster). Then, if the most common label for an engine contains at least one of the 13 keywords, the PUP counter is increased by one, otherwise the malware counter is increased by one. After evaluating all 11 engines on the cluster, if the PUP counter is larger or equal to the malware counter the cluster is considered PUP, otherwise malware.

**Sample classification.** The sample classification gets the label of the selected 11 engines for the sample. It can happen that some (and even all) of the selected engines do not detect the sample. If the label for an engine contains at least one of the 13 keywords, the PUP counter is increased by one, otherwise the malware counter is increased by one. After evaluating the labels, if the PUP counter is larger or equal to the malware counter the cluster is considered PUP, otherwise malware.

## 3.5 Evaluation

This section describes our datasets and the results of analyzing them through our infrastructure.

### 3.5.1 Datasets

Table 3.3 details the datasets used. The first two columns show the name of the dataset and its release date. Our main source of samples is VirusShare [114] from where we download 5 datasets between 2012 and February 2015. We also obtain from Italian collaborators a dataset of unlabeled older samples and collect a small dataset of samples

with encrypted network traffic. We include the publicly available Malicia dataset [108], which contains labeled samples that we use to evaluate our clustering. The last dataset contains samples downloaded from the CCSS Forum certificate blacklist [95] used for measuring our coverage.

The next two columns summarizes the executables in the dataset. First, it shows the number of PE executables in the dataset, after excluding other malicious files (e.g., HTML). Overall, our infrastructure processed 356,931 executables. Then, it shows the number and percentage of malicious and potentially unwanted executables in the dataset. An executable is malicious or unwanted if more than 3 AV engines flag it in VirusTotal [96]. As expected, the vast majority (96%) satisfy this condition.

The next group of 4 columns (*CS chain*) summarizes the signed executables and their code signing chains. It shows the number of signed malicious executables, the fraction of signed samples classified as PUP using the cluster classification, the number of unique certificate chains in those executables, and the number of distinct leaf certificates in those chains. Overall, 142,513 samples (42%) are signed of which 95% are PUP and 5% malware. Those signed samples contain 3,186 distinct chains. On average, 45 signed samples share the same certificate chain, which is an indication that they belong to the same operation. We detail the clustering into operations and PUP classification in Section 3.5.2. Those 3,186 chains contain 2,969 unique leaf (i.e., code signing) certificates.

The last group of 4 columns (*TS chain*) summarizes the timestamped executables, and their timestamping chains. It shows the number and percentage of timestamped malware over all signed malware, the fraction of those samples classified as PUP, the number of unique timestamping certificate chains in those executables, and the number of distinct leaf certificates in those timestamping chains. Overall, 42% of the signed samples are also timestamped. Those files contain only 76 distinct chains, with 49 unique leaf certificates. On average, 790 samples share the same timestamping chain, a significantly larger reuse compared to code signing chains indicating that TSA infrastructure is quite stable. Oftentimes, executables are signed by one CA and timestamped by a different CA.

**Dataset collection.** We know that the Malicia dataset was collected from drive-by downloads, which silently install malware on victim computers. Silent installs are characteristic of malware while PUP tends to be distributed through bundles or installers. Thus, the Malicia dataset is biased towards malware. In fact, given the available labels we know that it contains no PUP (neither signed or unsigned). Unfortunately, we do not know how datasets other than Malicia were collected, a common situation with third-party datasets. In particular we do not know whether the VirusShare datasets, which are the largest and dominate our corpus, may have some bias towards PUP or malware due to their collection methods. We leave as future work replicating the analysis in other datasets to compensate for any possible collection bias in VirusShare.

| Samples | Clusters | Singletons | Largest | Mean | Median |
|---|---|---|---|---|---|
| 142,513 | 2,288 | 1,432 | 42,711 | 62.3 | 1 |

Table 3.4: Clustering results on signed samples.

### 3.5.2 Clustering and PUP Classification

Table 4.4 summarizes the clustering results on the 142,513 signed samples. The clustering outputs 2,288 clusters of which 1,432 contain only one sample. The distribution is skewed with the largest cluster containing 42,711 samples, the average cluster 62.3, but the median only one due to the large number of singletons. We detail the top operations in Section 3.5.7. To evaluate the clustering accuracy we manually analyze the 235 clusters with more than 10 samples, which cover 97% of signed samples. We observe high precision but lower recall, i.e., some operations are split into multiple clusters typically one large cluster and one or two small clusters. This is consistent with the ground truth evaluation in Table 3.2.

**PUP cluster classification.** Our PUP cluster classification applied on the 2,288 clusters of signed samples outputs that 721 clusters are PUP and 1,567 malware. While a majority of clusters are labeled as malware, the largest clusters are labeled as PUP and overall the cluster classification considers 95% of the samples as PUP and 5% as malware. The median PUP cluster size is 188 samples and for malware clusters 4.4 samples. Over the top 235 clusters manually examined, we find 10 where our manual PUP classification differs from the automatic classification. The largest of these 10 clusters has 351 samples and altogether they comprise 890 potentially misclassified samples, 0.64% of all manually labeled samples.

**PUP sample classification.** The PUP sample classification is applied to 341,119 signed and unsigned samples, for which we have a VT report and they are detected by at least one of the selected 11 AV engines. Of those, 44% are labeled PUP and 56% malware. This indicates that our corpus is quite balanced on malware and PUP samples. For signed samples, 88% are labeled PUP and 12% malware. For unsigned samples, the results are almost opposite: 11% are labeled PUP and 89% malware. These numbers indicate that PUP is most often signed, but malware only rarely, an important conclusion of our work.

Table 3.5 summarizes the PUP classification. As expected, the cluster classification labels as PUP more signed samples (95% versus 88%) since it considers as PUP samples that may not be individually labeled as PUP, but belong to a PUP dominated cluster. Our manual analysis of samples with differing classification observes a higher accuracy for the cluster classification. When not mentioned explicitly throughout the evaluation, the PUP classification results are those of the cluster classification.

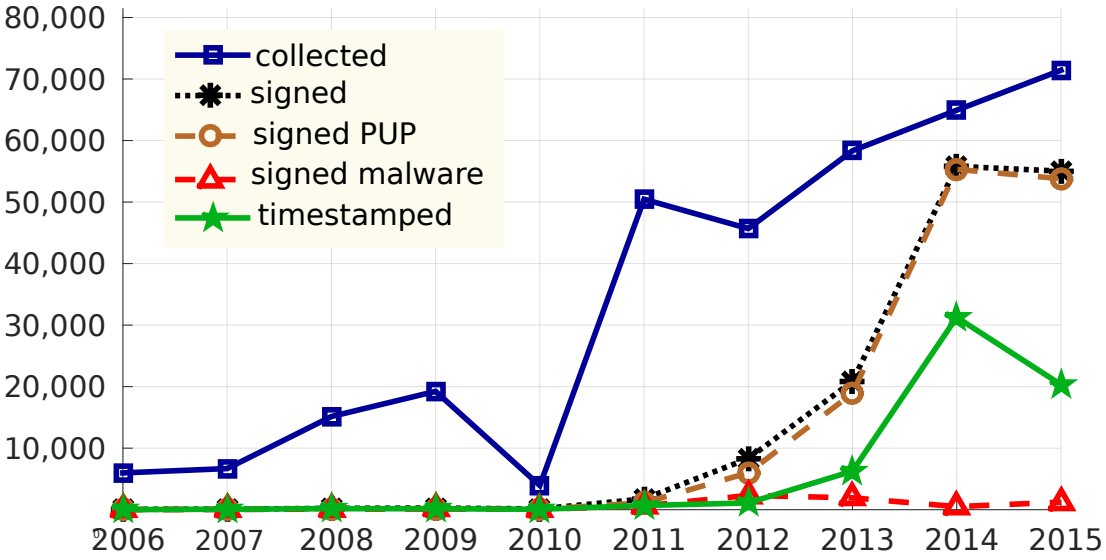



Figure 3.4: Number of collected, signed, timestamped, signed PUP, and signed malware samples over time. The cluster classification is used to label signed PUP and malware samples.

| | Signed | | Unsigned | | All samples | |
|---|---|---|---|---|---|---|
| **Classification** | **PUP** | **Mal.** | **PUP** | **Mal.** | **PUP** | **Mal.** |
| Per Sample | 88% | 12% | 11% | 89% | 44% | 56% |
| Per Cluster | 95% | 5% | - | - | - | - |

Table 3.5: Summary of PUP classification results.

### 3.5.3 Evolution over Time

We analyze if malware and PUP are increasingly being signed. To examine the evolution over time, we need to approximate when samples were created. The majority of dates embedded in executables (e.g., compilation time) are unauthenticated and can be forged. The timestamping date is authenticated, but only available in 42% of the signed samples. Thus, we approximate the creation date of each sample by the first submission to VirusTotal.

Figure 3.4 plots for each year between 2006 and 2015 the number of collected samples (signed and unsigned) in our corpus first seen by VT on that year, as well as the number of signed samples, signed PUP, signed malware, and timestamped samples (both malware and PUP). PUP and malware signed samples are labeled using the cluster classification. The figure shows that signed samples were rare in our corpus until 2011. Since then, they have steadily risen with the majority (87%) of all samples collected in 2014 being signed. This growth is due to the increase of signed PUP, as the number of signed malware has kept steadily low over time.

The figure also shows the increase of timestamped samples over time, which starts in 2012 and rises more slowly, achieving 49% of all collected samples being times-

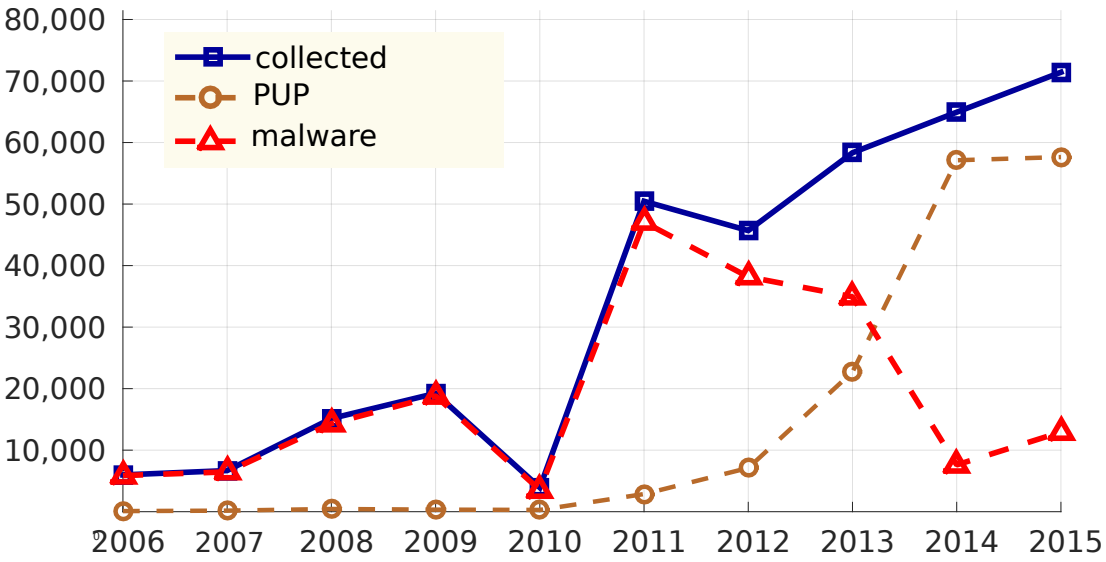



Figure 3.5: Number of collected, PUP, and malware samples over time including both signed and unsigned samples. The sample classification is used to label PUP and malware samples.

tamped in 2014. Note that the dip in 2010 is due to our corpus, not to less malware and PUP having been produced that year. In general, malware and PUP have been growing steadily over the years [115]. The dips in 2015 happen because only January and February are included.

Figure 3.5 is similar but it includes both signed and unsigned samples, labeled using the sample classification. It shows that in our corpus PUP has been increasing over time and the increase in PUP highly resembles the signed PUP increase in Figure 3.4, despite using different PUP classification metrics. In contrast, malware has been decreasing in our corpus since 2011. This could indicate that PUP is replacing malware over time, but could also be due to collection bias on the VirusShare datasets. We leave as future work examining this trend on other datasets.

### 3.5.4 Authenticode Validation

All signed samples are validated using the default Windows Authenticode policy. Table 3.6 summarizes the validation results. The majority (67%) of signed samples still validates correctly in Windows. The remaining 33% fail Windows Authenticode validation. The most common validation error is that a certificate has been revoked (*CERT_E_REVOKED*) returned for 16.5% of the signed samples. The second most common validation error is that a certificate in the chain has been expired (*CERT_E_EXPIRED*), which affects 13.3% of signed samples.

Note that revoked and expired code signing certificates were valid when they were issued. Thus, the total number of signed samples that used a CA-issued certificate is 97%. And, 73% of leaf certificates used to sign the samples have been issued by CAs.

| Validation Result | Signed Files | PUP | Mal. |
|---|---|---|---|
| OK | 95,277 (66.8%) | 69.2% | 21.7% |
| CERT_E_REVOKED | 23,550 (16.5%) | 16.9% | 9.7% |
| CERT_E_EXPIRED | 19,016 (13.3%) | 13.7% | 5.4% |
| TRUST_E_BAD_DIGEST | 2,798 (2.0%) | <0.1% | 38.3% |
| CERT_E_UNTRUSTEDROOT | 1,136 (0.8%) | <0.1% | 15.9% |
| TRUST_E_NOSIGNATURE | 503 (0.3%) | <0.1% | 7.0% |
| CERT_E_CHAINING | 170 (0.1%) | <0.1% | 1.0% |
| CERT_E_UNTRUSTEDTESTROOT | 47 (<0.1%) | <0.1% | 0.6% |
| TRUST_E_COUNTER_SIGNER | 8 (<0.1%) | 0% | 0.1% |
| TRUST_E_NO_SIGNER_CERT | 7 (<0.1%) | 0% | 0.1% |
| CERT_E_WRONG_USAGE | 1 (<0.1%) | 0% | <0.1% |
| Total | 142,513 (100%) | 100% | 100% |

Table 3.6: Validation results using the default Windows policy.

| | Issued | | | Revoked | | | Hard Revocations | | |
|---|---|---|---|---|---|---|---|---|---|
| **CA** | **Total** | **PUP** | **Malware** | **Total** | **PUP** | **Malware** | **Total** | **PUP** | **Malware** |
| Symantec/VeriSign | 708 | 70.5% | 29.5% | 76 (10.7%) | 7.2% | 19.1% | 23 (30.2%) | 44.4% | 17.5% |
| Symantec/Thawte | 510 | 66.0% | 34.0% | 109 (21.4%) | 24.6% | 15.0% | 4 (3.7%) | 2.4% | 7.7% |
| Comodo | 406 | 85.0% | 15.0% | 60 (14.8%) | 15.4% | 11.5% | 54 (90.0%) | 88.7% | 100% |
| GlobalSign | 153 | 80.0% | 20.0% | 14 (9.1%) | 9.8% | 6.4% | 0 | 0% | 0% |
| WoSign | 120 | 35.8% | 64.2% | 10 (8.3%) | 7.0% | 9.0% | 7 (70%) | 66.6% | 71.4% |
| GoDaddy/StarField | 99 | 85.0% | 15.0% | 28 (28.3%) | 31.0% | 13.3% | 6 (21.4%) | 23.0% | 0% |
| DigiCert | 85 | 68.2% | 31.8% | 37 (43.5%) | 36.2% | 59.2% | 9 (24.3%) | 14.3% | 37.5% |
| Certum | 32 | 65.6% | 34.4% | 7 (21.9%) | 14.3% | 36.4% | 0 | 0% | 0% |
| Symantec | 23 | 87.0% | 13.0% | 0 | 0% | 0% | 0 | 0% | 0% |
| StartCom/StartSSL | 10 | 60.0% | 40.0% | 2 (20%) | 16.6% | 25% | 0 | 0% | 0% |
| Total | 2,170 | 71.0% | 29.0% | 343 (15.8%) | 15.4% | 16.7% | 103 (30.0%) | 32.0% | 25.7% |

Table 3.7: Leaf certificates issued and revoked by CAs and used to sign PUP and malware.

The other are self-signed or bogus.

The two rightmost columns in Table 3.6 show the percentage of Authenticode validation results for PUP and malware respectively. Only 22% of signed malware still validates, compared to 69% of PUP. When including revoked and expired certificates we observe that 99.8% of PUP samples had at some point a valid signature, compared to 37% of malware. Thus, PUP authors have no trouble obtaining valid certificates from CAs. For malware authors, identity checks by CAs seems to present a higher barrier. Still, over one third of the signed malware had at some point a valid signature. The fact that less malware samples are revoked compared to PUP samples is due to PUP authors reusing certificates to sign a larger number of samples than malware authors. Certificate revocations are similar for both classes and detailed in Section 3.5.5.

The vast majority of other validation errors are due to malware. A significant (2.0%) fraction of samples have digital signatures that cannot be verified (*TRUST_E_BAD_DIGEST*) because the Authentihash in the PKCS7 structure does not match the file's Authentihash. This is the most common Authenticode validation result for malware samples and is often due to malware authors copying certificate chains from benign executables onto their malware. For example, the most common Subject CN of these leaf certificates is for Microsoft Corporation. Copying a benign certificate chain

on a malware sample changes the sample's file hash and also invalidates byte signatures on the certificates themselves, without changing the malware code. This may help to bypass some AV engines and explain why we observe multiple malware samples with the same Authentihash, but different certificate chains.

Another popular validation error among malware is an untrusted root certificate (*CERT_E_UNTRUSTEDROOT*) not included in the default Windows trust store. The majority of these (1,102/1,136) contain chains with only one self-signed certificate. Another 34 contain fake certificates for valid CAs and the rest are bogus.

There are 503 samples (491 malware) that Windows does not consider signed (*TRUST_E_NOSIGNATURE*). These contain misplaced Authenticode signatures, which Windows does not identify but our parsing code does. Another 170 (73 malware) samples contain chains where the certificates are not in the proper order (*CERT_E_CHAINING*), a phenomenon also observed in SSL certificate chains [116].

There are 47 samples whose chains end with a root certificate created by Microsoft's Certificate Creation Tool[2], used by developers to test code under development (*CERT_E_UNTRUSTEDTESTROOT*). Eight samples contain an invalid timestamping chain (*TRUST_E_COUNTER_SIGNER*). For seven samples Windows is not able to find the leaf certificate (*TRUST_E_NO_SIGNER_CERT*). The final sample contains a leaf certificate without the code signing flag (*CERT_E_WRONG_USAGE*).

**Malware and AV detection.** Malware can use Authenticode signatures (either valid or invalid) to apply polymorphism on their samples and evade AV detection. We examine three possible methods of Authenticode polymorphism: by performing small modifications on the Authenticode signature that do not make it invalid, by replacing the Authenticode signature with those of benign applications, or by completely removing the Authenticode signature from a signed executable. When we replace the Authenticode signature with that of another executable the Authenticode validation of the new sample returns the TRUST_E_BAD_DIGEST error. In all methods the original code of the samples remains intact and all changes are performed only on the Authenticode signature.

In our experiments we use two datasets. The first one contains 6 properly signed samples and the second 11 signed samples with invalid Authenticode signatures. The Authenticode validation of the samples in the first dataset returns OK and for the second dataset returns various error codes. All samples are detected by 20 to 43 AVs. Our approach is the same for all experiments and comprises three steps. First, we get the samples and scan them using VirusTotal against 56 AVs. Second, we perform the above methods of polymorphism on the samples and create new samples with slightly different characteristics than the originals and thus with different file hashes. Third, we submit the new samples to VirusTotal to be scanned and examine how the modifications we performed affect the detection rate.

Our results shows that all 3 methods can be used to successfully evade the detection

[2] https://msdn.microsoft.com/en-us/library/bfsktky3.aspx

of AVs. In addition, the results are similar either the original samples were properly signed or not. In total, 48 out of 56 AVs failed to detect at least one sample that was using one of these polymorphism methods. These AVs are detecting the original sample but they fail to detect the new one after the modifications on the Authenticode signature. The most effective method of polymorphism is to replace the original Authenticode signature with one from a benign application. Each sample manages to evade on average 7 AVs using this technique. These techniques are effective because some families (these may vary from one AV to another) are detected by AVs based only on their file hashes or strings contained in the Authenticode signature (e.g., fields of the code signing certificate chain). Replacing the Authenticode signature with a new one defeats both detection methods. AVs does not appear to differentiate between valid and invalid Authenticode signatures.

### 3.5.5 Revocation

In this section we examine the revocation of certificates used to sign PUP and malware. For this, we use OCSP and CRL revocation checks that our infrastructure performs for each certificate using OpenSSL. We do not use the *CERT_E_REVOKED* Authenticode validation error because it does not specify which certificate in the chain was revoked and because other errors may hide the revocation [117]. Of the 2,969 leaf certificates, 83% contain a CRL URL, 78% both CRL and OCSP URLs, and 17% neither[3]. Revocation checks are successful for 90% of the certificates with a CRL or OCSP URL, the remaining 10% fail. The most common errors are OCSP unauthorized (i.e., CA does not recognize the certificate typically because it is fake) and an empty CRL list.

Table 3.7 summarizes the code signing certificates issued by each CA and used to sign PUP or malware in our corpus, and their revocations. For each CA it shows the number of valid certificates issued (including those that still validate, have been revoked, and have expired but were valid otherwise), the number of certificates revoked, and the number of hard revocations performed by the CA. It also provides the split of those categories into certificates that sign PUP and malware respectively.

Overall, 2,170 out of 2,969 leaf certificates were issued by CAs, the rest are self-signed or bogus. Symantec's Verisign and Thawte brands issue most code signing certificates used to sign PUP and malware. This may be due to Symantec having the largest market share of the code signing market. Unfortunately, we did not find any public code signing CA market share figures to compare with. Of those 2,170 certificates, 71% are used to sign PUP and 29% malware. All CAs issue more certificates to PUP authors except WoSign, a Chinese CA. These results indicate that obtaining a CA-issued code signing certificate may be easier for PUP authors, but malware authors still often manage to obtain one.

All revocations are for leaf certificates. Overall, 343 code signing certificates have been revoked. Thus, CAs revoke less than 16% of the certificates they issue to PUP and

[3]One leaf certificate contains only OCSP URL.

malware authors. The PUP and malware percentages are computed over the number of certificates issued to PUP and malware authors, respectively. There is no significant difference in the percentage of PUP certificates that gets revoked (15.4%) compared to malware certificates (16.7%). Five CAs revoke a higher percentage of malware certificates and 4 a higher percentage of PUP certificates. Both results indicate that CAs revoke similarly certificates used by PUP and malware.

Thawte is the CA with most revoked certificates and DigiCert the CA revoking the largest fraction of malicious certificates it issued. No CA revokes more than 43% of their abused certificates. These numbers indicate that revocation is currently not an effective defense against abused code signing certificates. We further discuss this at the end of this subsection.

The average time to revoke a certificate is 133 days. Comodo is the fastest to revoke malicious certificates (21 days) although it only revokes 15% of them. Verisign is significantly slower (validity ¿ 9 months) than the other CAs to revoke malware-used code signing certificates and only revokes 11%.

All revocations are available through OCSP and only a handful through CRLs. The reason may be that expired certificates are removed from CRLs to prevent them from growing too large, a behavior allowed by RFC 2459 [118]. We find some revocations for GoDaddy/Starfield that appear in CRLs but not through OCSP. This inconsistency indicates the need to check both revocation methods for this provider.

The vast majority (96.2%) of revocations happen during a certificate's validity period. We only observe 13 certificates revoked after they have expired. A revocation after expiration has no effect in Windows validation.

**Revocation reason.** A revocation may optionally include a revocation reason [98, 99]. Table 3.8 details the revocation reasons returned by OCSP or in the CRL. The reason is unspecified or not provided at all in 47% of revocations. The most common revocation reason is key compromise used in 40% of revocations by two CAs: Thawte and VeriSign. The key compromise reason is used not only in cases where the certificate's owner may have reported a key compromise but also when the CAs were likely deceived to issue a certificate to a malicious publisher. For example, 30% of these certificates were issued to malware publishers, which are unlikely to report a key compromise. It seems that CAs do not care about giving precise revocation reasons and this field is currently not useful.

**Hard revocations.** We observe some CAs (WoSign, Comodo, VeriSign, GoDaddy, DigiCert, Thawte) performing some revocations on the certificate issue date. This could indicate that they are already performing hard revocations or that they want to hide when they discovered the certificate's abuse. We have not found any prior references on the need or use of hard revocations. Comodo (90%) and WoSign (70%) have the highest fraction of such revocations. Unfortunately, they never provide a revocation reason. Our analysis of these revocations reveals that they are not performed system-

| | Leaf Certificates | | | |
|---|---|---|---|---|
| **Reason** | **All** | **PUP** | **Malware** | **# CA** |
| Unspecified / NULL | 163 (47%) | 67.5% | 32.5% | 7 |
| Key Compromise | 137 (40%) | 69.3% | 30.7% | 2 |
| Cessation of Operation | 35 (10%) | 80.0% | 20.0% | 3 |
| Superseded | 6 (2%) | 50.0% | 50.0% | 2 |
| Affiliation Changed | 2 (<1%) | 100% | 0% | 2 |

Table 3.8: Summary of revocation reasons.

| **CA** | **Samples** | **Chains** |
|---|---|---|
| Symantec/VeriSign[4] | 43,295 (72%) | 12 |
| GlobalSign | 13,536 (22%) | 5 |
| Comodo | 1,878 (3%) | 8 |
| DigiCert | 630 (<1%) | 7 |
| GoDaddy/Starfield | 316 (<1%) | 3 |
| WoSign | 174 (<1%) | 7 |
| Entrust | 42 (<1%) | 5 |
| Microsoft | 126 (<1%) | 21 |
| Certum | 20 (<1%) | 2 |
| Yessign | 3 (<1%) | 2 |
| Daemon Tools | 2 (<1%) | 2 |
| GeoTrust | 2 (<1%) | 1 |

Table 3.9: Timestamping authorities used by malware and PUP: number of samples and timestamping chains for each TSA.

atically. For example, WoSign revokes two certificates from the same operation, with the same Subject CN and one gets a revocation on the issue date and the other does not.

**Summary of findings.** Our revocation analysis shows that less than 16% of CA-issued code signing certificates used by malware are revoked with no significant difference between certificates used by malware (17% revoked) and PUP (15%). The lack of revocation is widespread across CAs: no CA revokes over 43% of the abused code signing certificates it issued. In addition, CAs do not properly detail the reason for which a certificate was revoked, which makes it difficult to separate key compromises from certificates purposefully obtained to sign malware and PUP. Some CAs perform revocations on the issue date. They may have realized the need of hard revocations. But, we have not seen any references to this issue, most CAs seem unaware, and the ones performing them show inconsistencies in their use. These findings support that revocation of malicious code signing certificates is currently ineffective.

[4]Includes TC TrustCenter GmbH, acquired by Symantec

| | | Dates | | Samples | | Certificates | | | | | Certificate Subjects | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Name** | **Type** | **Certificates** | **Malware** | **Signed** | **TimeSt.** | **Issued** | **Revoked** | **Hard Rev.** | **Avg. Validity** | **CAs** | **CNs** | **Comp.** | **Ind.** | **CCs** | **Cost** |
| Firseria | PUP | 05/11 - 09/17 | 08/11 - 02/15 | 42,711 | 42,543 | 26 | 0 | 0 | 1.7 | 5 | 20 | 15 | 0 | 2 | $12,734 |
| SoftPulse | PUP | 02/14 - 01/16 | 07/14 - 02/15 | 21,083 | 1 | 43 | 4 | 0 | 1.0 | 6 | 20 | 13 | 0 | 2 | $15,959 |
| InstallRex | PUP | 03/11 - 07/16 | 10/11 - 02/15 | 12,574 | 0 | 51 | 21 | 20 | 1.1 | 3 | 45 | 2 | 43 | 4 | $10,394 |
| Tuguu | PUP | 05/12 - 06/15 | 01/13 - 02/15 | 7,891 | 3 | 34 | 22 | 10 | 1.0 | 6 | 15 | 7 | 0 | 4 | $8,771 |
| OutBrowse | PUP | 02/13 - 08/17 | 07/13 - 02/15 | 5,590 | 21 | 97 | 64 | 0 | 1.0 | 5 | 44 | 40 | 0 | 6 | $27,300 |
| LoadMoney | PUP | 12/11 - 03/16 | 08/12 - 02/15 | 5,285 | 38 | 14 | 9 | 8 | 1.2 | 2 | 13 | 12 | 0 | 1 | $3,554 |
| ClientConnect | PUP | 02/12 - 12/16 | 06/14 - 02/15 | 3,576 | 3,562 | 21 | 0 | 0 | 2.0 | 3 | 3 | 3 | 0 | 3 | $17,760 |
| InstallCore | PUP | 07/10 - 01/17 | 01/11 - 02/15 | 2,972 | 900 | 101 | 3 | 2 | 1.2 | 6 | 89 | 75 | 0 | 17 | $29,595 |
| Zango | PUP | 05/09 - 01/15 | 07/10 - 09/13 | 2,913 | 25 | 6 | 5 | 5 | 1.9 | 1 | 3 | 3 | 0 | 1 | $4,864 |
| Bundlore | PUP | 07/11 - 07/16 | 12/12 - 02/15 | 2,823 | 0 | 6 | 0 | 0 | 1.5 | 2 | 3 | 2 | 0 | 1 | $1,797 |

Table 3.10: Top 10 operations. The validity period is in years and the cost in US Dollars.

### 3.5.6 Timestamping

We have already shown (Table 3.3) that 42% of the signed samples in our corpus are timestamped and that timestamped samples are on the rise (Section 3.5.3). In this section we detail the usage of timestamping by PUP and malware. Table 3.9 shows the timestamp authorities (TSA) used by samples in our corpus. For each TSA, the table presents the number of samples that were timestamped by this TSA and the number of distinct timestamping certificates chains for the TSA.

The results show that Symantec/Verisign is the most popular TSA, used by 72% of the timestamped samples, followed by GlobalSign with 22%. Next, we show that TSAs do not perform checks on executables sent to be timestamped. Thus, the popularity of Symantec's and GlobalSign's TSAs among PUP and malware authors is not due to these providers performing less validation than other TSAs, but most likely due to a larger market share. Note that Microsoft and Daemon Tools are not publicly available TSAs, some authors copied the timestamping chains from other files into their executables. These samples do not validate.

**Lack of timestamping checks.** We perform an experiment to test whether TSAs perform any checks on executables they receive for timestamping. We select 22 signed samples from our corpus, two for each Authenticode validation result in Table 3.6. We use the Windows SignTool [119] to send those samples to the top 7 TSAs in Table 3.9. All 7 TSAs successfully timestamped 20 of the 22 samples. The only two samples that were not timestamped were those with Authenticode validation error TRUST_E_NO_SIGNATURE (Section 3.5.4). Those samples have their signatures in a wrong position. We also try timestamping an already timestamped file, which results in replacement of the old timestamp with a new one. In summary, we do not observe any restrictions imposed by TSAs on the executables to be timestamped, other than they should be signed. TSAs do not check that the executable's certificate chain validates and do not attempt to identify malicious or potentially unwanted software. Given that timestamping is a free service, TSAs may not have an incentive to invest in checks.

**Timestamped and revoked.** Timestamping is beneficial for authors since if a sample is timestamped before its code signing certificate is revoked, then Windows authenti-

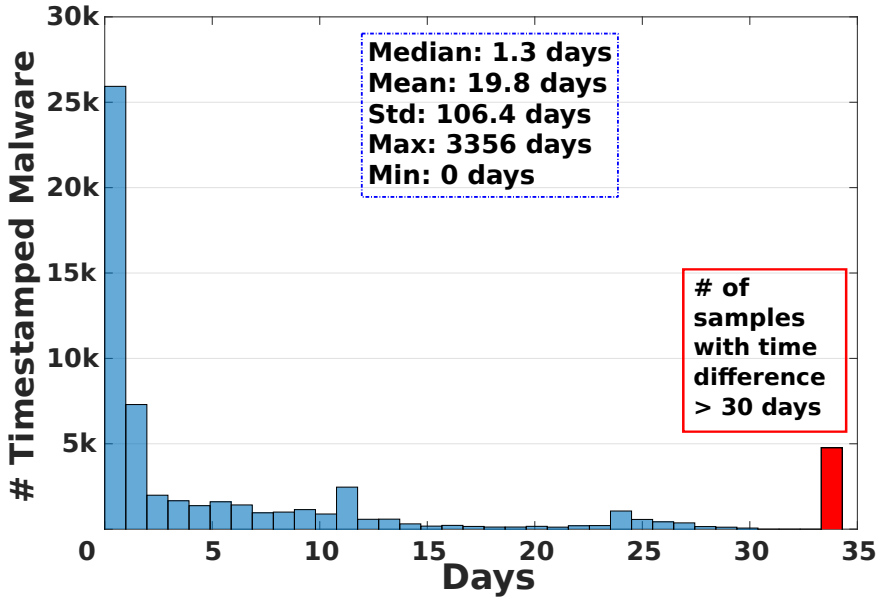



Figure 3.6: Time difference in days between a sample was timestamped and it was first observed in VirusTotal. There are 44 samples with a negative time difference of at most -10 minutes that are not shown in the figure.

cation will always succeed on that sample, regardless of the revocation. In our corpus we find 911 timestamped samples with a revoked certificate. A total of 118 revoked code signing certificates are used by these samples. The low number of samples in this category is due to less than 16% of abused code signing certificates being revoked. Of those samples, 655 (72%) are timestamped before their code signing certificate is revoked. These samples will continue to successfully validate after revocation. The remaining 28% are timestamped after revocation, up to 5.6 months after their code signing certificate was revoked. Thus, some authors keep using their code signing certificate long after it has been revoked. They still see value in signing their executables even when the signature does not validate, or did not realize that the revocation happened.

**Timestamping speed.** Next, we examine whether timestamping happens close to the creation time of a sample. For this we compare the timestamping date with the first time the timestamped sample was observed by VirusTotal (VT). As expected, the vast majority of samples are observed by VT after the timestamping date. Out of 60 K timestamped samples, only 44 are observed by VirusTotal before they are timestamped, and all those are seen by VT within 10 minutes of the timestamping date. These 44 samples are likely sent to VT to check if they are detected by AVs before timestamping them. This indicates that timestamping happens closely after a sample is signed and before it starts being distributed. Otherwise, we would expect

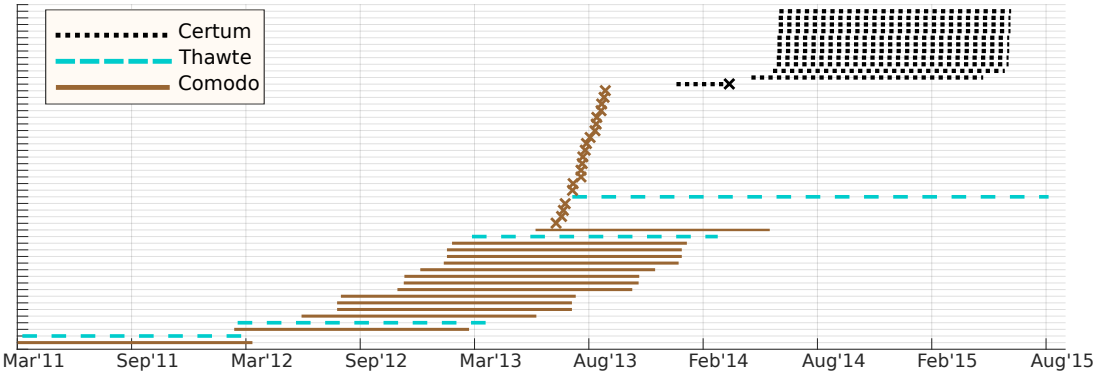



Figure 3.7: CA-issued certificates used by the InstallRex operation over time. Each line corresponds to a different certificate and its length marks the period between the certificate issuing date and its expiration or revocation (denoted by a cross) date. A single cross in one line indicates a hard revocation, i.e., a revocation on the certificate issuing date.

VT to see a larger number of samples distributed before timestamping and over a larger time frame. The consequence of this is that the timestamping date is a highly accurate estimation of the creation time. This is important because typically we have no reliable indication of when a sample is created. In practice, many works use the first-seen-on-the-wild date as an approximation.

We can use the timestamping date to evaluate how fast malware repositories collect samples, something that we are not aware has been measured earlier. Figure 3.6 shows the time difference in days between a sample was timestamped and it was first observed in VT. Overall, it takes VT a median of 1.3 days to observe a sample, but the distribution is long-tailed. The red bar on the right of the figure shows that 8% of the timestamped samples are seen by VT over a month after they are created. This happens more often with older samples created while VT did not have as good coverage as it does now. In the worst case, some samples are seen by VT more than 6 years after they were created. Thus, using the first-seen-on-the-wild date as an approximation of creation time for a sample works for the majority of recent samples, but can introduce large errors with a small percentage (<8%). Using the timestamping date is a more accurate estimation that does not rely on the distribution channel. While only 42% of our samples are timestamped, we have shown that timestamping is growing.

### 3.5.7 Largest Operations

In this section we use the clustering results to analyze the code signing infrastructure of the largest operations in our corpus. When sorting the clusters in Table 4.4 by number of signed samples they contain, the top 21 clusters correspond to PUP operations.

The first malware cluster at rank 22 corresponds to Zbot. However, aggregating all Zbot clusters would rank Zbot as 11th largest operation. When we sort clusters by the number of CA-issued leaf certificates the first malware cluster has rank 22 and uses 7 certificates.

Table 3.10 summarizes the top 10 operations, all PUP, in decreasing order of signed samples. The left half of the table shows, for each operation, the operation name, whether it corresponds to PUP or malware, the number of signed and timestamped samples, the number of certificates issued and revoked, and the average validity period in years of all certificates issued to the operation. The right half of the table details the subjects of the certificates issued to the operation, the number of CAs that issued those certificates, and the estimated certificate cost for the operation in US dollars.

**File polymorphism.** The 10 PUP operations in Table 3.10 distribute 75% of the signed samples in our corpus. The top operation (Firseria) distributes 30% of the signed samples alone, and the top 3 more than half. Thus, large PUP operations heavily use file polymorphism. For example, SoftPulse produces at least 21 K signed samples in 7 months, an average of 97 new signed samples per day. Such polymorphism is likely used to avoid AV detection and is a behavior often associated with malware.

Two of the top 10 families (Firseria and ClientConnect) timestamp the vast majority of their signed samples. Thus, some PUP authors have already realized the benefits of timestamping. The rest have no timestamped samples, or only a handful likely due to tests or third-party timestamping (like we did in Section 3.5.6).

**Certificates.** These 10 operations use from 6 code signing certificates (Zango, Bundlore) up to 84 certificates (OutBrowse). On average, they sign 445 samples with the same code signing certificate, amortizing the certificate cost over many samples. The average lifetime of their certificates ranges from one year for 3 operations to two years for 2 operations. Three of the operations favor 2-year certificates (validity larger than 1.5) and 6 favor 1-year certificates (validity less than 1.5). The longer the validity period the larger the investment loss if a certificate gets revoked.

**Certificate revocations.** Seven of the 10 families have multiple certificates revoked. It seems unreasonable that an entity would have 3–61 key compromises, so those revocations are likely due to malicious behavior. This indicates that CAs consider those 7 PUP operations malicious. For operations with revoked certificates, revocation does not work great since at most 66% of their certificates (OutBrowse) are revoked.

Interestingly, the two operations that timestamp their files do not have revocations and the 3 operations with zero revocations (Firseria, ClientConnect, and Bundlore) favor 2-year certificates. Their lack of revocations seems to give them enough confidence to commit to larger investments. Additionally, buying longer-lived certificates makes them look more benign, further contributing to the lack of revocations.

**Certificate polymorphism.** Eight of the 10 operations use over 10 code signing certificates and 9 buy certificates from multiple CAs. The right part of Table 3.10 examines who requested the code signing certificates (i.e., the certificate Subject field). First, it shows the number of distinct Subject CN fields in the certificates, then the grouping of those into unique companies or individuals that requested the certificates, and finally the number of countries for those subjects. These 10 operations use 399 certificates with 255 distinct Subject CNs. On average, 1.6 certificates have the same Subject CN. After grouping similar Subject CNs, (e.g., "Tuguu SL" and "Tuguu S.L.U.") those 399 certificates correspond to 172 corporations and 43 individuals. All individual certificates are used by the InstallRex operation. The other operations use corporations to buy the certificates. Five of the operations use more than 10 corporations. For some operations (e.g., Tuguu) we are able to check the company information on public business registers showing that the same person is behind multiple companies used by the operation. For each operation, the corporations and individuals are concentrated in a few countries, most often the United States and Israel.

These results show that PUP operations heavily rely on certificate polymorphism through the use of multiple CAs, small modifications of Subject CNs, and buying the certificates through multiple corporations or individuals. Such certificate polymorphism is likely used to bypass CA identity validation and revocation, increasing the resilience of their certificate infrastructure. For example, Comodo revokes a LoadMoney certificate issued for *LLC Monitor* but the family possess another one from Thawte issued for *Monitor LLC*, which due to the lack of CA synchronization is not revoked. Overall, operations have adapted to obtain new certificates when their current ones are revoked. We show an example for the InstallRex operation at the end of this subsection.

**Cost.** We estimate the cost of the certificate infrastructure for these operations by adding the cost of all certificates issued to the operation using the per CA and per validity period certificate costs in our market analysis. Certificate prices may have changed over the years and we may only have an incomplete view of the certificates used. Still, we believe this estimate provides a good relative ranking. The investment on code signing certificates by these operations varies from $1,797 (Bundlore) to $29,595 (InstallCore) with an average of $13,272.

**InstallRex.** Figure 3.7 shows the certificates of the InstallRex operation over time. Each line corresponds to a certificate's validity period. Crosses mark revocation dates. A single cross in a line indicates the CA performed a revocation on the issue date. InstallRex uses 51 certificates from 3 CAs. From March 2011 until April 2013 they bought 14 personal certificates from Comodo using different identities and one personal and another for a company ("Web Pick - Internet Holdings Ltd") from Thawte. Starting on June 2013 they acquired 22 personal certificates from Comodo and another from Thawte for the same company and different capitalization ("WEB PICK - IN-

TERNET HOLDINGS LTD"). This time Comodo realized and issued revocations on the expiration dates for all their certificates, but Thawte did not revoke theirs. A few months later they start acquiring personal certificates from Certum. The first one is revoked after some months, but a month later they succeed to buy 11 different certificates from Certum, which have not been revoked. This example illustrates how PUP operations exploit the lack of CA synchronization and multiple identities to survive revocations.

### 3.5.8 Blacklist Coverage

The certificate blacklist output by our infrastructure contains 2,170 CA-issued code signing certificates. In comparison, the CCSS blacklist [95] contained on May 2015 entries for 228 code signing certificates. Of those, only 197 provided a VirusTotal link to a malware sample signed with that certificate. We analyzed those 197 samples. Three of them were not considered malicious using our rule, another 19 are not really signed (according to both our infrastructure and VT), and 3 share certificate. Overall, our blacklist contains 9x more certificates. We further discuss blacklist coverage in Section 6.7.

## 3.6 Discussion

**Hard revocation deployment.** Hard revocations can be used without any changes to the Authenticode implementation in Windows. However, Microsoft could support its deployment by communicating to CAs both type of revocations and the recommended handling of key compromises and certificate abuse. One straightforward way to achieve this would be updating the 2008 Authenticode specification [36]. CAs can already use hard revocations, but it is important that they provide abuse email addresses to receive third-party notifications of abuse.

**PUP maliciousness.** PUP has been understudied in the research literature. Important questions such as the (lack of) behaviors that make a program PUP rather than malware remain open. This makes it possible for malware to disguise as PUP. We do not attempt to define what constitutes PUP, but rather rely on AV labels for that determination. However, our work is an important first step towards understanding PUP. We observe that PUP may be quickly growing and that it is typically signed. We also observe many PUP operations with suspicious behaviors such as high file and certificate polymorphism that could also be associated with malware.

**Identity checks.** CAs should implement checks to avoid identities to be reused with slight modifications. They should also provide correct revocation reasons to enable distinguishing revocations due to key compromise and abuse, which is important to build publisher reputation. Log-based PKI solutions [120] where CAs submit all their

issued certificates to a central repository would help identifying identity reuse across CAs.

**Blacklists.** Certificate blacklists would not be needed if revocation worked properly. However, they are an important stopgap solution and may help pushing CAs to improve revocation. We have shown that automatic approaches can build blacklists with an order of magnitude larger coverage than existing ones. To achieve larger coverage it is important that AV vendors and malware repositories contribute signed malware or their certificates to existing blacklists.

# 4 PUP Prevalence & Distribution in Consumer Hosts

## 4.1 Introduction

Potentially unwanted programs (PUP) are a category of undesirable software that includes adware and rogue software (i.e., rogueware). While not outright malicious (i.e., malware), PUP behaviors include intrusive advertising such as ad-injection, ad-replacement, pop-ups, and pop-unders; bundling programs users want with undesirable programs; tracking users' Internet surfing; and pushing the user to buy licenses for rogueware of dubious value, e.g., registry optimizers. Such undesirable behaviors prompt user complaints and have led security vendors to flag PUP in ways similar to malware.

There exist indications that PUP prominence has quickly increased over the last years. Already in Q2 2014, AV vendors started alerting of a substantial increase in collected PUP samples [17]. Recently, Thomas et al. [18] showed that ad-injectors, a popular type of PUP that injects advertisements into user's Web surfing, affects 5% of unique daily IP addresses accessing Google [18]. And, Kotzias et al. [19] measured PUP steadily increasing since 2010 in (so-called) malware feeds, to the point where nowadays PUP samples outnumber malware samples in those feeds. Still, the prevalence of PUP remains unknown.

A fundamental difference between malware and PUP is distribution. Malware distribution is dominated by *silent* installation vectors such as drive-by downloads [21, 22], where malware is dropped through vulnerability exploitation. Thus, the owner of the compromised host is unaware a malware installation happened. In contrast, PUP does not get installed silently because that would make it malware for most AV vendors. A property of PUP is that it is installed with the consent of the user, who (consciously or not) approves the PUP installation on its host.

In this work, we perform the first systematic study of PUP prevalence and its distribution through pay-per-install (PPI) services. PPI services (also called PPI networks) connect advertisers willing to buy installs of their programs with affiliate publishers

selling installs. The PPI services used for distributing PUP are disjoint from silent PPI services studied by prior work [56]. Silent PPI services are exclusively used for malware distribution, while the PPI services we study are majoritarily used for distributing PUP and benign software. In the analyzed PPI services, an affiliate publisher owns an original program (typically freeware) that users want to install. To monetize installations of its free program, the affiliate publisher bundles (or replaces) it with an installer from a PPI service, which it distributes to users looking for the original program. During the installation process of the original program, users are prompted with offers to also install other software, belonging to advertisers that pay the PPI service for successful installs of their advertised programs.

To measure PUP prevalence and its distribution through PPI services we use AV telemetry information comprising 8 billion events on 3.9 million hosts during a 19 month time period. This telemetry contains events where parent programs installed child programs and we focus on events where the publishers of either parent or child programs are PUP publishers. This data enables us to measure the prevalence of PUP on real hosts and to map the who-installs-who relationships between PUP publishers, providing us with a broad view of the PUP ecosystem.

We first measure PUP prevalence by measuring the installation base of PUP publishers. We find that programs from PUP publishers are installed in 54% of the 3.9M hosts examined. That is, more than half the examined hosts have PUP. We rank the top PUP publishers by installation base and compare them with benign publishers. The top two PUP publishers, both of them PPI services, are ranked 15 and 24 amongst all software publishers (benign or not). The top PUP publisher is more popular than NVIDIA, a leading graphics hardware manufacturer. The programs of those two top PUP publishers are installed in 1M and 533K hosts in our AV telemetry dataset, which we estimate to be two orders of magnitude higher when considering all Internet-connected hosts. We estimate that each top 20 PUP publisher is installed on 10M–100M hosts.

We analyze the who-installs-who relationships in the publisher graph to identify and rank top publishers playing specific roles in the ecosystem. This enables us to identify 24 PPI services distributing PUP in our analyzed time period. We also observe that the top PUP advertisers predominantly distribute browser add-ons involved in different types of advertising and by selling software licenses for rogueware. We measure PUP distribution finding that 65% of PUP downloads are performed by other PUP, that the 24 identified PPI services are responsible for over 25% of all PUP downloads, and that advertiser affiliate programs are responsible for an additional 19% PUP downloads.

We also examine the malware-PUP relationships, in particular how often malware downloads PUP and PUP downloads malware. We find 11K events (0.03%) where popular malware families install PUP for monetization and 5,586 events where PUP distributes malware. While there exist cases of PUP publishers installing malware, PUP–malware interactions are not prevalent. Overall, it seems that PUP distribution is largely disjoint from malware distribution. Finally, we analyze the top domains distributing PUP, finding that domains from PPI services dominate by number of down-

loads.

**Contributions:**

- We perform the first systematic study of PUP prevalence and its distribution through PPI services using AV telemetry comprising 8B events on 3.9M hosts over a 19-month period.
- We measure PUP prevalence on real hosts finding that 54% have PUP installed. We rank the top PUP publishers by installation base, finding that the top two PUP publishers rank 15 and 24 amongst all (benign and PUP) software publishers. We estimate that the top 20 PUP publishers are each installed on 10M-100M hosts.
- We build a publisher graph that captures the who-installs-who relationships between PUP publishers. Using the graph we identify 24 PPI services and measure that they distribute over 25% of the PUP.
- We examine other aspects of PUP distribution including downloads by advertiser affiliate programs, downloads of malware by PUP, downloads of PUP by malware, and the domains from where PUP is downloaded. We conclude that PUP distribution is largely disjoint from malware distribution.

## 4.2 Overview and Problem Statement

This section first introduces the PPI ecosystem (Section 4.2.1), then details the datasets used (Section 6.3), and finally describes our problem and approach (Section 6.4).

### 4.2.1 Pay-Per-Install Overview

The PPI market, as depicted in Figure 4.1, consists of three main actors: *advertisers*, *PPI services/networks*, and *affiliate publishers*. *Advertisers* are entities that want to install their programs onto a number of target hosts. They wish to *buy installs* of their programs. The PPI service receives money from advertisers for the service of installing their programs onto the target hosts. They are called advertisers because they are willing to pay to promote their programs, which are offered to a large number of users by the PPI service. Advertiser programs can be benign, potentially unwanted (PUP), and occasionally malware.

*Affiliate publishers* are entities that *sell installs* to PPI services. They are often software publishers that own programs (e.g., freeware) that users may want to install, and who offer the advertiser programs to those users installing their programs. This enables affiliate publishers to monetize their freeware, or to generate additional income on top of the normal business model of their programs. They can also be website owners that offer visitors to download an installer from the PPI service, thus selling

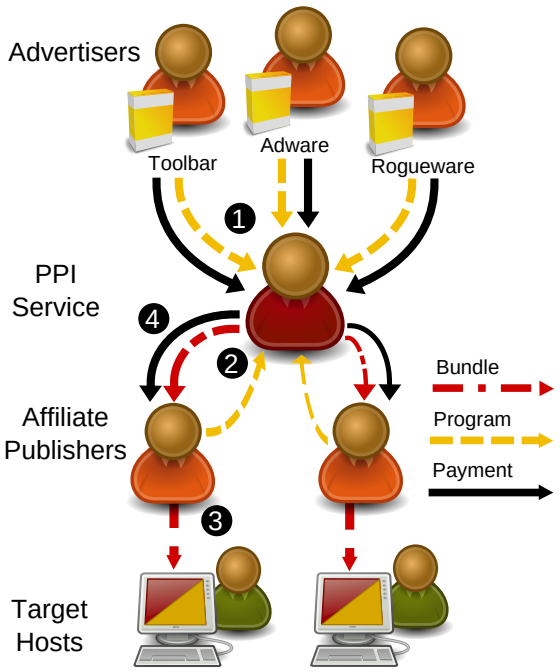



Figure 4.1: Typical transactions in the PPI market. (❶) Advertisers provide software they want to have installed, and pay a PPI service to distribute it. (❷) Affiliate publishers register with the PPI service, provide their program, and receive a bundle of their program with the PPI installer. (❸) Affiliate publishers distribute their bundle to target users. (❹) The PPI service pays affiliate publishers a bounty for any successful installations they facilitated.

installs on the visitor's machines. Affiliate publishers are often referred simply as publishers, but we use publishers to refer to software owners, and affiliate publishers for those signing up to PPI services.

The PPI service acts as a middle man that buys installs from affiliate publishers and sells installs to advertisers. The PPI service credits the affiliate publisher a *bounty* for each confirmed installation, i.e., affiliate displays an offer for an advertised program *and* the user approves and successfully installs the advertised program.

Affiliate publishers are paid between $2.00 and $0.01 per install depending on the geographic location. Prices vary over time based on offer and demand and the current price is typically only available to registered affiliate publishers. Table 4.1 shows the prices paid to affiliate publishers for the most demanded countries on June 25th, 2016 by 3 PPI services that publicly list their prices to attract affiliate publishers. The highest paid country is the United States with an average install price of $1.30, followed by the United Kingdom ($0.80), Australia and Canada ($0.40), and European countries starting at $0.30 for France. The cheapest installs are $0.03–$0.01 for Asian and African countries (typically part of a "Rest of the World" region). In comparison, prices paid to affiliate publishers by silent PPI services that distribute malware range $0.18-–$0.01 per install [56]. This shows that malware distribution can be an order of

| Country | Avg | Range |
|---|---|---|
| United States | $1.30 | $0.70-$2.00 |
| United Kingdom | $0.80 | $0.40-$1.50 |
| Australia | $0.40 | $0.30-$0.50 |
| Canada | $0.40 | $0.30-$0.50 |
| France | $0.28 | $0.15-$0.50 |
| Germany | $0.25 | $0.10-$0.40 |
| New Zealand | $0.23 | $0.15-$0.35 |
| Ireland | $0.19 | $0.15-$0.25 |
| Denmark | $0.18 | $0.15-$0.20 |
| Austria | $0.16 | $0.15-$0.20 |
| Netherlands | $0.16 | $0.10-$0.20 |
| Finland | $0.15 | $0.10-$0.20 |
| Norway | $0.15 | $0.05-$0.20 |
| Switzerland | $0.12 | $0.03-$0.20 |
| Spain | $0.11 | $0.03-$0.20 |

Table 4.1: Top 15 countries with the highest average price per install collected from 3 PPI services [1–3] on June 2016.

magnitude cheaper for the most demanded countries.

A common PPI model (depicted in Figure 4.1) is that the affiliate publisher provides the PPI service with its program executable and the PPI service *wraps* (i.e., bundles) the affiliate's program with some *PPI installer* software, and returns the bundle/wrapper to the affiliate publisher. The affiliate publisher is then in charge of distributing the bundle to users interested in the affiliate's program. The distribution can happen through different vectors such as websites that belong to the affiliate publisher or uploading the bundle to *download portals* such as Download.com [121] or Softonic [122]. When a user executes the wrapper, the wrapper installs the affiliate's program and during this installation it offers the user to install other advertised programs. If the user downloads and installs one of the offers, the PPI service pays a bounty to the affiliate's account.

An affiliate publisher can register with a PPI service even it if it does not own programs that users want to install. Some PPI services look for affiliate website owners whose goal is to convince visitors of their websites to download and run an installer from the PPI service. Furthermore, some PPI services offer a *pre-wrapped software* model where the PPI service wraps its own software titles with the advertiser offers, and provides the bundle to the affiliate publishers [123]. Some PPI services even allow affiliate publishers to monetize on third-party free programs (e.g., GNU).

Some download portals such as Download.com run their own PPI service. When publishers upload their programs to the portal (e.g., through Upload.com) they are offered if they want to monetize their programs. If so, the download portal wraps the original program and makes the bundle available for download. In this model the download portal is in charge of distribution.

Another distribution model are *affiliate programs* where an advertiser uses affiliate

| Dataset | Data | Count |
|---|---|---|
| WINE | Events Analyzed | 8 B |
| 01/2013 – 07/2014 | Events with Parent | 90 M |
| | Total number of Machines | 3.9 M |
| | All Files | 2.6 M |
| | Parent Files | 657 K |
| | Child Files | 2 M |
| | Signed Files | 982 K |
| | Publishers | 6 K |
| | Parent Publishers | 1.4 K |
| | Child Publishers | 6 K |
| | Events with URL | 1.1 M |
| | URLs | 290 K |
| | FQDNs | 13.4 K |
| | ESLDs | 7.5 K |
| Malsign | Signed executables | 142 K |
| VirusTotal | Reports | 12 M |
| | Feed Reports | 11 M |
| | WINE Reports | 1.1 M |
| | Malsign Reports | 142 K |

Table 4.2: Summary of datasets used.

publishers to distribute its software directly, without a PPI service. This is a one-to-many distribution model, in contrast with the many-to-many distribution model of PPI services.

### 4.2.2 Datasets

Our paper leverages several datasets to conduct a systematic investigation about PUP prevalence and distribution. We analyze WINE's binary downloads dataset [124] to trace PUP installations by real users and their parent/child (downloader/downloadee) relationships, the list of signed malicious executables from the Malsign project [19] to cluster together executables signed by different signers that belong to the same publisher, and VirusTotal [96] reports for enriching the previous datasets with additional file meta-data (e.g., AV detections, file certificates for samples not in Malsign). Table 5.1 summarizes these datasets.

**WINE.** The Worldwide Intelligence Network Environment (WINE) [125] provides researchers a platform to analyze data collected from Symantec customers that opt-in to the collection. This data consists of anonymous telemetry reports about security events (e.g., AV detections, file downloads) on millions of real computers in active use around the world.

In this work, we focus on the *binary downloads* dataset in WINE, which records meta-data about all executable files (e.g., EXE, DLL) and compressed archives (e.g., ZIP, RAR, CAB) downloaded by Windows hosts regardless if they are malicious or benign. Each *event* in the dataset can correspond to (1) a download of an executable file or compressed archive over the network, or (2) the extraction of a file from a compressed archive. For our work, we analyze the following fields: the server-side timestamp of the event, the unique identifier for the machine where the event happens, the SHA256 hash of the child file (i.e., downloaded or extracted), the SHA256 hash of the parent process (i.e., downloader program or decompressing tool), the certificate subject for the parent and child files if they are signed, and, when available, the URL from where the child file was downloaded. The files themselves are not included in the dataset.

We focus our analysis on the 19 months between January 1st 2013 and July 23rd 2014. As our goal is to analyze PUP (i.e., executables from PUP publishers), we only monitor the downloads of PUP and the files that are downloaded by PUP, i.e., events where either the child or the parent is PUP. This data corresponds to 8 billion events. The details of the data selection methodology are explained in Section 4.3. Out of 8 B events, 90 M events have information about the parent file that installed the PUP. Those events comprise 2.6 M distinct executables out of which 982 K (38%) are signed by 6 K publishers.

A subset of 1.1 M events provide information about the URL the child executable was downloaded from. These events contain 290 K unique URLs from 13.4 K fully qualified domain names (FQDNs). To aggregate the downloads initiated from the same domain owner, we extract the effective second-level domain (ESLD) from the FQDN. For example, the ESLD of `www.google.com` is the 2LD `google.com`, however, the ESLD of `www.amazon.co.uk` is the 3LD `amazon.co.uk` since different entities can request `co.uk` subdomains. We extract the ESLDs of the domains by consulting Mozilla's public suffix list [126].

**Malsign.** To cluster executables in the WINE binary downloads dataset signed by different entities that belong to the same publisher, we leverage a dataset of 142 K signed malware and PUP from the Malsign project [19]. This dataset includes the samples and their clustering into families. The clustering results are based on statically extracted features from the samples with a focus on features from the Windows Authenticode signature [36]. These features include: the leaf certificate hash, leaf certificate fields (i.e., public key, subject common name and location), the executable's hash in the signature (i.e., Authentihash), file metadata (i.e., publisher, description, internal name, original name, product name, copyright, and trademarks), and the PE-hash [107]. From the clustering results we extract the list of publisher names (subject common name in the certificates) in the same cluster, which should belong to the same publisher.

**VirusTotal.** VirusTotal [96] is an online service that analyzes files and URLs submitted by users. One of its services is to scan the submitted binaries with anti-virus products. VirusTotal also offers a web API to query meta-data on the collected files including the AV detection rate and information extracted statically from the files. We use VirusTotal to obtain additional meta-data about the WINE files, as well as from 11 M malicious/undesirable executables from a feed. In particular, we obtain: AV detection labels for the sample, first seen timestamp, detailed certificate information, and values of fields in the PE header. This information is not available otherwise as we do not have access to the WINE files that are not in Malsign, but we can query VirusTotal using the file hash. We consider that a file is malicious if at least 4 AV engines in the VT report had a detection label for it, a threshold also used in prior works to avoid false positives [19].

### 4.2.3 Problem Statement

In this paper we conduct a systematic analysis of PUP prevalence and its distribution through PPI services. We split our measurements in two main parts. First, we measure how prevalent PUP is. This includes what fraction of hosts have PUP installed, which are the top PUP publishers, and what is the installation base of PUP publishers in comparison with benign publishers. Then, we measure the PPI ecosystem including who are the top PPI services and PUP advertisers, what percentage of PUP installations are due to PPI services and advertiser affiliate programs, what are the relationships between PUP and malware, and what are the domains from where PUP is downloaded.

We do not attempt to differentiate what behaviors make a program PUP or malware, but instead rely on AV vendors for this. We leverage the prior finding that the majority of PUP (and very little malware) is properly signed. In particular, signed executables flagged by AV engines are predominantly PUP, while malware rarely obtains a valid code signing chain due to identity checks implemented by CAs [19]. Using that finding, we consider PUP any signed file flagged by at least 4 AV engines. Thus, the term PUP in this paper includes different types of files that AV vendors flag as PUP including undesirable advertiser programs, bundles of publisher programs with PPI installers, and stand-alone PPI installers.

To measure PUP prevalence, we first identify a list of dominant PUP publishers extracted from the code signing certificates from the 11M VT reports from the malware feed (Section 4.3). Then, we group publisher names (i.e., subject strings in code signing certificates) from the same entity into publisher clusters (Section 4.4). Finally, we use the WINE binary reputation data to measure the PUP installation base, as well as the installation base of individual PUP and benign publisher clusters (Section 4.5). Since we focus on signed executables, our numbers constitute a lower bound on PUP and publisher prevalence.

To measure the PPI ecosystem, we build a publisher graph that captures the who-installs-who relationships among PUP publishers. We use the graph for identifying PPI services and PUP advertisers (Section 4.6). Then, we measure the percentage of PUP

installations due to PPI services and advertiser affiliate programs (Section 4.7). Next, we analyze the downloads of malware by PUP and the downloads of PUP by malware (Section 4.8). Finally, we examine the domains from where PUP is downloaded (Section 4.9).

## 4.3 Identifying PUP Publishers

The first step in our approach is to identify a list of dominant PUP publishers. As mentioned earlier, prior work has shown that signed executables flagged by AV engines are predominantly PUP, while malware is rarely properly signed. Motivated by this finding, we identify PUP publishers by ranking publishers of signed binaries flagged by AV vendors, by the number of samples they sign.

For this, we obtain a list of 11M potentially malicious samples from a "malware" feed and query them in VirusTotal to collect their VT reports. From these reports, we keep only executables flagged by at least 4 AV vendors to make sure we do not include benign samples in our study. We further filter out executables with invalid signatures, i.e., whose publisher information cannot be trusted. These filtering steps leave us with 2.5M binaries whose signatures validated at the time of signing. These include executables whose certificate chain still validates, those with a revoked certificate, and those with expired certificates issued by a valid CA.

From each of the 2.5M signed executables left, we extract the publisher's subject common name from the certificate information in its VT report. Hereinafter, we will refer to the publisher's subject common name as publisher name. Oftentimes, publisher names have some variations despite belonging to the same entity. For example, MyRealSoftware could use both "MyRealSoftware S.R.U" and "MyRealSoftware Inc" in the publisher name. Thus, we perform a normalization on the publisher names to remove company suffixes such as Inc., Ltd. This process outputs a list of 1,440 normalized PUP publisher names. Table 4.3 shows the top 20 normalized PUP publisher names by number of samples signed in the feed. These 20 publishers own 56% of the remaining signed samples after filtering.

Clearly, our list does not cover all PUP publishers in the wild. This would not be possible unless we analyzed all existing signed PUP. However, the fact that we analyze 2.5M of undesirable/malicious signed samples gives us confidence that we cover the top PUP publishers. Those 1,440 PUP publisher names are used to scan the file publisher field in WINE's binary downloads dataset to identify events that involve samples from those PUP publishers, i.e., where a parent or child file belongs to the 1,440 PUP publishers. As shown in Table 5.1, there are 8 B such events.

Note that at this point we still do not know whether different publisher names (i.e., entries in Table 4.3) belong to the same PUP publisher. For example, some popular publisher names such as Daniel Hareuveni, Stepan Rybin, and Stanislav Kabin are all part of Web Pick Internet Holdings Ltd, which runs the InstalleRex PPI service. The process to cluster publisher names that belong to the same publisher is described in

| Rank | Publisher | Samples | |
|---|---|---|---|
| 1 | Popeler System | 326,530 | 13.2% |
| 2 | Daniel Hareuveni | 138,159 | 5.6% |
| 3 | Start Now | 117,930 | 4.8% |
| 4 | Mail.Ru | 117,920 | 4.8% |
| 5 | Softonic International | 69,233 | 2.8% |
| 6 | Bon Don Jov | 68,937 | 2.8% |
| 7 | Stepan Rybin | 68,390 | 2.8% |
| 8 | WeDownload | 66,332 | 2.7% |
| 9 | Payments Interactive | 41,128 | 1.7% |
| 10 | Tiki Taka | 37,072 | 1.5% |
| 11 | Stanislav Kabin | 36,893 | 1.5% |
| 12 | Safe Software | 36,602 | 1.5% |
| 13 | Vetaform Developments | 36,001 | 1.5% |
| 14 | Outbrowse | 35,832 | 1.4% |
| 15 | appbundler.com | 34,895 | 1.4% |
| 16 | Rodion Veresev | 34,696 | 1.4% |
| 17 | Mari Mara | 31,031 | 1.3% |
| 18 | Firseria | 29,940 | 1.2% |
| 19 | Give Away software | 26,541 | 1.1% |
| 20 | Jelbrus | 23,457 | 0.9% |

Table 4.3: Top 20 publishers in the feed of 11M samples by number of samples and percentage over all samples signed and flagged by at least 4 AV engines.

Section 4.4.

## 4.4 Clustering Publishers

PUP authors use certificate polymorphism to evade detection by certification authorities and AV vendors [19]. Two common ways to introduce certificate polymorphism are applying small variations to reuse the same identity / publisher name (e.g. apps market ltd, APPS Market Inc., Apps market Incorporated) and using multiple identities (i.e., companies or persons) to obtain code signing certificates. We cluster publisher names that belong to the same publisher according to similarities on the publisher names, domain names in events with URLs, and Malsign clustering results.

**Publisher name similarity.** This feature aims to group together certificates used by the same identity that have small variations on the publisher name. Since the WINE binary downloads dataset contains the publisher name for parent and child files, this feature can be used even when a signed sample has no VT report and we do not have the executable (i.e., not in Malsign). The similarity between two publisher names is computed in two parts: first derive a list of normalized tokens from each publisher

| Publishers | Clusters | Singletons | Largest | Median |
|---|---|---|---|---|
| 6,066 | 5,074 | 4,534 | 103 | 1 |

Table 4.4: Publisher clustering results.

name through four steps and then compute similarity between the token lists.

To obtain the token list of a publisher name, the first step is to extract parenthesized strings as separate tokens. For example, given the publisher name "Start Playing (Start Playing (KnockApps Limited))" this step produces 3 tokens: "Start Playing", "Start Playing", and "KnockApps Limited". The second step converts each token to lowercase and removes all non-alphanumeric characters from the token. The third step removes from the tokens company extensions (e.g., ltd, limited, inc, corp), geographical locations (e.g., countries, cities), and the string "Open Source Developer", which appears in code signing certificates issued to individual developers of open source projects. Finally, tokens that have less than 3 characters and duplicate tokens are removed.

To compute the similarity between two token lists, for each pair of publisher names $P_1$ and $P_2$, we calculate the normalized edit distance among all token pairs $(t_i, t_j)$ where $t_i$ belongs to $P_1$ and $t_j$ to $P_2$. If the edit distance between $P_1$ and $P_2$ is less than 0.1, we consider these two publishers to be the same. We selected this threshold after experimenting with different threshold values over 1,157 manually labeled publisher names. The edit distance threshold of 0.1 allowed us grouping the 1,157 publisher names into 216 clusters with 100% precision, 81.9% recall, and 86.4% F1 score.

**Child download domains.** If child executables signed by different publisher names are often downloaded from the same domains, that is a good indication that the publisher names belong to the same entity. To capture this behavior, we compute the set of ESLDs from where files signed by the same publisher name have been downloaded. Note that we exclude ESLDs that correspond to file lockers and download portals as they are typically used by many different publishers. The publisher names whose Jaccard Index of their ESLD sets is over 0.5 are put to the same cluster.

**Parent download domains.** Similarly, if parent files signed by different publisher names download from a similar set of domains, this indicates the publisher names likely belong to the same entity. This feature first computes the set of ESLDs from where parent files signed by the same publisher name download (excluding file lockers and download portals). Publisher names whose Jaccard Index is over 0.5 are put to the same cluster.

**Malsign clustering.** For each Malsign cluster we extract the list of distinct publisher names used to sign executables in the cluster, i.e., Subject CN strings extracted from certificates for files in the cluster. We consider that two publisher names in the same Malsign cluster belong to the same publisher.

**Final clustering.** We group publisher names into the same cluster if they satisfy at least one of the first 3 features explained above or are in the same Malsign cluster. Table 4.4 summarizes the clustering, which produces 5,074 clusters from 6,066 publisher names.

## 4.5 PUP Prevalence

In this Section, we measure the prevalence of PUP, based on the number of hosts in the WINE binary downloads dataset (i.e., WINE hosts) that have installed programs from PUP publishers. We measure the total number of WINE hosts affected by PUP, rank PUP publishers by installation base, and compare the installation base of PUP publishers to benign publishers.

We first compute the *detection ratio* (DR) for each cluster, which is the number of samples signed by publishers in the cluster flagged by at least 4 AVs, divided by the total number of samples in the cluster for which we have a VT report. We mark as PUP those clusters with $DR > 5\%$, a threshold chosen because is the lowest that leaves out known benign publishers. From this point on, when we refer to PUP publishers, we mean the 915 publisher clusters with $DR > 5\%$.

Note that the number of WINE hosts with installed programs from a publisher cluster constitutes a quite conservative lower bound on the number of hosts across the Internet that have programs installed from that publisher. It captures only those Symantec customers that have opted-in to share data and have been sampled into WINE. If we take into account that Symantec only had 8% of the AV market share in January 2014 [127] and that only $\frac{1}{16}$ of Symantec users that opt-in to share telemetry are sampled into WINE [128], we estimate that the number of WINE hosts is two orders of magnitude lower than the corresponding number of Internet-connected hosts. Furthermore, we do not count WINE hosts with only unsigned PUP executables installed.

**PUP prevalence.** We find 2.1M WINE hosts, out of a total 3.9M WINE hosts in our time period, with at least one executable installed from the 915 PUP clusters. Thus, 54% of WINE hosts have PUP installed. This ratio is a lower bound because we only count signed PUP executables (i.e., we ignore unsigned PUP executables) and also because our initial PUP publisher list in Section 4.3 may not be complete. Thus, PUP is prevalent: more than half of the hosts examined have some PUP installed.

**Top PUP publishers.** Table 4.5 shows the top 20 PUP publishers by WINE installation base and details the cluster name, whether the publisher is a PPI service (this classification is detailed in Section 4.6), the number of publisher names in the cluster, detection ratio, and host installation base. The number of publishers ranks from singleton clusters up to 48 publishers for IronSource, an Israeli PPI service. The installation bases for the top 20 PUP publishers range from 200K up to over 1M for Perion Network, an Israeli PPI service that bought the operations of the infamous Conduit

| # | Cluster | PPI | Pub | DR | Hosts |
|---|---|---|---|---|---|
| 1 | Perion Network | ✓ | 5 | 52% | 1.0M |
| 2 | Mindspark | ✗ | 1 | 85% | 533K |
| 3 | Bandoo Media | ✗ | 5 | 46% | 373K |
| 4 | Web Pick | ✓ | 21 | 79% | 346K |
| 5 | IronSource | ✓ | 48 | 81% | 332K |
| 6 | Babylon | ✗ | 1 | 38% | 330K |
| 7 | JDI BACKUP | ✗ | 1 | 56% | 328K |
| 8 | Systweak | ✗ | 3 | 37% | 320K |
| 9 | OpenCandy | ✓ | 1 | 55% | 311K |
| 10 | Montiera Technologies | ✗ | 2 | 54% | 303K |
| 11 | Softonic International | ✗ | 2 | 70% | 292K |
| 12 | PriceGong Software | ✗ | 1 | 18% | 292K |
| 13 | Adknowledge | ✓ | 7 | 75% | 277K |
| 14 | Adsology | ✗ | 2 | 77% | 276K |
| 15 | Visual Tools | ✗ | 2 | 70% | 275K |
| 16 | BitTorrent | ✗ | 1 | 40% | 271K |
| 17 | Wajam | ✗ | 2 | 87% | 218K |
| 18 | W3i | ✓ | 4 | 93% | 216K |
| 19 | iBario | ✓ | 15 | 84% | 208K |
| 20 | Tuguu | ✓ | 14 | 94% | 200K |

Table 4.5: Top 20 PUP publishers by installation base.

toolbar in 2013. As explained earlier, these numbers are a quite conservative lower bound. We estimate the number of Internet-connected computers for these publishers to be two orders of magnitude larger, in the range of tens of millions, and up to a hundred million, hosts. We have found anecdotal information that fits these estimates. For example, an adware developer interviewed in 2009 claimed to have control over 4M machines [129].

**Comparison with benign publishers.** Table 4.6 shows the top 20 publisher clusters, benign and PUP, in WINE. The most common publishers are Microsoft and Symantec that are installed in nearly all hosts. The Perion Network / Conduit PPI network ranks 15 overall. That is, there are only 14 benign software publishers with a larger installation base than the top PUP publisher. Perion Network is more prevalent than well known publishers such as Macrovision and NVIDIA. The second PUP publisher (Mindspark Interactive Network) has the rank 24. This highlights that top PUP publishers are among the most widely installed software publishers.

A reader may wonder if we could also compute the installation base for malware families. Unfortunately, due to malware being largely unsigned and highly polymorphic, we would need to first classify millions of files in WINE (without having access to the binaries) before we can perform the ranking.

| # | Cluster | PUP | Hosts |
|---|---|---|---|
| 1 | Microsoft | ✗ | 3.9M |
| 2 | Symantec | ✗ | 3.8M |
| 3 | Adobe Systems | ✗ | 3.5M |
| 4 | Google | ✗ | 3.1M |
| 5 | Apple | ✗ | 1.8M |
| 6 | Intel | ✗ | 1.6M |
| 7 | Sun Microsystems | ✗ | 1.6M |
| 8 | Cyberlink | ✗ | 1.6M |
| 9 | GEAR Software | ✗ | 1.5M |
| 10 | Hewlett-Packard | ✗ | 1.5M |
| 11 | Oracle | ✗ | 1.4M |
| 12 | Skype Technologies | ✗ | 1.3M |
| 13 | Mozilla Corporation | ✗ | 1.0M |
| 14 | McAfee | ✗ | 1.0M |
| 15 | Perion Network / Conduit | ✓ | 1.0M |
| 16 | WildTangent | ✗ | 941K |
| 17 | Macrovision Corporation | ✗ | 802K |
| 18 | LEAD Technologies | ✗ | 775K |
| 19 | NVIDIA Corporation | ✗ | 722K |
| 20 | Ask.com | ✗ | 624K |
| 24 | Mindspark Interactive Network | ✓ | 533K |

Table 4.6: Top publishers by install base (benign and PUP).

## 4.6 Classifying Publishers

Among the 5,074 PUP publisher clusters obtained in Section 4.4 we want to identify important clusters playing a specific role in the ecosystem. In particular, we want to identify clusters that correspond to PPI services and to examine the type of programs distributed by the dominant advertisers. For this, we first build a *publisher graph* that captures the who-installs-who relationships. Then, we apply filtering heuristics on the publisher graph to select a subset of publishers that likely hold a specific role, e.g., PPI service. Finally, we manually classify the filtered publishers into roles by examining Internet resources, e.g., publisher web pages, PPI forums, and the Internet Archive [130].

**Publisher graph.** The publisher graph is a directed graph where each publisher cluster is a node and an edge from cluster $C_A$ to cluster $C_B$ means there is at least one event where a parent file from $C_A$ installed a child file from $C_B$. Self-edges are excluded, as those indicate program updates and downloads of additional components from the same publisher. Note that an edge captures download events between parent and child clusters across all hosts and the 19 months analyzed. Thus, the publisher graph captures the who-installs-who relationships over that time period, enabling a birds-eye

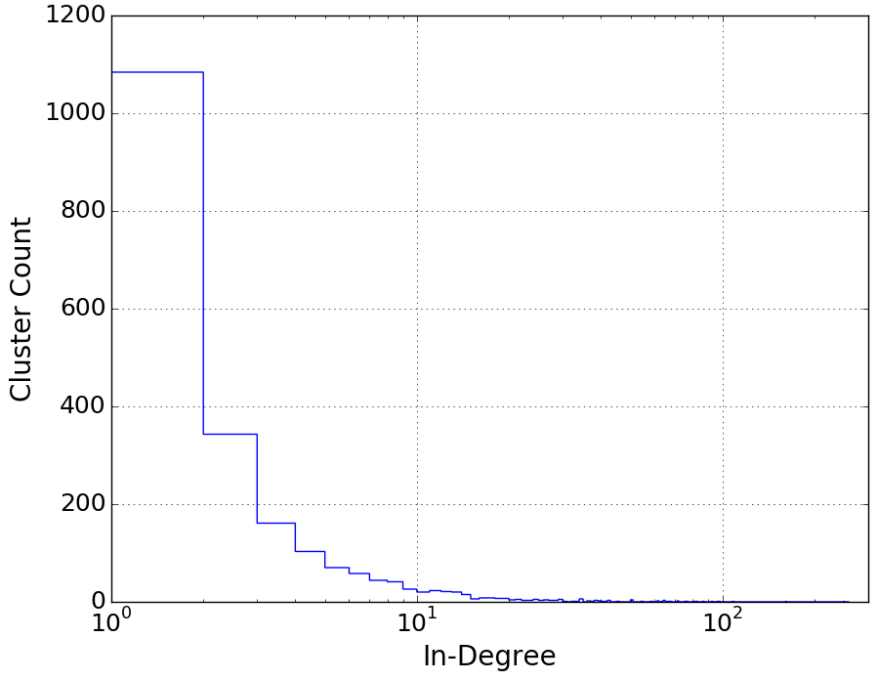



Figure 4.2: Cluster in-degree distribution.

view of the ecosystem.

**In-degree and out-degree.** We first measure the in-degree and out-degree of each cluster in the publisher graph. The in-degree is the count of distinct parent publisher clusters that install programs from a child publisher cluster. Intuitively, publishers with a high in-degree are installed by many other publishers, which indicates that they are buying installs. The out-degree is the count of distinct child publisher clusters installed by a parent publisher cluster. Intuitively, publishers with a high out-degree install many other publishers, which indicates that they are selling installs.

To compute a cluster's in-degree we filter out 12 benign parent clusters that correspond to tools that download large numbers of executables from different publishers such as browsers, BitTorrent clients, and Dropbox. To compute a cluster's out-degree we exclude benign child publishers ($DR < 5\%$) that are typically dependencies.

Figure 4.2 shows the in-degree distribution. 57% of the clusters have no parents (i.e., installed by unsigned files only). Another 21.5% have one. These are typically installed only by parents in the same cluster. Only 224 (4.4%) clusters have an in-degree larger than 10. We call these high in-degree clusters. Figure 4.3 shows the out-degree distribution. 572 clusters (11%) have an out-degree larger than zero and only 133 (2.6%) clusters have an out-degree larger than 10. We call these high out-degree clusters.

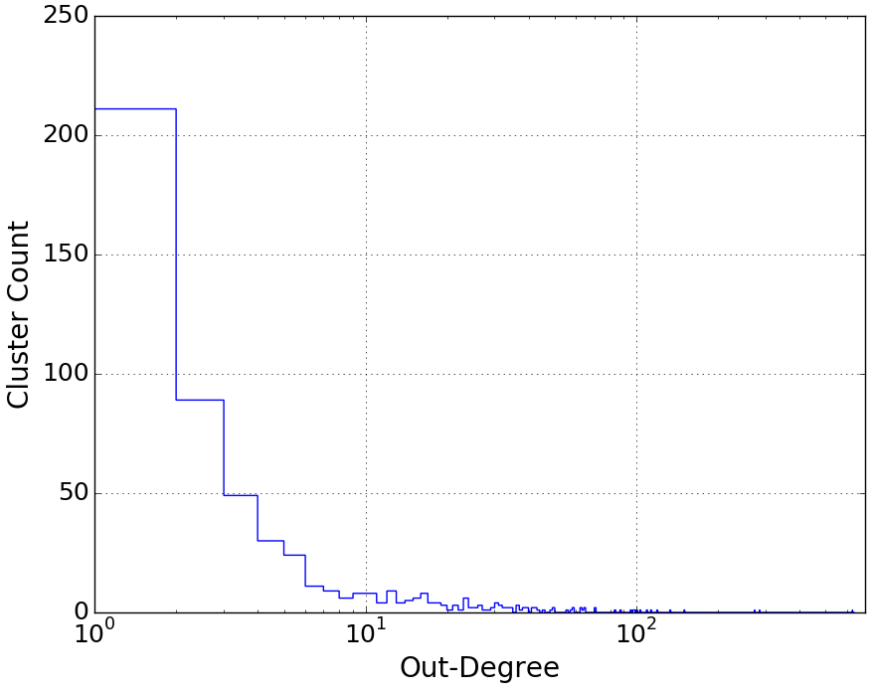



Figure 4.3: Cluster out-degree distribution.

**PPI services.** To identify PPI services in the publisher graph, we first select all PUP publisher clusters with both high in-degree and high out-degree (i.e., $DR \geq 5\% \wedge ID \geq 10 \wedge OD \geq 10$), which indicate these publishers are buying and selling installs. This rule reduces the 5,074 clusters to 49 candidate publisher clusters. Next, we manually classify those 49 clusters through extensive analysis using PPI forums, publisher websites, and the Internet Archive. Of those 49 clusters, we classify 22 as PPI services, 12 as advertisers that run an affiliate program, 8 as advertisers without an affiliate program; 3 as download portals (Download.com, BrotherSoft, Softonic), and 4 as PUP publishers that distribute free download tools (e.g., BitTorrent clients). The latter tools inflate the out-degree of their publishers and were not included in our whitelist of download tools due to the high DR of their publishers. Our manual analysis also reveals two additional PPI services (7install and Install Monster) that were missed by our rule because they do not achieve high enough in-degree and out-degree, either because of low popularity or because they appear at the end of our observation period.

Table 4.7 summarizes the 24 identified PPI services sorted by installation base. For each cluster, it shows the name of the PPI service, in-degree, out-degree, installation base, detection ratio, and number of publishers in the cluster. The classification reveals that 3 of the top 5 PUP publishers by installation base in Table 4.5 are PPI services. Thus, the most popular PUP publishers are PPI services. Some of the PPI services identified no longer work at the time this paper is published, e.g., OneInstaller, but their PPI service front-ends are present in the Internet Archive.

During our manual analysis we keep track of all PPI services we find advertised

| # | Cluster | PPI Service | ID | OD | Hosts | DR | Pub. |
|---|---|---|---|---|---|---|---|
| 1 | Perion Network/Conduit | CodeFuel [131] | 168 | 63 | 1 M | 52% | 5 |
| 2 | Yontoo | Sterkly [132] | 53 | 17 | 601 K | 93% | 103 |
| 3 | iBario | RevenueHits [133] | 62 | 36 | 479 K | 84% | 16 |
| 4 | Web Pick | InstalleRex [134] | 65 | 22 | 346 K | 79% | 21 |
| 5 | IronSource | InstallCore [135] | 73 | 112 | 332 K | 81% | 48 |
| 6 | OpenCandy | OpenCandy [136] | 91 | 36 | 311 K | 55% | 1 |
| 7 | Adknowledge | Adknowledge [137] | 53 | 48 | 277 K | 75% | 7 |
| 8 | W3i | NativeX [138] | 38 | 49 | 216 K | 93% | 4 |
| 9 | Somoto | BetterInstaller [139] | 60 | 70 | 209 K | 96% | 5 |
| 10 | Firseria | Solimba [140] | 41 | 30 | 209 K | 94% | 9 |
| 11 | Tuguu | DomaIQ [141] | 49 | 16 | 200 K | 94% | 14 |
| 12 | Download Admin | DownloadAdmin [142] | 25 | 16 | 192 K | 73% | 2 |
| 13 | Air Software | AirInstaller [143] | 33 | 41 | 191 K | 79% | 1 |
| 14 | Vittalia Internet | OneInstaller [144] | 27 | 29 | 155 K | 71% | 18 |
| 15 | Amonetize | installPath [145] | 50 | 63 | 154 K | 93% | 2 |
| 16 | SIEN | Installbay [146] | 34 | 33 | 139 K | 80% | 2 |
| 17 | OutBrowse | RevenYou [147] | 22 | 41 | 86 K | 94% | 4 |
| 18 | Verti Technology Group | Verti [148] | 17 | 39 | 47 K | 44% | 1 |
| 19 | Blisbury | Smart WebAds [149] | 19 | 30 | 46 K | 77% | 2 |
| 20 | Nosibay | Nosibay [150] | 19 | 20 | 30 K | 75% | 1 |
| 21 | ConversionAds | ConversionAds [151] | 10 | 38 | 24 K | 72% | 1 |
| 22 | Installer Technology | InstallerTech [152] | 10 | 14 | 11 K | 56% | 1 |
| 23 | 7install | 7install [153] | 2 | 0 | 75 | 12% | 1 |
| 24 | Install Monster | Install Monster [154] | 3 | 1 | 9 | 100% | 1 |

Table 4.7: PPI services services identified sorted by installation base.

on the Internet, e.g., on PPI forums. In addition to the 24 PPI services in Table 4.7 we identify another 12 PPI services, shown in Table 4.8.

There are several reasons for which we do not observe those 12 PPI services in our data. First, some of them are simply resellers that pay affiliate publishers to distribute bundles or downloaders for other PPI services. Second, PPI services may have been launched (or gained popularity) after the end of our observation period (e.g., AdGazelle). Third, some PPI services may distribute unsigned bundles or downloaders. For example, we examined over 30K samples that AV engines label as belonging to the InstallMonetizer PPI service, of which only 8% were signed. Finally, some PPI services may have so low volume that they were not observed in our initial 11 M sample feed.

**Advertisers.** To identify advertisers in the publisher graph, we first select PUP clusters with high in-degree, low out-degree, and for which at least one parent is one of the 24 PPI services (i.e., $DR \geq 5\% \wedge ID \geq 10 \wedge OD \leq 9 \wedge PPPI > 0$). Advertisers pay to have their products installed (i.e., buy installs) and may not install other publishers

| # | PPI Service | Reseller |
|---|---|---|
| 1 | AdGazelle [155] | |
| 2 | EarnPerInstall [156] | |
| 3 | GuppyGo [157] | |
| 4 | Installaxy [158] | ✓ |
| 5 | InstallMonetizer [123] | |
| 6 | MediaKings [159] | |
| 7 | NetCashRevenue [160] | ✓ |
| 8 | PayPerInstall [161] | |
| 9 | PerInstallBox [162] | |
| 10 | PerInstallBucks [3] | ✓ |
| 11 | PerInstallCash [163] | |
| 12 | PureBits [164] | ✓ |

Table 4.8: PPI services found through manual analysis on PPI forums and other Internet resources that are not present in our dataset. The reseller data comes from [4].

for monetization as they know how to monetize the machines themselves. Since buying installs costs money, they need to generate enough income from the installations to offset that cost. This filtering identifies 77 clusters, which we manually examine to identify the main product they advertise (they can advertise multiple ones) and whether they run an affiliate program where they pay affiliates to distribute their programs. We also include in this analysis the 20 advertiser clusters manually identified in the PPI service identification above.

Table 4.9 shows the top 30 advertiser clusters by installation base. The table shows the cluster name, whether it runs an affiliate program, in-degree, out-degree, detection ratio, installation base, the number of parent PPI service nodes, the number of child PPI service nodes, the main product they install, and whether they install browser add-ons (BAO). The latter includes any type of browser add-ons such as toolbars, extensions, plugins, browser helper objects, and sidebars.

The data shows that 18 of the 30 top advertisers install browser add-ons. Those browser add-ons enable monetization through Web traffic, predominantly through different types of advertisement. Common methods are modifying default search engines to monetize searches (e.g., SearchResults, Delta Toolbar, Imminent Toolbar), shopping deals and price comparisons (e.g., PriceGong, PricePeep, DealPLY, SupremeSavings), and other types of advertisement such as pay-per-impression and pay-per-action (e.g., Widgi Toolbar, Inbox Toolbar).

The 12 advertisers that focus on client applications monetize predominantly through selling licenses and subscriptions. The main group is 6 publishers advertising rogueware claiming to improve system performance (Regclean Pro, Optimizer Pro, SpeedUpMyPC, PC Speed Maximizer, Advanced System Care, DAEMON Tools). These rogueware try to convince users to buy the license for the full version. We also observe multimedia tools (Free Studio, GOM Player), backup tools (MyPC Backup), game

| # | Cluster | Aff | ID | OD | DR | Hosts | PPPI | CPPI | Main Product | BAO |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Xacti | ✗ | 57 | 9 | 22% | 563 K | 13 | 1 | RebateInformer | ✓ |
| 2 | Mindspark | ✓ | 62 | 17 | 85% | 533 K | 3 | 5 | Mindspark Toolbar | ✓ |
| 3 | Bandoo Media | ✓ | 86 | 108 | 46% | 373 K | 7 | 18 | MediaBar | ✓ |
| 4 | Babylon | ✓ | 83 | 14 | 38% | 330 K | 16 | 3 | Babylon Toolbar | ✓ |
| 5 | JDI Backup Limited | ✓ | 71 | 19 | 56% | 328 K | 17 | 3 | MyPC Backup | ✗ |
| 6 | Systweak | ✓ | 81 | 24 | 37% | 320 K | 7 | 2 | Regclean Pro | ✗ |
| 7 | Montiera Technologies | ✗ | 37 | 2 | 66% | 303 K | 8 | 1 | Delta Toolbar | ✓ |
| 8 | PriceGong Software | ✗ | 12 | 0 | 17% | 292 K | 6 | 0 | PriceGong | ✓ |
| 9 | Adsology | ✓ | 62 | 12 | 77% | 276 K | 17 | 1 | OptimizerPro | ✗ |
| 10 | Wajam | ✗ | 42 | 5 | 87% | 218 K | 11 | 2 | Wajam | ✓ |
| 11 | Visicom Media | ✗ | 13 | 2 | 14% | 185 K | 4 | 0 | VMN Toolbar | ✓ |
| 12 | Linkury | ✗ | 46 | 2 | 54% | 174 K | 13 | 0 | SmartBar | ✓ |
| 13 | Uniblue Systems | ✓ | 64 | 13 | 11% | 160 K | 10 | 1 | SpeedUpMyPC | ✗ |
| 14 | Search Results | ✗ | 35 | 3 | 79% | 159 K | 12 | 2 | SearchResults | ✓ |
| 15 | Bitberry Software | ✓ | 13 | 64 | 88% | 130 K | 1 | 7 | BitZipper | ✗ |
| 16 | Iminent | ✗ | 13 | 1 | 74% | 118 K | 4 | 1 | Iminent Toolbar | ✓ |
| 17 | DealPly Technologies | ✗ | 43 | 0 | 93% | 108 K | 16 | 0 | DealPly | ✓ |
| 18 | Smart PC Solutions | ✓ | 38 | 0 | 32% | 106 K | 13 | 0 | PC Speed Maximizer | ✗ |
| 19 | DVDVideoSoft | ✗ | 15 | 2 | 18% | 101 K | 3 | 1 | Free Studio | ✗ |
| 20 | Spigot | ✓ | 17 | 1 | 39% | 101 K | 1 | 1 | Widgi Toolbar | ✓ |
| 21 | Web Cake | ✗ | 34 | 2 | 98% | 97 K | 16 | 2 | Desktop OS | ✓ |
| 22 | GreTech | ✓ | 13 | 1 | 21% | 90 K | 3 | 1 | GOM Player | ✗ |
| 23 | Digital River | ✓ | 17 | 0 | 10% | 80 K | 1 | 0 | DR Download Manager | ✓ |
| 24 | Widdit | ✓ | 20 | 16 | 27% | 79 K | 4 | 2 | HomeTab | ✓ |
| 25 | EpicPlay | ✗ | 12 | 4 | 90% | 77 K | 3 | 1 | EpicPlay | ✗ |
| 26 | Iobit Information Technology | ✓ | 18 | 8 | 6% | 73 K | 3 | 1 | Advanced SystemCare | ✗ |
| 27 | DT Soft | ✗ | 14 | 2 | 22% | 68 K | 2 | 1 | DAEMON Tools | ✗ |
| 28 | Innovative Apps | ✗ | 14 | 1 | 68% | 60 K | 7 | 0 | Supreme Savings | ✓ |
| 29 | Woolik Technologies | ✗ | 13 | 9 | 70% | 50 K | 4 | 1 | Woolik Search Tool | ✓ |
| 30 | Visual Software Systems | ✓ | 22 | 12 | 62% | 42 K | 5 | 3 | VisualBee | ✗ |

Table 4.9: Top 30 advertiser clusters by installation base. For each publisher cluster it shows: whether we found an affiliate program (Aff), the in-degree (IN), out-degree (OD), detection ratio (DR), installation base (Hosts), number of parent PPI services (PPPI), number of child PPI services (CPPI), the main product advertised, and whether that product is a browser add-on (BAO) including toolbars, extensions, sidebars, and browser helper objects.

promotion (EpicPlay), compressors (BitZipper), and presentation tools (Visual Bee).

## 4.7 PUP Distribution Methods.

This section measures the distribution of PUP through PPI services and affiliate programs. The relevant data is provided in Table 4.10. From the 90 M events with parent information, we first find the events with child files that are signed by PUP publishers (40.1M events). Then, we investigate the parents that installed them. In 28.6M (71%) of these events, parents were signed, therefore allowing us to go further in our search for finding the parents who are PPIs. 7.4M (35%) of the these parents correspond to Web browsers and other benign download programs such as BitTorrent clients and Dropbox. The remaining 21.2M (65%) events have a PUP parent. This indicates that the majority of PUP is installed by other PUP. In particular, for 7.3M out of 21.2M events (34%) with PUP parent, the parent corresponds to one of the 24 PPI services

| Event Type | Count |
|---|---|
| All PUP downloads | 40.1M |
| Unsigned parent | 11.5M |
| Signed parent | 28.6M |
| Benign parent | 7.4M |
| PUP parent | 21.2M |
| PPI | 7.3M |
| Adv. affiliate program | 5.5M |

Table 4.10: Analysis of PUP download events.

identified in Table 4.7. And, for another 5.5M (26%) events the parent corresponds to one of the 21 affiliate programs identified in Section 4.6. From these statistics, we can conclude that PUPs are generally installed by other PUPs and moreover, over 25% of the PUP download events are sourced by PPI services, and another 19% by advertisers with affiliate programs.

## 4.8 PUP–Malware Relationships

We are interested in understanding if there is any form of relationship between PUP and malware and if malware uses the PPI services we identified. In particular we would like to measure the percentage of PUP that installs malware or is installed by malware. Here, the obvious challenge is to accurately label malware in the WINE dataset. While the majority of properly signed executables flagged by AV engines are PUP, unsigned executables flagged by AV engines can be PUP or malware and there are a few malware that are signed.

To address these issues, we use AVClass, a recently released malware labeling tool [165]. Given the VT reports of a large number of executables, AVClass addresses the most important challenges in extracting malware family information from AV labels: label normalization, generic token detection, and alias detection. For each sample, it outputs a ranking of the most likely family names ranked by the number of AV engines assigning that family to the sample. Since AV labels can be noisy [112], we focus on executables for which the top family AVClass outputs is in a precomputed list of 70 malware families that includes prevalent families such as zbot, zeroaccess, reveton, virut, sality, shylock, and vobfus. Clearly, our methodology is not 100% accurate, but allows us to gain insight on the relationships between malware and PUP.

**PUP downloading malware.** One way malware authors could relate to PUP could be by signing up as advertisers to PPI services to distribute their malware. To identify such cases, we look for PUP publishers that download executables from one of the 70 malware families considered. What we have found out is that there is a link between 71 of the PUP publisher clusters to malware. Those publishers distribute malware

from 40 families through 5,586 download events. Out of those 71 clusters, 11 are classified as PPI services in Section 4.6. Those PPI services generate 35% of the 5,586 malware downloads by PUP. For example, Perion Network, the most popular PPI service, downloads instances of zbot, shylock, and andromeda trojans. We also observe at the end of 2013 iBario downloading instances of sefnit clickfraud malware as reported by TrendMicro [11]. Clearly, 5,586 downloads is a low number, which may indicate that malware favors silent distribution vectors and that PPI services are careful to avoid malware to preserve their reputation towards security vendors. We only observe occasional events spread amongst multiple PPI services, possibly due to insufficient checks by those PPI services. Another factor of influence may be that installs through these PPIs can be an order of magnitude more expensive than those from silent PPIs, as shown in Section 4.2.1.

**Malware downloading PUP.** Malware authors could also sign up as affiliate publishers to PPI services to monetize the compromised machines by selling installs. To capture this behavior, we analyzed PUP downloaded by samples from the 70 malware families considered. We found 11K downloads by malware from 25 families. These malware samples downloaded executables from 98 PUP publisher clusters. 88% of these downloads were generated by 3 malware families: vobfus, badur, and delf. 7 of the 98 PUP publisher clusters belong to the PPI services category. For example, we observe zeroaccess installing files from the DomaIQ PPI service. Overall, malware downloading PUP is a more common event than PUP downloading malware, but still rare, affecting only 0.03% of all events where PUP is downloaded.

The conclusion of this analysis is that while PUP–malware interactions exist, they are not prevalent and malware distribution seems disjoint from PUP distribution. Observed malware–PPI service interactions do not focus on a few misbehaving PPI services, but rather seem to occasionally affect many PPI services.

## 4.9 Domain Analysis

In this section we analyze the 1.1 M events that contain a URL, and in particular the domains (ESLDs) in those URLs. The events that contain a URL allow us to identify publishers that download from and are downloaded from a domain. Note that the domains we extract from this dataset are used for hosting and distributing executables and do not cover all of the domains used by PUP. We identify 3 main types of domains from our analysis:

- **File lockers.** Cloud storage services used for backup or sharing executables between users. They exhibit a high number of client publishers being downloaded from them, most of which are benign (e.g., Microsoft, Adobe, AutoDesk). These ESLDs also host a front-end website for users.

- **Download portals.** They also distribute programs from a high number of publishers, predominantly free software publishers and their own PPI services. They

also host a front-end website.

- **PPI services.** Used by PPI services to host their wrappers and advertised programs. These ESLDs do not host a front-end website as they are accessed by PPI installers, rather than humans.

**Rank by downloaded publishers.** Table 4.11 shows the top 20 ESLDs by number of child publishers signing files downloaded from that ESLD. The 4 tick-mark columns classify the domain as file locker (FL), download portal (DP), PPI service (PPI), or other (Oth). Of the 20 ESLDs, 15 correspond to file lockers, 2 to download portals, and another 2 to PPI services. The remaining domain is `file.org`, a portal where users can enter a file extension to find a tool that can open files with that extension. The publisher behind this portal uses it to promote its own free file viewer tool, which is offered as the best tool to handle over 200 file extensions.

If we give a vote to the top 3 publishers downloaded from each of the 15 file lockers (45 votes), Microsoft gets 13, Adobe 11, Cyberlink 4, and AutoDesk 3. The rest are popular benign publishers such as Ubisoft, VMWare, and Electronic Arts. Thus, file lockers predominantly distribute software from reputable publishers.

For the two download portals, the publishers downloaded from them correspond to their own PPI service (i.e., bundles signed by "CBS Interactive" from cnet.com), free software publishers, and PPI services. For `edgecastcdn.net` all 67 publishers are part of the same PPI service run by the Yontoo group. The domain `d3d6wi7c7pa6m0.cloudfront.net` belongs to the Adknowledge PPI service and distributes their advertiser programs. Among those advertiser programs we observe bundles signed by other PPI services, which may indicate arbitrageurs who try to take advantage of pricing differentials among PPI services [56].

**Rank by downloads.** Table 4.12 ranks the top 20 domains by number of downloads. It shows the ESLD, the type (file locker, download portal, PPI service, advertiser, other), the cluster that owns the domain, the number of downloads, the number of publishers of the downloaded executables, and the number of distinct files downloaded. We label each domain as belonging to the cluster that signs most executables downloaded from the domain. The publisher in the other category is Frostwire, which distributes a popular free BitTorrent client.

Table 4.12 shows that PPI domains dominate in terms of downloads, but distribute a smaller number of child publishers compared to file lockers and download portals that dominate Table 4.11. It also shows that it is possible to link download domains to the publishers that own them based on the signature of files they distribute, despite the domains being typically registered by privacy protection services.

| ESLD | FL | DP | PPI | Oth | Pub |
|---|---|---|---|---|---|
| uploaded.net | ✓ | | | | 366 |
| cnet.com | | ✓ | | | 142 |
| extabit.com | ✓ | | | | 128 |
| share-online.biz | ✓ | | | | 125 |
| 4shared.com | ✓ | | | | 120 |
| rapidgator.net | ✓ | | | | 90 |
| depositfiles.com | ✓ | | | | 76 |
| mediafire.com | ✓ | | | | 73 |
| edgecastcdn.net | | | ✓ | | 67 |
| chip.de | | ✓ | | | 53 |
| zippyshare.com | ✓ | | | | 49 |
| uloz.to | ✓ | | | | 48 |
| file.org | | | | ✓ | 47 |
| putlocker.com | ✓ | | | | 47 |
| d3d6wi7c7pa6m0.cf | | | ✓ | | 44 |
| turbobit.net | ✓ | | | | 44 |
| freakshare.com | ✓ | | | | 41 |
| rapidshare.com | ✓ | | | | 40 |
| ddlstorage.com | ✓ | | | | 38 |
| bitshare.com | ✓ | | | | 38 |

Table 4.11: Top 20 ESLDs by number of distinct publishers of downloaded executables. FL means file locker, DP download portal, PPI pay-per-install service, and Oth other. For brevity, d3d6wi7c7pa6m0.cf stands for d3d6wi7c7pa6m0.cloudfront.net.

## 4.10 Discussion

**Unsigned PUP.** Our work focuses on signed PUP executables based on the prior observation that most signed samples flagged by AV engines are PUP [19]. However, this means that we will miss PUP publishers if they distribute only unsigned executables. Also, our PUP prevalence measurements are only a lower bound since there may be hosts with only unsigned PUP installed. In concurrent work, Thomas et al. [4] infiltrate 4 PPI services observing that only 58% of the advertiser software they distribute is signed. Thus, we could be missing as much as 42% of PUP software, but we expect a much smaller number of hosts will only have unsigned PUP installed.

**Affiliate publisher analysis.** We have classified publisher clusters as PPI services and advertisers, but we have not examined affiliate publisher clusters. One challenge with affiliate publishers is that when distribution happens through a stand-alone PPI installer (rather than bundles) both the advertiser program and the affiliate publisher program may appear as children of the PPI service in the publisher graph. It may be possible to measure the number of affiliates for some PPI services by analyzing URL parameters of download events. We leave this analysis to future work.

| ESLD | FL | DP | PPI | Ad | Oth | Cluster | Downl. | Pub. | Children |
|---|---|---|---|---|---|---|---|---|---|
| conduit.com | | | ✓ | | | Perion Network | 138,480 | 2 | 727 |
| edgecastcdn.net | | | ✓ | | | Yontoo | 106,449 | 67 | 1,148 |
| frostwire.com | | | | | ✓ | Frostwire | 53,592 | 1 | 2,511 |
| ask.com | | | | ✓ | | Ask | 40,939 | 6 | 125 |
| imgfarm.com | | | | ✓ | | Mindspark | 26,498 | 6 | 3,209 |
| ilivid.com | | | | ✓ | | Bandoo Media | 25,429 | 5 | 905 |
| conduit-services.com | | | ✓ | | | Perion Network | 21,149 | 8 | 1,345 |
| adpk.s3.amazonaws.com | | | | ✓ | | Adpeak | 14,513 | 2 | 36 |
| airdwnlds.com | | | ✓ | | | Air Software | 14,342 | 1 | 13,389 |
| ncapponline.info | | | ✓ | | | Web Pick | 13,974 | 11 | 13,252 |
| uploaded.net | ✓ | | | | | Cyando | 10,886 | 366 | 7,816 |
| storebox1.info | | | ✓ | | | Web Pick | 10,109 | 13 | 9,561 |
| oi-installer9.com | | | ✓ | | | Adknowledge | 8,360 | 4 | 7,892 |
| 4shared.com | ✓ | | | | | 4shared | 8,222 | 120 | 5,649 |
| systweak.com | | | | ✓ | | Systweak | 8,104 | 4 | 509 |
| mypcbackup.com | | | | ✓ | | JDI Backup Limited | 7,837 | 1 | 43 |
| greatfilesarey.asia | | | ✓ | | | Web Pick | 7,699 | 8 | 7,296 |
| incredimail.com | | | ✓ | | | Perion Network | 7,408 | 3 | 2,571 |
| softonic.com | | ✓ | | | | Softonic | 6,980 | 36 | 3,869 |
| nicdls.com | | | ✓ | | | Tuguu | 6,908 | 14 | 1,704 |

Table 4.12: Top ESLDs by number of downloads from them. The two rightmost columns are the number of publishers and files of the downloads.

**Other distribution models.** We have examined PUP distribution through PPI services and advertiser affiliate programs. However, other distribution models exist. These include bilateral distribution agreements between two parties (e.g., Oracle's Java distributing the Ask toolbar [166]) and pre-installed PUP (e.g., Superfish on Lenovo computers [167]). We observe Superfish distributed through PPI services prior to the Lenovo agreement, which started in September 2014 after our analysis period had ended. We leave the analysis of such distribution models to future work.

**Observation period.** Our observation period covers 19 months from January 2013 to July 2014. Unfortunately, WINE did not include newer data at the time of our study. Thus, we miss newer PUP publishers that joined the ecosystem after our observation period. However, the vast majority of PUP publishers examined are still alive at the time of writing.

**Internet population.** We have measured the installation base of PUP (and benign) publishers on WINE hosts. We have also estimated that our measured WINE population may be two orders of magnitude lower than that of hosts connected to the Internet. But, we concede that this estimation is rough and could be affected by different factors such as selection bias.

# 5 PUP Prevalence in Enterprise Hosts

## 5.1 Introduction

Despite all efforts of the cyber security community, malware and other cyber attacks run rampant on the Internet. In recent years, there is almost not a single day we do not come across new incidents, such as data breaches [168] and ransomware attacks [169, 170]. Such incidents typically involve malware and affect both enterprises and consumers. While the security posture of consumers against malware and other cyber threats has been explored by security vendors [71–73] and the academic community [5, 37], the security posture of enterprises against those threats has been significantly under-studied. This is problematic because enterprises own a significant fraction of the hosts connected to the Internet and possess valuable assets, such as financial data and intellectual property that may be the objective of (targeted) attacks. Enterprises may differ from consumers in important ways, such as using the same software across hosts, establishing security policies, installing multiple security products, educating their employees, and having departments dedicated to securing their assets. However, there exists a large variety of enterprises in terms of size, industries they belong to, financial assets, and security investment. Thus, it is very likely that the best practices mentioned above do not equally apply to all of them.

Currently, it is not clear how the security posture of enterprises differs according to different factors and whether enterprises are indeed more secure than consumers, i.e., if their security investment is paying off. In this paper, we aim to shed light into these questions by conducting what is, to the best of our knowledge, the largest and longest measurement study of enterprise security. Our data covers nearly 3 years and is collected from 28K enterprises with over 82M client hosts and 73M public-facing servers. We analyze the enterprise threat landscape including the prevalence of malware and PUP in enterprise client hosts and how common security practices, such as vulnerability patching and operating system updates are handled. We use a wealth of datasets collected from a large cyber security company (Symantec) and public sources. At the core of our study are file reputation logs that capture the installation of files in

82M real enterprise client hosts. These logs enable us an *internal view* of enterprise security. We complement these logs with a classification of the 28K enterprises into 67 industries and with AV labels of low-reputation files for classification. To analyze the security of the externally visible enterprise servers, we supplement our internal view, with an *outside-in view* using data from public sources, such as Internet-wide scans and blacklists.

Most related to our work is a study by Yen et al. [171] on the security of a large multinational enterprise comprising of 85,000 hosts for which they had four months of logs from an AV engine deployed in the enterprise. Similar to that work, we have an internal view of enterprise security, but our study analyzes a time frame that is eight times longer and covers 28K enterprises across 67 industries. Other related works have studied the network hygiene and security posture of enterprises using an outside-in view based on Internet-wide scans and blacklists [69, 70, 172, 173]. The limitations of an outside-in view is that it only applies to externally reachable servers or is based on coarse-grained blacklists. Thus, its ability to provide an accurate view of the enterprise security posture remains to be proven. In contrast, we only use the outside-in view to complement and compare with our internal view of the enterprises. We find that the enterprise threat landscape looks very different from the inside and from the outside.

This paper comprises two main parts: an analysis of the enterprise threat landscape and an analysis of the enterprise vulnerability patching behavior. The first part of the paper, on analyzing the enterprise threat landscape, studies the encounter rate of malware and PUP in enterprises. It examines low reputation files installed on enterprise client hosts; classifies them into malware and PUP families; measures their prevalence; identifies the top families overall and per industry; examines whether some families target specific industries; analyzes the temporal evolution of the encounter rate; performs a case study on ransomware; and finally analyzes the outside-in view of enterprises by cross-checking blacklists with the publicly-facing IP addresses (including externally-facing servers).

The second part of the paper consists in the analysis of the vulnerability patching behavior of enterprises. We measure the time needed to patch 50% and 90% of the vulnerable population for 12 client-side applications installed on the 82M enterprise client hosts and 112 services installed on the 73M enterprise servers. For this, we first identify the list of vulnerabilities and vulnerable versions for those applications using NVD [174]. Then, we examine the time when those vulnerable versions are updated using the file reputation logs for client applications and Internet-wide scans for server applications. We also rank industries based on their patch deployment agility. Prior work has performed a similar study on client applications installed on consumer hosts [5] and has analyzed specific server vulnerabilities (e.g., Heartbleed) and misconfigurations [175–177]. However, to our knowledge, we are the first to measure the patch deployment behaviour of such a large number of enterprises, and to combine both client-side and server-side perspectives.

Below we list the most significant findings of our study:

- Between 91% (conservative estimate) and 97% (lax estimate) of the enterprises, 13% and 41% of the client hosts respectively, encountered at least one malware or PUP over the length of our study. Thus, despite their differences almost all enterprises will encounter some malware or PUP in three years.
- The 10 most-affected industries have 69%–76% of hosts affected, while the 10 least-affected have 15%–36%, highlighting that some industries, e.g., banks and finance-related, are definitely doing better than others.
- 73% of the low reputation files installed on enterprise hosts are unknown to VirusTotal, despite many being high prevalence. This questions how representative VirusTotal data may be of the enterprise landscape.
- Enterprises encounter malware (34% lax) much more often than PUP (8% lax). This is in contrast to prior works on consumer hosts that have shown that 54% had some PUP installed [37].
- Cracking tools for Microsoft products (e.g., KMSPico [178]) are found on 34% of all enterprises.
- Despite its notoriety, we observe ransomware affecting only a modest 0.02% of all enterprise client hosts.
- It takes over 6 months on average to patch 90% of the population across all vulnerabilities in the 12 client-side applications. This shows that patching still remains an issue even in enterprise settings.
- Enterprise computers are faster to patch vulnerabilities compared to consumer hosts.
- The patching of servers is overall much worse than the patching of client applications. On average a server application remains vulnerable for 7.5 months. Furthermore, it takes more than nine months for 90% of the enterprise server population to be patched.

## 5.2 Datasets

This section details the datasets used in our work, summarized in Table 5.1. We use *file reputation logs* to identify malicious files installed in 82M hosts across 28K enterprises; *file appearance logs* to identify the installation of 12 benign applications in the enterprises; *enterprise classification* to place enterprises into industries; *VirusTotal* (VT) reports to obtain AV labels to classify the malicious files; the *National Vulnerability Database* (NVD) to identify vulnerabilities in client-side and server-side applications and the range of versions affected; *Censys* [179] Internet-wide IPv4 scans

| Dataset | Data | Count |
|---|---|---|
| File Reputation Logs | Hosts | 82.1 M |
| 04/2015 – 12/2017 | Enterprises | 28 K |
| | Countries | 137 |
| | Total Reports | 375 B |
| | Total Distinct Files | 326 M |
| | Low Reputation Files | 14.6 M |
| | Low Reputation Executables | 7.3 M |
| File Appearance Logs | Hosts | 23M |
| | Enterprises | 25 K |
| VirusTotal | Reports | 1.3 M |
| NVD | Client Apps | 12 |
| 01/2015 – 12/2017 | Client CVE | 1,850 |
| | Server Apps | 112 |
| | Server CVE | 988 |
| Internet Scans | Protocols | 8 |
| 10/2015 – 11/2017 | | |
| Blacklists | IP and Domain blacklists | 38 |
| 07/2015 – 12/2017 | | |
| Enterprise-to-IP mapping | Enterprises | 28 K |
| 07/2015 – 12/2017 | | |

Table 5.1: Summary of datasets used.

to analyze externally-facing servers in the enterprises; *blacklists* to identify compromised hosts in the enterprises; and *enterprise-to-IP mapping* to check ownership of IP addresses.

**File reputation logs.** These logs capture metadata about the presence of files in 82M Windows client hosts across 28K enterprises in 137 countries and their corresponding reputation scores. These logs are collected from real client hosts in use by enterprise customers of the cyber security company. The enterprises opted-in to sharing their data and the hosts and enterprises are anonymized to preserve the privacy of the customers. The dataset covers nearly three years from April 2015 to December 2017.

Each host in the collection regularly queries a centralized system to obtain the reputation of files installed in the host. The query includes file metadata such as file hash, file size, and publisher (if the file is signed). The response includes a reputation score that ranges between $128$ and $-127$ with higher (positive) scores indicating good reputation and lower (negative) scores indicating lack of trust. The reputation score is computed using input from different security products and covers a large variety of features including file characteristics, dynamic behaviours, file prevalence, download source, and signer information.

We use this dataset to analyze the presence of malicious files in the enterprises. To

identify malicious files, we first select a subset of 14.6M low reputation files (out of the 326M reported files) with a reputation score less than $-20$. We selected this threshold experimentally to minimize the number of benign files included, while balancing the amount of data to be processed. The low reputation files can be of different types including executables (.exe, .dll, .sys), documents (.pdf, .docx), and archives (.zip, .rar). Overall, out of the 375B reports in the dataset, the low reputation files appear in 135M reports.

In detail, each report in the file reputation logs contains a timestamp, an anonymized enterprise identifier, an anonymized host identifier, a SHA256 file hash, and the file path where the file was installed. For each file hash, the logs also contain reputation score, AV detection label (if the file was flagged as malicious by the cyber security company), and file signer subject and code signature validation result (if the file is signed). For each anonymized host identifier, the logs also contain the Windows version installed (i.e, major and minor OS and Service pack versions).

**File appearance logs.** These logs capture metadata about all executables and archives installed in 23M real hosts belonging to 25K enterprises, a subset of the hosts and enterprises in the file reputation logs. Each event in the dataset can correspond to a download of an executable file or a compressed archive over the network, or the extraction of an executable file from a compressed archive. File appearance logs are very similar to file reputation logs, but differ in that they include all files installed in the host (regardless of their reputation score or potential maliciousness), they are collected from a smaller set of security products, they only include executables and archives, and they provide a more accurate timestamp of the first appearance of the file in the host. Again, the enterprises opted-in to sharing their data and the hosts and enterprises are anonymized to preserve privacy. The file appearance logs contain a timestamp, anonymized enterprise and host identifiers, SHA2 file hash, file signer, file path, and file version fields. We use these logs to identify the presence of specific versions of 12 selected benign client applications in the enterprise hosts.

**Enterprise classification.** Each anonymized enterprise identifier has associated its industry, number of employees, and country they are based in. This information was obtained from an specialized external company. The classification comprises of 67 industries. Table 5.2 shows the number of enterprises, hosts, IP addresses, employees, and country codes for the top 20 industries by number of hosts in the file reputation logs. These top 20 industries cover 65% of the hosts. Banking is the top industry with 16.6M hosts across 1.1K banks in 85 countries, followed by IT services, and healthcare providers. Overall, the dataset shows good industry coverage with 55 (82%) of the industries having at least 100 enterprises and over 100K hosts.

**VirusTotal.** We query the hash of low reputation files in VirusTotal [96] (VT), an online service that analyzes files and URLs submitted by users using a large number of

| Industry | Ent. | Hosts | IPs | Emp. | CC |
|---|---|---|---|---|---|
| Banks | 1.1K | 16.6M | 7.6M | 5.5M | 85 |
| IT Services | 1.0K | 7.5M | 3,500M | 3.0M | 52 |
| Healthcare Providers | 1.1K | 6.5M | 2.9M | 2.3M | 46 |
| Professional Services | 875 | 3.8M | 374.1M | 1.4M | 39 |
| Commercial Services | 1.2K | 3.2M | 366.5M | 2.0M | 49 |
| Insurance | 597 | 3.2M | 1.4M | 1.2M | 52 |
| Capital Markets | 851 | 2.0M | 4.1M | 596K | 55 |
| Software | 832 | 2.0M | 803.8M | 497K | 43 |
| Electronic Equipment | 1.0K | 1.7M | 304.1M | 1.7M | 45 |
| Machinery | 1.4K | 1.5M | 13.3M | 1.6M | 49 |
| Specialty Retail | 601 | 1.5M | 17.9M | 1.6M | 51 |
| Construction & Engineering | 1.3K | 1.1M | 471.9K | 1.3M | 52 |
| Media | 971 | 1.5M | 96.6M | 1.3M | 44 |
| Chemicals | 850 | 1.0M | 1.3M | 909K | 54 |
| Food Products | 846 | 872K | 594.0K | 1.6M | 61 |
| Financial Services | 602 | 827K | 749.1K | 317K | 47 |
| Hotels Restaurants & Leisure | 567 | 752K | 1.2M | 2.8M | 46 |
| Trading Companies | 718 | 714K | 12.4M | 542K | 40 |
| Internet Software & Services | 567 | 572K | 407.8M | 207K | 34 |
| Metals & Mining | 874 | 506K | 1.9M | 1.8M | 56 |

Table 5.2: Number of enterprises, hosts, IPv4 addresses, employees, and country codes for the top 20 industries sorted by number of hosts. The high number of IPs for IT Services is due to that industry including ISPs and hosting providers.

security tools. VT offers a commercial API that given a file hash returns metadata on the file including the list of detection labels assigned by a large number of AV engines used to scan the file. We use the AV labels as input to our malicious file classification. Unfortunately, given the API restrictions, we are only able to collect VT reports for 1.3M low reputation executables, corresponding to 18% of the 7.3M executables found among the 14.6 low reputation files.

**NVD.** We use the National Vulnerability Database (NVD) [174] to obtain the list of vulnerabilities, found between April 2015 and December 2017, in the selected benign client and server applications, For each vulnerability, we use the NVD to obtain the list of application versions affected by the vulnerability.

**Internet scans.** To identify vulnerabilities on servers belonging to the enterprises, we use data from IPv4 Internet-wide scans from Censys.io [179]. The scans were performed on multiple ports between October 2015 and November 2017. We use raw protocol banners from FTP, SSH, SMTP, IMAP(S), POP(S), and HTTP(S) scans. We extract application names and versions from these banners and match them against NVD data to identify vulnerable servers.

**IP and domain blacklists.** We also identify compromised hosts inside the enterprises using archives of 38 public and commercial IP and domain reputation blacklists. These blacklists include, among others, Abuse.ch [180], Cymru's botnet tracking feeds [181], DShield [182], Phishtank [183], ShadowServer [184], Spamhaus DNSBLs [185], and Uceprotect DNSBLs [186]. These blacklists capture different types of malicious behaviors from clients and servers including, among others, spam, botnet infections, malicious server hosting, and brute-force login attacks. Each blacklist is downloaded on a hourly or daily basis depending on its update policy. The archives span 2.5 years between July 2015 and December 2017.

**Enterprise-to-IP mapping.** To analyze blacklisted hosts and vulnerabilities of the externally-facing enterprise servers, we need to identify the public IP addresses an enterprise uses. These include IP addresses allocated to the enterprise, as well as IP addresses leased from cloud hosting providers. We obtain the blocks of IP addresses allocated to an enterprise from the Internet Routing Registries (IRRs). To identify the cloud hosting infrastructure used by an enterprise, we first use domain Whois data to identify domains that have been registered by the company or its subsidiaries. Then we use Rapid7's passive DNS [187] to identify IP addresses that those domains have resolved to. Any IP address that is also a target for a domain from a different enterprise is removed to prevent pollution. To minimize the impact of IP address churn, we recompute the whole enterprise-to-IP mapping every week using archives of the data sources that cover the analysis period. We match blacklists and network scans with the enterprise-to-IP mapping for the corresponding week. We have verified the correctness of our mapping by manually validating it for 100 companies. We selected companies of different sizes and industries to account for potentially different IT administration practices.

### 5.2.1 Selection Bias

Our datasets may introduce selection bias. First, they only include enterprises investing in security products. Enterprises with no security products should have a worse security posture, making our results conservative. Also, our datasets only cover enterprises with security products of a specific vendor and that opted-in to share their data. Products from other vendors may provide different security, and enterprises that opted-out due to privacy concerns could be more security conscious. Furthermore, the file reputation and appearance logs contain only Windows client hosts. Client hosts running other OSes (e.g., macOS, Android) may have a different security posture. To analyze enterprise servers we use blacklists and network scans, but may miss internal servers not facing the Internet.

| Data | Count |
|---|---|
| Low Reputation Files | 14.6M |
| Low Reputation Executables | 7.3M |
| Benign Files | 729K |
| Total VT reports collected | 1.3M |
| Executables with vendor label | 3.3M |
| Total Labeled | 2.0M |
| Total families | 19K |
| Malware Families | 15.5K |
| PUP Families | 3.5K |

Table 5.3: Breakdown of low reputation files.

### 5.2.2 Ethical and Privacy Considerations

The file reputation logs and file appearance logs were collected from enterprises who opted in to sharing their data. Those logs are anonymized to preserve the privacy of the enterprises and their users. They do not contain any identifiable data about the origin of the log entries. Machines and enterprises are referred to using anonymized machine and enterprise identifiers. The outside-in analysis requires the list of enterprise customers of the cyber security company to identify their external-facing IP addresses. That analysis was performed by an employee of the cyber security company and the customer list was not shared with the external authors. To further prevent deanonymization of the enterprises and their users, we present our findings on an aggregated level and on anonymized case studies.

## 5.3 Threat Landscape

This section presents our analysis of the enterprise threat landscape. We start by analyzing the security posture of enterprise client hosts from inside (Sections 5.3.1 through 5.3.5). First, we describe our family classification of malicious files in Section 5.3.1. Then, we analyze the prevalence of malware and PUP (Section 5.3.2) and how specific families are to industries and enterprises (Section 5.3.3). Next, we perform a longitudinal analysis of malware and PUP encounters (Section 5.3.4) and present a case study on the prevalence of ransomware in the enterprises (Section 5.3.5). Finally, we analyze the security posture of enterprises from the outside (including externally-facing servers) in Section 5.3.6.

### 5.3.1 Family Classification

To analyze the most prevalent threats enterprise client hosts face, we identify the malicious files in our dataset and classify them into families. We start with 14.6M low-

reputation files described in Section 6.3. We first filter out the benign files that might have been assigned a low reputation, e.g., due to their low prevalence. This step removes executables signed by benign publishers – using a manually curated whitelist of 948 popular publishers – as well as executables for which a VT report is available and are considered malicious by less than 4 AV engines. As a result, we filter out 729K executables.

Table 5.3 summarizes our classification. Out of the 7.3M executables among the low-reputation files, we collected 1.3M VT reports. Not all 7.3M files were queried to VT due to API restrictions. Among those queried, VirusTotal only knew 27%. This is important because our community largely assumes VT data adequately represents the malware ecosystem.

Our threat classification methodology analyzes the AV labels in the 1.3M VT reports, as well as the labels assigned by the cyber security company, available for another 3.3M executables. We feed the AV labels as input to AVClass [165]. AVClass outputs the most likely family name for each sample and also classifies it as malware or PUP based on the presence of PUP-related keywords in the AV labels (e.g., *adware*, *unwanted*). In addition to the files flagged as PUP by AVClass, we further identified PUP samples by matching their publisher information with 3.8K known PUP publishers. The original version of AVClass is designed to take as input VT reports, which include labels from multiple AV vendors. For the 3.3M executables for which we only have one AV label, we had to modify AVClass by removing the check that requires a family to appear in at least two AV engines to be considered. While no longer using a plurality vote for those 3.3M files, AVClass still enables us to remove noise and generic tokens from the cyber security company's labels. Overall, we labeled 2.1M (29%) executables belonging to 19K families. For the remaining 2.5M samples for which labels were available, no family was identified as their labels were generic.

One advantage of our classification over prior works that classified malware obtained from malware feeds (e.g., [20]) is that we can rank malware families based on their prevalence on real hosts, while samples in malware feeds may be biased towards highly polymorphic families. Table 5.4 shows the top 20 malware and PUP families the enterprises encountered over the analysis period. From the top 20 families, 12 are PUP and 8 are malware families. The most prevalent family is *opencandy*, a well-known commercial pay-per-install service [37], which we observe installed in 1.1M hosts in over 19K enterprises. The most popular malware family is *winactivator*, a label used by AVs for Microsoft Windows crack tools. Activators are found on 34% (9.4K) of all enterprises across all industries. These enterprises have a median size of 490 hosts, although there are also 98 large enterprises (over 100K hosts). Furthermore, 10% of these 9.4K enterprises had *winactivator* installed in over 15% of their client hosts. Further analysis reveals that the majority of the *winactivator* executables belong to KMSPico [178], a popular Microsoft Windows and Office crack tool. The publishers of KMSPico claim that the cracked software can get all the available updates by Microsoft.

| Family | Type | Hosts | Ent. | Files |
|---|---|---|---|---|
| opencandy | pup | 1.1 M | 19.5K | 12.0K |
| winactivator | malware | 470.8K | 9.4K | 5.3K |
| installcore | pup | 453.4K | 17.3K | 54.6K |
| autoit | malware | 398.4K | 6.5K | 12.2K |
| remoteadmin | pup | 333.0K | 8.7K | 1.7K |
| sogou | pup | 282.8K | 2.2K | 813 |
| mictraylog | pup | 264.0K | 3.1K | 21 |
| asparnet | pup | 232.8K | 13.7K | 238 |
| elex | pup | 218.3K | 7.1K | 6.9K |
| donex | pup | 179.0K | 2.0K | 49 |
| dealply | pup | 176.5K | 12.8K | 23.9K |
| nssm | malware | 171.2K | 441 | 41 |
| ramnit | malware | 142.8K | 7.6K | 737.2K |
| qjwmonkey | pup | 142.3K | 2.0K | 281 |
| asprox | malware | 139.7K | 2.1K | 1.4K |
| flystudio | malware | 126.9K | 3.0K | 5.7K |
| conficker | malware | 125.6K | 5.0K | 2.4K |
| spigot | pup | 114.2K | 10.0K | 1.3K |
| fusioncore | pup | 111.2K | 9.3K | 901 |
| ursu | malware | 108.2K | 2.3K | 559 |

Table 5.4: Top 20 families by number of hosts.

| Preval. | | All | Mal. | PUP |
|---|---|---|---|---|
| Lax | Host | 33.6M (41%) | 28.2M (34%) | 6.2M ( 8%) |
| | Ent. | 27.2K (97%) | 26.8K (96%) | 24.8K (89%) |
| Con. | Host | 10.8M (13%) | 8.3M (10%) | 5.2M ( 6%) |
| | Ent. | 25.5K (91%) | 24.5K (87%) | 24.6K (87%) |

Table 5.5: Lax and conservative PUP and malware prevalence estimates.

### 5.3.2 Malware and PUP Prevalence

In this section, we analyze malware and PUP encounters in enterprises. We first establish their prevalence using a lax and a conservative estimate. The lax estimate measures the prevalence of all low reputation files minus the benign files, a total of 13.9M files. The conservative estimate measures the prevalence of only the executables for which we have a VT report and are not benign. Table 5.5 summarizes the prevalence results. We find that using the lax estimate 41% of the hosts and 97% of the enterprises have suffered at least one malware or PUP encounter during the nearly three years analyzed. Using the conservative estimate the numbers are 13% of hosts and 91% of the enterprises. Thus, regardless of the estimate used, the vast majority (91%–97%) of enterprises have suffered at least one malware or PUP encounter. Only 3%–9% of the enterprises never encountered malware or PUP in our analysis period. All these clean

enterprises had less than 100 hosts and the vast majority had only one host. These findings highlight the difficulty of securing enterprises against malicious software. Any reasonable-sized enterprise can be expected to encounter malicious software in three years.

We compare our measured prevalence with prior works. Yen et al. [171] observed that 15% of hosts in a single enterprise encountered malware over a four-month period in 2014. Microsoft reports a 2017 malware encounter rate of 14% in Canada [71], however, without making the distinction between enterprise and consumer hosts. While our conservative estimate is close to those prevalence rates, our lax estimate shows a higher prevalence than those prior works.

The split between malware and PUP shows a higher impact of malware than PUP, with malware affecting 10%–34% of hosts and 87%–96% of enterprises, compared to 6%–8% and 87%–89% for PUP. These findings indicate that both malware and PUP affect the vast majority of enterprises, although malware impacts a larger number of hosts. We find that PUP encounters in enterprises are considerably lower of what is reported in previous works for consumer hosts. Kotzias et al. [37] measured PUP prevalence in consumer hosts for the period Jan 2013 - July 2014 and found out that 54% of the 3.9M analyzed hosts had had some PUP installed. There are four PUP families in Table 5.4 whose prevalence was measured for consumer hosts in [37]. They all show much higher prevalence in consumer hosts: *opencandy* (8% in consumer hosts vs 1.3% in enterprise hosts), *installCore* (8.5% vs 0.55%), *dealply* (2.8% vs 0.2%), and *spigot* (2.6% vs 0.1%). This confirms that PUP is significantly less prevalent in enterprises. This could be due to stricter security policies about what programs can be installed applied by enterprises, which may affect PUP, but not malware (since PUP typically requires user acceptance for installation). Other explanations could be differences on the awareness of corporate users and the different time period of the two studies, as prior work shows PUP prevalence dropping at the end of 2015 [20].

Figure 5.1 shows the cumulative distribution of the number of distinct families observed per host with at least one encounter. Nearly 50% of the hosts are only affected by one family and 75% by less than 5 families. The fact that 25% of hosts encountered more than 5 families is surprising and can be due to pay-per-install relationships [37, 56] or to machines that are periodically re-infected.

**Industry prevalence.** Table 5.6 presents the top ten (most-affected) and bottom ten (least-affected) industries ranked by malware and PUP prevalence. There is a significant difference between both groups of industries. The most-affected industries have 76%–69% of hosts affected, while the least-affected industries have 36%–15%. That is, the ten most-affected industries have more than twice the prevalence of malware and PUP compared to the least-affected ten industries. This shows that there are industries that take security more seriously than others. Four of the ten least-affected industries are finance-related including Banks and Consumer Finance, which are the two least-affected industries. This matches reports that banking is the industry that invests the most in cyber security products [38]. However, note that Banks have the most

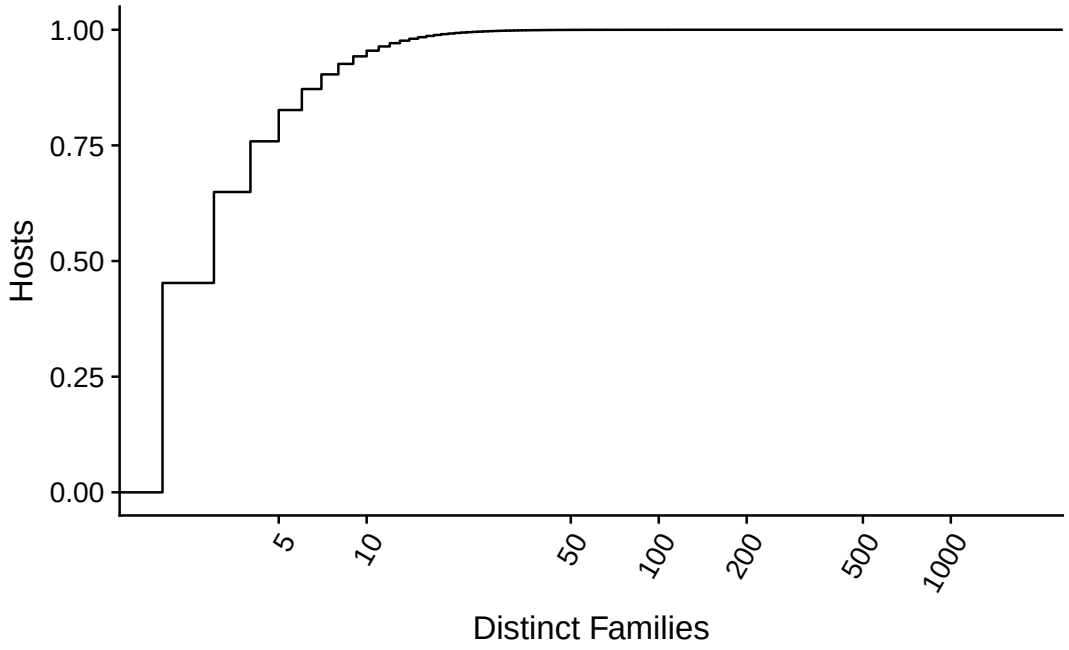



Figure 5.1: Number of families per host

hosts in our dataset, thus a prevalence of 16% represents over 2.5M encounters. The most-affected industries are Electrical Equipment, Automobiles, and Construction Materials. In general, this group seems dominated by industries related to manufacturing and consumer products.

### 5.3.3 Malware and PUP Specificity Analysis

We rank the top 20 PUP and malware families affecting each industry, whose union comprises of 221 malware and 86 PUP families. We observe a few families that appear in the top 20 of almost all industries. These include both malware (`winactivator`, `ramnit`, `autoit`) as well as PUP (`dealply`, `installcore`, `spigot`, `amonetize`, `opencandy`, `asparnet`, `remoteadmin`). On the other hand, 117 of the malware families and 57 of the PUP families were not found on 90% of the industries. This is an interesting observation showing that there are many PUP and malware families only seen in one or a small number of industries. Furthermore, 17 malware families were found only on one industry and those families were on that industry's top 3 malware list. For example, the remote access trojan `xtrat` is only in the top 20 malware of the Construction and Engineering industry, but encountered in 2% (22K) of those hosts.

We perform the same investigation at per-enterprise level to identify families targeting specific enterprises. The number of malware families seen in only one enterprise is 1,911 (37%), while for PUP is 446 (26%). Thus, the specificity of malware families

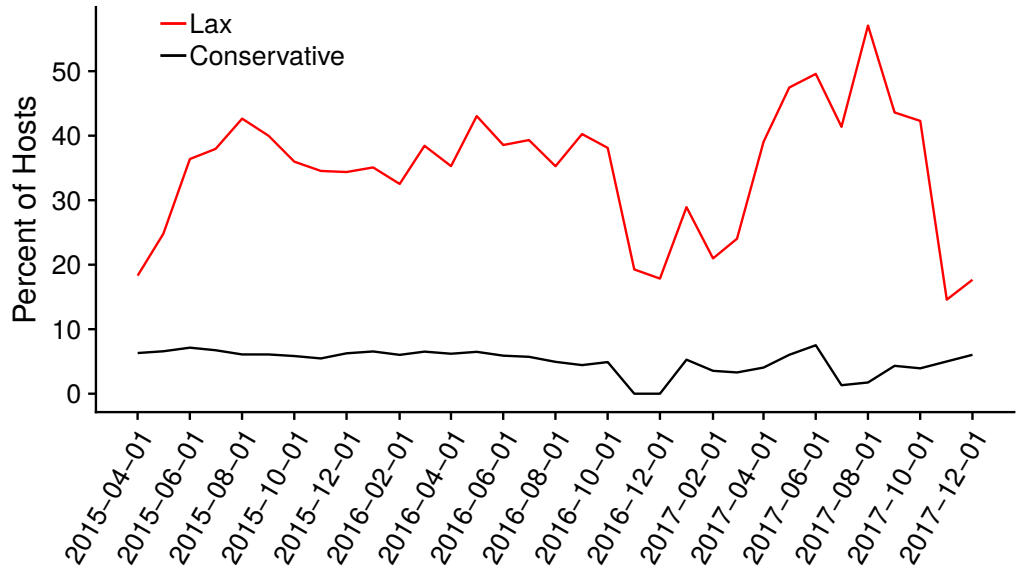
| Industry | All | Mal. | PUP |
|---|---|---|---|
| Electrical Equipment | 76.4% | 69.7% | 22.0% |
| Automobiles | 75.5% | 70.4% | 13.9% |
| Construction Materials | 74.4% | 66.7% | 18.5% |
| Marine | 74.3% | 67.4% | 30.8% |
| Semiconductors | 72.9% | 66.8% | 19.9% |
| Industrial Conglomerates | 72.8% | 67.5% | 26.0% |
| Communications Equipment | 71.3% | 63.8% | 22.0% |
| Healthcare Equipment | 70.8% | 64.1% | 15.6% |
| Leisure Products | 69.4% | 60.6% | 11.5% |
| Beverages | 69.3% | 61.0% | 10.9% |
| Thrifts and Mortgage Finance | 36.5% | 30.2% | 8.3% |
| Diversified Financial Services | 35.7% | 30.3% | 4.6% |
| Specialty Retail | 35.3% | 29.6% | 6.4% |
| Healthcare Providers | 34.0% | 27.4% | 3.1% |
| Professional Services | 32.6% | 27.2% | 4.5% |
| Real Estate | 31.8% | 25.7% | 2.4% |
| Wireless Telecommunication | 28.6% | 23.3% | 6.6% |
| Biotechnology | 20.5% | 15.1% | 1.1% |
| Consumer Finance | 15.9% | 11.4% | 1.9% |
| Banks | 15.7% | 13.6% | 1.2% |

Table 5.6: Top 10 (most affected) and bottom 10 (least affected) industries by malware and PUP prevalence.

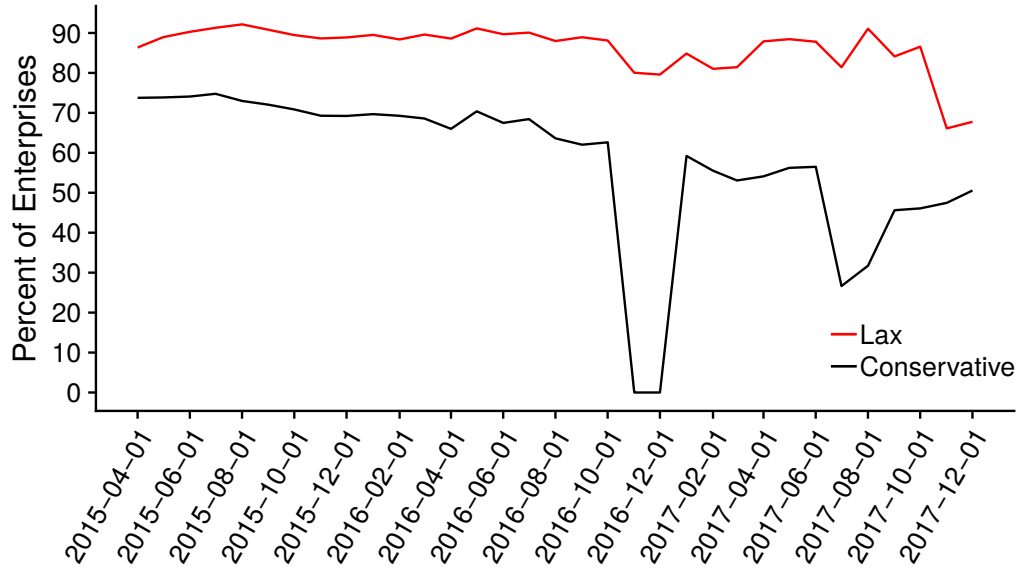
is higher than for PUP families. On the other hand, in contrast with the industry-based results, we do not observe any malware or PUP family encountered in the majority of enterprises. Among the enterprise-specific malware families, 78 are ranked as the top malware family encountered in that enterprise, which may indicate targeting. Of those 78, 13 affect a large enterprise. One example is the `zcrypt` ransomware. It affected only one enterprise in our dataset (from the Hotels, Restaurants, and Leisure industry). In conclusion, we find a significant number of enterprise-specific malware families and observe indications of targeting for 78 families.

### 5.3.4 Longitudinal Analysis

Figure 5.2 shows the monthly encounter rate using the conservative and lax estimates. The percentage of hosts that encountered malware (Figure 5.2a) using the conservative estimate does not change drastically over the years, remaining on average around 7%. On the other hand, we observe larger fluctuations on the lax estimate. Between mid-2015 and the end of 2016, the monthly encounter rate was stable around 30–40%. Then, in November 2016 the percentage drops by approximately 20%. In May 2017, the monthly encounter rate increases drastically reaching over 50% in August. Finally, in November 2017 it drops by 25%. The two large drops one year apart are also visible




(a) By hosts




(b) By enterprises.

Figure 5.2: Monthly malware and PUP prevalence by number of hosts and enterprises with at least one encounter.

| Family | Hosts | Ent. | Ind. | Files |
|---|---|---|---|---|
| wannacry | 30.1K | 872 | 65 | 2.2K |
| locky | 20.3K | 5.2K | 67 | 4.6K |
| petya | 11.2K | 155 | 46 | 72 |
| ransomkd | 10.2K | 1.1K | 66 | 70 |
| teslacrypt | 9.4K | 2.9K | 66 | 5.9K |
| cryptolocker | 8.7K | 1.7K | 66 | 714 |
| cerber | 6.1K | 2.2K | 66 | 1.7K |
| cryptowall | 2.6K | 1.4K | 66 | 359 |
| dcryptor | 2.0K | 468 | 59 | 36 |
| torrentlocker | 785 | 443 | 62 | 207 |
| All | 103K | 8.8K | 67 | 16K |

Table 5.7: Top 10 ransomware families by number of hosts.

in the percentage of enterprises encountering malware (Figure 5.2b). The reasons for these two large malware encounter drops remain unknown. We checked that the abrupt increase in mid-2017 is not due to *wannacry* and *petya* that emerged during that time (see Section 5.3.5). However, it could be due to other malware families exploiting the same EternalBlue vulnerability.

### 5.3.5 Case Study: Ransomware

In this section we present a case study on the prevalence of ransomware in enterprise networks. There are 28 ransomware families among the classified low reputation executables. In total, we identify 103K hosts in 8.8K enterprises across all 67 industries affected by ransomware. This is a pretty low prevalence of 0.02%. Assuming an average ransom payment of $500, these encounters amount to a modest $51.5M in direct costs, plus possibly an order of magnitude larger indirect costs including remediation [188].

Table 5.7 presents the top 10 ransomware families by number of affected hosts. `wannacry` and `locky` are the ransomware families found in most hosts, 30K and 20K, respectively. `wannacry` only ranks seventh in terms of enterprises affected due to the fact that its worm-like behavior exploited the previously known *Eternal Blue* SMB vulnerability that was patched for Windows 7 and above. Thus, it likely only affected enterprises with hosts using earlier Windows versions, but spread quickly within those enterprises. In fact, from the 12K enterprises with at least one Windows XP host, 50% experienced a ransomware attack, while the average encounter rate was only 31%.

Most ransomware families are found in the majority of industries, indicating that ransomware operators currently do not target specific industries. Figure 5.3 shows the monthly number of hosts and enterprises with ransomware encounters in our analysis period. We observe the first large peak on March 2016 where affected hosts reach

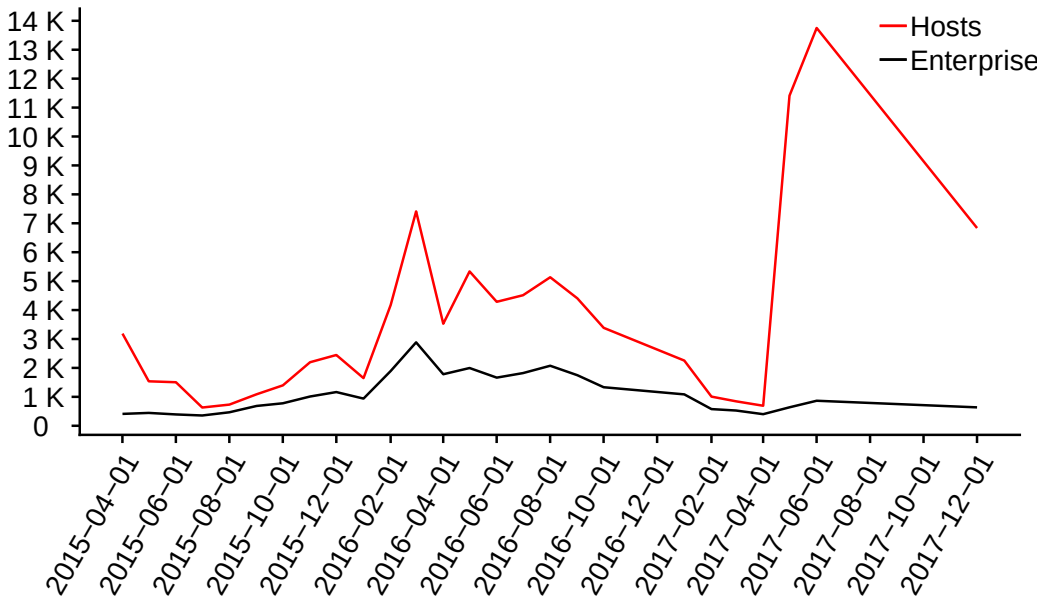



Figure 5.3: Monthly number of hosts and enterprises with ransomware appearances.

7K and affected enterprises 3K. This peak is mostly due to `locky`. The second and largest peak occurs on June 2017 and is due to `wannacry` and `petya`. It affects more than 13K hosts, but (as explained above) the number of affected enterprises does not significantly increase. Note however that this peak in June 2017 is much smaller than the one observed in Figure 5.2 for all malware encounters. Thus, these two families by themselves cannot explain that large increase.

### 5.3.6 Outside-in Perspective

In this section, we look at the enterprise threat landscape from an *outside-in* perspective. We extract symptoms of malware encounters inside an enterprise by (i) uncovering all public, Internet-facing IP addresses owned or used by an enterprise and (ii) correlating them with datasets of external indicators of compromise (IoCs), essentially blacklists of Internet hosts associated with different types of malicious activity (e.g., spam senders, C&C servers of known botnets, malware distributing and phishing web servers).

Table 5.8 presents the breakdown of the malicious activity observed from the top and bottom 10 industries in terms of number of blacklisted IP addresses. The first big trend we can observe from the blacklisted hosts inside enterprises is that, as of today, *spam* is still the predominant type of malicious activity sourced by allegedly compromised machines. In most companies of most industries, spam largely dominates any other type of malicious activity. This phenomenon can be in part explained by the fact

| Industry | Ent. | Botnet Infect. | Brute-f. Logins | C&C Host. | Malware Distri. | Phish. | Scan. | Spam | Total |
|---|---|---|---|---|---|---|---|---|---|
| Media | 598 | 108.8K | 1.8K | 3.5K | 30.0K | 6.9K | 949 | 610.7K | 797.2K |
| Communications Equipment | 141 | 13.6K | 55 | 763 | 2.8K | 1.3K | 14 | 456.5K | 486.5K |
| Software | 543 | 7.5K | 1.8K | 3.4K | 9.2K | 6.2K | 581 | 336.5K | 391.5K |
| Technology Hardware | 129 | 5.0K | 1.4K | 2.3K | 8.4K | 5.2K | 533 | 222.4K | 275.4K |
| IT Services | 675 | 6.8K | 1.8K | 8.1K | 25.0K | 13.0K | 194 | 94.9K | 220.7K |
| Internet Software and Services | 371 | 7.0K | 1.9K | 3.1K | 16.8K | 13.7K | 361 | 85.8K | 146.9K |
| Electronic Equipment | 563 | 2.0K | 46 | 1.8K | 4.7K | 3.4K | 13 | 116.8K | 136.3K |
| Diversified Consumer Services | 118 | 8.5K | 75 | 574 | 4.6K | 849 | 43 | 90.9K | 107.2K |
| Commercial Services | 756 | 502 | 156 | 2.6K | 4.5K | 3.8K | 16 | 35.0K | 56.0K |
| Professional Services | 614 | 764 | 159 | 3.4K | 7.4K | 5.0K | 14 | 10.5K | 38.2K |
| Multi-Utilities | 17 | 194 | 3 | 51 | 101 | 59 | 0 | 688 | 1.2K |
| Airlines | 47 | 20 | 2 | 119 | 268 | 226 | 0 | 329 | 1.1K |
| Construction Materials | 100 | 6 | 0 | 170 | 263 | 260 | 0 | 263 | 1.1K |
| Paper and Forest Products | 92 | 14 | 1 | 132 | 255 | 215 | 0 | 252 | 1.0K |
| Transportation Infrastructure | 80 | 14 | 2 | 53 | 156 | 125 | 0 | 194 | 628 |
| Multiline Retail | 13 | 58 | 0 | 23 | 53 | 53 | 1 | 127 | 349 |
| Gas Utilities | 34 | 2 | 0 | 46 | 85 | 68 | 0 | 76 | 332 |
| Marine | 38 | 4 | 0 | 33 | 70 | 60 | 0 | 86 | 299 |
| Water Utilities | 25 | 1 | 0 | 40 | 68 | 61 | 0 | 68 | 284 |
| Tobacco | 5 | 2 | 0 | 45 | 55 | 53 | 0 | 62 | 257 |

Table 5.8: Breakdown of malicious activity exhibited by industries (top 10 and bottom 10).

that spam is heavily monitored and might be easier to detect than machines hosting malware or C&C servers inside an enterprise. The high prevalence of *malware*, *phishing* and *C&C server hosting* highlights the serious threat that compromised machines inside enterprises can pose.

Comparing the top and bottom 10 industries from the perspective of malware encounters (Table 5.6) and blacklisted hosts (Table 5.8), we notice some obvious differences. Only one industry – Communications Equipment – appears both in the top 10 malware industries and the top 10 blacklisted hosts industries. Moreover, two industries – Construction Materials and Marine – are found in the top 10 malware industries and in the bottom 10 blacklisted hosts industries. There are two likely reasons for this: (i) blacklists have limited visibility into much malware encountered in enterprise client hosts, and (ii) most malware do not exhibit external IoCs captured by blacklists.

The effectiveness of blacklists for operational threat detection has already been extensively studied in previous works [20, 189–193]. Some of these studies have also assessed the quality of blacklists. The general take-away message from these prior studies is that *the quality of blacklists can vary drastically from one another* so care should be taken when selecting them, *more specialized datasets should be preferred, when available*, and that *despite their limitations, they remain a useful source of malicious activity*. As we have seen in our results here-above, IP- and domain-based blacklists can be useful to provide a general trend on the security posture of an enterprise and, by extension, the industry it belongs to. However, to study malware encounters in enterprises, we can see that blacklists cannot match the granularity and accuracy of more specialized datasets, such as the file appearance logs used in Section 5.3. We understand that, in the absence of other datasets, blacklists may be the only source to

study the (enterprise) threat landscape. However, care should be taken when deriving conclusions solely based on blacklists.

## 5.4 Vulnerability Patching Behavior

In this section we analyze the presence of vulnerabilities and their patching behavior in enterprise networks, which prior work has shown to be fairly correlated with future security incidents [74, 172]. In particular, we study vulnerability patching practices carried out by different industries. Our goal is to understand the security posture of the enterprises and whether particular industries are less or more agile to patch their vulnerabilities and therefore, are less or more secure against cyber threats. We conclude the section with an analysis of OS upgrade behavior in enterprises.

We analyze both vulnerabilities in client and server applications. In Section 5.4.1 we use the file appearance logs to analyze the patching speed of 12 popular client-side applications in the enterprise hosts. In Section 5.4.2 we use periodic IPv4 Internet-wide scans to analyze the patching speed of vulnerabilities in 112 server applications and libraries. Since these are externally-facing servers installed in the enterprises, they are easier to be discovered by the attackers and are greatly exposed to external threats.

| | | | | | | Unpatched | Ent. PT | | Cons. PT [5] | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Program** | **Vendor** | **Versions** | **Hosts** | **Ent.** | **CVE** | **Hosts** | **50%** | **90%** | **50%** | **90%** |
| Chrome | Google | 267 | 10.2M | 24K | 454 | 1.7M | 18 | 78 | 15 | 246 |
| Firefox | Mozilla | 205 | 4.2M | 20K | 308 | 1.1M | 25 | 161 | 36 | 179 |
| Thunderbird | | 10 | 159K | 6K | 40 | 15K | 23 | 98 | 27 | 129 |
| Skype | Microsoft | 41 | 1.1M | 18K | 2 | 8K | 17 | 89 | - | - |
| Internet Explorer | | 1K | 15.8M | 24K | 428 | 11M | 47 | 138 | - | - |
| .NET | | 197 | 8.5 M | 22K | 21 | 2.5M | 60 | 162 | - | - |
| Silverlight | | 43 | 8.9M | 23K | 17 | 5M | 82 | 182 | - | - |
| Media Player | | 141 | 9.5M | 23K | 1 | 7.5M | 147 | 314 | - | - |
| JRE | Oracle | 340 | 5.7M | 22K | 21 | 1.4M | 56 | 141 | - | - |
| Air | Adobe | 11 | 1.2M | 15K | 316 | 216K | 44 | 152 | - | - |
| Reader | | 47 | 13.9M | 23K | 221 | 6.2M | 78 | 234 | 188 | 219 |
| MariaDB | - | 35 | 13.5K | 1K | 53 | 3K | 75 | 246 | - | - |
| | TOTAL | 2K | 23M | 25.4K | 1.8K | AVG | 67 | 200 | | |

Table 5.9: Client application patching summary. It shows the number of application versions, the number of hosts and enterprises where the application was installed, the number of vulnerabilities analyzed, the number of hosts unpatched at the end of the analysis, the 50% and 90% enterprise patch time in days measured in this work, and the 50% and 90% consumer patch time in days measured in previous work [5].

### 5.4.1 Analysis of client-side vulnerabilities.

Our analysis of client-side vulnerabilities focuses on 12 client applications and frameworks: .NET, Adobe Air, Adobe Reader, Chrome, Firefox, Internet Explorer, Java Runtime Environment (JRE), MariaDB, Silverlight, Skype, Thunderbird, and Windows Media Player. We selected these 12 applications because they are popular; they

cover both stand-alone applications (e.g., Chrome, Adobe Reader) and frameworks (.NET, JRE); they include proprietary programs from five vendors (Adobe, Google, Microsoft, Mozilla, Oracle) and one open-source application (MariaDB); their executables are signed; and they embed the program version in their executables.

To identify the presence of these applications on the enterprise hosts, we follow a methodology similar to that proposed by Nappa et al. [5]. We first identify the main executable for each application (e.g., firefox.exe), then we examine the file appearance logs to obtain the hashes and file versions of all executables with that name and signed by the right publisher (e.g., Mozilla). This step outputs for each application, a mapping from file hash to the application version corresponding to that hash. Using this mapping we can identify hosts in the file appearance logs where those versions were installed, as well as their installation time. We then use the NVD to obtain the vulnerabilities, disclosed between April 2015 and December 2017, in those 12 applications and the list of vulnerable program versions for each vulnerability.

For each vulnerability we compute the *patch time*, i.e., the time needed to patch a certain fraction (50% and 90% in this work) of the vulnerable hosts. To compute the patch time, we exclude hosts that never patched a vulnerability, e.g., because they left the population.

Table 5.9 summarizes the client application patching results. The left side of the table captures, for each application, the name, the vendor, the number of versions identified, the number of hosts with one of those versions installed, the number of enterprises those hosts belong to, and the number of vulnerabilities analyzed. Overall, we analyze 1,882 vulnerabilities, of which 50% are critical (CVSS $\geq$ 9) and 90% have high impact (CVSS $\geq$ 7). The most popular application is Internet Explorer installed in 69% of the hosts in the file appearance logs, followed by Adobe Reader (60%), and Chrome (44%). Eight of the 12 applications are installed in over 20K enterprises highlighting their popularity.

The middle part of the table summarizes our enterprise patching measurements. It shows the average number of hosts that never patched and the average time in days to patch 50% and 90% of the vulnerable hosts. The results show that Chrome is the fastest application being patched requiring on average 18 days to patch 50% of the vulnerable hosts and 78 days to patch 90%. On the other hand, the slowest application is Windows Media Player which takes nearly 5 months to patch 50% of the vulnerable hosts and over 10 months to patch 90%. Overall, it takes over 6 months on average to patch 90% of the population across all applications and vulnerabilities, highlighting the limitations of patch deployment in enterprises.

The right side of the table shows the patch time reported by Nappa et al. [5] in their analysis of 8.4M consumer hosts. We use a similar methodology to that work and examine four applications in common: Chrome, Firefox, Thunderbird, and Adobe Reader. The comparison shows that three of the four applications (Chrome, Firefox, Thunderbird) reach 90% patching faster in enterprises and another three (Firefox, Thunderbird, Reader) reach 50% patching also faster in enterprises. These results seem to indicate that enterprises are on average faster to apply patches than consumers.

One caveat is that the period of analysis differs between both works, 2008–2013 for the work on consumer hosts and 2015–2017 in this work. There may be different reasons behind the improvement in patching in enterprise hosts including enterprises being more security aware, having deployed security software that may detect the need to update, having teams dedicated to securing their hosts, or that enterprise hosts may be online more often than consumer hosts (enabling the patches to be downloaded earlier).

Table 5.10: Industry ranking of vulnerability patching time (in days).

| | | IE | | Chrome | | Adobe Reader | | Firefox | | JRE | | All Apps | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Rank** | **Industry** | **50%** | **90%** | **50%** | **90%** | **50%** | **90%** | **50%** | **90%** | **50%** | **90%** | **50%** | **90%** |
| 1 | Communications Equipment | 42 | 91 | 20 | 71 | 85 | 201 | 20 | 111 | 73 | 148 | 53 | 152 |
| 2 | Consumer Finance | 45 | 123 | 18 | 67 | 72 | 193 | 20 | 126 | 67 | 156 | 52 | 152 |
| 3 | Diversified Financial Services | 46 | 142 | 14 | 67 | 71 | 191 | 27 | 119 | 50 | 140 | 56 | 164 |
| 4 | Diversified Telecommunication Services | 48 | 134 | 17 | 70 | 90 | 247 | 22 | 130 | 30 | 127 | 49 | 141 |
| 5 | Capital Markets | 47 | 120 | 22 | 48 | 81 | 228 | 25 | 133 | 77 | 157 | 64 | 159 |
| 6 | Software | 47 | 140 | 15 | 66 | 63 | 196 | 20 | 118 | 51 | 155 | 55 | 160 |
| 7 | Trading Companies and Distributors | 50 | 138 | 17 | 76 | 88 | 210 | 23 | 155 | 63 | 104 | 56 | 152 |
| 8 | IT Services | 42 | 114 | 18 | 78 | 69 | 214 | 24 | 152 | 31 | 109 | 60 | 156 |
| 9 | Health Care Technology | 46 | 147 | 18 | 78 | 62 | 172 | 23 | 144 | 61 | 149 | 49 | 133 |
| 10 | Diversified Consumer Services | 46 | 124 | 16 | 67 | 64 | 185 | 29 | 159 | 78 | 195 | 65 | 151 |
| 57 | Containers and Packaging | 54 | 143 | 14 | 74 | 93 | 300 | 27 | 188 | 53 | 167 | 58 | 183 |
| 58 | Multi-Utilities | 54 | 136 | 17 | 74 | 71 | 470 | 58 | 306 | 63 | 159 | 62 | 210 |
| 59 | Road and Rail | 47 | 132 | 16 | 86 | 95 | 266 | 32 | 204 | 42 | 161 | 64 | 184 |
| 61 | Real Estate Management and Development | 50 | 177 | 18 | 79 | 80 | 295 | 22 | 163 | 39 | 162 | 55 | 178 |
| 62 | Textiles, Apparel and Luxury Goods | 52 | 187 | 16 | 76 | 78 | 259 | 24 | 177 | 58 | 183 | 63 | 178 |
| 63 | Industrial Conglomerates | 58 | 201 | 15 | 75 | 82 | 281 | 22 | 172 | 78 | 177 | 65 | 196 |
| 64 | Air Freight and Logistics | 52 | 174 | 20 | 80 | 95 | 371 | 31 | 240 | 63 | 150 | 58 | 185 |
| 65 | Gas Utilities | 60 | 187 | 22 | 93 | 108 | 256 | 31 | 179 | 70 | 162 | 68 | 197 |
| 66 | Construction Materials | 49 | 169 | 18 | 100 | 107 | 341 | 36 | 189 | 84 | 193 | 66 | 187 |
| 67 | Multiline Retail | 60 | 276 | 15 | 78 | 55 | 256 | 32 | 251 | 88 | 219 | 61 | 193 |

**Client patching by industry.** Table 5.10 ranks the top and bottom ten industries by vulnerability patching time. We provide detailed patching time (50% and 90%) for the five client applications that are installed the most on enterprise hosts. We also provide results averaged across all applications. The results are obtained by cummulating the ranking for each application and ordering the list over the cumulative ranking. As it can be seen, the industries that invest the most in cyber security products, and encounter higher amount of malware in count (not in percentage), such as finance, software and communications are considerably faster at patching their vulnerable applications. On the other hand, the industries whose majority of machines encounter malware are worst at patching their vulnerabilities on a timely manner making the window of their exposure to cyber threats larger. Seeing industries such as gas and electricity utilities in the bottom part of the list is especially worrisome as successful attacks in this kind of industries could have physical impacts. When we perform a similar analysis on the percentage of unpatched hosts and the length of their vulnerability windows in each industry, we obtain different rankings. While 90% of the machines from the top best (i.e. Banks, Household Products, Multi-Utilities) industries remain vulnerable for an average of four months, hosts from the bottom of the list (i.e, Tobacco, Multiline Rail, Energy Equipment and Services, and Marine) remain vulnerable for 15 months.

**Disabling automatic updates.** We examine whether enterprises may have disabled the auto-update functionality of applications, which is mandatory but can be disabled through configuration options. For this, we compute the average time it takes each host to install a new version of applications (for all versions, not only the vulnerable ones). Then we examine the distribution across hosts to identify outlier hosts that update applications slower. For this, we first calculate the median of each application's update speed distribution. To identify outliers, we calculate the absolute deviation which was proposed as an optimal way for outlier detection [194]. Using the absolute deviation, hosts that on average take more than $median + absolute\_deviation$ days to update their apps are considered outliers. We then look for enterprises for which the majority of hosts are outliers, which would indicate an enterprise-wide policy to disable auto-updates. We only find a limited number of enterprises that satisfy that condition. For example, for Chrome we found two, for Adobe Reader four and for Firefox only one enterprise where more than 75% of the machines were outliers. Thus, disabling auto-updates on the client applications we analyzed is in general a rare policy.

**Best and worst patchers** We identify the best and the worst patchers in our data to compare their malware encounter rate with their patching behavior. We choose enterprises that have at least 1000 machines for this measurement. The top 10 enterprises that patch their vulnerable applications the fastest patch 90% of their machines in less than 10 days. Note that here we take the average patch time for all of the applications we analyzed in our study. On the other side of the scale, the 90% patch time of the worst patchers is 500 days on average. While the best patcher is an enterprise from the Hotels, Restaurants and Leisure industry that patches most hosts in only 5 days, the remaining best patchers are from the Financial and Insurance industry. The worst ones are from the Capital Market, Media, Speciality Retail, Textiles, Apparel and Luxury Goods and Healthcare. Having the worst patcher from an industry which was ranked as the 5th best in patching and the best patcher from an industry that is ranked as the 4th worst illustrates the big variation in patching behavior among companies. We also looked at the malware prevalence in these enterprises and found out that the worst patchers encounter more malware compared to the best patchers. This simple investigation on the best and worst patchers supports that patching applications on time has a significant effect on the number of malware encounters.

### 5.4.2 Analysis of server-side vulnerabilities

In this section we analyze the patching of vulnerabilities in servers belonging to the 28K enterprises. Each server corresponds to an IP address and may run multiple services on different ports. Each service is an instance of one of the 112 server software packages analyzed. To identify the specific software and version of a service, we use a set of 2,664 regular expressions that are applied on the protocol banners collected through Internet-wide scans. One difference with the client application analysis is that here we do not know the exact timestamp when a service was updated. Instead, we

approximate it with the time when the new version is first observed, which happens later as scans take place with at most daily granularity.

In the remainder of this section we use the same metrics to measure the patching behavior of enterprise servers than those we used for client machines, i.e., the *patch time* measuring the time it takes for a vulnerable server application to be updated once a patch is released, and the *vulnerability window* defining the time period during which a server application remains vulnerable to a known vulnerability.

| **Property** | **Count** | **Avg** | **50%** | **90%** |
|---|---|---|---|---|
| Servers | 73.1M | - | - | - |
| Vul. servers | 17.9M | - | - | - |
| Patched at least once | 16.4M | - | - | - |
| Never patched | 1.5M | - | - | - |
| CVEs/server | - | 6.82 | 5.00 | 15.00 |
| CVE CVSS score | - | 5.42 | 5.00 | 7.60 |

Table 5.11: Summary of the server-side applications vulnerability assessment. These results are computed for the 28 K enterprises and the 112 server-side applications.

**Overview.** Table 5.11 provides an overview of the server vulnerability analysis results. These results are computed for the 28K enterprises and the 112 server-side applications. Out of 73.1M servers mapped to the 28K enterprises, 17.9M have had at least one vulnerable service. On average, each server is affected by more than six vulnerabilities. Even more worryingly, at least 10% of vulnerable servers are affected by more than 15 vulnerabilities. One important observation is that 1.5M servers in 11,905 enterprises have never been upgraded throughout the 2.5 years analysis period.

| | | | **Vulnerable** | | | **Patch Time** | | | **Avg. Vul.** |
|---|---|---|---|---|---|---|---|---|---|
| **Rank** | **Program** | **Service** | **Mach.** | **Ent.** | **CVEs** | **Avg.** | **50%** | **90%** | **Window** |
| 1 | OpenSSH | SSH | 4.5M | 11K | 84 | 96 | 22 | 317 | 132 |
| 2 | Apache Httpd | HTTP | 2.7M | 11K | 182 | 108 | 24 | 323 | 165 |
| 3 | Microsoft IIS | HTTP | 2.7M | 14K | 22 | 140 | 32 | 552 | 208 |
| 4 | Lighttpd | HTTP | 1.1M | 908 | 26 | 78 | 15 | 233 | 88 |
| 5 | vsftpd | FTP | 825K | 2K | 5 | 59 | 7 | 216 | 89 |
| 6 | mini_httpd | HTTP | 811K | 349 | 2 | 89 | 15 | 253 | 111 |
| 7 | Nginx | HTTP | 414K | 5K | 14 | 175 | 162 | 346 | 191 |
| 8 | ProFTPD | FTP | 267K | 2K | 27 | 70 | 7 | 287 | 106 |
| 9 | Apache Coyote | HTTP | 208K | 3K | 1 | 168 | 71 | 575 | 241 |
| 10 | Exim | SMTP | 52K | 2K | 13 | 135 | 16 | 480 | 211 |
| | | | | Total 112 Apps. | | 108 | 56 | 282 | 230 |

Table 5.12: Summary of the server-side applications and patching behavior of the enterprise servers. Results per application are given for the top 10 vulnerable applications in number of affected servers. The average, 50%, 90% patch time and the average vulnerability window are also provided for the total 112 server-side applications.

**Server patching by application.** Table 5.12 presents the top 10 vulnerable server software in terms of number of vulnerable servers found. The table is dominated by popular SSH and Web servers. The top 3 server software (OpenSSH, Apache, IIS) had at least one vulnerable version installed on over 2.5M servers across more than 10K enterprises. On average, for the 112 server programs, it takes eight weeks (56 days) to patch 50% of the servers and over nine months (282 days) to patch 90%. While the server patch time for 50% is slightly shorter (56 days) that the patch time for 50% of the client applications (67 days), when considering 90% of the servers, it is almost 50% worse than the 200 days (90%) observed on the 12 client applications. One possible reason for the slower server patching is the lack of automatic updates on server programs. There is a stark contrast between the 50% and 90% patch time. While seven of the top 10 software have a 50% patch time of 24 days or lower (significantly better than the average), their 90% patch time is 10 months (the average). Thus, even for the most popular server software it is hard for enterprises to fully deploy patches. Finally, only 2.2K out of 28K (7.9%) enterprises have a 50% patch time below or equal to the average 50% patch time across all applications (56 days). While a significant fraction of the enterprises are diligent in patching their servers, the rest are quite slow making it very hard to completely eliminate a vulnerability. This situation creates points of entry for cyber-criminals to penetrate corporate networks by leaving Internet-facing servers vulnerable for very long periods of time.

Table 5.13: Industry ranking of server-side applications vulnerability patching time (in days). Blank fields indicate industries in which a server-side application was not found.

| | | | FTP | | SSH | | SMTP | | HTTP(S) | | POP(S) | | IMAP(S) | | All Apps | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Rank** | **Industry** | **Machines** | **50%** | **90%** | **50%** | **90%** | **50%** | **90%** | **50%** | **90%** | **50%** | **90%** | **50%** | **90%** | **50%** | **90%** |
| 1 | Multi-Utilities | 1.0K | 18 | 98 | 183 | 435 | | | 70 | 322 | 340 | 340 | | | 78 | 412 |
| 2 | Communications Equipment | 77.8K | 57 | 433 | 155 | 543 | 127 | 705 | 155 | 695 | 377 | 532 | 319 | 535 | 159 | 679 |
| 3 | Thrifts and Mortgage Finance | 262 | 119 | 274 | 211 | 492 | 69 | 236 | 200 | 698 | | | | | 162 | 705 |
| 4 | Beverages | 596 | 130 | 619 | 316 | 695 | 188 | 724 | 169 | 668 | | | | | 171 | 695 |
| 5 | Automobiles | 4.0K | 64 | 473 | 237 | 561 | 209 | 510 | 188 | 678 | 546 | 695 | 179 | 280 | 172 | 659 |
| 6 | Technology Hardware | 889.9K | 137 | 621 | 155 | 540 | 226 | 390 | 183 | 660 | 169 | 629 | 39 | 593 | 172 | 660 |
| 7 | Electric Utilities | 22.8K | 109 | 657 | 218 | 590 | 124 | 392 | 200 | 582 | 182 | 468 | 114 | 205 | 181 | 589 |
| 8 | Multiline Retail | 222 | 74 | 481 | 114 | 342 | 127 | 127 | 190 | 714 | 557 | 557 | | | 190 | 709 |
| 9 | Food and Staples Retailing | 2.6K | 32 | 623 | 295 | 603 | 176 | 436 | 184 | 705 | 176 | 310 | | | 196 | 705 |
| 10 | Internet Software and Services | 665.4K | 155 | 674 | 174 | 574 | 134 | 595 | 200 | 674 | 196 | 716 | 148 | 588 | 199 | 674 |
| 58 | Construction Materials | 196 | 151 | 496 | 310 | 543 | 188 | 226 | 234 | 590 | 134 | 303 | 141 | 141 | 226 | 610 |
| 59 | Electrical Equipment | 1.5K | 134 | 736 | 285 | 579 | 58 | 472 | 200 | 599 | 328 | 479 | 550 | 550 | 230 | 606 |
| 60 | Internet and Catalog Retail | 2.8K | 137 | 428 | 157 | 543 | 71 | 520 | 238 | 705 | | | | | 233 | 704 |
| 61 | Containers and Packaging | 1.1K | 36 | 369 | 317 | 691 | 188 | 473 | 181 | 614 | 408 | 670 | | | 237 | 691 |
| 62 | Gas Utilities | 114 | 323 | 469 | 277 | 543 | 226 | 226 | 200 | 589 | | | | | 252 | 582 |
| 63 | Construction and Engineering | 2.6K | 49 | 417 | 317 | 691 | 210 | 399 | 200 | 614 | 169 | 348 | | | 253 | 685 |
| 64 | Personal Products | 356 | 90 | 287 | 284 | 683 | 226 | 480 | 260 | 660 | | | 185 | 185 | 268 | 671 |
| 65 | Energy Equipment and Services | 264 | 22 | 399 | 317 | 513 | 241 | 350 | 203 | 630 | 78 | 306 | 74 | 74 | 279 | 625 |
| 66 | Transportation Infrastructure | 399 | 82 | 357 | 317 | 695 | 151 | 304 | 272 | 689 | 297 | 554 | 372 | 667 | 279 | 703 |
| 67 | Marine | 156 | 120 | 175 | 317 | 487 | 155 | 452 | 299 | 686 | 203 | 203 | 264 | 478 | 292 | 691 |

**Server patching by industry.** We now focus on the patching behavior of enterprises per industry. Table 5.13 presents the top and bottom 10 industries based on their overall server-side service patch time. Overall, the patching behavior of enterprises in the top 10 industries, i.e., the best patchers, is not good, especially when compared to the

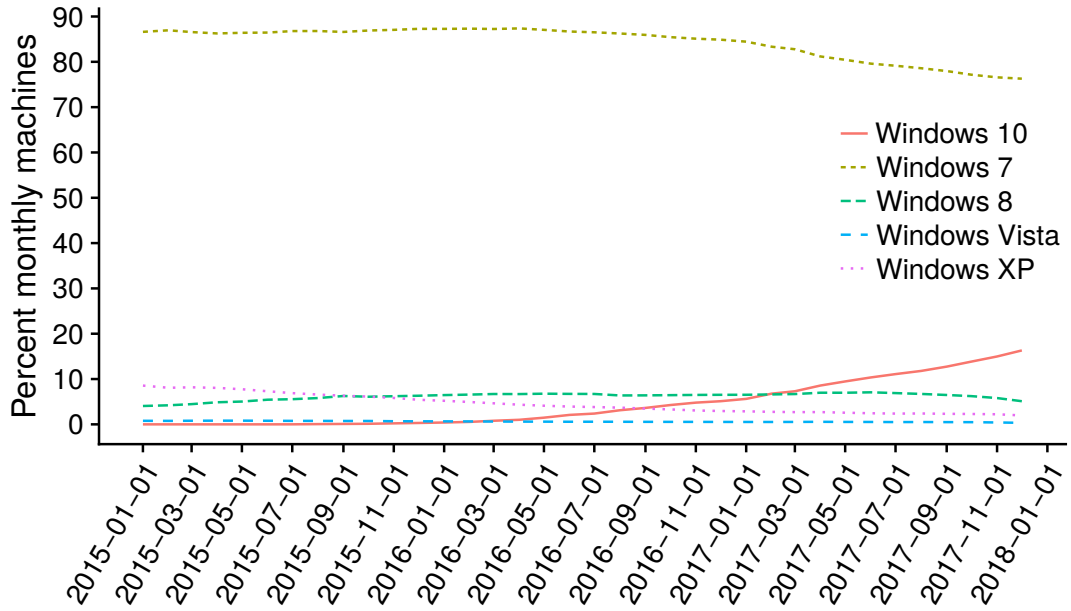



Figure 5.4: Percentage of monthly enterprise hosts per Windows OS version.

per-client application patch time reported in Table 5.9. For instance, we can see that the 50% patch time across all applications for the top 10 industries (between 78 and 199 days) is way above the average 50% patch time across all applications (56 days). The same applies to the 90% patch time per industry (between 412 and 709 days) when compared to the average 90% patch time across all applications (282 days). Looking at the patch time per protocol, we can see that some very popular services like Web servers and SSH servers bear the worst patch time (both 50% and 90%) in the top and bottom 10 industries. Indeed, the top 10 industries take almost six months (174 days) to patch 50% of their SSH servers and almost seven months (205 days) to patch 50% of their web servers. This is more than three times the average 50% patch time (56 days) reported across all applications. Similar to what we observe for client-side applications, we witness some industries associated with critical infrastructures, such as Gas Utilities, Transportation Infrastructure, and Marine among the worst-patching industries. Overall, our conclusion is that the patching behavior of servers in enterprises is worryingly bad, across all server applications and services.

### 5.4.3 Operating System Upgrade Behavior

In this section, we analyze the Windows upgrade behavior in the enterprise client hosts. Figure 5.4 shows the monthly percentage of hosts that use Windows XP, Vista, 7, 8, and 10. There is no big change on Windows version usage between 2015 and 2017. For most of the period Windows 7 dominated with over 80% of the hosts using them.

On March 2017, an increase of Windows 10 hosts occurs, raising from 10% to 20% by end of 2017. A simultaneous drop of Windows 7 machines indicates a slow shift from Windows 7 to 10. The percentage of Windows Vista and 8 remains constantly below 10%; there is no significant adoption of these versions by enterprises. Windows XP usage is already low in the beginning of 2015 (around 10%) and declines until the end of 2017.

Microsoft ended support of Windows XP in April 2014, but we still see an alarmingly large number of enterprises that use it. During 2017, we see a total of 466K Windows XP hosts in more than 43% (12.2K) of the enterprises. Most of these are medium to large enterprises; 73% of those have a total of more than 100 hosts, and 25% more than 1K hosts. All 67 industries have at least some companies with outdated OS hosts. The three industries with the largest number of XP hosts are *Electronic Equipment, Instruments and Components*, *Specialty Retail*, and *Banks*. Interestingly, banks have the lowest percentage of hosts with malware appearances but still more than 500 of those operate Windows XP hosts. This possibly indicates the difficulty of decommissioning legacy systems.

We see far less Windows Vista hosts, compared to XP hosts, in enterprises during 2017; a total of 86K hosts in 7K enterprises. Microsoft ended support for Vista in April 2017. The low number of hosts is probably due to the small adoption of Windows Vista by enterprises (Figure 5.4). As in the case of XP, enterprises that still use Vista are medium to large. In fact, 76% (5.5K) of enterprises with Vista hosts have also XP hosts.

# 6 PUP Economics

## 6.1 Introduction

Potentially unwanted programs (PUP) are a category of undesirable software that includes adware performing ad-injection, ad-replacement, pop-ups, and pop-unders, as well as rogue software (i.e., rogueware) that pushes users through scary warnings to buy licences of the rogueware, despite its limited functionality. PUP's undesirable behaviors prompt user complaints and have led security vendors to flag PUP in ways similar to malware. PUP prominence has quickly increased over the last years. Thomas et al. [18] showed that ad-injectors, a popular type of PUP that injects advertisements into user's Web surfing, affect 5% of unique daily IP addresses accessing Google. They also measured that Google's Safe Browsing generates 60 million warnings related to PUP - three times that of malware [4]. And, Kotzias et al. measured that over 54% of the 3.9 M real hosts they examined had PUP installed [37] and that PUP dominates so-called malware feeds [19] .

PUP is often distributed through commercial pay-per-install (PPI) services [4, 37]. A commercial PPI service acts as an intermediary between advertisers that want to distribute their programs and affiliates that own programs (typically freeware) that users want to install. To monetize installations of its free program, an affiliate bundles the free program with a downloader from a PPI service, which it distributes to users looking for the free program. Affiliates are paid by the PPI service \$2.00–\$0.01 per installation, depending on the geographic location of the user. During the installation process of the free program, users are prompted with offers to also install programs from the PPI advertisers. Advertisers pay the PPI service for successful installs of their advertised programs. Commercial PPI services are often used (or abused) by PUP publishers to advertise their programs and play an important role in PUP distribution [4,37]. Undesirable programs from advertisers, commercial PPI downloaders, and affiliate programs bundled with the PPI downloader are all typically flagged as PUP by AV vendors. PPI services also exist for distributing malware [28], but we call those *underground PPI services* to differentiate them from the commercial PPI services that

PUP uses (which for short we simply call PPI services in this paper). Prices paid to affiliates by underground PPI services range $0.18–$0.01 per install [28], showing that malware distribution can be an order of magnitude cheaper than PUP distribution for the most demanded countries. Prior work has shown that both types of PPI services are largely disjoint [4, 37].

This work has two goals. Our main goal is to *measure the economics* of the commercial PPI services used to distribute PUP, e.g., how profitable they are. Understanding their economic evolution over time is an essential step for evaluating the effect of deployed defenses [29]. To measure their economics, we first need to perform *attribution*, i.e., identify the entities behind them.

A fundamental difference between PUP and malware is that PUP is often published by companies, and companies also run the commercial PPI services used to distribute PUP. In contrast, malware publishers are cybercriminals with hidden identities and use underground PPI services also run by cybercriminals. Since companies are behind PUP and the PPI services PUP uses, attribution is potentially easier (compared to malware) because those companies may be required to publish information about them, their business activities, and the people that manage them. For example, legislation may require companies to register in a national company register, to submit financial reports, and in some cases to be the subject of external audits. Although PUP attribution is arguably easier than malware attribution, it is still challenging because behind PUP and commercial PPI services there are often networks of companies [19] and those companies are created, dissolved, and renamed over time. It is also challenging because company information widely varies among countries, comes from different sources, is often incomplete, and only available as text documents (e.g., PDF).

In this work we call *operation* the network of persons and companies that operate a commercial PPI service. To perform attribution of an operation we propose *entity graphs*. Nodes in an entity graph represent companies and persons. An edge from a person to a company indicates that the person is part of the company's management. Company nodes are annotated with corporate information such as creation date, address, country where registered, fiscal identification number, list of names used over time, and type of economic activity. An entity graph enables structured attribution by tracking the business relationships among persons and companies in an operation. Our approach to build an entity graph takes as input an initial list of companies, possibly only one, known to belong to an operation. It uses company registers for obtaining company information, identifying the persons managing the company, and finding other companies also managed by those persons. This approach discovers new companies in the operation, not present in the input set of companies, thus expanding the operation's coverage.

Once we have an entity graph for an operation, we obtain financial and audit reports for the identified companies, and use them to analyze the operation's economics. We analyze the revenue, net income (i.e., profits and losses), and EBITDA. We also examine the number of employees and, when available, expenses and revenue split by source. We focus on the 2013–2015 time period. As far as we know, this is the

first work that looks at the economics of PUP operations, and in particular of the commercial PPI services used to distribute PUP. Prior work has analyzed the economics of diverse malicious activities, e.g., [30, 79–81]. A key difference is that those papers analyze revenue data obtained from data leaks or estimated through external measurements. However, high revenues do not necessarily mean high profits, since the operational expenses can also be high. In contrast, by using company financial statements, we have not only revenue data, but also profits and losses, and in some cases expenses split by category. Thus, we can truly analyze how profitable the operations running commercial PPI services are.

While our approach is generic, in practice it is challenging to obtain the data for building entity graphs for the reasons exposed above. Therefore, in this work we focus on operations based in Spain because for this country we are able to collect, in a semi-automated fashion, the company information for building the entity graphs and analyzing the operations' economics. In particular, we analyze three Spain-based operations. All three operations run a PPI service for Windows programs, but are also involved in other parts of the PUP ecosystem such as publishing their own PUP (e.g., system cleaning utilities) and managing freeware download portals.

Our analysis addresses the following 6 main questions:

1. *How profitable are commercial PPI services and the operations behind them?* We measure that the three operations have a total revenue of 202.5M €, net income of 23M €, and EBITDA of 24.7M €. The most profitable operation has revenue of 92.2M € and net income of 11M € obtained in 2013–2015. Most of the revenue of each operation comes from a small subset of companies. There is a large gap between revenue and net income in all operations, indicating large expenses and low margins.

2. *What are the revenue sources?* The largest source of revenue for all three operations is the PPI service, which provides up to 90% of an operation's revenue. But, we also observe the operations to draw revenue from other sources such as advertising, download portals, PUP products they develop, and video streaming services.

3. *How has the PPI business evolved?* Peak revenue and net income happened in 2013. We observe a sharp decrease on both revenue and income for all three operations starting mid-2014, leading to all three operations to have losses in 2015. We conclude that improved PUP defenses deployed by different vendors in mid-2014 [39–41] significantly impacted the PPI market, which did not recover afterwards.

4. *How many companies are involved in an operation?* We find that each operation runs from 15 up to 32 companies, but most of them are *shell companies* that have no employees, no revenue, share address with other companies, are often created in batches, and have no website. We observe those shell companies being used to

obtain code signing certificates from certification authorities, later used to sign the distributed executables. While all three operations are based in Spain, two of them also use 2–5 companies registered abroad, namely in Israel and the state of Delaware in the US, a known tax haven [195].

5. *How many persons run an operation?* We find that a small number of 1–6 persons manages the large number of companies in each operation. One of the operations is run by a single person that manages 21 companies.

6. *How long have they been in operation?* The lifetime of each operation is 7–13 years, with companies created as early as 2003. However, the companies that run the PPI service in each operation were created in 2010–2011. Prior to 2010, the revenue came from other activities such as PUP licenses and download portals.

Our contributions are the following:

- We perform the first economic analysis of PUP operations and specifically of commercial PPI services used to distribute PUP. We acquire financial and audit reports for the companies involved and use them to analyze the revenue, net income, and EBITDA. When available, we also analyze expenses and sources of revenue.

- We propose a novel approach to perform PUP attribution using entity graphs. Nodes in an entity graph are companies or persons and edge from a person to a company indicates the person holds a management position in the company.

- We generate the entity graphs for three Spain-based operations, each running a commercial PPI service and involved in other related activities. The entity graphs comprise of 15–32 companies and 1–6 persons.

## 6.2 Overview

In this section we describe privacy and legal considerations (Section 6.2.1), introduce the operations analyzed (Section 6.2.2), define the entity graph (Section 6.2.3), and present the input company lists (Section 6.2.4).

### 6.2.1 Privacy & Legal Considerations

Our main goal is to analyze the economics of operations running commercial PPI services. For this, we build entity graphs for three Spain-based PUP operations. At no point we aim to point the finger to these particular operations or the people behind them. They have been chosen simply because they are Spain-based and thus we can obtain the needed data for the analysis. Any other operation could have been analyzed if their country of origin makes available the needed data.

| Operation | PPI | DP | PUP |
|---|---|---|---|
| OP1 | ✓ | 16 | 1 |
| OP2 | ✓ | 1 | 4 |
| OP3 | ✓ | 2 | 1 |

Table 6.1: Whether each operation runs a PPI service, download portals, and PUP software.

Our ethics advisory board has mandated that we anonymize the operations to prevent putting the spotlight on the people running these three operations, and to avoid time-consuming legal actions. Specifically, we anonymize the names of the operations, as well as the names of the companies and persons involved in each operation. For the rest of the paper we refer to the three operations as: OP1, OP2, and OP3. And, we refer to specific companies using the operation and a company identifier, e.g., OP1.C02.

We strive to achieve a balance between the privacy of the persons behind the operations and the value of the information provided. Our anonymization is best-effort since all the data analyzed is public and accessible freely or by paying small fees. Thus, the raw sources could be analyzed independently of our results. We understand that providing information about the operations' rankings (Section 6.2.2) reduces the anonymity set for the operations, but we believe that it is not much additional information given the availability of the raw data. Furthermore, we believe that the rankings are fundamental for readers to understand how representative the three operations are and thus the extent of our results.

We note that the anonymization process does not affect our analysis since it is performed a posteriori. We also believe that it does not significantly impact the presentation of our results, while helping protect the privacy of the persons behind the operations.

We also note that providing a definition of what behaviors make a program PUP (or malware) exceeds the scope of this paper. Instead, to determine if a sample is PUP, malware, or benign we use a previously proposed approach that examines PUP-related keywords that appear in the labels output by AV engines during scanning of suspicious samples [19]. In a nutshell, we rely on AV vendors to identify PUP samples, and use the digital signatures in those samples (when available) to identify the companies in charge of the PUP.

### 6.2.2 PUP Operations Analyzed

All three operations run a commercial PPI service during our analysis period. The three PPI services have been ranked by prior work among the top 15 commercial PPI services by user installation base, and have been estimated to affect a few millions of users in total [37]. Other prior work ranks two of these operations among the Top 10 PUP operations by number of signed samples and the other in the Top 30 [19]. Thus, while we do not know exactly what fraction of the commercial PPI market the three

operations represent, we do know that they play a significant role, i.e., they run some of the largest commercial PPI services and affect a large number of users. Thus, we believe that the insights gained on the commercial PPI market from these operations are representative of the ecosystem.

In addition to running a PPI service, the operations have been involved in other related activities. Specifically, all operations have developed at least one PUP product and have managed at least one download portal to assist in the distribution of their PPI downloaders and PUP products.

Table 6.1 summarizes the number of download portals and PUP products we have identified. These numbers are only a lower bound since we may have missed other software products and download portals. OP1 develops a download manager used to offer advertiser programs to users that install it. They also manage a large number of download portals that offer freeware bundled with their PPI downloader. Of their 16 download portals, 9 are blocked by SafeBrowsing as unsafe. OP2 develops several rogueware, namely system cleaning utilities and media players, and operated until 2015 a download portal. The audit report for OP3.C18, the company that runs the PPI service in OP3, has a nice description of how the company operates. Translated to English, it states: "The company obtains its revenue predominantly from offering to users visiting their download portals third-party applications from which they receive a payment for each installation or a share of the revenues the application generates. In the first case, the revenue is accounted for when the application is installed by the final user; in the second case, the revenue is accounted for as it is confirmed by our clients". Thus, advertisers can opt for a pay-per-install or a revenue sharing model. The download portals are used by the PPI service to attract users looking for freeware to have them install the PPI downloader.

### 6.2.3 Entity Graph

The entity graph is an undirected graph where a node is either a person or a company. An edge from a person to a company means that the person holds, or held in the past, a directive position in the company. A person may have (or have had) multiple positions in a company (e.g., administrator and treasurer). Companies are uniquely identified by their fiscal identification number because they may change names over time. The left part of Table 6.2 summarizes the node and edge attributes of the entity graph. For each attribute it shows the attribute type (i.e., string, integer, float, list, boolean), the type of object where stored (i.e., node or edge), and the node type (i.e., person, company, roles). We detail node and edge attributes below.

**Node attributes.** Persons have only one attribute, the name of the person. Companies have generic, economic, and code signing certificate attributes. Generic attributes include the list of company names, company type (e.g., limited liability), economic activity, number of employees, fiscal identification number, telephone number, address,

| Attributes | | Objects | | Datasets | | |
|---|---|---|---|---|---|---|
| **Attribute** | **Type** | **Object** | **Type** | **BE** | **HP** | **IF** |
| Person name | str | Node | Person | ✓ | ✗ | ✗ |
| Fiscal ID | str | Node | Comp. | ✓ | ✗ | ✓ |
| Company names | str list | Node | Comp. | ✓ | ✗ | ✗ |
| Company type | str | Node | Comp. | ✓ | ✗ | ✗ |
| Economic activity | str | Node | Comp. | ✓ | ✗ | ✓ |
| Employees | int | Node | Comp. | ✗ | ✗ | ✓ |
| Telephone number | str | Node | Comp. | ✓ | ✗ | ✓ |
| Address | str | Node | Comp. | ✓ | ✗ | ✓ |
| City | str | Node | Comp. | ✓ | ✗ | ✓ |
| Country | str | Node | Comp. | ✓ | ✗ | ✓ |
| Creation date | date | Node | Comp. | ✓ | ✗ | ✓ |
| Dissolution date | date | Node | Comp. | ✓ | ✗ | ✓ |
| Last modification date | date | Node | Comp. | ✓ | ✗ | ✗ |
| Capital | float | Node | Comp. | ✓ | ✗ | ✓ |
| Earnings | float | Node | Comp. | ✗ | ✗ | ✓ |
| Revenue | float | Node | Comp. | ✗ | ✗ | ✓ |
| EBITDA | float | Node | Comp. | ✗ | ✗ | ✓ |
| Certificates | str list | Node | Comp. | ✗ | ✓ | ✗ |
| Revoked certificates | str list | Node | Comp. | ✗ | ✓ | ✗ |
| Active roles | str list | Edge | Roles | ✓ | ✗ | ✗ |
| Inactive roles | str list | Edge | Roles | ✓ | ✗ | ✗ |

Table 6.2: Attributes used in the entity graph, the objects holding the attribute, and the datasets used to obtain their information. The datasets are described in Section 6.3 and correspond to BORME (BE), HerdProtect (HP), and Infocif (IF).

city, country, creation date, dissolution date (if any), and the date of the last modification of the company data. Economic information attributes include parts of the annual balance for all available years. These attributes are initial capital, revenue, net income, and EBITDA (Earnings Before Interest, Taxes, Depreciation, and Amortization). At last, there are two code signing certificate attributes: the list of known code signing certificates issued to the company and the list of those certificates that have been revoked. If the company has not been used for obtaining a code signing certificate both attributes are empty.

**Edge attributes.** Edges have two attributes: the list of active roles (if any) that the person currently has in the company, and the list of past roles (if any) that the person had in the company.

### 6.2.4 Input Company List

Our approach takes as input an initial list of companies that are part of a PUP operation. This initial list can be obtained from different sources such as the contact information and privacy policies of PUP websites, Whois registration data for PUP domains, or the digital signatures of PUP samples. In this work, we obtain the list of initial companies from prior work that clustered PUP samples into operations using information from their digital signatures [19].

The intuition of using the digital signatures from PUP samples to identify compa-

nies is that PUP publishers are constantly looking for ways to make their programs look benign in order to convince the user to install them and to avoid detection. One such way is code signing, where the software is distributed with a digital signature which, if valid, certifies the integrity of the software and the identity of the publisher. Signed programs look more benign and may be assigned higher reputation by security products. In Windows, properly signed programs avoid scary warnings when a user executes them and are assigned higher reputation when downloaded through Internet Explorer [91]. Furthermore, kernel-mode code is required to be signed. To sign Windows programs, publishers need to obtain a valid *code signing certificate* from a Certification Authority (CA), which requires providing the publisher's identity to the CA and paying a fee ($500–$60 for 1-year certificates). While not all PUP samples are signed [18], prior work has shown that properly signed samples detected by AV engines are predominantly PUP [19], as identity validation by CAs poses an important barrier for malware.

The input lists obtained from Malsign [19] comprises of 15 companies for OP1, 9 companies for OP2, and 29 companies for OP3. Despite the large number of input companies, our approach still discovers 15 previously unknown companies, as well as the people managing the companies. Furthermore, we have also evaluated our approach by using an input list with a single company for each operation. With this reduced input list, the produced entity graphs still contain all companies registered in Spain present in the entity graphs obtained with the larger input lists.

## 6.3 Datasets

We leverage a variety of datasets for this work. We use company registers for obtaining company information and for establishing the persons managing a company; audit and business reports for obtaining company financial data; a dataset of signed PUP executables for identifying the initial list of companies in each operation; the website of a security company for determining which companies have been used for obtaining code signing certificates; a malware repository to measure the prevalence of samples from each operation over time; and certificate transparency logs to identify websites belonging to the operations.

**Company registers.** Company registers collect information about companies in the jurisdiction they operate under. Each country has its own norms regarding the existence of such register, whether the register is centralized or distributed (e.g., to its regions), what type of information it collects on companies, and how publicly available the data is. Countries may provide public access to some of the data collected by their company registers, for example Germany [196], Israel [197], Spain [198], and United States [199].

In Spain, there exist 52 regional registers and a central register called *Registro Mercantil Central* [200]. The regional registers collect the information on the companies

in their region. The central register is in charge of providing access to the information collected by the regional registers since January 1st, 1990. The central register has a publication, called *Boletín Oficial del Registro Mercantil* (BORME), which provides free public access to much of the collected company information. Every day, BORME publishes a PDF document with all changes that occurred in the regional registers. Among others, BORME reports the following company events: creation, dissolution, changes in the type of economic activity identified by a CNAE[1] code (e.g., 6312 - Web portals), changes in administrators, capital increases and reductions, balances, company name changes, corporate split-ups, and adsorptions.

Not all company data collected by the regional registers appears in BORME. In particular, Spanish law requires all companies to submit an annual financial report to their regional register. These financial reports are not included in BORME, but can be acquired from the central or regional registers for a fee. The financial report contains a balance sheet, an income statement, a statement of changes in equity, a profit and loss statement, and a statement of cash flows. Depending on specific criteria such as company size and net turnover, a company may be required to submit only an abbreviated version of the above documents [201].

To access the information from BORME, we leverage LibreBORME [202], an open source project that provides a public API to query BORME data published since 2009. LibreBORME parses the PDF files published every day and stores their data in a central database. It provides a clean interface, but only to a subset of the data from BORME. By querying LibreBORME with a company name, we can obtain: the fiscal identification number (NIF) that uniquely identifies a company in Spain, its creation date, the last modification date of the company's data, and the names of the persons with management positions (e.g., administrator, secretary, liquidator). By querying LibreBORME with a person name, we can obtain the list of companies in which the person holds management positions.

**Audit reports.** Countries may require companies satisfying certain criteria (e.g., income or number of employees above some threshold) to have periodic audits of their financial reports by external certified professional accountant (CPA) firms, i.e., auditors. The auditors examine and validate the financial reports of the company and perform a detailed analysis of its business activities, as well as a comparison with the previous year. Audit reports often contain details not included in other financial reports. For example, they may detail the sources of income for a company (e.g., company products), the type of expenses, the reasons behind big changes on revenue or net income, and an analysis of the business risks.

In Spain, companies need to perform an audit if they fulfill two of the three following requirements for two consecutive years: (a) revenue over 5.7M €, (b) total assets over 2.8M €, (c) an average number of employees higher than 50 [203]. Audit reports can be acquired from business portals (see below). We use the Infocif [204] Web por-

[1] *Clasificación Nacional de Actividades Económicas*

tal to obtain 5 audit reports for the three companies that fulfill the above requirements: OP1.C02, OP2.C08, OP3.C18. Those three companies correspond to the largest company for each operation. Each audit report covers two years, the year being audited and the previous year for comparison. If a company is part of a corporate group, the audit report also contains financial information for the other group companies.

For OP1.C02 and OP2.C08, we obtain the audit reports for 2014 and 2015, which cover 2013–15. For OP3.C18, only the 2014 audit report is available, which covers 2013–14. Each audit report costs 10 € and is performed by one of the following CPA firms: PricewaterhouseCoopers [205], Deloitte [206], Audalia Laes Nexia [207], or AFP Audit & Consulting [208]. The audit reports have varying degree of detail. example, the PricewaterhouseCoopers and Deloitte reports contain a revenue split by income source, but the ones from Audalia Laes Nexia and AFP Audit & Consulting do not.

**Business reports.** Many online services offer company reports comprising of corporate and financial data. Such reports are typically compiled by aggregating data from multiple sources including financial statements from company registers, public audit reports, financial reports published by the company, mandatory statements from listed companies, and Internet data such as news clips. These reports can be acquired by paying a fee. They are typically not as detailed as audit reports and their information can widely vary among services. One advantage is that their aggregation may cover longer time periods (e.g., up to 3 years), which is useful since we focus on the 2013–2015 period.

We use the Infocif service [204] to acquire reports for all companies in our entity graphs. Infocif reports contain financial data such as revenue, net income, EBITDA, number of employees, and an abbreviated version of the profit and loss statement. They also contain corporate information such as company address, phone number, CNAE code, and the company's website. Corporate information, except the company's website, comes from BORME, but is not accessible through LibreBORME.

Overall, we were able to acquire reports from Infocif for 59, out of 68 companies, with a total cost of 215 €. For the remaining 9 companies we could not obtain reports because 8 are registered outside of Spain and the other one is created in 2016, and thus had not yet filed their first financial statement when we ordered the reports. From the 59 acquired reports, 5 are empty. An empty report indicates that a company has not submitted their financial report to the company register, or that the report is pending approval (or digitalization) by the company register.

**Malsign.** The Malsign dataset consists of 142 K signed PUP (and a few malware) samples, as well as their clustering into operations/families [19]. The clustering results are based on statically extracted features from the samples with a focus on features from the Windows Authenticode signature [36]. These features include: the leaf certificate hash, leaf certificate fields (i.e., public key, subject common name and location),

the executable's hash in the signature (i.e., Authentihash), file metadata (i.e., publisher, description, internal name, original name, product name, copyright, and trademarks), and the PEhash [107]. From the Malsign clustering results, we select the three operations based in Spain. For each of these three operations, we use the clustering results to extract an initial list of companies, which comes from the subject common names (CN) in the certificates of samples in the cluster, after being normalized to remove duplicates. Overall, Malsign contains 15 companies for OP1, 9 for OP2, and 29 for OP3.

**HerdProtect.** HerdProtect [209] is a security company that provides a host-based defense against PUP and malware. Their website provides detailed threat information including certificate information from PUP samples they observe in their users' hosts. For each company in an entity graph, we leverage HerdProtect's website to query for all code signing certificates they have observed issued to the company. For each certificate found, we collect the Subject, the Issuer (i.e., certification authority), the validity period, and the certificate's serial number. The data from HerdProtect enables us to identify which companies have been used to obtain code signing certificates from CAs. While we could use Malsign for this step, HerdProtect's coverage is larger, containing many certificates not included in Malsign.

**VirusShare.** We collect 27.7 M hashes of malware and PUP executables from the VirusShare repository [114]. We query those hashes to VirusTotal [96] (VT) to get their detection labels by multiple AV engines, as well as the timestamp when they were first submitted to VT. We use the AV labels as input to AVClass [165], a malware labeling tool that outputs for each sample the most likely family name and a confidence factor based on the agreement across engines. We use the AVClass results to identify samples that belong to the three operations and the VT first seen timestamp to measure the fraction of samples of each operation in VirusShare over time.

**Certificate transparency logs.** We analyze 38.3 M HTTPS certificates from Google's Certificate Transparency logs [210] to check if companies in entity graphs have a website.

The right part of Table 6.2 summarizes which dataset is used to extract each attribute in the entity graphs.

## 6.4 Building Entity Graphs

This section describes how an entity graph is built given as input an initial list of companies known to belong to a PUP operation. This process may identify additional companies, not present in the initial company list, which are also part of the operation. In addition, it identifies the persons managing the companies in the operation. Building an entity graph comprises of four steps: building an initial graph, collecting

certificate data, trimming the initial graph, and acquiring financial data. The first three steps we have been able to automate, so that they are reusable for other Spanish PUP operations. Acquiring the financial data and incorporating it into the entity graph is a manual process.

**Building an initial graph.** To build the initial graph we leverage LibreBORME to obtain information about companies and the persons managing them. For each company in the input list, our approach first queries LibreBORME to obtain its data (fiscal identification number, creation date, last modification date) as well as the names and positions of the people managing the company. If the company is found, a node is added for it in the initial graph. In addition, one node for each person managing the company is also added, as well as edges from each person node to the company node. If the company is not found, for example because it is registered in a country other than Spain, a node is still added for the company, but no person nodes or edges are added. Thus, foreign companies introduce disconnected components into the graph. Then, for each person found in the previous step, LibreBORME is queried again to obtain any other companies where the person has managing positions. For each additional company identified, not yet present in the initial graph, the process recurses and the two steps above are repeated to obtain the new company's data and possibly identify new persons managing the new company. The process recurses until no new companies or persons are found.

During the recursion, if a person is found to hold managing positions in an unusually large number of companies (i.e., more than 100), the person is not added to the entity graph. This rule prevents lawyers to appear in the entity graph. Such lawyers are used by one operation to perform the initial registration of a company, after which the company is transferred to the real managers that are part of the operation. If we included the lawyers, we would also include the companies they register for other clients, which often number in the hundreds and are unrelated to the operation.

**Collecting certificate data.** For each company in the initial graph, our approach queries HerdProtect using the company name to obtain the list of code signing certificates issued to the company that HerdProtect has observed being used to sign PUP executables. For each certificate, we check the revocation status using the OCSP protocol and certificate revocation lists (CRLs). HerdProtect does not provide us with the raw certificate, but rather with its metadata. To query the revocation status we use the certificate's serial number and the CA that issued the certificate, both obtained from HerdProtect.

**Trimming the initial graph.** The initial graph goes through a trimming process to remove persons and companies unrelated to the PUP operation. The trimming applies two heuristic rules in sequence. The first rule aims at removing persons unrelated to the PUP operation, which did not satisfy the lawyer rule while building the initial graph.

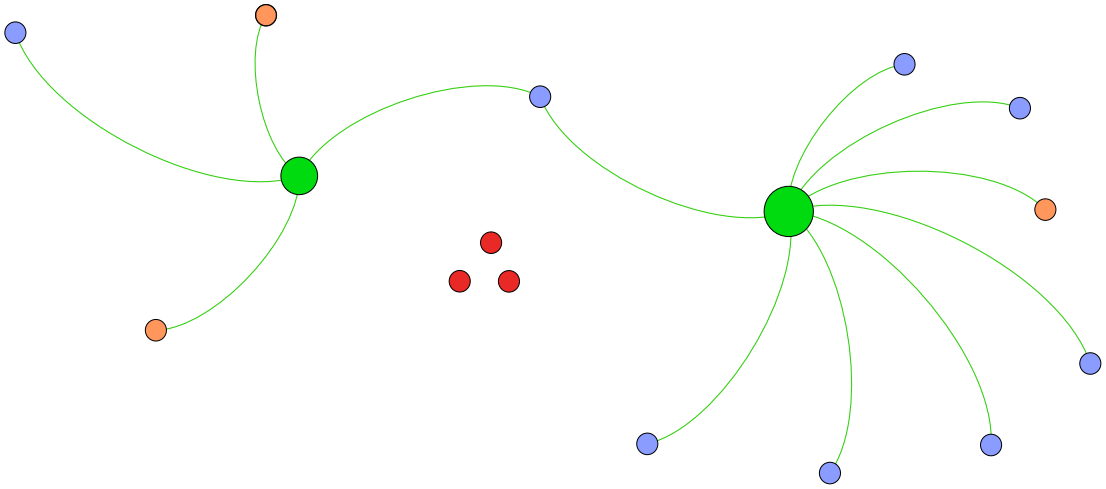


Figure 6.1: Mock example of an entity graph.

Specifically, this rule removes persons that only have positions in companies for which no certificates have been collected from HerdProtect. After removing those persons, any (Spanish) companies with no managing persons are also trimmed, since the reason they were initially added was the person no longer considered part of the operation. The second rule aims at trimming companies managed by people that are part of the operation, but that are used for purposes different than the PUP operation. This is important because persons involved in the PUP operation may also be involved in other unrelated activities such as real estate or finance. Specifically, this rule removes companies for which their business area is unrelated to information technology and which have not been used for obtaining code signing certificates.

**Acquiring financial data.** The trimmed graph corresponds to the entity graph for the PUP operation. For each company in the entity graph, we try to acquire a business report (and an audit report if applicable) from Infocif to obtain its financial data, as well as additional corporate information (e.g., address and phone number) not available through LibreBORME.

**Visualization.** Figure 6.1 shows a mock example of an entity graph visualized using Gephi 0.9.1 [211]. We differentiate nodes using colors. Green nodes are persons and we use three colors for company nodes: orange, purple, and red. Orange companies have no code signing certificate; purple companies have at least one certificate; and red companies are not registered in Spain.

| | Graph | | | Companies | | | Certificates | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Operation** | **Comp.** | **Per.** | **Edges** | **Addr.** | **Renames** | **CC** | **Comp.** | **Certs** | **CAs** | **Revoked** |
| OP1 | 21 (6) | 1 | 21 | 7 | 3 | 1 | 18 | 48 (22) | 5 | 2 |
| OP2 | 15 (6) | 3 | 20 | 8 | 5 | 3 | 12 | 85 (51) | 8 | 24 |
| OP3 | 32 (3) | 6 | 55 | 15 | 3 | 2 | 14 | 54 (26) | 7 | 16 |

Table 6.3: Summary of entity graphs for the three operations.

## 6.5 PUP Entity Graphs

This section describes the entity graphs produced for the three operations. We first compare the three entity graphs and then analyze each operation in more detail in its own subsection.

Table 6.3 summarizes the entity graphs. The table is split in three parts. The leftmost part shows the number of company nodes, person nodes, and edges in the entity graph. The numbers in parentheses indicate how many companies are new in the entity graph, i.e., they were not present in the initial list of companies used as input to build the entity graph. The middle part contains company data: the number of distinct addresses for the companies, the number of companies that had at least one name change since their creation, and the number of countries that these companies are registered in (a value larger than one means non-Spanish companies appear in the entity graph). The rightmost part summarizes the code signing certificates: the number of distinct companies used for obtaining certificates, the number of certificates issued to those companies, the number of CAs that issued those certificates, and the number of revoked certificates. The numbers in parentheses indicate certificates issued to new companies not in the initial list.

The entity graphs show that OP3 is the largest operation with 32 companies and 6 persons managing them. All operations have a large number of companies, ranging from 15 for OP2 up to 32 for OP3. But only a handful of people manage those companies, from 1 in OP1 up to 6 persons in OP3. In the case of OP1, a single person is the sole manager for 21 companies.

The number of new companies (in brackets) show that our approach to build entity graphs enables discovering additional companies that were not present in the initial list of companies used as input. Specifically, we discover 15 previously unknown companies: 6 in OP1, 6 in OP2, and 3 in OP3. Thus, in addition of capturing the relationships between companies and their managers, the entity graphs amplify the coverage for all three operations. This amplification happens despite the initial list of companies in each operation, obtained from Malsign, being fairly large. We expect that for other operations the initial list of companies may come from less complete sources and be much smaller, perhaps even a single company. In fact, we have also tested building the entity graphs for all three operations starting with only the main company in the initial list. The produced entity graphs are identical to the ones in Table 6.3 except in that companies not registered in Spain are not identified since they do not appear in BORME.

For all three operations the number of distinct street addresses is much lower than the number of companies. This indicates that multiple companies share the same address. For example, OP1 uses 7 addresses for all 21 companies. This points towards some of the companies not having real activity. We confirm this in Section 6.6 by examining the number of employees and other financial data.

Not shown in Table 6.3 is that both OP1 and OP2 often create multiple companies on the same day. For example, three OP2 companies (OP2.C12, OP2.C13, OP2.C14) were created on the same day in April 2014. Such batch registrations are often performed by lawyers that later transfer the companies to the real managers.

Renaming companies is a common behavior. All operations have companies that have changed names since their creation. Renaming a company is cheaper than creating a new company. The new company name can be used to obtain new code signing certificates, e.g., if a CA does not verify that the fiscal identity number of the requesting company matches a company with a certificate already issued. Even when a CA already issued a certificate for the company under a different name, it is logical that the managers may want to update their certificate after a company name change, making it difficult to deny the request. In OP2, five companies have been renamed and two of the new names have been used for obtaining new certificates. In each of the other two operations, three companies have been renamed and one new name has been used to obtain a new certificate.

It is also not uncommon for the operations to set up companies in multiple countries. OP2 and OP3 have companies registered outside of Spain, specifically in Israel and the United States. Interestingly, 6 companies (1 in OP2 and 5 in OP3) are registered in the US state of Delaware, a known tax haven [195].

**Certificates.** The number of code signing certificates issued to all three operations is four times larger than the number of companies across the three operations. This indicates that companies are used to obtain multiple code signing certificates (four on average). For OP2 this ratio goes up to 7 certificates per company. Certificates may be issued to the same company by different CAs. We also observe multiple certificates for the same company from the same CA using slight variations in the company name, e.g., *FakeComp SL* and *Fake Comp S.L.* All operations obtain certificates from multiple CAs (from 5 to 8). Focusing on a small number of CAs reduces the effort for obtaining the information required for the identity validation process. All three operations have certificates revoked, but the revocation ratio is quite low ranging from 4% for OP1 to 29% for OP3. The number of revoked certificates is especially low considering the lifetime of the operations. For example, OP1 had only 2 certificates revoked over 7 years.

**Coverage.** While we have shown that entity graphs amplify coverage, an important question is how much coverage do they achieve. We evaluate the coverage by comparing the companies in the entity graphs with the companies listed in the audit reports

as members of a corporate group. The audit reports identify 21 OP1 companies, 11 OP2 companies, and 6 OP3 companies. Overall, the audit reports identify 39 companies, compared with 68 companies in our entity graphs. Thus, the entity graphs have significantly higher coverage. Of all the companies identified in the audit reports, only three OP3 companies are missing in the entity graphs. One of those three companies is registered in Ireland and it was not identified because there are no certificates in Malsign for that company. A manual check for the other two missed companies reveals that they are Spanish companies, but LibreBorme has not properly parsed the person that created the company. Thus, our approach did not identify the companies. On the other hand, there are 29 companies in the entity graphs that do not appear in the audit reports. One company in OP1 was created in 2016, i.e., after the audit reports were issued. The other 28 companies are not listed in the audit reports as being part of the corporate group, however the entity graphs reveals that they are connected to the operations. In fact, many of them have been issued certificates used to sign PUP samples of the operation. This indicates that solely relying on the information reported by the company in the audit reports is not enough to understand the full scope of the operation.

**Graph building rules.** The building of the entity graphs excluded 5 persons (3 in OP1 and 2 in OP3) that manage an unusually large number of companies. They correspond to lawyers that, if included, would add hundreds of unrelated companies to the entity graphs. Additionally, the two trimming rules applied to the initial entity graph removed 50 companies and 12 persons across the three operations (4 companies and 2 persons in OP1, 2 companies and 3 persons in OP2, and 44 companies and 7 persons in OP3).

### 6.5.1 OP1 Analysis

Figure 6.2 shows the OP1 entity graph. It's the simplest entity graph among the three PUP operations with one person controlling the 21 companies in the operation. In contrast with the other two entity graphs, it does not contain any disconnected nodes indicating that all companies are registered in Spain.

Figure 6.3 presents a timeline of the 21 OP1 companies. The length of each line represents the lifetime of a company from creation to dissolution (or January 2017 if still active). A circle marks the issuing date of the first certificate for a company (if any). A star marks a date when a company was renamed (if any). The timeline shows that OP1 has existed for seven years, with the first company (*OP1.C00*) being created on March 2009. The company that runs the PPI service (OP1.C02, the one for which we have audit reports) was created in June 2010. For the first five years, at least one new company was created each year. In 2014, the rate of company creation increases significantly, with 14 companies created in the span of one year. These recent registrations often happen in batches with multiple companies being created simultaneously on the same date. The high rate of company registrations in 2014, and thus

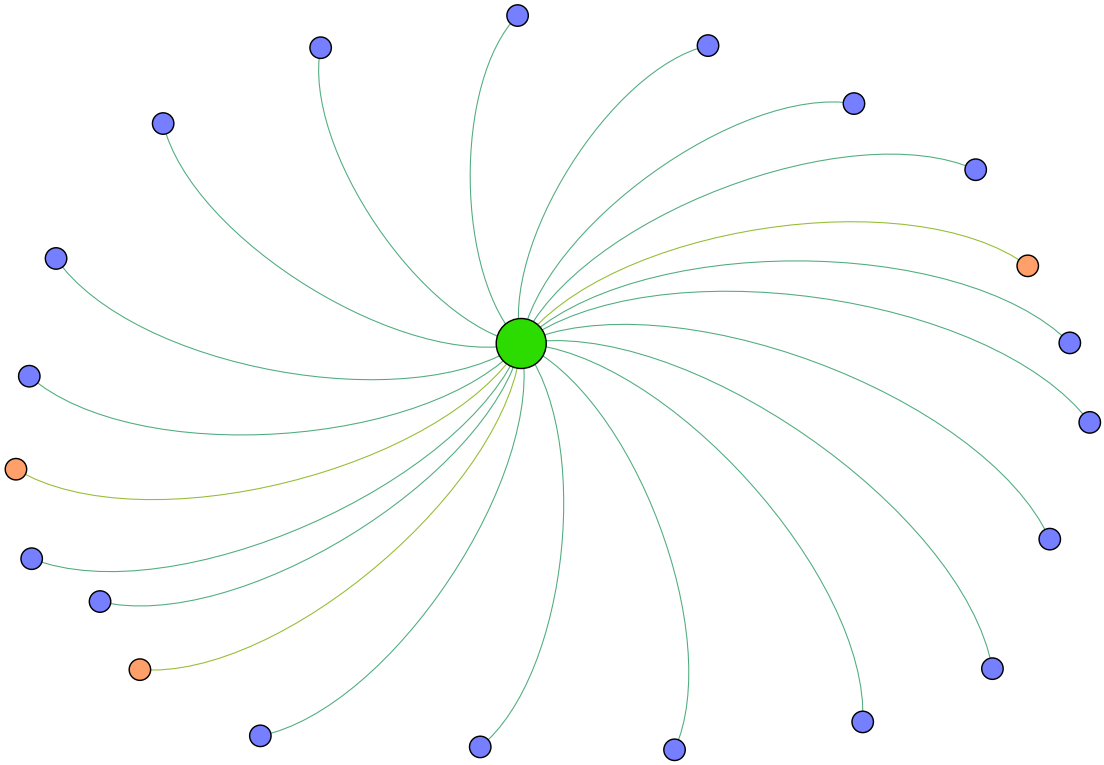


Figure 6.2: Anonymized OP1 entity graph. Green nodes represent persons, orange nodes companies without code signing certificates, and purple nodes companies with certificates.

of certificates used, may indicate increased pressure by security vendors during that time period. Since January 2015, only one new company has been created. This may be due to the recently observed shift of focus in OP1 towards other activities such as distribution of mobile applications.

OP1 follows a unique company registration pattern, not present in the other operations. Initially, an employee of a law firm creates the companies, but after a few months the company is transferred to the real manager that appears in the entity graph. When created, all companies mark the type of activity (i.e., CNAE code) as real estate. But, when ownership is transferred, the type of activity is modified to be development of web portals, which is one of OP1's activities. Similar to the company creation, changes of ownership occur in batches. For example, five companies (OP1.C08, OP1.C09, OP1.C10, OP1.C11, and OP1.C12) changed ownership on the same day in September 2014.

The first OP1 certificate was issued in 2011 (for OP1.C02). Of the 21 companies, 18 have been used to obtain certificates. We observe that certificates are often issued in batches using multiple CAs to request certificates on the same day. For example, on the same day in September 2014, 5 certificates were issued to 4 companies (OP1.C08, OP1.C09, OP1.C10, OP1.C12). We also observe that, especially since 2014, the code

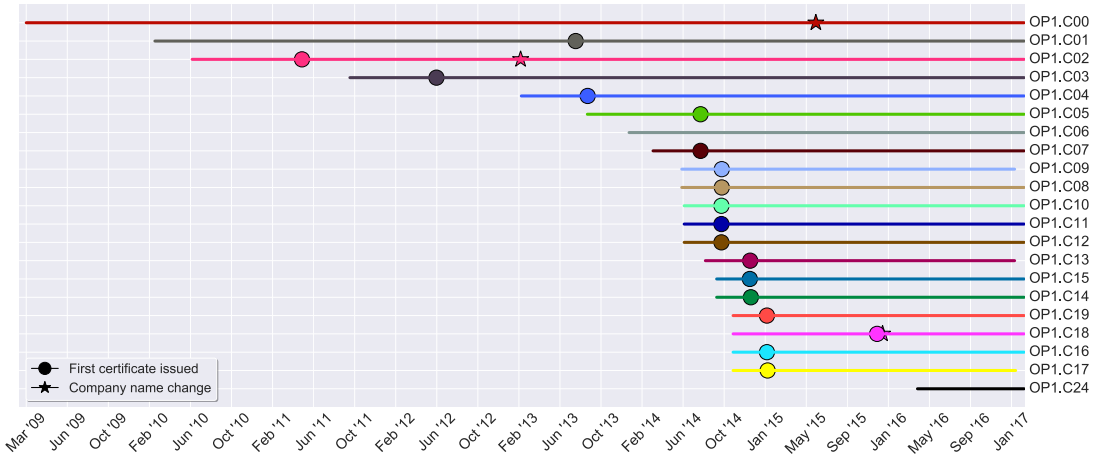



Figure 6.3: Each line represents the lifetime of an OP1 company. Circles mark the date of the first issued certificate of a company (if any) and stars mark the date of the company name change (if any).

signing certificates are issued close to the registration date of the company. We measure how fast a company is used for obtaining a certificate by measuring the difference in days between the company creation date and the issuing date of the first certificate for the company. The median delay is 117 days with the shortest being 97 days (OP1.C15) and the longest 1,230 days (OP1.C01). Instead, if we measure the delay starting from the date the company is transferred to the real manager, the median time drops to 26.5 days, with the fastest being 14 days after the company transfer. This indicates that once the real manager is in charge of the company, within a month he obtains a code signing certificate for the company. This may indicate that obtaining such certificate is one of the main reasons for the company creation.

### 6.5.2 OP2 Analysis

Figure 6.4 shows the OP2 entity graph, which comprises of 3 persons and 15 companies. One company (OP2.C08) connects the 3 persons. Beyond OP2.C08, each person manages a quite independent set of companies, except for OP2.C17 which is managed by two persons. There are 3 disconnected nodes, which correspond to companies registered outside of Spain: two in Israel, and another in Delaware, US.

Interestingly, all the companies in the entity graph appear in Malsign. However, in Malsign the companies were split among two different clusters. Even if we only used one of the Malsign clusters as input for the entity graph creation, our approach identifies that the companies in the other Malsign cluster (not used as input) also belong to the operation.

Figure 6.5 shows the company timeline for OP2. The operation has been active for 9 years, with the first company (OP2.C05) being created on December 2007. The company in the operation that runs the PPI service (OP2.C08, the one for which we have

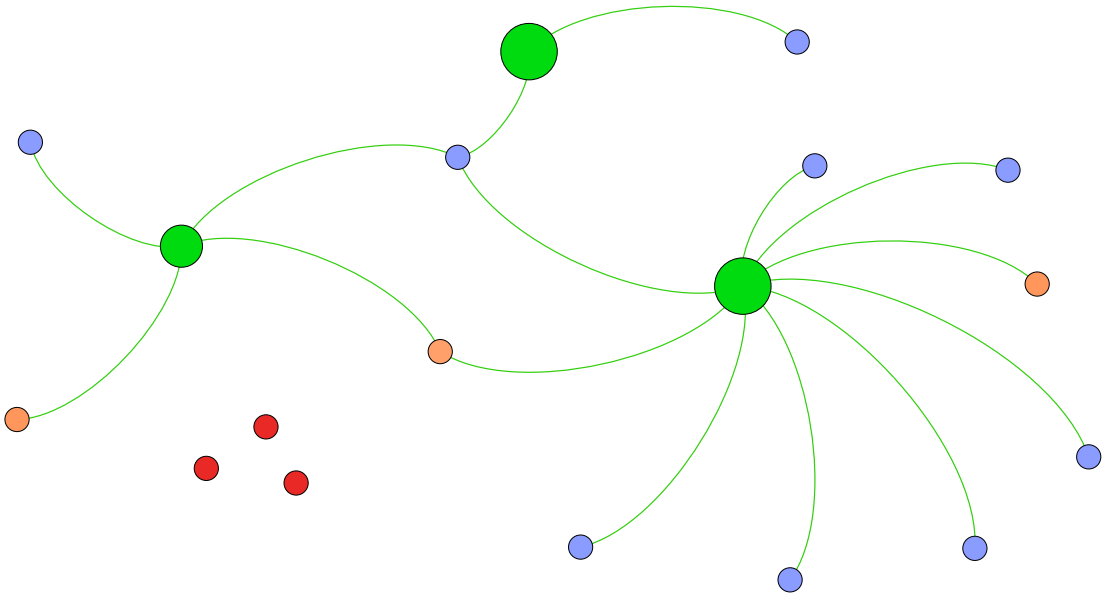


Figure 6.4: Anonymized OP2 entity graph. Nodes are colored similar to Figure 6.2.

audit reports) is created in February 2011. Similar to OP1 company creation happens in batches, but in contrast with OP1, lawyers are not used by OP2. For example, on the same day in April 2014 three companies are created (OP2.C12, OP2.C13, OP2.C14). Interestingly, OP2.C09 and OP2.C10 are also created on the same day, although they are managed by different persons in the operation. We see a spike on company registrations between September 2013 and May 2014, which largely coincides with the spike in OP1. Similar to OP1, we do not observe any new companies created in the second half of 2014 and throughout 2015.

Overall, OP2 has been issued 85 certificates, the largest number among the three operations. The first certificate for OP2 was issued in March 2011 (for OP2.C05) and it was revoked one day later by Comodo. The second certificate was issued in September 2011 for the main company (OP2.C08). Since then, 83 other certificates have been issued for this operation with a peak in 2014 with 46 issued certificates. This operation also requests certificates in batches, but each certificate in a batch is issued on a separate, but consecutive, day. The median time between the creation of a company and the issuing of the first certificate is 123 days, with the shortest being 45 days (OP2.C12) and the longest 4.7 years (OP2.C06).

Similar to the other operations, multiple OP2 companies are registered on the same address of the same city. Interestingly, three companies (OP2.C18, OP2.C05, OP2.C17) are registered by different people in different years, but all three on the same address.

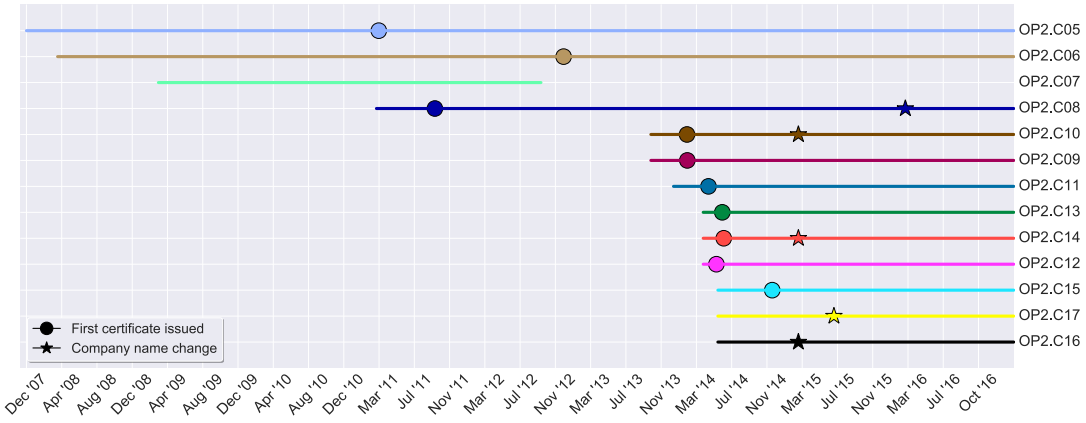



Figure 6.5: Each line represents the lifetime of an OP2 company. Circles mark the date of the first issued certificate of a company (if any) and stars mark the date of the company name change (if any).

### 6.5.3 OP3 Analysis

Figure 6.6 represents the OP3 entity graph. It is the largest and most complex entity graph with 32 companies and 6 persons. Of the 6 persons, 4 are connected to at least 4 companies and the other two are less central, being only connected to 1 or 2 companies. The company with the highest degree is OP3.C09, which connected 4 persons in the graph, but was dissolved in March 2014. The company that operates the PPI service (and for which we have an audit report) is OP3.C18, which connects 3 of the persons. The entity graph has five disconnected nodes for companies registered in Delaware, US.

The timeline in Figure 6.7 shows that OP3 has operated for 13 years, making it the longest-lived operation. The first OP3 company (OP3.C00) was created in July 2003 and was soon followed by three other companies (OP3.C01, OP3.C02, OP3.C03). Since then, 1–3 new companies were created every year. The company running the PPI service was created on June 2011. In 2012, 4 new companies are created and 8 new companies are added in 2013. The last company was created in May 2014. Compared to the other operations, OP3 companies are more spread over the years and the spike occurs earlier, in the second half of 2012 and the first half of 2013. We do not observe batch company registrations in OP3.

Overall, 54 certificates have been issued for OP3 companies. The first certificate was issued in 2004 for OP3.C04 and the highest number is 22 certificates issued during 2013. Each company is used for obtaining 3.8 certificates on average. The highest use is for OP3.C18, which has been issued 12 certificates. The median time between the creation of a company and the issuing of the first certificate is 236 days with the fastest being 13 days (OP3.C33) and the longest 9.5 years (OP3.C00). Thus, OP3 is the slowest among the three operations in using new companies to obtain certificates.

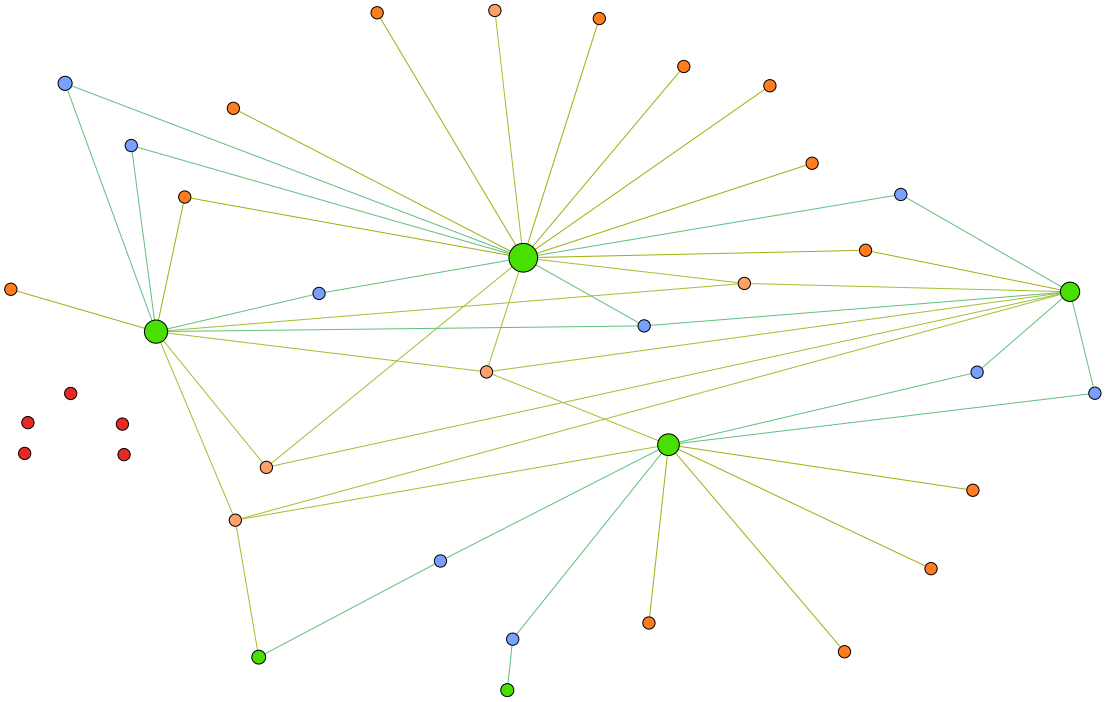


Figure 6.6: Anonymized OP3 entity graph. Nodes are colored similar to Figure 6.2.

| Op. | Period | Employees | Revenue | Income | EBITDA |
|---|---|---|---|---|---|
| OP1 | 2012-15 | $\leq$ 40 (3) | 81.8 M | 8.2 M | 7.3 M |
| OP2 | 2013-15 | $\leq$ 66 (4) | 92.2 M | 11.0 M | 12.3 M |
| OP3 | 2008-14 | $\leq$ 65 (8) | 28.5 M | 3.8 M | 5.1 M |
| **Total** | **2008-15** | **$\leq$ 171 (15)** | **202.5 M** | **23.0 M** | **24.7 M** |

Table 6.4: Summary of financial data for all operations. Revenue, net income, and EBITDA are provided in Euros.

## 6.6 PUP Economics

In this section we analyze the financial data obtained from the business and audit reports. We first provide a summary of the three operations and then detail each operation in its own subsection.

Table 6.4 summarizes the financial data. For each operation, it shows the period covered by the financial data, an upper bound on the number of employees across all companies in the operation (and the number of companies reporting at least one employee in brackets), and the total revenue, net income, and EBITDA across the whole period and for all companies in the operation. All currency values are in Euros.

The number of employees in Table 6.4 is the sum of the maximum number of em-

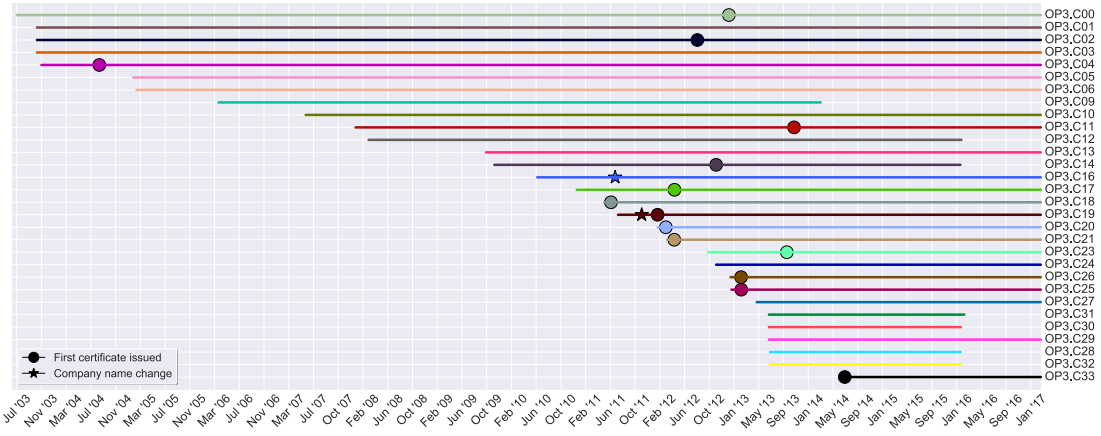



Figure 6.7: Each line represents the lifetime of an OP3 company. Circles mark the date of the first issued certificate of a company (if any) and stars mark the date of the company name change (if any).

ployees reported by each company across the period. It is an upper bound because there could be overlaps between employees in different companies and because not all employees may have been contracted at the same time. The numbers in brackets show that the majority of the companies in all three operations have no employees. In each operation, there is one company that provides almost all employees and a few companies with a very low number of employees. For example, of the 21 OP1 companies only 3 have reported any employees, and of those two have reported only one employee, with the main company in the operation providing the other 38 employees.

In total, the three operations have revenue of 202.5M €, net income of 23M €, and EBITDA of 24.7M €. These amounts are lower bounds since we do not have financial data for every year for each company. We provide the period covered for each individual company in Tables 6.6–6.10. OP2 is the most profitable operation with net income of 11M € and EBITDA of 12.3M €. OP1 ranks second with profits of 8.2M €and OP3 third with profits of 3.8M €. Thus, the number of companies in the operation does not directly influence its financial data as OP2 has the fewest companies, followed by OP1 and OP3. In each operation, there is a small subset of companies that brings the most revenue. For example, OP1 has 26 companies but five of them are responsible for 93% of the total revenue and 98% of the total net income. Similarly, 3 OP2 companies and 5 OP3 companies are responsible for 99% and 95% of the total revenue of those operations. We examine the individual companies of each operation in Sections 6.6.1–6.6.3.

**Expenses.** The large difference between revenue and net income in all operations indicates large expenses. The expenses data comes from the financial reports submitted by the companies. While many categories exist, the declared expenses are typically under one of three categories: personnel, supplies, and a generic other costs. The per-

| Category | OP1.C02 | OP2.C08 | OP3.C18 |
|---|---|---|---|
| Personnel | 205K (<1%) | 5.7M ( 6%) | 2.7M (28%) |
| Advertising | - | 64.9M (72%) | 2.9M (30%) |
| Supplies | 39.3M (75%) | - | - |
| Other | 5.2M (10%) | 8.0M (9%) | 2.6M (27%) |

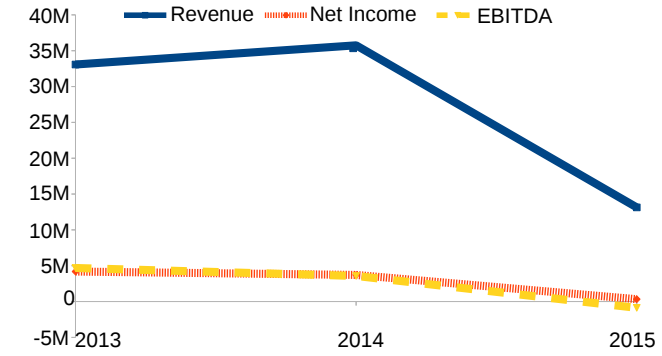
Table 6.5: Expenses of the 3 audited companies for 2013–15. Percentages are calculated over the company's yearly revenue.

sonnel expenses for the whole operation are highest for OP3 reaching 17% (5M) of the total revenue of the operation, followed by OP1 with 9% (7.5M) and OP2 with 6% (6M). Unfortunately, the other two categories are too generic to understand the nature of the expenses. For the three companies required to have their financial statements audited by a CPA firm, i.e., the ones running the PPI services, the audit reports provide more detail into the generic other costs category. That generic category may include, among others, advertisement, office rentals, maintenance, insurance fees, banking fees, taxes, and provisions for losses. Table 6.5 summarizes the expenses declared in the audit reports by the three audited companies. Although the main business activity of all three companies is to operate a PPI service, the declared expenses differ significantly. OP1.C02 declares that 75% of the revenue is spent in (unspecified) supplies. However, some parts of the audit report label these supplies as external services provided to the company. For OP2.C08 and OP3.C18, the largest expenses are in advertising, which correspond to 72% of all revenue for OP2.C08 and 30% for OP3.C18. We suspect that the supplies expenses in OP1.C02 and the advertising expenses in OP2.C08 and OP3.C18 include the payments to the PPI affiliates. Overall, the data indicates commercial PPI services have high expenses and low margins.

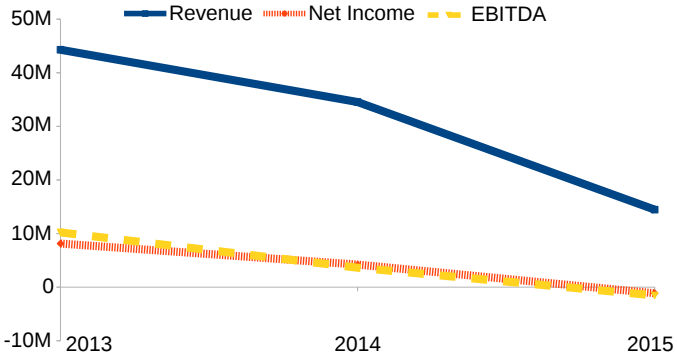
**Evolution over time.** Figure 6.8 shows how the revenue, net income, and EBITDA of each operation has evolved in the period 2013–2015. The figures illustrate the large gap between revenue and net income (or EBITDA). In fact, OP1 and OP2 have losses in 2015, despite revenue of over 10M each. OP3 has losses in both 2014 and 2015, although the EBITDA is slightly positive in both years. Overall, the trend is that in 2014 revenue stabilized with respect to 2013 with OP1 showing a small increase, OP3 a small decrease, and OP2 a larger decrease. Then, in 2015 all 3 operations have a steep decrease in revenue.

We know that on June 2014 Symantec announced that their AV engines would start to flag PUP [39], that Microsoft enabled stricter PUP detection rules on July 1st 2014 [40], and that on August 2014 Google introduced policies against PUP in SafeBrowsing [41]. Since we only have yearly financial data it is possible that those events are responsible for the drop in revenue and profits, which does not clearly manifest in Figure 6.8 in 2014 because the PPI market was still growing during the first half of 2014.

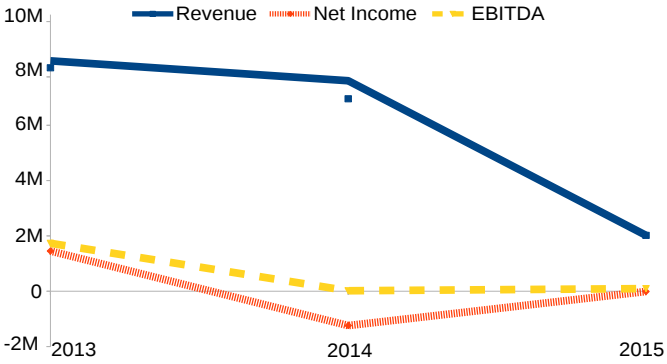
We further investigate this assumption using the VirusShare repository. Figure 6.9




(a) OP1




(b) OP2




(c) OP3

Figure 6.8: Economic data for the three PUP operations for the period 2013-15.

shows the fraction of samples in VirusShare that belong to each of the three operations over the period 2013–15 (using the AVClass family classification and VirusTotal first seen timestamp). For the two largest operations (i.e., OP1, OP2) we observe growth until May–June 2014, followed by a steep drop in June–August 2014. A drop on the PPI samples observed in the wild is an indication that fewer programs are distributed through PPI services, which in turn indicates less revenue for the PPI services.

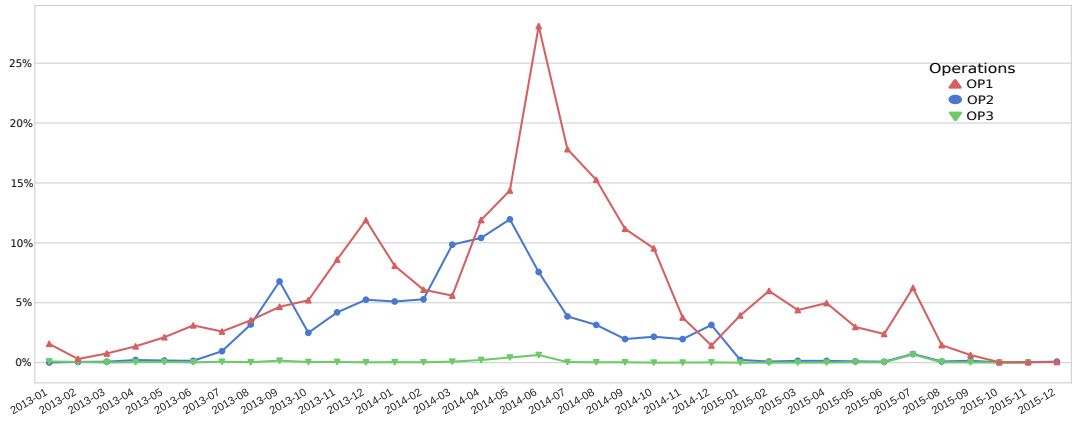



Figure 6.9: Percentage of samples in VirusShare that belong to each of the three operations for the period 2013–2015, over the total number of samples collected by VirusShare in that month. The two largest operations show growth until Summer 2014 where the number of samples sharply declines and does not later recover.

While correlation in time does not necessarily mean causality, we also observe that the 2014 audit report of OP3.C18 identifies as a main business risk the changes in the policies of both Google and Microsoft, which it states can significantly affect the installations of their customers products. The audit also mentions that their R&D department plans to recover from the losses by developing better techniques that can address the installation demands of their clients.

Thus, we conclude that improved PUP defenses deployed by different vendors in mid-2014 significantly impacted the PPI market, which did not recover afterwards.

**Web presence.** We check if the companies have a website using three sources. First, the business reports may include the company's official website. Second, we query search engines using the company names. Third, we search for domains belonging to the companies in the certificate transparency logs. From the business reports and search engines we identify that 8 of the 68 companies have an official website. From the CT logs, we identify 31 HTTPS certificates for 6 companies containing 42 domains[2]. From these 42 domains, only four have a website, in each case describing a product rather than a company, e.g., a registry cleaner offered by OP2.C11. The results show that only a minority of the companies have a Web presence. This is surprising since according to the declared type of activity they provide Internet services like website development or online marketing. The lack of Web presence, in addition to the lack of employees and the minimal business activity, indicates that most of the companies are shell companies.

[2]Some certificates contain additional domains in the Subject Alternative Name extension.

| Company | Creation | Diss. | Cert. | Web | Period | Emp. | Revenue € | Net Inc. € | EBITDA € |
|---|---|---|---|---|---|---|---|---|---|
| OP1.C00 | 03/09 | - | ✗ | ✓ | 2013-15 | 20-38 | 11.9 M | 2.2 M | -923 K |
| OP1.C01 | 03/10 | - | ✓ | ✗ | 2013-15 | 1 | 8.6 M | 285.5 K | 399.3 K |
| OP1.C02 | 06/10 | - | ✓ | ✓ | 2012-15 | 0-1 | 52.6 M | 6.0 M | 7.7 M |
| OP1.C03 | 10/11 | - | ✓ | ✗ | 2014-15 | 0 | 2.1 M | 237.0 K | 371.1 K |
| OP1.C04 | 02/13 | - | ✓ | ✗ | 2014-15 | 0 | 2.0 M | 96.7 K | 193.6 K |
| OP1.C05 | 08/13 | - | ✓ | ✗ | 2014-15 | 0 | 3.1 M | 106.4 K | 220.2 K |
| OP1.C06 | 12/13 | - | ✗ | ✗ | 2014-15 | 0 | 33.7 K | 5.2 K | 7.2 K |
| OP1.C07 | 03/14 | - | ✓ | ✗ | 2014-15 | 0 | 384.1 K | -14.6 K | -6.2 K |
| OP1.C08 | 05/14 | - | ✓ | ✗ | 2014-15 | 0 | 648.4 K | -485.2 K | -643.9 K |
| OP1.C09 | 05/14 | 01/17 | ✓ | ✗ | 2014-15 | 0 | | -16.5 K | |
| OP1.C10 | 06/14 | - | ✓ | ✗ | 2014-15 | 0 | 277.5 K | -57 | 8.1 K |
| OP1.C11 | 06/14 | - | ✓ | ✗ | 2014-15 | 0 | | -2.7 K | |
| OP1.C12 | 06/14 | - | ✓ | ✗ | 2014-15 | 0 | 59.9 K | -8.5 K | -8.9 K |
| OP1.C13 | 08/14 | 01/17 | ✓ | ✗ | 2014-15 | 0 | 26.6 K | -1.5 K | -1.9 K |
| OP1.C14 | 09/14 | - | ✓ | ✗ | 2014-15 | 0 | 538 | -1.3 K | -1.8 K |
| OP1.C15 | 09/14 | - | ✓ | ✗ | 2014-15 | 0 | 85.8 K | 3.2 K | 2.1 K |
| OP1.C16 | 10/14 | - | ✓ | ✗ | 2015 | 0 | 2.3 K | -1.5 K | -2.1 K |
| OP1.C17 | 10/14 | 01/17 | ✓ | ✗ | 2015 | 0 | | -4.3 K | |
| OP1.C18 | 10/14 | - | ✓ | ✓ | 2015 | 0 | | -2.5 K | |
| OP1.C19 | 10/14 | - | ✓ | ✗ | 2015 | 0 | | -185.6 K | |
| OP1.C24 | 04/16 | - | ✗ | ✗ | | | | | |
| **Total** | | | | | **2012-15** | **0-38** | **81.8 M** | **8.2 M** | **7.3 M** |

Table 6.6: OP1 financial data.

### 6.6.1 OP1 Economics

Table 6.6 summarizes the financial data obtained from the business reports of each OP1 company. The left part of the table contains general company data: the creation and dissolution (if any) dates, whether the company has been issued at least one code signing certificate, and whether the company has any website (corporate or product). The right part of the table shows financial data: the period covered in the business report, the number of employees reported, the revenue, the net income, and the EBITDA. From the 21 companies, 5 companies do not report any revenue, and one was created in 2016 and thus had not filed any financial report.

The lead company in the operation is OP1.C02 with 52.6M in revenue and 6M in net income. These correspond to 64% of the total revenue and 73% of the net income of the whole operation. In general, the companies created before 2014 show significant activity, while companies created in 2014–2015 have little business activity and mostly report losses. The three largest companies by revenue are the ones that report some employees, but the top company (OP1.C02) reports a single employee.

The content of OP1's download portals shows that different companies in the operation run the portals. The companies behind the download portals are the older ones, created before May 2014, and thus the ones that have largest activity.

**Audit reports.** We have acquired the 2014 and 2015 audit reports for OP1.C02, which were performed by Audalia Laes Nexia [207] and AFP Audit & Consulting [208] respectively. Unfortunately, these audit reports do not include as much information as the audit reports for the other two operations. In particular, they do not detail how much specific products and services are contributing to the bottom line. The audit re-

| Category | 2013 | 2014 | 2015 | Total |
|---|---|---|---|---|
| PPI | 39.4M (91%) | 34M (99%) | 10M (76%) | **83.4M** |
| Mobile Adv. | 2.0M ( 5%) | 34K (<1%) | - | **2.0M** |
| Down. Portal | 1.4M ( 3%) | - | - | **1.4M** |
| Stream. Portal | - | - | 3.1M (24%) | **3.1M** |
| Rogueware | - | 4K (<1%) | 41K (<1%) | **45.1K** |
| Other | 0.5M ( 1%) | 80K (<1%) | - | **0.6M** |

Table 6.7: OP2.C08 revenue split. Percentages are calculated over the company's yearly revenue.

| Company | Creation | Diss. | Cert. | Web | Period | Emp. | Revenue € | Net Inc. € | EBITDA € |
|---|---|---|---|---|---|---|---|---|---|
| OP2.C05 | 12/07 | - | ✓ | ✓ | 2013-14 | 0-1 | 1.3 M | 86.1 K | 154.0 K |
| OP2.C07 | 03/09 | 10/12 | ✗ | ✓ | 2009-10 | 0-2 | | -200.4 K | |
| OP2.C08 | 03/11 | - | ✓ | ✓ | 2013-15 | 58 | 90.6 M | 11.3 M | 12.2 M |
| OP2.C09 | 10/13 | - | ✓ | ✗ | 2014-15 | 0 | 88.5 K | 3.0 K | 4.3 K |
| OP2.C10 | 10/13 | - | ✓ | ✗ | 2014-15 | 0 | 209 K | 5.9 K | 14.4 K |
| OP2.C11 | 12/13 | - | ✓ | ✓ | 2014-15 | 0 | 41.8 K | -2.3 K | -2.7 K |
| OP2.C12 | 04/14 | - | ✓ | ✗ | 2014-15 | 0 | 4.9 K | -460 | -579 |
| OP2.C13 | 04/14 | - | ✓ | ✓ | 2014-15 | 0 | 4.7 K | -471 | -624 |
| OP2.C14 | 04/14 | - | ✓ | ✗ | 2014-15 | 0 | 1.8 K | -887 | -1.2 K |
| OP2.C15 | 05/14 | - | ✓ | ✗ | 2014-15 | 0 | 1.7 K | 679 | -930 |
| OP2.C16 | 05/14 | - | ✗ | ✗ | 2014-15 | 0-5 | 1.1 M | -69.4 K | -55.5 K |
| OP2.C17 | 05/14 | - | ✗ | ✗ | 2015 | 0 | | -91.6 K | |
| **Total** | | | | | **2013-15** | **0-58** | **92.2 M** | **11.0 M** | **12.3 M** |

Table 6.8: OP2 financial data.

port identifies 21 companies in the corporate group. OP1.C02 reports transactions of 1.9M with 11 of the companies in the corporate group, which are typically services provided to the auditee by those other companies.

### 6.6.2 OP2 Economics

Table 6.8 shows the financial data for the 12 OP2 companies registered in Spain, with the same structure as Table 6.6. Of those 12 companies, two report no revenue.

The lead company in the operation is OP2.C08 with 90.6M in revenue and 11.3M in net income for the period 2013–2015. These correspond to 98% of the total revenue and 100% of the net income of the operation. Similar to OP1, the companies that report employees are the ones with most business activity, save for OP2.C07 that does not report any revenue.

**Audit reports.** We have acquired the 2014 and 2015 audit report of OP2.C08, both performed by PricewaterhouseCoopers [205]. The two audit reports cover the 2013–2015 period. The audit reports contain a detailed revenue split, summarized in Table 6.7. The revenue split reveals that the main source of revenue is the PPI service, which generates revenue of 83.4M for the period 2013–15. This is 92% of the total revenue of OP2.C08, and 90% of the total revenue of the operation in that period. The

| Category | 2013 | 2014 | Total |
|---|---|---|---|
| PPI | 5.6M (87%) | 2.3M (68%) | **7.9M** |
| Advertising | 0.4M ( 6%) | 0.3M (10%) | **0.7M** |
| Software | 44K (<1%) | - | **44K** |
| Other | 0.4M ( 7%) | 0.7M (22%) | **1.1M** |

Table 6.9: OP3.C18 revenue split. Percentages are calculated over the company's yearly revenue.

next largest revenue source is a video streaming service launched in 2015. The video streaming service targets the US and offers a free 5-day unlimited content trial that automatically renews to $59.95 per month when the trial ends (compared to Netflix $9 monthly fee for a similar service). Other relevant sources of income are mobile advertising and a download portal that generated 5% and 3% of the 2013 revenue, respectively, but did not generate significant revenue in 2014–2015.

The audit reports also split the revenue of OP2.C08 by geographical area. From the 90.6M of revenue, 2.5% (2.3M) comes from Spain, 10% (9.1M) from other countries in the European Union, and 87.5% (79.2M) from the rest of the world. Thus, most business comes from outside Spain and is produced in US dollars. We believe this split represents where the advertisers using the PPI service to promote their programs originate from.

Finally, the audit report shows transactions of 10.7M with 9 other OP2 companies. The largest transactions are performed with OP2.C20, which is registered in Israel. No transactions are reported with the other Israel-based company or with the company registered in Delaware, US.

### 6.6.3 OP3 Economics

Table 6.10 shows the financial data for the 27 OP3 companies registered in Spain, with the same structure as Tables 6.6–6.8. Of those 27 companies, 5 have no business reports and 10 report no revenue.

The largest company by revenue is OP3.C09 with 10.6M in 2008–2009, but the central company in the operation is OP3.C18, which has most employees and runs the PPI service. OP3.C18 has 9.7M in revenue and 443K in net income during 2013–2014, which represent 34% of the total revenue and 11% of the net income. Compared to the other operations, the revenue of OP3 is more diversified and less reliant on the revenue of the PPI service. Once again, the companies that report employees are the most active ones.

**Audit report.** We have acquired the 2014 audit report for OP3.C18, performed by Deloitte [206]. There is no audit report for 2015. The revenue split in the audit report is summarized in Table 6.9. The table shows that most revenue (7.9M) comes from the PPI service, which represents 80% of the revenue for OP3.C18 and 28% of the revenue

| Company | Creation | Diss. | Cert. | Web | Period | Emp. | Revenue € | Net Inc. € | EBITDA € |
|---|---|---|---|---|---|---|---|---|---|
| OP3.C00 | 07/03 | - | ✓ | ✗ | 2013-14 | 0 | | -107.4 K | |
| OP3.C01 | 10/03 | - | ✗ | ✗ | 2013-14 | 0 | 6.2 K | 141.1 K | |
| OP3.C02 | 10/03 | - | ✓ | ✗ | 2013-14 | 1 | 156.3 K | 22.3 K | -48.3 K |
| OP3.C03 | 10/03 | - | ✗ | ✗ | 2013-14 | 0 | | 124.9 K | |
| OP3.C04 | 11/03 | - | ✓ | ✗ | 2014-15 | 0 | | -17.0 K | |
| OP3.C05 | 02/05 | - | ✗ | ✗ | 2009-10 | 0-5 | 224.0 K | 923.1 K | -475.8 K |
| OP3.C06 | 02/05 | - | ✗ | ✓ | | | | | |
| OP3.C09 | 03/06 | 03/14 | ✗ | ✓ | 2008-09 | 0 | 10.6 M | 2.6 M | 3.8 M |
| OP3.C10 | 05/07 | - | ✗ | ✓ | 2014-15 | 12 | 3.8 M | 345.4 K | 335.6 K |
| OP3.C11 | 01/08 | - | ✓ | ✗ | | | | | |
| OP3.C12 | 03/08 | 02/16 | ✗ | ✗ | 2013-14 | 0 | | -17.8 K | |
| OP3.C13 | 10/09 | - | ✗ | ✗ | | | | | |
| OP3.C14 | 11/09 | 02/16 | ✓ | ✗ | 2013-14 | 0 | | -6.6 K | |
| OP3.C16 | 06/10 | - | ✗ | ✗ | 2013-14 | 0 | | -10.3 K | |
| OP3.C17 | 12/10 | - | ✓ | ✓ | | | | | |
| OP3.C18 | 05/11 | - | ✓ | ✓ | 2013-14 | 31 | 9.7 M | 443.6 K | 1.7 M |
| OP3.C19 | 07/11 | - | ✓ | ✗ | 2013-14 | 6 | 3.3 M | 82.0 K | 144.3 K |
| OP3.C23 | 09/12 | - | ✓ | ✓ | | | | | |
| OP3.C24 | 11/12 | - | ✗ | ✗ | 2013-14 | 0-1 | 405.6 K | -60.2 K | -50.1 K |
| OP3.C27 | 05/13 | - | ✗ | ✗ | 2013-14 | 0 | 28.2 K | 52.4 K | -52.4 K |
| OP3.C28 | 07/13 | 02/16 | ✗ | ✗ | 2013 | 0 | | -1.2 K | |
| OP3.C29 | 07/13 | - | ✗ | ✗ | 2013-14 | 0 | 68.8 K | -2.7 K | -2.7 K |
| OP3.C30 | 07/13 | 02/16 | ✗ | ✗ | 2013 | 0 | | -1.2 K | |
| OP3.C31 | 07/13 | 02/16 | ✗ | ✗ | 2013-14 | 0 | 47.0 K | -5.7 K | -5.7 K |
| OP3.C32 | 07/13 | 02/16 | ✗ | ✗ | 2013 | 0 | | -1.2 K | |
| OP3.C34 | | - | ✗ | ✓ | 2014-15 | 1 | 12.9 K | -729.3 K | -345.9 K |
| OP3.C36 | | - | ✗ | ✗ | 2013-14 | 0-8 | 191.2 K | 56.8 K | 126.2 K |
| **Total** | | | | | **2008-14** | **0-31** | **28.5 M** | **3.8 M** | **5.1 M** |

Table 6.10: OP3 financial data.

of the whole operation. Advertising provides an additional 7% of the revenue, while software revenue is minimal (44K in 2013). Of the 9.7M revenue of OP3.C18, 8% (824K) comes from Spain, 20% (1.9M) from other countries in the European Union, and 72% (7M) from the rest of the world. Similar to OP2, most revenue comes from outside Spain. This geographical split likely represents where advertisers using the PPI service for distribution come from. The company has transactions of 1.3M with 6 other OP3 companies. No transactions are reported with the 5 companies registered in Delaware, US.

## 6.7 Discussion

This section discusses different aspects of the operations and limitations of our approach.

**Defenses.** The economic analysis of malicious and undesirable operations has two main applications: evaluating the deployment of defenses and proposing new defenses [29]. Our results are useful towards the first goal by demonstrating the impact

on the PPI market of PUP defenses deployed in mid-2014 by different vendors. A possible defense using entity graphs would be that once the persons behind an operation are identified using an entity graph, company registers could be periodically queried to find new companies created by those persons and put them in a watchlist. Those watchlists could be used by CAs for identifying certificate requests from PUP operations.

**Shell companies.** Our analysis shows that the three operations employ a large number of companies, but most of them have no employees, use the address of other companies, report no revenue, and have no Web presence. While we cannot be certain of the purpose of such shell companies, we do observe them being used to obtain code signing certificates that are later used to sign PUP samples.

**False positives.** Our approach to generate entity graphs went through successive iterations to define the trimming steps to avoid including unrelated persons and operations. The resulting entity graphs have undergone extensive manual curation by the authors to verify that no unrelated entities are included. While our approach to build entity graphs can be applied to automatically produce entity graphs for other (Spain-based) operations, given the high cost of wrong attribution, we recommend the final entity graphs are manually curated, as we did, to guarantee that no unrelated entities are included.

**Financial data trustworthiness.** Our economic analysis is based on the yearly financial statements filed by the companies, and a few audit reports by CPA firms. A limitation of this approach is that it is possible for companies to falsify their results in financial statements, e.g., for fiscal reasons [212, 213]. However, such manipulation constitutes a fraud in countries like Spain that mandate yearly financial statements. Verifying the accuracy of financial data is a complex task that requires full access to the finances of a company and is outside the scope of this work.

**Certificates.** We observe operations using 48–85 code signing certificates, but those are rarely revoked by CAs. This raises the question of why large numbers of certificates are needed. We believe that certificate changes help evading detection by security products. Specifically, it is common for AV engines to include detection signatures that focus on the certificate information, e.g., the signature may correspond to the subset of the certificate's Subject field that captures the company name. Re-signing a program with a clean certificate for another company bypasses those signatures. We have performed experiments (not detailed in the paper) demonstrating that by removing the certificate chain from a detected PUP sample, the number of AVs detecting the sample reduces significantly. In addition, new companies can be used to reset the reputation for PUP programs. For example, changing the name of a program and its publisher (i.e., certificate) makes it difficult for a user to check if a suspicious program has already been reported by other users as undesirable.

# 7 Conclusions & Future Work

Early work on PUP from 2005 has focused on the deceptive methods employed by PUP and tried to give preliminary definitions of the term. Since then, the PUP ecosystem remained unexplored by the academia, and largely ignored as a threat by the industry. This allowed PUP publishers to expand their operation by developing novel distribution and monetization mechanisms. Almost a decade later, indications appear of the PUP prominence quickly increasing over time to the point where in 2015, PUP samples outnumber malware samples in malware feeds [19,20] and in Google Safe Browsing [4]. This thesis addresses this gap by empirically and systematically analyzing in both breadth and depth the PUP abuse, prevalence, distribution, and economics.

**Prevalence.** Despite the early evidence of PUP prominence on malware feeds, the impact of PUP on real users was unknown. Our work measured PUP prevalence on real consumer and enterprise hosts using millions of hosts from various geographic regions. Our results showed that PUP affects 54% of the analyzed consumer hosts. We estimated the affected Internet-connected consumer hosts to be two orders of magnitude higher than our measurements, reaching 210M hosts. On the other hand, PUP prevalence on enterprise hosts is much lower compared to consumer hosts. Enterprises encounter malware much more often than PUP. This can be due to stricter enterprise security policies about what programs can be installed, which may affect PUP but not malware.

**Distribution.** The high PUP prevalence is a result of the effective PUP distribution mechanisms. Our analysis of the commercial PPI ecosystem reveals a dynamic ecosystem with a large number of PPI services that play a major role in PUP distribution. We also verify that malware distribution through commercial PPI services is very limited. This makes sense considering the fact that distribution through silent PPI services can be an order of magnitude cheaper.

**Abuse.** It is common for PUP to be digitally signed, this way the code looks more benign and helps convincing users to complete the installation process. Code signing on Windows requires PUP publishers to obtain valid code signing certificates by CAs. Our work evaluated existing CA defenses against PUP and malware and concluded that current defenses are largely ineffective for PUP. CA identity checks pose some barrier to malware, but do not affect PUP. Revocation is equally low for malware and PUP certificates. Moreover, we witness PUP heavily leveraging certificate polymorphism in order to trick CAs to issue them new certificates while their previous certificates have been revoked. Certificate polymorphism is applied by using multiple enterprise or individual identities or by performing simple modifications on their company names (e.g., capitalization, introducing commas and spaces). This demonstrates that even simple tricks can undermine the effectiveness of the CA identity checks.

Finally, we discover PUP families taking advantage of a problematic scenario on Windows code signing that allows their timestamped executables to remain valid even after the revocation of their certificate. We propose as a solution to this problem the concept of hard revocation. Hard revocation can be immediately used by CAs without any additional cost or change in the existing revocation procedures.

**Economics.** Our work reveals many operational and economical details of PUP that can be used for evaluating existing PUP defenses. We show that a small number of people is managing a large number of shell companies that are used for obtaining code signing certificates. We measure profitability of three large Spanish-based PUP operations and present both their revenues and net incomes. We analyze the revenue sources and reveal that PPI services is the main revenue source, while download portals and rogueware software is of secondary importance. Also, we witness a large drop of both PUP revenue and net income in mid-2014 that we attribute to the improved PUP defenses deployed by various vendors that have significantly impacted the PPI market, which did not recover afterwards.

**Future Work.** This thesis reveals the central role of commercial PPI services in the distribution of PUP. It also demonstrates that the PPI market has been significantly impacted forcing some of these services to either seize operation or discover new distribution mechanisms. Recent work has analyzed the online survey scam ecosystem and discovered that survey scams are predominantly used for PUP distribution. This illustrates the arms race between defenders and PUP publishers as well as the need of constant monitoring of the evolving PUP ecosystem.

Moreover, our work provides evidence that PUP publishers are moving from Windows to mobile platforms such as Android and iOS. Although this thesis focused on Windows, future work should extend PUP analysis on mobile platforms.

PUP host-based detection remains an open challenge. This is due to three big limitations of the current malware detector systems. First, PUP detection requires fine grained detection approaches, while current detector systems are binary, flagging an

executable either as benign or malware. Second, PUP is being distributed as installers and current dynamic analysis approaches often fail to complete the installation process in an automated way. Third, current detector systems do not analyze installation and uninstallation together, so they cannot establish a relationship between the two phases. Detector systems should be able to examine if an unistallation process is complete by verifying that all installed applications have been completely removed from the system. Thus, future work should focus on building new detection techniques tailored to PUP specific characteristics.

# Funding Acknowledgments

This research was partially supported by the European Union Horizon 2020 ElasTest (ICT-10-2016-731535) project; by the Regional Government of Madrid through the BLOQUES-CM (S2018/TCS-4339) and N-GREENS Software-CM (S2013/ICE-2731) projects; and by the Spanish Government through the MINECO DEDETIS (TIN2015-7013-R) and StrongSoft (TIN2012-39391-C04-01) projects. This research is supported by the EIE Funds of the European Union.

# Bibliography


[1] "Cinstaller." http://cinstaller.com/.

[2] "CashMyLinks." http://www.cashmylinks.com/.

[3] "PerInstallBucks." https://perinstallbucks.com/.

[4] K. Thomas, J. A. E. Crespo, R. Rastil, J.-M. Picodi, L. Ballard, M. A. Rajab, N. Provos, E. Bursztein, and D. Mccoy, "Investigating Commercial Pay-Per-Install and the Distribution of Unwanted Software," in *USENIX Security Symposium*, 2016.

[5] A. Nappa, R. Johnson, L. Bilge, J. Caballero, and T. Dumitras, "The attack of the clones: A study of the impact of shared code on vulnerability patching.," in *IEEE Symposium on Security and Privacy*, pp. 692–708, 2015.

[6] CheckPoint, "Fireball – the chinese malware of 250 million computers infected," 2017. https://blog.checkpoint.com/2017/06/01/fireball-chinese-malware-250-million-infection/.

[7] L. Abrams, "Microsoft warnds adware developers to stop using man-in-the-middle attacks," 2015. https://www.bleepingcomputer.com/news/microsoft/microsoft-warns-adware-developers-to-stop-using-man-in-the-middle-attacks/.

[8] L. Abrams, "Adware uploads screenshot of your active windows without your permission," 2016. https://www.bleepingcomputer.com/news/security/adware-uploads-screenshot-of-your-active-windows-without-your-permission/.

[9] V. Valeros, "In plain sight: Credential and data stealing adware," 2016. https://blogs.cisco.com/security/in-plain-

sight-credential-and-data-stealing-adware?CAMPAIGN=Security&Country_Site=us&POSITION=Social+Media&REFERRING_SITE=Twitter&CREATIVE=CiscoSecurity.

[10] L. Abrams, "Wajam browser add-on serves malvertising," 2016. https://blog.malwarebytes.com/threat-analysis/2016/02/wajam-browser-add-on-serves-malvertising/.

[11] T. Micro, "On the actors behind mevade/sefnit," 2014. http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp-on-the-actors-behind-mevade-sefnit.pdf.

[12] L. Abrams, "Adware bundlers pushing cryptocurrency miners on unsuspecting victims," 2016. https://www.bleepingcomputer.com/news/security/adware-bundlers-pushing-cryptocurrency-miners-on-unsuspecting-victims/.

[13] L. Abrams, "Microsoft drops the hammer on coercive registry cleaners & system optimizers," 2018. https://www.bleepingcomputer.com/news/security/ask-com-toolbar-updater-abused-to-download-malware://www.bleepingcomputer.com/news/microsoft/microsoft-drops-the-hammer-on-coercive-registry-cleaners-and-system-optimizers/.

[14] Microsoft, "How Microsoft antimalware products identify malware and unwanted software," 2018. https://www.microsoft.com/en-us/wdsi/antimalware-support/malware-and-unwanted-software-evaluation-criteria.

[15] Google, "Unwanted Software Policy," 2018. https://www.google.com/about/unwanted-software-policy.html.

[16] McAfee, "McAfee Potentially Unwanted Programs (PUP) Policy," 2018. https://www.mcafee.com/enterprise/en-us/assets/legal/pup-policy.pdf.

[17] P. Security, "Malware still generated at a rate of 160,000 new samples a day in Q2 2014." http://www.pandasecurity.com/mediacenter/press-releases/malware-still-generated-rate-160000-new-samples-day-q2-2014/.

[18] K. Thomas, E. Bursztein, C. Grier, G. Ho, N. Jagpal, A. Kapravelos, D. McCoy, A. Nappa, V. Paxson, P. Pearce, N. Provos, and M. A. Rajab, "Ad Injection at Scale: Assessing Deceptive Advertisement Modifications," in *IEEE Symposium on Security and Privacy*, May 2015.

[19] P. Kotzias, S. Matic, R. Rivera, and J. Caballero, "Certified PUP: Abuse in Authenticode Code Signing," in *ACM Conference on Computer and Communication Security*, 2015.

[20] C. Lever, P. Kotzias, D. Balzarotti, J. Caballero, and M. Antonakakis, "A Lustrum of Malware Network Communication: Evolution and Insights," in *2017 IEEE Symposium on Security and Privacy*, pp. 788–804, May 2017.

[21] N. Provos, P. Mavrommatis, M. A. Rajab, and F. Monrose, "All Your Iframes Point To Us," in *USENIX Security Symposium*, (San Jose, CA), July 2008.

[22] Grier et al., "Manufacturing Compromise: The Emergence Of Exploit-as-a-service," in *ACM Conference on Computer and Communications Security*, (Raleigh, NC), October 2012.

[23] L. Heddings, "Yes, Every Freeware Download Site is Serving Crapware," Jan. https://www.howtogeek.com/207692/yes-every-freeware-download-site-is-serving-crapware-heres-the-proof/, accessed November 2017.

[24] L. Heddings, "Here is what happens when you install the top 10 download.com apps.." http://www.howtogeek.com/198622/heres-what-happens-when-youinstall-the-top-10-download.com-apps, accessed November 2017.

[25] EmsiSoft, "Mind the PUP: Top download portals to avoid." http://blog.emsisoft.com/2015/03/11/mind-the-pup-top-download-portals-to-avoid/, accessed December 2016.

[26] R. Rivera, P. Kotzias, A. Sudhodanan, and J. Caballero, "Costly Freeware: A Systematic Analysis of Abuse in Download Portals," *IET Information Security*, June 2018.

[27] A. Geniola, M. Antikainen, and T. Aura, "A Large-Scale Analysis of Download Portals and Freeware Installers," in *Nordic Conference on Secure IT Systems*, 2017.

[28] J. Caballero, C. Grier, C. Kreibich, and V. Paxson, "Measuring Pay-per-Install: The Commoditization of Malware Distribution," in *USENIX Security Symposium*, 2011.

[29] K. Thomas, D. Yuxing, H. David, W. Elie, B. C. Grier, T. J. Holt, C. Kruegel, D. Mccoy, S. Savage, and G. Vigna, "Framing Dependencies Introduced by Underground Commoditization," in *Workshop on Economics of Information Security*, 2015.

[30] D. McCoy, A. Pitsillidis, G. Jordan, N. Weaver, C. Kreibich, B. Krebs, G. M. Voelker, S. Savage, and K. Levchenko, "Pharmaleaks: Understanding the business of online pharmaceutical affiliate programs," in *Proceedings of the 21st USENIX conference on Security symposium*, pp. 1–1, USENIX Association, 2012.

[31] M. A. Rajab, L. Ballard, P. Mavrommatis, N. Provos, and X. Zhao, "The nocebo effect on the web: An analysis of fake anti-virus distribution.," in *LEET*, 2010.

[32] D. Y. Huang, M. M. Aliapoulios, V. G. Li, L. Invernizzi, E. Bursztein, K. McRoberts, J. Levin, K. Levchenko, A. C. Snoeren, and D. McCoy, "Tracking ransomware end-to-end," in *2018 IEEE Symposium on Security and Privacy (SP)*, pp. 618–631, IEEE, 2018.

[33] ANA and WhiteOps, "The bot baseline: Fraud in digital advertising 2017 report," 2017. https://www.ana.net/content/show/id/botfraud-2017.

[34] J. Finkle, "Inside a global cybercrime ring," 2010. https://www.reuters.com/article/us-technology-scareware/inside-a-global-cybercrime-ring-idUSTRE62N29T20100324.

[35] B. F. Rubin, "Perion sees soaring 2014 earnings following merger," 2014. https://www.wsj.com/articles/perion-sees-soaring-2014-earnings-following-merger-1393869981.

[36] Microsoft, "Windows authenticode portable executable signature format," Mar. 21 2008. http://download.microsoft.com/download/9/c/5/9c5b2167-8017-4bae-9fde-d599bac8184a/Authenticode_PE.docx.

[37] P. Kotzias, L. Bilge, and J. Caballero, "Measuring PUP Prevalence and PUP Distribution through Pay-Per-Install Services," in *USENIX Security Symposium*, August 2016.

[38] "Investment in cyber security by industry sector." https://www.savoystewart.co.uk/blog/firms-investment-on-cyber-security-by-industry.

[39] "Symantec security products will soon detect and remove Potentially Unwanted Programs (PUPs)." http://botcrawl.com/symantec-security-products-will-soon-detect-and-remove-potentially-unwanted-programs-pups/.

[40] "Adware: A new approach," April 2017. https://blogs.technet.microsoft.com/mmpc/2014/04/02/adware-a-new-approach/.

[41] "That's not the download you're looking for...." https://security.googleblog.com/2014/08/thats-not-download-youre-looking-for.html.

[42] B. Edelman, "Spyware Installation Methods." http://www.benedelman.org/spyware/installations/.

[43] N. Good, R. Dhamija, J. Grossklags, D. Thaw, S. Aronowitz, D. Mulligan, and J. Konstan, "Stopping Spyware at the Gate: A User Study of Privacy, Notice and Spyware," in *ACM Symposium on Usable Privacy and Security*, 2005.

[44] N. S. Good, J. Grossklags, D. K. Mulligan, and J. A. Konstan, "Noticing Notice: A Large-Scale Experiment on the Timing of Software License Agreements," in *ACM SIGCHI Conference on Human Factors in Computing Systems*, 2007.

[45] C. Pickard and S. Miladinov, "Rogue software: Protection against potentially unwanted applications," in *Malicious and Unwanted Software (MALWARE), 2012 7th International Conference on*, pp. 1–8, IEEE, 2012.

[46] T. Urban, D. Tatang, T. Holz, and N. Pohlmann, "Towards understanding privacy implications of adware and potentially unwanted programs," in *European Symposium on Research in Computer Security*, pp. 449–469, Springer, 2018.

[47] T. Nelms, R. Perdisci, M. Antonakakis, and M. Ahamad, "Towards Measuring and Mitigating Social Engineering Malware Download Attacks," in *USENIX Security Symposium*, August 2016.

[48] A. Kharraz, W. Robertson, and E. Kirda, "Surveylance: Automatically detecting online survey scams," in *2018 IEEE Symposium on Security and Privacy (SP)*, pp. 70–86, IEEE, 2018.

[49] N. Jagpal, E. Dingle, J.-P. Gravel, P. Mavrommatis, N. Provos, M. A. Rajab, and K. Thomas, "Trends and Lessons from Three Years Fighting Malicious Extensions," in *USENIX Security Symposium*, 2015.

[50] S. Banescu, A. Pretschner, D. Battré, S. Cazzulani, R. Shield, and G. Thompson, "Software-based protection against changeware," in *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*, pp. 231–242, ACM, 2015.

[51] B. J. Kwon, J. Mondal, J. Jang, L. Bilge, and T. Dumitras, "The Dropper Effect: Insights into Malware Distribution with Downloader Graph Analytics," in *ACM Conference on Computer and Communications Security*, October 2015.

[52] B. J. Kwon, V. Srinivas, A. Deshpande, and T. Dumitras, "Catching Worms, Trojan Horses and PUPs: Unsupervised Detection of Silent Delivery Campaigns," in *Network and Distributed Systems Security Symposium*, February 2017.

[53] B. Andow, A. Nadkarni, B. Bassett, W. Enck, and T. Xie, "A study of grayware on google play," in *2016 IEEE Security and Privacy Workshops (SPW)*, pp. 224–233, IEEE, 2016.

[54] I. Ideses and A. Neuberger, "Adware detection and privacy control in mobile devices," in *2014 IEEE 28th Convention of Electrical & Electronics Engineers in Israel (IEEEI)*, pp. 1–5, IEEE, 2014.

[55] A. Moschuk, T. Bragin, S. D. Gribble, and H. Levy, "A Crawler-based Study of Spyware in the Web," in *Network and Distributed System Security Symposium*, (San Diego, CA), 2006.

[56] J. Caballero, C. Grier, C. Kreibich, and V. Paxson, "Measuring Pay-per-Install: The Commoditization of Malware Distribution," in *Proceedings of the 20th USENIX Security Symposium*, (San Francisco, CA, USA), August 2011.

[57] B. J. Kwon, J. Mondal, J. Jang, L. Bilge, and T. Dumitras, "The Dropper Effect: Insights into Malware Distribution with Downloader Graph Analytics," in *ACM SIGSAC Conference on Computer and Communications Security*, 2015.

[58] A. Apvrille, D. Gordon, S. Hallyn, M. Pourzandi, and V. Roy, "Digsig: Runtime authentication of binaries at kernel level," in *USENIX Conference on System Administration*, 2004.

[59] G. Wurster and P. C. van Oorschot, "Self-signed Executables: Restricting Replacement of Program Binaries by Malware," in *USENIX Workshop on Hot Topics in Security*, 2007.

[60] Y. Wu and R. H. C. Yap, "Towards a Binary Integrity System for Windows," in *ACM Symposium on Information, Computer and Communications Security*, 2011.

[61] C. Gates, N. Li, J. Chen, and R. Proctor, "CodeShield: Towards Personalized Application Whitelisting," in *Annual Computer Security Applications Conference*, 2012.

[62] D. Barrera, D. McCarney, J. Clark, and P. van Oorschot, "Baton: Certificate Agility for Android's Decentralized Signing Infrastructure," in *ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*, 2014.

[63] M. Wood, "Want my autograph? The use and abuse of digital signatures by malware," in *Virus Bulletin Conference*, 2010.

[64] J. Niemala, "It's signed, therefore it's clean, right?," May 2010. Presentation at the CARO 2010 Workshop.

[65] D. Kim, B. J. Kwon, and T. Dumitraş, "Certified malware: Measuring breaches of trust in the windows code-signing pki," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1435–1448, ACM, 2017.

[66] I. Glucksmann, "Injecting custom payload into signed Windows executables," in *REcon*, 2012.

[67] E. Law, "Caveats for Authenticode Code Signing," September 2014. http://blogs.msdn.com/b/ieinternals/archive/2014/09/04/personalizing-installers-using-unauthenticated-data-inside-authenticode-signed-binaries.aspx.

[68] D. Stevens, "Playing with authenticode and md5 collisions," 2009. http://blog.didierstevens.com/2009/01/17/playing-with-authenticode-and-md5-collisions/.

[69] J. Zhang, Z. Durumeric, M. Bailey, M. Liu, and M. Karir, "On the Mismanagement and Maliciousness of Networks," in *NDSS*, 2014.

[70] B. Edwards, J. Jacobs, and S. Forrest, "Risky Business: Assessing Security with External Measurements." https://bjedwards.github.io/assets/papers/EdwardsRisky.pdf, August 2016.

[71] "Microsoft Security Intelligence Report," 2018. https://info.microsoft.com/rs/157-GQE-382/images/EN-US_CNTNT-eBook-SIR-volume-23_March2018.pdf.

[72] "Internet Security Threat Report," 2018. https://www.symantec.com/security-center.

[73] "Threats Report by McAfee Labs," 2018. https://www.mcafee.com/enterprise/en-us/assets/infographics/infographic-threats-report-mar-2018.pdf.

[74] L. Bilge, Y. Han, and M. Dell'Amico, "Riskteller: Predicting the risk of cyber incidents," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, 2017.

[75] J. Levine, R. LaBella, H. Owen, D. Contis, and B. Culver, "The Use of Honeynets to Detect Exploited Systems across Large Enterprise Networks," in *IEEE Information Assurance Workshop*, 2003.

[76] T.-F. Yen, A. Oprea, K. Onarlioglu, T. Leetham, W. Robertson, A. Juels, and E. Kirda, "Beehive: Large-Scale Log Analysis for Detecting Suspicious Activity in Enterprise Networks," in *Annual Computer Security Applications Conference*, 2013.

[77] A. Oprea, Z. Li, T.-F. Yen, S. H. Chin, and S. Alrwais, "Detection of Early-Stage Enterprise Infection by Mining Large-Scale Log Data," in *Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2015.

[78] P. D. McDaniel, S. Sen, O. Spatscheck, J. E. van der Merwe, W. Aiello, and C. R. Kalmanek, "Enterprise Security: A Community of Interest Based Approach," in *Network and Distributed Security Symposium*, 2006.

[79] B. Stone-Gross, R. Abman, R. A. Kemmerer, C. Kruegel, D. G. Steigerwald, and G. Vigna, "The Underground Economy of Fake Antivirus Software," in *Workshop on Economics of Information Security*, 2011.

[80] K. Thomas, D. McCoy, C. Grier, A. Kolcz, and V. Paxson, "Trafficking Fraudulent Accounts: The Role of the Underground Market in Twitter Spam and Abuse," in *USENIX Security Symposium*, 2013.

[81] P. Pearce, V. Dave, C. Grier, K. Levchenko, S. Guha, D. McCoy, V. Paxson, S. Savage, and G. M. Voelker, "Characterizing Large-Scale Click Fraud in Zeroaccess," in *ACM SIGSAC Conference on Computer and Communications Security*, 2014.

[82] K. Liao, Z. Zhao, A. Doupé, and G.-J. Ahn, "Behind Closed Doors: Measurement and Analysis of CryptoLocker Ransoms in Bitcoin," in *APWG Symposium on Electronic Crime Research*, 2016.

[83] Z. Li, Q. Liao, and A. Striegel, "Botnet Economics: Uncertainty Matters," in *Workshop on Economics of Information Security*, 2008.

[84] C. Herley and D. Florêncio, "Nobody Sells Gold for the Price of Silver: Dishonesty, Uncertainty and the Underground Economy," in *Workshop on Economics of Information Security*, 2009.

[85] R. Anderson, C. Barton, R. Böhme, R. Clayton, M. J. Van Eeten, M. Levi, T. Moore, and S. Savage, "Measuring the Cost of Cybercrime," in *Workshop on Economics of Information Security*, 2012.

[86] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. M. Voelker, and S. Savage, "Re: CAPTCHAs-Understanding CAPTCHA-Solving Services in an Economic Context," in *USENIX Security Symposium*, 2010.

[87] K. Thomas, D. Iatskiv, E. Bursztein, T. Pietraszek, C. Grier, and D. McCoy, "Dialing Back Abuse on Phone Verified Accounts," in *ACM SIGSAC Conference on Computer and Communications Security*, 2014.

[88] G. Stringhini, M. Egele, C. Kruegel, and G. Vigna, "Poultry Markets: On the Underground Economy of Twitter Followers," in *ACM Workshop on Online Social Networks*, 2012.

[89] E. De Cristofaro, A. Friedman, G. Jourjon, M. A. Kaafar, and M. Z. Shafiq, "Paying for Likes?: Understanding Facebook Like Fraud using Honeypots," in *ACM Internet Measurement Conference*, 2014.

[90] K. Kozák, B. J. Kwon, D. Kim, C. Gates, and T. Dumitraş, "Issued for abuse: Measuring the underground trade in code signing certificate," *arXiv preprint arXiv:1803.02931*, 2018.

[91] MSDN, ""Stranger Danger" - Introducing SmartScreen Application Reputation." http://blogs.msdn.com/b/ie/archive/2010/10/13/stranger-danger-introducing-smartscreen-application-reputation.aspx.

[92] "Stuxnet Under the Microscope." http://www.eset.com/us/resources/white-papers/Stuxnet_Under_the_Microscope.pdf.

[93] "Unveiling Careto - The Masked APT." http://kasperskycontenthub.com/wp-content/uploads/sites/43/vlpdfs/unveilingthemask_v1.0.pdf.

[94] "Malware Analysis Report - W64/Regin, Stage 1." https://www.f-secure.com/documents/996508/1030745/w64_regin_stage_1.pdf.

[95] "Ccss forum: Common computing security standards." http://www.ccssforum.org/.

[96] "VirusTotal." http://www.virustotal.com/.

[97] B. Kaliski, "Pkcs7: Cryptographic message syntax version 1.5." RFC 2315 (Proposed Standard), 1998.

[98] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet x.509 public key infrastructure certificate and certificate revocation list (crl) profile." RFC 5280 (Proposed Standard), 2008. Updated by RFC 6818.

[99] S. Santesson, M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP." RFC 6960 (Proposed Standard), June 2013.

[100] "Catalog files and digital signatures." https://msdn.microsoft.com/en-us/library/windows/hardware/ff537872%28v=vs.85%29.aspx.

[101] "Practical windows code and driver signing." http://www.davidegrayson.com/signing/.

[102] "Allowing only signed application to run." https://technet.microsoft.com/en-us/library/dd723683%28v=ws.10%29.aspx.

[103] MSDN, "Driver Signing Policy." https://msdn.microsoft.com/en-us/library/windows/hardware/ff548231.aspx.

[104] "Cross-certificates for kernel mode code signing." https://msdn.microsoft.com/en-us/library/windows/hardware/dn170454%28v=vs.85%29.aspx.

[105] "Ca security council." https://casecurity.org/.

[106] "Ca/browser forum." https://cabforum.org/.

[107] G. Wicherski, "pehash: A novel approach to fast malware clustering," in *2nd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2009.

[108] A. Nappa, M. Z. Rafique, and J. Caballero, "The MALICIA Dataset: Identification and Analysis of Drive-by Download Operations," *International Journal of Information Security*, vol. 14, pp. 15–33, February 2015.

[109] Netcraft, "Keys left unchanged in many Heartbleed replacement certificates!," April 2014. http://news.netcraft.com/archives/2014/05/09/keys-left-unchanged-in-many-heartbleed-replacement-certificates.html.

[110] "Malwarebytes PUP Reconsideration Information," April 2014. http://blogs.technet.com/b/mmpc/archive/2014/04/03/adware-a-new-approach.aspx.

[111] "Malwarebytes PUP Reconsideration Information." https://www.malwarebytes.org/pup/.

[112] M. Bailey, J. Oberheide, J. Andersen, Z. Mao, F. Jahanian, and J. Nazario, "Automated Classification and Analysis of Internet Malware," in *RAID*, September 2007.

[113] A. Mohaisen and O. Alrawi, "AV-Meter: An Evaluation of Antivirus Scans and Labels," in *Detection of Intrusions and Malware, and Vulnerability Assessment*, July 2014.

[114] "Virusshare.com repository." http://virusshare.com/.

[115] M. Labs, "Threat Report," November 2014. http://www.mcafee.com/us/resources/reports/rp-quarterly-threat-q3-2014.pdf.

[116] Z. Durumeric, J. Kasten, M. Bailey, and J. A. Halderman, "Analysis of the HTTPS Certificate Ecosystem," in *ACM Internet Measurement Conference*, 2013.

[117] C. Brubaker, S. Jana, B. Ray, S. Khurshid, and V. Shmatikov, "Using Frankencerts for Automated Adversarial Testing of Certificate Validation in SSL/TLS Implementations," in *IEEE Symposium on Security & Privacy*, 2014.

[118] R. Housley, W. Ford, W. Polk, and D. Solo, "Rfc 2459: Internet x. 509 public key infrastructure certificate and crl profile," 1999.

[119] "Signtool." https://msdn.microsoft.com/en-us/library/windows/desktop/aa387764%28v=vs.85%29.aspx.

[120] T. H.-J. Kim, L.-S. Huang, A. Perring, C. Jackson, and V. Gligor, "Accountable key infrastructure (aki): A proposal for a public-key validation infrastructure," in *International Conference on World Wide Web*, 2013.

[121] "Download.com." http://www.download.com/.

[122] "Softonic." www.softonic.com.

[123] "InstallMonetizer." http://www.installmonetizer.com/.

[124] T. Dumitraş and D. Shou, "Toward a Standard Benchmark for Computer Security Research: The Worldwide Intelligence Network Environment (WINE)," in *EuroSys Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*, April 2011.

[125] T. Dumitras and P. Efstathopoulos, "The Provenance Of Wine," in *European Dependable Computing Conference*, May 2012.

[126] "Public Suffix List." https://publicsuffix.org/.

[127] "Opswat Antivirus and Threat Report," January 2014. https://www.opswat.com/resources/reports/antivirus-january-2014.org/.

[128] L. Bilge and T. Dumitras, "Before We Knew It: An Empirical Study of Zero-day Attacks in the Real World," in *ACM Conference on Computer and Communications Security*, 2012.

[129] S. Davidoff, "Interview with an adware author," 2009. http://philosecurity.org/2009/01/12/interview-with-an-adware-author.

[130] "Internet Archive WayBack Machine." https://archive.org/web/.

[131] CodeFuel, "7 reasons codefuel beats all other pay per install companies," 2015. http://www.codefuel.com/blog/7-reasons-perion-codefuel-beats-all-other-pay-per-install-companies/.

[132] "Sterkly." http://www.sterkly.com/.

[133] "RevenueHits." https://web.archive.org/web/20130805140617/http://www.revenuehits.com/.

[134] "InstalleRex." https://installerex.com/.

[135] "InstallCore." https://www.installcore.com/.

[136] "Open Candy." http://opencandy.com/.

[137] M. Geary, "Adknowledge apps distribution opportunities," 2013. http://ppitalk.com/showthread.php/49-Adknowledge-Apps-Distribution-Opportunities.

[138] "NativeX." http://nativex.com/.

[139] "BetterInstaller." http://betterinstaller.somotoinc.com/.

[140] "Solimba." https://solimba.com/.

[141] "DomaIQ." http://www.domaiq.com/en/.

[142] "Download Admin." https://web.archive.org/web/20140208040640/http://www.downloadadmin.com/.

[143] "AirSoftware." https://airinstaller.com/.

[144] "Oneinstaller." https://web.archive.org/web/20150220020855/http://oneinstaller.com/.

[145] "installPath." http://www.installpath.com.

[146] "InstallBay." http://www.visibay.com/installbay.

[147] "RevenYou." http://www.revenyou.com/.

[148] "Verti." http://www.vertitechnologygroup.com.

[149] "Smart WebAds." http://www.smartwebads.com/.

[150] "Nosibay." http://www.nosibay.com/.

[151] "ConversionAds." https://web.archive.org/web/20160217095842/http://www.conversionads.com/.

[152] "Installertech." http://www.installertech.com/.

[153] "7install." https://web.archive.org/web/20160306081435/http://7install.com/.

[154] "InstallMonster." http://installmonster.ru/en.

[155] "AdGazelle." http://adgazelle.com/.

[156] "EarnPerInstall." https://web.archive.org/web/20160419013909/http://www.earnperinstall.com/.

[157] "GuppyGo." http://www.guppygo.com/.

[158] "Installaxy." https://web.archive.org/web/20151105011933/http://installaxy.com/.

[159] "Mediakings." https://web.archive.org/web/20140517213640/http://media-kings.com/.

[160] "Net Cash Revenue." http://netcashrevenue.com/.

[161] "PayPerInstall." http://payperinstall.com/.

[162] "Perinstallbox." http://www.setupbundle.com/index.php.

[163] "PerInstallCash." http://www.perinstallcash.com/.

[164] "Purebits." http://purebits.net/.

[165] M. Sebastian, R. Rivera, P. Kotzias, and J. Caballero, "Avclass: A tool for massive malware labeling," in *Research in Attacks, Intrusions, and Defenses*, pp. 230–253, 2016.

[166] O. Java, "What are the Ask Toolbars?." https://www.java.com/en/download/faq/ask_toolbar.xml.

[167] D. Goodin, "Lenovo pcs ship with man-in-the-middle adware that breaks https connections," 2015. http://arstechnica.com/security/2015/02/lenovo-pcs-ship-with-man-in-the-middle-adware-that-breaks-https-connections/.

[168] "Top 15 data breaches in 2018." http://www.businessinsider.fr/us/data-breaches-2018-4.

[169] "Ransom.wannacry." https://www.symantec.com/security-center/writeup/2017-051310-3522-99, 2017.

[170] "Petya ransomware outbreak: Here's what you need to know." https://www.symantec.com/blogs/threat-intelligence/petya-ransomware-wiper, 2017.

[171] T.-F. Yen, V. Heorhiadi, A. Oprea, M. K. Reiter, and A. Juels, "An Epidemiological Study of Malware Encounters in a Large Enterprise," in *ACM SIGSAC Conference on Computer and Communications Security*, 2014.

[172] Y. Liu, A. Sarabi, J. Zhang, P. Naghizadeh, M. Karir, M. Bailey, and M. Liu, "Cloudy with a Chance of Breach: Forecasting Cyber Security Incidents," in *24th USENIX Security Symposium (USENIX Security 15)*, (Washington, D.C.), pp. 1009–1024, USENIX Association, 2015.

[173] Y. Liu, J. Zhang, A. Sarabi, M. Liu, M. Karir, and M. Bailey, "Predicting Cyber Security Incidents Using Feature-Based Characterization of Network-Level Malicious Activities," in *Proceedings of the 2015 ACM International Workshop on International Workshop on Security and Privacy Analytics*, IWSPA '15, (New York, NY, USA), pp. 3–9, ACM, 2015.

[174] "National vulnerability database." https://nvd.nist.gov/.

[175] Z. Durumeric, J. Kasten, D. Adrian, J. A. Halderman, M. Bailey, F. Li, N. Weaver, J. Amann, J. Beekman, M. Payer, and V. Paxson, "The matter of Heartbleed," in *Proceedings of the Internet Measurement Conference*, (Vancouver, Canada), Nov 2014.

[176] 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)*, (Austin, TX), pp. 1033–1050, USENIX Association, 2016.

[177] D. Springall, Z. Durumeric, and J. A. Halderman, "FTP: The Forgotten Cloud," in *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 503–513, June 2016.

[178] "KMSPico." http://kmspico10.com/.

[179] "Censys." https://censys.io/.

[180] Abuse.ch, "Fighting malware and botnets." http://www.team-cymru.com/.

[181] "Team Cymru." http://www.team-cymru.com/.

[182] Internet Storm Center, "DShield DNSBL." https://www.dshield.org/.

[183] "Phishtank." https://www.phishtank.com/.

[184] The ShadowServer Foundation, "ShadowServer." https://www.shadowserver.org/.

[185] Spamhaus, "The Spamhaus Project." https://www.spamhaus.org/.

[186] "Uceprotect DNSBLs." http://www.uceprotect.net/.

[187] Rapid7 Labs, "Forward DNS (FDNS)." https://opendata.rapid7.com/sonar.fdns_v2/.

[188] R. Anderson, C. Barton, R. Böhme, R. Clayton, M. J. G. van Eeten, M. Levi, T. Moore, and S. Savage, "Measuring the Cost of Cybercrime," in *Workshop on the Economics of Information Security*, (San Francisco, CA, USA), May 2013.

[189] L. Metcalf and J. M. Spring, "Blacklist Ecosystem Analysis: Spanning Jan 2012 to Jun 2014," in *Proceedings of the 2Nd ACM Workshop on Information Sharing and Collaborative Security*, WISCS '15, pp. 13–22, ACM, 2015.

[190] M. Kührer, C. Rossow, and T. Holz, "Paint it Black: Evaluating the Effectiveness of Malware Blacklists," in *Proceedings of the 17th International Symposium on Research in Attacks, Intrusions and Defenses*, September 2014.

[191] S. Sinha, M. Bailey, and F. Jahanian, "Shades of Grey: On the Effectiveness of Reputation-based "blacklists"," in *Proceedings of the 3rd International Conference on Malicious and Unwanted Software (MALWARE '08)*, (Fairfax, Virginia, USA), pp. 57–64, October 2008.

[192] A. Pitsillidis, C. Kanich, G. M. Voelker, K. Levchenko, and S. Savage, "Taster's Choice: A Comparative Analysis of Spam Feeds," in *Proceedings of the 2012 Internet Measurement Conference*, IMC '12, pp. 427–440, ACM, 2012.

[193] C. A. Shue, A. J. Kalafut, and M. Gupta, "Abnormally Malicious Autonomous Systems and Their Internet Connectivity," *IEEE/ACM Trans. Netw.*, vol. 20, pp. 220–230, Feb. 2012.

[194] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata, "Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median," *Journal of Experimental Social Psychology*, vol. 49, no. 4, pp. 764 – 766, 2013.

[195] R. Phillips, "What Makes Delaware an Onshore Tax Haven," December 2015. http://www.taxjusticeblog.org/archive/2015/12/what_makes_delaware_an_onshore.php.

[196] "Handelsregister - Commercial Register of Germany." https://www.handelsregister.de/rp_web/welcome.do.

[197] "Israeli Corporations Authority." havarot.justice.gov.ilIIs.

[198] "Boletín Oficial del Registro Mercantil (BORME)." https://www.boe.es/diario_borme/.

[199] "Electronic Data Gathering, Analysis, and Retrieval (EDGAR)." https://www.sec.gov/edgar/searchedgar/companysearch.html.

[200] "Registro Mercantil Central." http://www.rmc.es/.

[201] C. Horwath, "Country by Country Financial Reporting and Auditing Framework Spain," 2014. https://www.crowehorwath.net/uploadedfiles/crowe-horwath-global/services/audit/financial_reporting_frameworks/financial%20reporting%20-%20spain%20may%202014.pdf.

[202] "LibreBorme." https://libreborme.net/.

[203] "Requisitos para auditoria obligatoria." http://bcsconsultoresasociados.blogspot.com.es/2012/01/requisitos-para-auditoria-obligatoria.html.

[204] "Infocif - Informes de empresas." http://www.infocif.es/.

[205] "PricewaterhouseCoopers." https://www.pwc.com/.

[206] "Deloitte." https://www2.deloitte.com/global/en.html.

[207] "Audalia Laes Nexia." http://www.audalialaesnexia.com/en/.

[208] "AFP Audit and Consulting." http://www.afpaudit.com/.

[209] "Herdprotect." http://www.herdprotect.com/index.aspx.

[210] "Certificate Transparency." https://www.certificate-transparency.org/.

[211] "The Open Graph Viz Platform." https://gephi.org/.

[212] "Financial Statement Manipulation An ever-present problem for investors." http://www.investopedia.com/articles/fundamental-analysis/financial-statement-manipulation.asp.

[213] "Here's why you cannot trust a company's financials." http://fortune.com/2015/10/19/auditors-financial-reports/.