Vanity, Cracks and Malware

Insights into the Anti-Copy Protection Ecosystem

Markus Kammerstetter, Christian Platzer and Gilbert Wondracek
Vienna University of Technology
Vienna, Austria
{mk,cplatzer,gilbert}@iseclab.org

ABSTRACT

Today, a large amount of software products include mechanisms to counter software piracy. However, most protection mechanisms can be easily circumvented by applying software patches (*cracks*) or license key generators (*keygens*) with seemingly no financial incentives. Our research shows that the distribution of cracks and keygens not only allows miscreants to generate revenue (e.g. through advertising or malware infections), but it also leads to high risks for the end-users of pirated software. We collected more than 43,900 download links and analyzed more than 23,100 (3,551 unique) real-world cracks, showing that these tools are heavily used by criminals to spread malware. Our results indicate that even state of the art virus scanners can not fully protect users from these threats. Moreover, we conducted a manual analysis, showing how many cracks and keygens actually work and how much effort is necessary to acquire them. In addition, we made our data-set publicly available to the research community.

Categories and Subject Descriptors

K.6.5 [**Management of computing and information systems**]: Security and Protection—*Invasive software*; C.2.0 [**Computer-Communication Networks**]: GeneralSecurity and Protection; D4.6 [**Operating Systems**]: Security and Protection; Invasive software

General Terms

Economics, Measurement, Security

Keywords

Piracy, Malware, Internet measurements, Underground economy

1. INTRODUCTION

Software copyright infringement or *software piracy* is a serious threat to commercial software developers worldwide. While deriving exact numbers is hard, it is commonly assumed that the accumulated economic damage from software copyright infringement ranges in the order of several tens of billions of dollars [15, 22].

As a consequence, software manufacturers regularly integrate technical protection mechanisms in their products that aim at preventing unauthorized copying and redistribution. Examples of such protection mechanisms include serial number or license key checks that are performed when installing the protected software products. Some manufacturers even go as far as requiring special hardware called "dongles" to be present in the user's system to successfully run the software.

While these and similar countermeasures have been established in the industry since more than three decades [28], a recent study covering 116 countries [15] claims that about 42% of the globally installed software products are "pirated".

Even considered conservatively, this indicates the relative ineffectiveness of these measures and concurrently supports the claim that software piracy is a very common phenomenon. On a technical level, hackers and criminals typically try to circumvent anti-piracy measures by creating and distributing specific software patches or even custom reverse-engineered serial number generators for commercial products, called *cracks* and *keygens* in the scene jargon.

Users who look for "cracked" software products can typically access either the unmodified original software through established filesharing systems like Bittorrent, One-Click-Hosters (OCH), or via Usenet downloads, or download bundles of the original software and cracks or keygens. Additionally, some types of cracks target legitimate evaluation copies of software products that are published by the actual manufacturer to attract potential customers. In these cases, the attacker's aim is to emulate or unlock restricted features, thus upgrading the evaluation version to the full-priced software product.

Interestingly, relatively little academic research has been published in recent years on this topic. In particular, anecdotal evidence strongly hints at criminals using cracks and keygens as an attack vector to distribute malicious software such as trojans or viruses.

In this work, we first elaborate on how we collect a large test-set of real-world cracks and keygens, and then perform a number of experiments to evaluate the maliciousness of this software. We demonstrate that, in addition to the direct economic damage caused by software copyright infringe-

ment, there are significant risks for many end-users who use pirated software. Finally, we conclude that there is a rewarding opportunity for malware researchers and anti-virus companies to collect novel malware samples from sources related to software piracy. Furthermore, we show that the circumvention of copy protections is embedded in an active ecosystem, describing its actors and interdependencies. In particular, the main contributions of this work are:

- We collected more than 43,900 download links and performed automated and manual analysis on more than 23,100 (3,551 unique) resulting real-world crack downloads, providing deep insights into how cracks and keygens are used to spread malware.
- To gain a systematical understanding of the ecosystem surrounding cracks and keygens, we describe the motivations and possible sources of revenue of its participants.
- We performed a dynamic behavior analysis of infected binary samples to determine *how malicious* these samples really are and present results.
- We made our data-set publicly available to the research community for download to drive future research in that direction.

1.1 Motivation

Recent publications [24, 27, 26, 25] have shown that websites and binaries related to software piracy are more likely to be infected with malware. This is supported by findings on the thriving underground economy and its links to software piracy, which has grown into a profitable, global business [17, 18, 19]. Gullible users who download anti copy-protection tools such as cracks and key generators are among the typical targets of shady business practices and cybercrime. Criminals covertly infect their systems with malware and transform them into malicious bots that are controlled by criminals to commit or support attacks or sending spam. Given the large scale of software piracy, we decided to further investigate on the risks that users are facing through the use of these tools, the possible monetary gains for the criminals, and the supporting roles and actors in the piracy ecosystem.

2. RELATED WORK

In 2008, Iklinci, Holz and Freiling introduced the honeyclient system *Monkey-Spider* [24] to detect malicious websites. Their results show that web content specific to piracy or games is more likely to be malicious than benign content and domains related to anti-copy protection tools are among the most malicious domains. However, they do not specifically target anti-copy protection tools such as cracks and key generators.

Moshchuk, Bragin et al. did a Crawler-based Study of Spyware on the Web [27]. Similar to our approach, they employ a virtual machine-based analysis platform to scrutinize collected executables. Their study proves that there is a strong correlation between software piracy and Spyware infections, but unlike our work, does not focus on cracks and keygens.

In a sociological study [26], Limayem et al. show that there are many factors motivating software piracy and that the usual software pirate is in fact no computer expert. Their work indicates that software piracy is not only a very common problem, but also suggests that due to the limited amount of computer literacy, systems of software pirates might not be protected well enough to combat malware found in cracks and keygens.

In [25], Gantz et al. investigate the security risks involved with using cracks and key generators, showing that the vast amount of these tools is infected with malware. Although they use an approach similar to our manual analysis approach, their analysis concentrates on a relatively small set of samples. In contrast, our work covers the underground economy aspects and employs both manual and automated collection and analysis approaches on a large set of samples.

3. THE SOFTWARE PIRACY ECOSYSTEM

While *cracks* and *key generators* are specific tools to circumvent copy protections, they only represent a small fraction of the software piracy ecosystem. In the following, we give an overview of the involved actors and groups as well as their incentives and interdependencies.

3.1 Warez Groups

Underground Warez groups focus on the distribution of protected or copyrighted media, such as movies, music, or software ("warez") in the scene.

Usually, individual members of these groups are also the authors of cracks and keygens [23]. Unlike typical cybercriminals, for example, botnet operators, the primary motivation of these groups is not monetary gain. Rather, it seems to be some type of "vanity contest", based on competition among different warez groups, with the aim of being the first - and thus, most respected - group to release a crack or keygen for a specific software product [23, 21, 29].

To be able to actually write cracks and keygens, access to the original software products is necessary. To this end, warez groups share original media that they acquired among each other, usually via private sites and servers. This is significantly different to public end-users, who have to resort to public file-sharing networks like One Click Hosters (OCHs), peer-to-peer networks or binary Usenet groups [23, 20]. As we will see later, the intermediate distribution steps between warez groups and the end-users of pirated media allow other miscreants to spread malware and gain monetary profits.

3.2 Hosting Providers

To distribute cracks and keygens to end-users, third-party hosting providers are used by warez groups. There are several types of such hosting providers, typically they all allow revenue to be made through the subsequent downloads of cracks and keygens. *Pay-Per-Download (PPD)* hosters such as Sharecash [12] or Honeycontent [7] allow publishers to earn money by sharing files or links to content. If users want to access the content on a PPD hoster, they usually need to complete manual tasks, like filling out surveys, to unlock the download. The advertisement fees for these surveys are then distributed among the PPD hoster and the content publisher. *One-Click Hosters (OCHs)* follow a similar scheme: Content publishers can upload files, but downloading users either have to pay for premium accounts or they are punished with slow download speeds, long waiting times or other limitations supported by advertising. *Dedicated Crack Hosters* like Crackstorage.net are websites that specialize in

hosting cracks and keygens. Typically, they do not allow visitors to search for the stored files, which can be considered a counter-measure against crawling or enumeration of their content. Instead, downloads are available via direct, deep links, for example obtained through a third party search engine like Astalavista.box.sk [2] or forums. Through advertising and by partnering with indexing sites, they can generate revenue from crack and keygen downloads. *Malware* hosters typically set up sites for hosting malware, often disguised as cracks or keygens. Unsuspecting users then download and execute the malware and thus generate revenue for the malware hoster. Other hosting types like Usenet newsgroups or P2P networking do not directly generate revenue through cracks and keygens. However, through combinations with other hosting types like PPD hosting, the uploader can still make money. Recent studies show that contributions to P2P networks can be profit-driven as well, for instance by seeding content that includes advertisements for web sites or private BitTorrent portals [18].

3.3 Malware Distributors

The distribution of malware has grown into a booming underground economy. A recent study [17] states that due to market forces in the underground economy, a significant amount of malware no longer incorporates spreading mechanisms on its own. Instead, a whole service culture has evolved, leading to providers that offer a wide range of specialized services ranging from malware or packaging toolkits for Anti-Virus evasion to infection services (*Pay-Per-Install*). In fact, out of the world's top 20 most prevalent families of malware, 12 employ PPI services to buy infections [17]. In general, a PPI provider receives malware executables from clients and charges money for a requested number of malware installs. The actual installs are mostly done through financially compensated affiliates that focus on different infection vectors. Our results indicate that uploading infected cracks and keygens is at least one of them. Since the malware comes from clients, it can be virtually anything, ranging from botnet bots to ad- and spyware distribution or click fraud. However, in general the income generated through the malware will outweigh the costs for the PPI service.

4. ANALYSIS

Here we describe how we retrieved and analyzed pirated programs and key generators. To provide a realistic set of samples, we utilized several different facilities to gather them. At the same time we aimed to follow roughly the same procedures a real user would perform to download the desired crack.

4.1 Manual Data Acquisition

We followed two approaches to acquire software cracks and key generators. First, we placed ourselves in the role of a user manually searching for a specific crack and key generator. This allowed us to conduct experiments that are related to the user experience (e.g. how much effort is necessary to successfully circumvent a copy protection mechanism) and the risks involved with the execution of anti-copy protection tools.

As there are thousands of different software products and anti-protection tools available, we restricted our search to the most prominent products in the categories games and applications. To this end, we combined commercial rankings (i.e. Amazon.com [1] and Download.cnet.com [6]) with download numbers from Thepiratebay.org [3], one of the top providers for pirated software. A listing of the top five applications and games can be found in Table. 1.

Application	Game
Adobe Photoshop CS5.1	Brink
Ahead Nero 10	Crysis 2
Microsoft Office 2010	Fable III
Norton 360	Portal 2
WinRAR 3.93	The Sims 3

Table 1: Top 5 Applications and Games

Just as any user would, we first utilized the Google search engine to issue a query. We also used Astalavista.box.sk [2], a well known search engine for anti-copy protection tools. From the search results, we followed links that were most likely to directly lead to a crack or key generator for a specific product. We assumed that this was especially the case, when the links included the precise version information for the product we were looking for (e.g. *"Adobe CS5.1 All products Crack By tEAM RED HOT"*). Accompanying our manual analysis (see Section 4.3.1), we manually collected 363 direct download links over a period of 3 months. In total, this led to 242 executable cracks or key generators, from which 141 were unique and 101 were duplicates. We considered an executable a duplicate, if there was another executable with the same MD5 sum within our downloads. The overall download results are visible in Table 8 in the Appendix.

Our approach led to a wide range of different sites with downloads from regular web hosters, dedicated crack hosters like Crackstorage.net, One Click Hosters (OCHs) and the BitTorrent P2P Network. At this point, we observed how many links we had to follow (i.e. the *link depth*) from the search engine result to the actual download. The result is visible in Figure 1. Our observations indicate, that in order to reach 80% of the downloads, a user needs to follow at least 4 links from the actual search result. If up to 3 links are followed, only about 43% of the downloads can be reached.

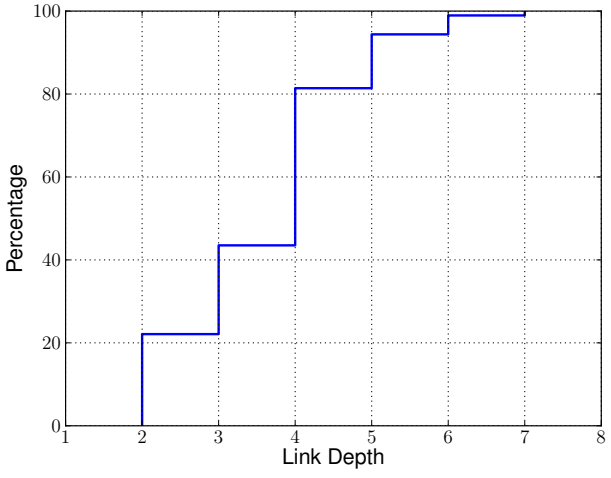

Figure 1: Average Link Depth to reach a Crack/Keygen Download (CDF)

Throughout our observations, we had to follow up to 8 links to finally reach the real download site. The web sites between the search result and the actual download site are frequently either advertising or link collection sites, allowing the site operators to generate additional revenue, presumably with the generated traffic. In general, search engine-powered downloading of cracks and keygens is a cumbersome task.

4.2 Automated Data Acquisition

Following up to the manual approach, we automatically retrieved a high number of cracks and key generators for a wide range of different products. The goal was to gain insight into a wider area of the anti copy protection landscape by analyzing how many of our downloads contained malicious code and whether there is a difference in the infection ratio between games and applications based on their initial source.

To this end, we constructed an automated acquisition system that is shown in Figure 2. The system uses three data sources: The Usenet, One Click Hosters (OCHs) and the BitTorrent network. We chose these data sources due to the potentially high amount of cracks and key generators available on these networks, the accessibility of indexing services, the possibility to automate the collection process and their general pervasiveness. For each of these data sources, we wrote custom crawlers that leverage existing indexing services. Specifically, we wrote crawlers for the Usenet indexing service Nzbindex.nl [11], the OCH search engine Filestube.com [5] and the BitTorrent search engine Isohunt.com [8].

4.2.1 Applications vs. Games

To distinguish between anti-copy protection tools for games and for software applications, we again created a list with the most popular products. Since an automated approach is not subject to the same restrictions as a manual one, we compiled a new list of ranked products which is similar to Table 1 but more comprehensive. The final list comprises a total of 292 non-free software products (151 computer games and 141 applications).

For every product, our crawlers collected download links for the search terms *"[PRODUCT] crack"* and *"[PRODUCT] keygen"*, where *[PRODUCT]* refers to the name of the current software product from our list. Since cracks and key generators are small programs, we limited our search to downloads with 10MB or less. In addition, due to the distributed nature of P2P networks, we only collected BitTorrent download links that had at least 5 seeders. Table 2 shows the crawl results with and without the manually collected download links. In total we collected 43,972 download links (43,609 crawled links and 363 manually collected links). In comparison to the Usenet and OCH download links, the number of crawled BitTorrent links was lower, because many downloads had no seeders and thus were not collected by our crawler in the first place.

For each network, we forwarded the collected download links to a corresponding download client.

4.2.2 Usenet Downloads

For Usenet downloads, we chose to use the NZBGet [10] client. In addition to its easy integrability into our system architecture, it automatically processes the downloads once it is finished. Since Usenet downloads are usually compressed (mostly RAR or ZIP archives), post-processing was required for downloads that included archives. For each decompressed download, we located all potential cracks and keygens (i.e. executable files) and added them to our repository. We discarded any downloads that included no executable files or downloads we could not decompress (i.e. due to corrupted archives or password protection). From 21,574 collected download links, we extracted a total of 17,434 executables, meaning that 80.81% of our download links resulted in an executable file.

	# Web	# OCH	# BT	# Usenet	# Total
Crawler:	0	20,233	1,802	21,574	43,609
Manual:	185	85	93	0	363
Total:	185	20,318	1,895	21,574	43,972

Table 2: Overall Collected Download Links

OCH	# Links	Link %	# DL	DL %
Megashares.com	14,345	70.60	1,121	21.32
Letitbit.net	2,041	10.04	1,928	36.66
Mediafire.com	1,087	5.35	710	13.50
Rapidshare.com	957	4.72	1,036	19.70
4lastfile.com	401	1.97	77	1.46
Uploading.com	171	0.84	133	2.53
Others	1,318	6.48	254	4.83
Total	20,320	100.00	5,259	100.00

Table 3: Distribution of Crack and Keygen Crawl Links and Successful Downloads on OCHs

4.2.3 One Click Hosting Downloads

For OCH downloads, we used the JDownloader [9] download management tool. This Java-based framework supports over 100 different One Click Hosters and comes with an integrated proxy-functionality to facilitate rotating IP addresses. Furthermore, it features an automatic CAPTCHA solver and a built-in decompression tool for the most common archive types, including basic password-guessing for protected archives. While these features were sufficient for most Hosters, we discovered that a significant amount of our downloads were hosted on Letitbit.net [4] (see Table 3). To overcome unacceptably long wait times for this Hoster, we decided to acquire a premium account for Letitbit.net. In contrast to our expectations, with the premium account, we still experienced download speed limitations and waiting times, but in general, the downloads worked as intended. Table 3 shows the distribution of crawl links and successful downloads for One Click Hosters. Out of 20,320 downloads links, 5,259 downloads (25.88%) were successfully retrieved and unpacked. The rest of the downloads was not available for various reasons (i.e. removed downloads, server errors, password protected downloads, etc.).

Interestingly, more than 70% (14,345) of our crawled links were for Megashares.com, but due to the high number of removed downloads, we could only retrieve 7.8% (1,121) of them. We discovered, that the majority of our successful crack and keygen downloads (36.66%) came from Letitbit.net.

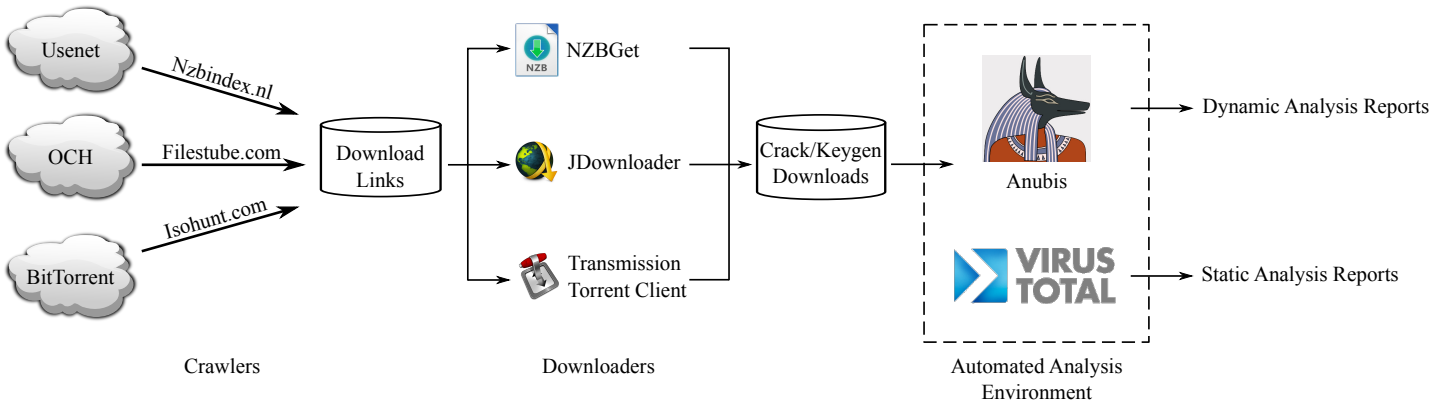

Figure 2: Automated Acquisition and Analysis Architecture

4.2.4 *BitTorrent Downloads*

For BitTorrent downloads, we used the Transmission BitTorrent Client [13] as it includes a remote control interface that allowed integration into our automated acquisition system (Fig. 2). Our Isohunt.com crawler collected and retrieved torrent files for downloads with a size of at most 10MB and with at least 5 seeders. Our crawler scripts immediately added the collected torrent files to the Transmission Client to minimize the time between searching and downloading. In addition, our scripts constantly removed finished downloads from the Transmission Client. Similar to our approach for Usenet and OCH downloads, we used custom post processing scripts to decompress downloads and to add potential crack and keygen executables to our repository. We ran our BitTorrent client for a period of 1 month to ensure that downloads with very few sources could still succeed eventually. However, it turned out, that the number of the seeders returned by Isohunt.com was not up to date and, as a result, many of our downloads had no seeders at all within the given time frame. In total, out of the collected 1,900 torrent files, only 356 (18.74%) downloads succeeded. To avoid distributing copyright-protected or illegal software, we suppressed uploads from our clients altogether. The negative impact on overall download speed was severe but acceptable.

In total, we collected 43,979 download links and obtained 23,131 potential crack and keygen executables.
However, in conjunction with the executable downloads from all other data sources (Web, OCH and BitTorrent), only 3,561 (15.39%) of the overall executables were unique. We used these executables as basis for our further analyses. It is also important to note here, that manually and automatically downloaded samples overlap. To be specific, 43% of the manually downloaded cracks can also be found in the automatically acquired ones. This number suggests that files distributed over various sources are largely the same. Furthermore, the detailed analysis performed on the manually downloaded samples are representative for the larger, automated sample base.

4.3 Analysis Setup

To gain more insight into anti-copy protection software tools, we decided to follow both manual and automatic analysis approaches. While automatic analysis can cover a large amount of samples easily, it also has a number of limitations. For instance, vital information such as whether a specific crack or keygen can actually unlock a software product or whether the software product still runs stable after applying a patch, can not be gained through automatic analysis. The reasons for this are manifold. Most cracks and keygens need to be applied in specific ways and the accompanying instructions need to be closely followed by the user. Some of the tools might not even run out of the box, for example due to missing dependencies. In general, the necessary user interactions closely depend on the anti-copy protection tool and the software application. Similarly, testing whether a tool managed to successfully break the copy-protections and whether the unlocked software product is still in a usable and stable condition afterwards, closely depends on the software product itself. As a result, we decided to combine automatic analysis with manual analysis results. However, the manual approach has limitations on its own. Most notably, due to the extensive amount of manual work involved, only a fraction of all available samples can be analyzed in detail.

	Web	OCH	BT	Usenet	Total
# DL Links	185	20,320	1,900	21,574	43,979
# EXE	82	5,259	356	17,434	23,131
# Unique	67	1,379	230	2,054	3,561

Table 4: Overall Download Results

4.3.1 *Manual Analysis*

We focused our manual analysis approach on the software products that we covered in the manual acquisition phase (see Table 8). Our manual analysis setup is shown in Fig. 3.

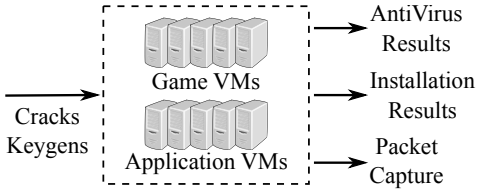

Figure 3: Manual Analysis Setup

For each of the software products, we set up a separate virtual machine, where the product and all its software dependencies (e.g. .NET, DirectX, etc.) were installed. Due to the less stringent security features in comparison to Windows 7, we chose to use Windows XP SP2 32-bit as the operating system for the virtual machines. However, especially for some games, it was required to install Windows 7 instead. Besides, we installed a state of the art virus scanner (AVG free [30]) with up to date virus definitions and real-time (i.e. Resident Shield) protection. We set up each of the software products to the point where either license information had to be entered or the software refused to work due to expired evaluation licenses. After that, we performed a whole system virus scan within all virtual machines to ensure that the system was free from malware infections. In this state, we created a snapshot (which we denote *base snapshot*) for each virtual machine, forming the basis for any subsequent experiments.

We manually tested all of the 141 unique cracks and keygens within our analysis setup. For each anti-copy protection tool, we reverted the according machine to the base snapshot. We started our packet capture tool (Wireshark) on the host machine and then copied the anti-copy protection tool to the virtual machine. Before executing the crack or keygen, we scanned the files with the AV software. If infections could be detected, we noted them and temporarily disabled the AV software. After that, we executed the crack or keygen and precisely followed the accompanying instructions. In some cases, for the anti-copy protection tool to run, it was necessary to resolve software dependencies, make registry modifications or perform multiple steps both in the software product and the crack/keygen. After installation, we checked if the tool actually unlocked the software product and whether the software product was still in a usable and stable running state. This was followed by a full system AV scan, so that we could identify possible malware infections. We distinguished between *file-* and *system infections*: If the AV software discovered infections in the anti-copy protection tool, but running the tool did not infect any other files on the system, we denote this a *file infection*. If other files on the system were infected, we denote this a *system infection*. In order to capture slow network traffic as well, we let each machine run for at least 10 minutes. After that, we shut down the virtual machine, stopped the packet capture tool and analyzed the captured network traffic if applicable. This allowed us to manually collect AV scan results, installation results (i.e. whether the tool could unlock the software product) and packet captures for each crack and keygen. Overall, the manual analysis took between 15 and 20 minutes per single crack or keygen executable. To the best of our knowledge, we are the first who conducted a manual analysis of cracks and keygens to that extent.

4.3.2 Automated Analysis

We analyzed all available unique samples (including the manually acquired ones) in our automated analysis environment (Fig. 2). In order to conduct both static and dynamic analysis, we utilized the Virustotal [14] service and the Anubis [16] environment.

Virustotal [14] is a publicly available service that allows concurrent scanning of submitted files with a large number of AntiVirus scanners. At the time of writing, the service accepted files up to 32MB in size and conducted scans with 43 different AntiVirus scanners. Each AntiVirus scanner reports that either no infection or an infection with a given malware could be identified in the submitted file. We wrote a set of scripts that retrieved existing scan reports for files based on their MD5 sum. This way, we could retrieve existing reports and determine the number of cracks or keygens that were already known to Virustotal. In the next step, we submitted all files to Virustotal and rescanned them in case there were existing analysis reports. The difference between older existing reports and fresh reports is that for fresh reports, the latest virus definitions information is used. Thus it is possible, that old reports did not indicate malware infections since there were no AV signatures for the specific malware at that time, whereas, due to the more recent AV signatures, the fresh scan report would be able to identify these infections. In order to determine whether a file was infected, we calculated the overall percentage of the virus scanners that indicated malware infections. This allowed us to limit the effect of false positives for individual AV scanners. Out of the 3,561 unique crack and keygen executables, we were able to retrieve 2,281 (64.06%) existing reports and 3,471 (97.47%) fresh scan reports. For 90 files (2.53%) we did not receive a report from Virustotal, either because the submitted file was too large (31 files, 0.87%) or due to internal Virustotal issues (59 files, 1.66%).

Anubis is a dynamic analysis environment, originally based on TTAnalyze [16], that performs virtual machine introspection (VMI). It allows to upload executables with up to 8MB in size and considers a wide range of possible threats, including malicious registry and file modification, process creation, or network activities. Based on this information, Anubis creates a detailed analysis report including a *severity score* in the range $\{0, \ldots, 10\}$. Similar to the Virustotal results, the severity score corresponds to the *maliciousness* of the executed sample, whereas a value of 0 indicates no malicious behavior at all and a value of 10 suggests, that the sample is malware. For our experiments, we submitted all cracks and keygen samples with a size of up to 8MB to Anubis. Due to this limitation, we could not analyze 340 (9.55%) out of the 3,561 unique crack and keygen executables. The remaining 3,221 samples resulted in 3,145 (97.64%) valid Anubis reports. Anubis identified 45 samples (1.40%) as no valid Windows PE executables and hence did not generate a report. For 31 samples (0.96%), we received incomplete Anubis reports without severity score.

5. RESULTS AND DISCUSSION

The following Sections show the results we obtained through manual and automated approaches.

5.1 Manually Collected Samples

By following the methodology described in Section 4.1, we were able to download 141 unique cracks and keygens. However, since some of the files appeared for multiple products, it was necessary to test these duplicates for each product in its specific virtual environment. Although the overall number of unique manual downloads is 141, we had to test 157 cracks and keygens.

Table 5 shows the overall results of our manual analysis based on the dataset from Table 9 (see Appendix). For each product, the table summarizes the results for the unique cracks and keygens, whereas the first 5 products are applications and the latter ones are games. Our downloads

Product	real C/K	wrong C/K	file inf.	sys. inf.	VT avg.	Anubis avg.	NW traf.	bare MW.	work.	work. clean
Photoshop	66.67	60.00	66.67	13.33	62.06	11.43	20.00	33.33	13.33	0.00
Nero 10	78.95	15.79	68.42	52.63	56.41	32.78	42.11	15.79	63.16	15.79
Office 2010	64.29	42.86	42.86	35.71	47.32	43.33	35.71	35.71	21.43	14.29
Norton 360	43.75	25.00	31.25	12.50	39.67	31.33	50.00	43.75	12.50	0.00
Winrar	68.18	18.18	22.73	4.55	32.87	17.27	18.18	13.64	50.00	31.82
Brink	69.23	30.77	7.69	7.69	6.64	15.71	7.69	0.00	38.46	30.77
Crysis 2	88.24	35.29	23.53	11.76	29.43	1.25	5.88	11.76	29.41	29.41
Fable 3	66.67	0.00	55.56	66.67	50.03	33.33	55.56	22.22	11.11	0.00
Portal 2	42.86	0.00	21.43	35.71	13.20	10.00	28.57	21.43	42.86	28.57
The Sims 3	88.89	27.78	44.44	11.11	36.91	10.00	11.11	5.56	38.89	27.78
Total App.	65.12	30.23	45.35	23.26	47.66	27.23	32.56	26.74	34.88	13.95
Total Game	73.24	21.13	29.58	22.54	27.24	14.06	18.31	11.27	33.80	25.35
Total	68.79	26.11	38.22	22.93	37.45	20.65	26.11	19.75	34.39	19.11

Table 5: Manual Analysis Results in Percent of Samples

indicate that even though they were all labeled as crack or keygen for a specific product, the binaries included did not necessarily have this functionality. Other binaries were in fact cracks or keygens, but for different versions or even different products. Some of the binaries even consisted of the bare malware itself. Table 5 reflects these findings in the *real C/K*, *wrong C/K* and *bare MW.* columns. We divided our extensive AV scanning results into two categories: *file infections* and *system infections*. A *file infections* means that the AV engine reported, that a file is infected with malware. However, this does not necessarily imply that executing a purportedly infected file leads to malicious activities. On the other hand, a *system infection* means that during execution of the program, malicious activity could be observed and at least one other file on the system was infected with malware. In our test environment, successful malware infection was usually accompanied with infection of existing or creation of new files and adding custom startup entries in the Windows registry. These infections and modifications could then be found through the full system scan that we performed at the end of each test run (see Section 4.3.1 for details). Moreover, we analyzed all manual samples with the Virustotal and Anubis environments as well. The columns *VT avg.* and *Anubis avg.* show the average results based on VT and Anubis scores. The setup is described in more detail in Section 5.1.1. The column *NW traf.* reflects all samples causing network traffic. The number of working cracks and keygens is visible in column *work.*, whereas samples with clean AV results and no malicious activity are represented by *work. clean*. To exemplary clarify the columns of Table 5, we consider the Photoshop application. From all unique binaries that we downloaded, 66.67% were indeed a real crack or keygen (*real C/K*), but out of these, 60.00% were either not for Photoshop or for a different Photoshop version (*wrong C/K*). Scanning the binaries inside the virtual environment with AVG free [30] showed that 66.67% are infected with malware (*file inf.*), but only 13.33% managed to actually infect the system (*sys. inf.*). The column *bare MW.* shows that out of all Photoshop samples, 33.33% were no crack or keygen at all and just consisted of the bare malware itself. Note that the percentages of *real C/K* and *bare MW.* do not necessarily need to add up. For instance, if an alleged crack turns out to be a harmless malware free decompression tool, it does not show up in the *real C/K* or *wrong C/K* columns as it is no crack or keygen and it also doesn't show up in the *bare MW.* column. Scanning the samples with VirusTotal showed that 62.06% of the virus scanners reported an infection (*VT avg.*), while the average Anubis score (i.e. the maliciousness of the binaries) was 11.43% (*Anubis avg.*). From all Photoshop cracks and keygens, 20.00% established one or more network connections to the Internet (*NW traf.*). The column *work.* indicates that 13.33% of the cracking tools successfully unlocked Photoshop, but 0.0% of the tools worked and were also free from malware (*work. clean*).

Our results indicate that cracks and keygens are an effective distribution channel for malware and, as a result, a valuable source of revenue for underground entrepreneurs like Pay Per Install (PPI) providers or other actors in the software piracy ecosystem like PPD or One-Click hosters, dedicated crack hosters, specialized crack search engines and similar (see Section 3). Both, for applications and games, the chance of manually finding a working crack or keygen is roughly one third (33%). However, for all applications we tested, only 13.95% worked and were free from malware. Similarly, for all games, only 25.35% worked and were free from malware. Even though not each purportedly infected file leads to malicious activities, we could observe that more than 22% of our samples managed to successfully infect the whole system.

5.1.1 Control Analysis

We submitted all 141 manually collected unique samples to the Virustotal and Anubis analysis environments to get both static and dynamic analysis results. For 91.49% of our samples, there were existing Virustotal reports indicating that these cracks and keygens had been scanned by other users before. Virustotal uses a set of AV scanners (Section 4.3.2) and outputs their individual results. The more AV scanners report infections, the higher is the probability that a sample is indeed infected with malware. In Virustotal reports, this ratio is commonly known as *detection ratio*. In contrast, Anubis calculates a *severity score* (Section 4.3.2) that reflects *how malicious* a sample behaved inside the analysis environment. Table 5 shows the average results from these analysis environments on a *per product* basis. A high average percentage indicates that a high number of samples is very likely to be infected with malware (*VT avg.*) or behaves maliciously (*Anubis avg.*). For instance, the cracks and keygens for Photoshop were more likely to contain malware than for Office 2010, but the Office 2010 cracks and keygens behaved much more maliciously on average.

5.1.2 Sample Age

Recent results of Caballero et al. [17] show that malware is frequently repacked to avoid detection by static analysis environments. Consequently, if cracks and keygens are used to spread malware, a malicious uploader would frequently upload existing tools bundled with freshly repacked malware. To determine the approximate age of our samples, we analyzed all existing Virustotal reports for the manually collected cracks and keygens. We distinguished between *all samples* and *infected samples*, whereas a sample is considered to be infected if at least 30% of the Virustotal AV scanners reported malware infections, an admittedly conservative threshold. The result is depicted in the Cumulative Distribution Function (CDF) in Figure 4. We manually collected cracks an keygens from May to September 2011 which can be seen from the graph as well. The graph indicates that a significant amount of our samples was new, suggesting that the samples were either freshly repacked or even 0-day malware.

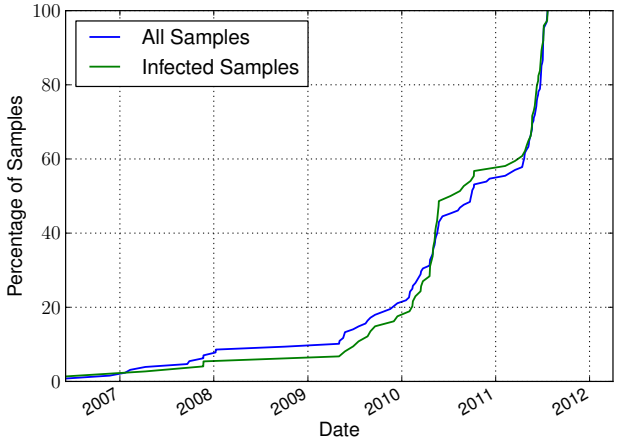

Figure 4: VirusTotal First Seen Dates for All and Infected Manual Downloads (CDF)

5.1.3 Malware Packing

Moreover, we analyzed whether the use of packers could be detected within our samples and which packers were the most frequent. The results are visible in Table 6. The table shows the packers we could identify for infected samples. We consider a sample to be infected if at least 30% of the VirusTotal scanners reported infections.

Packer	# Freq.	% Freq.
PECompact 2.xx	10	37.04
UPX 2.90 [LZMA]	4	14.81
Armadillo v1.71	3	11.11
FSG v2.0	2	7.41
UPX 2.93 [LZMA]	2	7.41
ASPack v2.12	1	3.70
BobSoft Mini Delphi	1	3.70
kkrunchy	1	3.70
PECompact v1.4x	1	3.70
Others	2	7.41

Table 6: Most Frequent Packers in Infected Manual Downloads

5.1.4 Malware Distribution on Download Sources

For each download source (Web, OCH and BitTorrent), we compared the overall number of unique downloads with the number of infected ones (i.e. samples with a Virustotal detection percentage $> 30\%$). Based on these percentages, we calculated the distribution of infected samples on download sources. The result is visible in Figure 5. It can be seen that for our manually collected unique cracks and keygens, we had slightly more infected downloads from web sources than from BitTorrent sources or One Click Hosters (OCHs). Since the goal of malware distributors is to spread the malware as widely as they can, we believe that they focus more on download sources that have many users and are easy to use with minimal effort. Both is more the case for web downloads than for OCH or BitTorrent downloads.

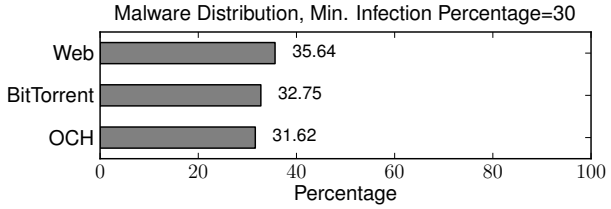

Figure 5: Malware Distribution for Manual Downloads

5.2 Automatically Collected Samples

In addition to the analysis of manually collected cracks and keygens with small sample size, we acquired a high number of samples through custom crawlers (Section 4.2) and analyzed them with automated analysis environments (Section 4.3.2). Our analysis of automatically collected samples is based on a collection of more than $43,600$ download links for Usenet, BitTorrent and OCH downloads leading to $3,491$ unique cracks and keygen executables.

5.2.1 Results

Similar to the analysis of manually collected downloads, we submitted our samples to the Virustotal and Anubis analysis environments to obtain both static and dynamic analysis results. Surprisingly, more than 36.32% ($1,268$ samples) of our cracks and keygens were previously unknown to Virustotal and had no existing report. Due to the large number of games and applications, we represent the results of Virustotal and Anubis scans as Cumulative Distribution Function (CDF). Although we could observe that cracks and keygens for games are slightly less likely to be infected with malware, we decided to combine the results in Figure 6 due to the high similarity of the resulting graphs. The graph shows both the Virustotal existing and fresh scan results for *all* collected samples. For Virustotal results, the X axis represents the average Percentage of AV scanners that reported infections with malware (i.e. the Virustotal detection percentage). Hence, if we consider existing reports and a sample to be malicious if at least 30% of the employed AV scanners reported that the sample is malware, then about 50% of all cracks and keygens we collected would be infected with malware. In contrast, fresh reports are based on more recent AV signatures and, as a result, also include more malware infections. Thus, considering fresh reports with the same

Virustotal detection percentage, about 55% of our samples would include malware.

Moreover, the graph includes the Anubis severity percentage (i.e. the percentage for Anubis severity levels in the range $\{0, .., 10\}$). It provides a good overview of *how malicious* the samples behaved inside the analysis environment, but also allows to classify the samples. As for classification, if we consider all samples with an Anubis severity percentage of at least 30% (i.e. an Anubis severity score of 3) to be malicious, then roughly 35% of the cracks and keygens would be malicious. The comparison between Virustotal and Anubis results can also be seen as sanity check. If the percentage of infected samples rises, it also means that the overall Anubis severity increases due to the more malicious behavior of the samples in comparison to benign ones.

5.2.2 Malware Families

Based on fresh Virustotal reports, we analyzed the most frequent malware families for cracks and keygens. While the most frequent malware `HEUR:Trojan.Win32.Generic` is not a malware itself, but rather a heuristic covering unknown Trojans, our results show that the majority of malware families are indeed Trojans. A closer look at these malware samples reveals that they frequently connect to the Internet, suggesting that they might act as valuable tool for Pay Per Install (PPI) providers. That is, the samples might download additional malware from remote locations or function as remote controlled bots by connecting to command and control (C&C) servers (i.e. `Backdoor.Win32.IRCBot.tyy`). As a result, the most frequent malware families we found can act as valuable source of revenue for underground economy. More information regarding the most frequent malware families we could identify in cracks and keygens can be found in Tables 10 and 11 in the Appendix section.

5.2.3 Sample Age

Based on existing and fresh Virustotal reports, we analyzed the *First Seen Dates* to get an overview of the age of our all samples. The results can be seen in Figure 7, whereat we consider a sample to be infected if the Virustotal detection percentage is 30% or more. We automatically collected cracks and keygens between February and March 2012. The graph reveals a similar picture as for the sample age for manually collected samples. Especially the infected samples are very new, suggesting that the samples might be freshly repacked. It also closely reflects the fact, that more than 36.32% of our samples were new to Virustotal.

5.2.4 Malware Packing

We analyzed Virustotal reports for detected packers that are frequently used in malware. Table 7 gives an overview of the most frequent packers used for infected samples, whereas we consider a sample to be infected if the Virustotal detection percentage is 30% or more. Due to the close similarity, the table reflects the combined results for cracks and keygens for applications and for games. Our results show that *UPX* is the most frequent packer that is used in infected cracks and keygens. However, we also need to note that out of 2,285 possibly infected samples, only 243 (10.63%) used a packer that was detected by Virustotal. We suspect that the number of malicious samples that employs packing mechanisms that are not detected is significantly higher.

Packer	# Freq.	% Freq.
UPX 2.90 [LZMA]	90	37.04
FSG v2.0	32	13.17
PECompact 2.xx	31	12.76
Armadillo v1.71	26	10.70
BobSoft Mini Delphi	19	7.82
UPX 2.93 [LZMA]	15	6.17
ASPack v2.12	8	3.29
FSG v1.33 (Eng)	3	1.23
tElock 0.99 - 1.0 private	2	0.82
MEW 11 SE v1.2	2	0.82
ASProtect v1.23 RC1	2	0.82
themida 1.0.0.5	1	0.41
PECompact v1.56	1	0.41
UPX 0.72	1	0.41
Others	10	4.13

Table 7: Most Frequent Packers in Infected Automated Downloads

5.2.5 Malware Distribution on Download Sources

Our automated crack and keygen acquisition system downloads from three downloads sources: The Usenet, BitTorrent and One-Click Hosting (OCH). Similar to the analysis for manual downloads, we compared the number of all downloads per download source with the number of corresponding infected downloads.

A sample is considered to be infected, if the Virustotal detection percentage was at least 30%. Utilizing this data, we calculated the overall malware distribution on download sources. The result is visible in Figure 8.

Surprisingly, the percentage of infected downloads was significantly higher for Usenet downloads than for Bittorrent or OCH downloads. We believe that this is due to the way the Usenet works. If a sample is uploaded to the Usenet, it can be downloaded as long as the upload is not older than the retention time of the Usenet provider. Hence, if a crack or keygen with freshly packed malware is uploaded, it will be available even years later (considering that most binary Usenet providers have retention times of 1,000 days and more). In contrast, the lifetime of BitTorrent and OCH downloads is more limited. BitTorrent downloads no longer work if there are not enough seeders whereas OCH downloads are frequently removed in case of complaints.

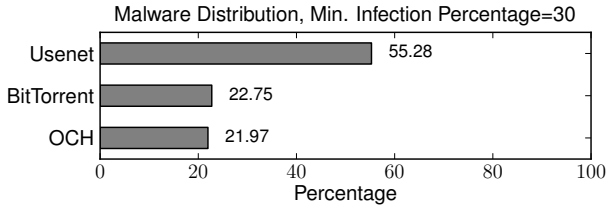

Figure 8: Malware Distribution for Automated Downloads

6. DISCUSSION

The key message from our evaluation and the numeric results is quite simple. Downloading cracks and keygens from any source on the Internet is very dangerous. Even with our conservative form of measurement, which considers only samples where 30% or more of all scanners raised an alert,

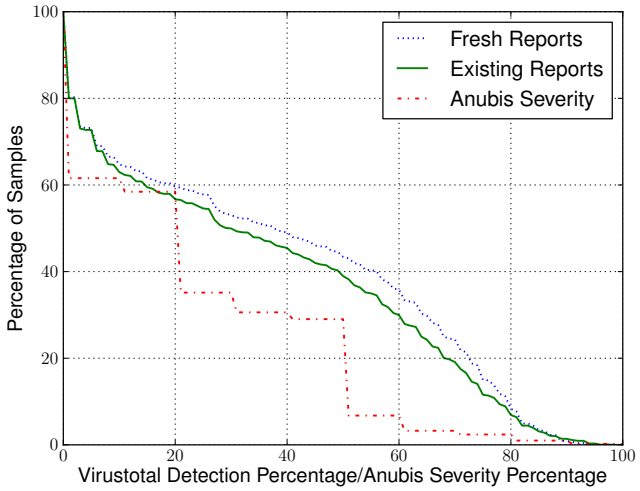

Figure 6: Virustotal and Anubis Scan Results for Automated Downloads (CDF)

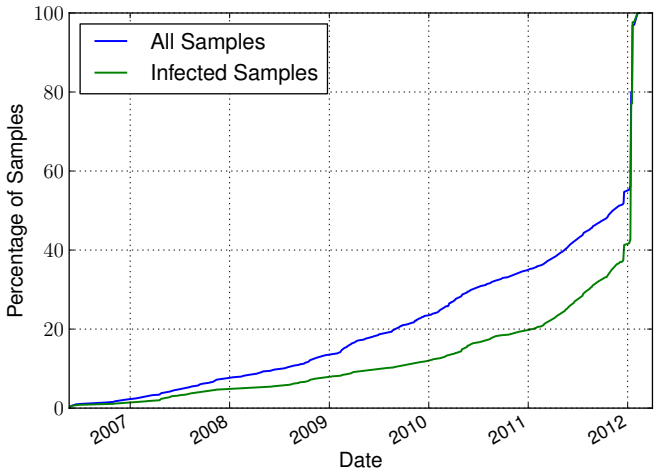

Figure 7: VirusTotal First Seen Dates for Automated Downloads (CDF)

this claim holds true. With more than every second file being infected, the level of malware exposure for users downloading cracks is yet unmatched. Simultaneously, the numbers suggest, that this form of malware distribution is popular with various miscreants like PPI providers or underground entrepreneurs. The reasons for the attractiveness of keygens and cracks are manifold. One one hand, the technical barriers for a miscreant are minimal. The target programs are typically small which saves bandwidth, popular titles can be easily deduced by commercial rankings and the victims already operate on the brink of legality and are therefore more willing to execute shady downloads. On the other hand, the users are no experts and rely on their Anti-Virus software to protect their system. A precaution that often turns out to be futile. Furthermore, most of the investigated samples turned out to be younger than expected.

Every third scrutinized file was not known before, suggesting two important conclusions:

1. Malware samples are constantly repacked or even newly created to fool the victim's AV scanners. Signature-based detection can only protect a system to a certain extend. Detection heuristics, on the other hand, are of limited effectiveness against packers. For this effort to pay off, the infection rate must be correspondingly high.

2. By closely monitoring the crack and keygen distribution channels covered in this paper, Anti Virus Companies could access a rich pool of 0-day malware or repacked viruses. Since our results show that roughly one third of malware is not detected at all, we believe that there is room for improvement for AV companies.

Interestingly, the intentions of the individuals initially producing the crack/keygen are not malicious. Their main motivation is not the money but group reputation and other sociological aspects. As soon as the cracked software is made public though, it is loaded with malware through various procedures. For malware distributors or PPI providers, these cracks are a simple asset that wait to be exploited. An undertaking that has been perfected to an almost exorbitant degree.

7. CONCLUSION

In this paper, we collected more than 43,900 download links and analyzed more than 23,100 (3,551 unique) resulting real-world cracks and serial number generators to get an idea how dangerous it is for an individual to circumvent a program's copy protection. Our results indicate that a majority of these programs aim to infect the target machine with one or more types of malware. Furthermore, a good percentage of the scrutinized samples have never been seen before, suggesting that the scene is heavily used for malware distribution and maintained accordingly. For the end-user it means that the chance to be exposed to malicious code when dealing with cracked applications or games is more than 50 percent. A risk that can only mildly be mitigated by an up-to-date Anti-Virus scanner.

8. DATA-SET

We created a website to make the full data-set available to the research community. The website can be reached at `http://amnesiac.seclab.tuwien.ac.at`.

9. ACKNOWLEDGEMENTS

We would like to thank the JDownloader developers for providing valuable support regarding our specialized download automation requirements. We would also like to thank the anonymous reviewers for their helpful feedback and improvement suggestions. The research leading to these results has received funding from the European Union Seventh Framework Programme under grant agreement n. 257007 (SysSec) and the Austrian Research Promotion Agency (FFG) under grant 820854 (TRUDIE).

10. REFERENCES

[1] Amazon.com: Top 100 software products. *http://www.amazon.com/best-sellers-software/zgbs/software*.

[2] Astalavista.box.sk. *http://astalavista.box.sk*.

[3] Download music, movies, games, software! the pirate bay - the galaxy's most resilient bittorrent size. *http://thepiratebay.org*.

[4] File hosting letitbit.net. *http://letitbit.net*.

[5] filestube - search & download files. *http://www.filestube.com*.

[6] Freee software downloads and software reviews - cnet download.com. *http://download.cnet.com*.

[7] honeyconcent - we are here - honey content sharing for peace & love. *http://honeycontent.com*.

[8] isohunt > the bittorrent & p2p search engine. *http://isohunt.com*.

[9] Jdownloader.org. *http://jdownloader.org/*.

[10] Nzbget. *http://nzbget.sourceforge.net*.

[11] Nzbindex - we index, you search. *http://nzbindex.nl*.

[12] Sharecash.org - make money uploading files! *http://www.sharecash.org*.

[13] transmission - a fast, easy and free bittorren client. *http://www.transmissionbt.com*.

[14] Virustotal - free onlince virus, malware and url scanner. *http://www.virustotal.com*.

[15] B. S. Alliance. 2010 piracy study. 2010.

[16] U. Bayer, C. Kruegel, and E. Kirda. TTAnalyze: A Tool for Analyzing Malware.

[17] 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*, Aug. 2011.

[18] R. Cuevas, M. Kryczka, A. Cuevas, S. Kaune, C. Guerrero, and R. Rejaie. Is content publishing in bittorrent altruistic or profit-driven? In *Proceedings of the 6th International COnference*, Co-NEXT '10, pages 11:1–11:12, New York, NY, USA, 2010. ACM.

[19] N. Doshi, A. Athalye, and E. Chien. Pay-per-install: The new malware distribution network. April 2010.

[20] Engimax. Top pirate reveals warez scene secrets, attracts mpaa lawyer's attention. *http://torrentfreak.com/top-pirate-reveals-warez-scene-secrets-071119 (retrieved 2011-09-12)*.

[21] E. Goldman. Warez trading and criminal copyright infringement. *Journal of the Copyright Society of the U.S.A.*, 51, 2004.

[22] R. D. Gopal and G. L. Sanders. International software piracy: Analysis of key issues and impacts. *Info. Sys. Research*, 9(4):380–397, Apr. 1998.

[23] R. Honick. *Software Piracy Exposed*. Syngress Publishing, 2005.

[24] A. Ikinci, T. Holz, and F. Freiling. Monkey-spider: Detecting malicious websites with low-interaction honeyclients. In *In Proceedings of Sicherheit, Schutz und Zuverlaessigkeit*, 2008.

[25] A. G. John F. Gantz, Christian A. Christiansen. The risks of obtaining and using pirated software. 2006.

[26] M. Limayem, M. Khalifa, and W. Chin. Factors motivating software piracy: a longitudinal study. *Engineering Management, IEEE Transactions on*, 51(4):414 – 425, nov. 2004.

[27] E. Moshchuk, T. Bragin, S. D. Gribble, and H. M. Levy. A crawler-based study of spyware on the web. 2006.

[28] P. C. V. Oorschot. P.c.: Revisiting software protection. In *ISC 2003. LNCS*, pages 1–13. Springer, 2003.

[29] A. Rehn. The politics of contraband: The honor economies of the warez scene. *The Journal of Socio-Economics*, 33(3):359–374, 2004.

[30] A. Technologies. Avg free. *http://free.avg.com*.

APPENDIX

Product	Category	# Web DL	# OCH DL	# BT DL	# Total DL
Adobe Photoshop CS5.1	Application	15	0	4	19
Ahead Nero 10	Application	2	6	16	24
Microsoft Office 2010	Application	12	4	3	19
Norton 360	Application	12	6	1	19
WinRAR 3.93	Application	20	4	6	30
Brink	Game	1	5	8	14
Crysis 2	Game	6	4	15	25
Fable III	Game	0	8	11	19
Portal 2	Game	3	28	15	46
The Sims 3	Game	11	15	3	29
	Total:	82	80	82	244

Table 8: Manual Crack and Key Generator Downloads

Product	Category	# Web	# OCH	# BitTorrent	# Total
Photoshop	App.	13	0	2	15
Nero 10	App.	2	5	12	19
Office 2010	App.	8	3	3	14
Norton 360	App.	10	5	1	16
Winrar	App.	17	3	2	22
Brink	Game	1	4	8	13
Crysis 2	Game	6	3	8	17
Fable 3	Game	0	2	7	9
Portal 2	Game	3	7	4	14
The Sims 3	Game	7	8	3	18
	Total:	67	40	50	157

Table 9: Manual Distinct Downloads per Product

Malware Family	#	%
HEUR:Trojan.Win32.Generic	76	48.10%
Worm.Win32.VBNA.b	6	3.80%
UDS:DangerousObject.Multi.Generic	6	3.80%
Trojan.Win32.Jorik.Skor.pi	4	2.53%
Backdoor.Win32.IRCBot.tyy	4	2.53%
Trojan.Win32.Swisyn.bfdh	3	1.90%
Trojan.Win32.Llac.ajkz	3	1.90%
Trojan-PSW.Win32.Dybalom.dhc	3	1.90%
Trojan-Dropper.MSIL.Agent.vfw	3	1.90%
Backdoor.MSIL.Ubot.b	3	1.90%
others	47	29.75%

Table 10: 10 most frequent Malware Families found in Cracks and Keygens for Applications

Malware Family	#	%
Trojan.Win32.VBKrypt.iwbp	70	22.01%
HEUR:Trojan.Win32.Generic	59	18.55%
Trojan.Win32.Inject.cnht	34	10.69%
Trojan.Win32.Llac.zjo	23	7.23%
not-a-virus:PSWTool.Win32.NetPass.df	19	5.97%
not-a-virus:Downloader.Win32.SwiftCleaner.ay	18	5.66%
Trojan.Win32.Agent.aggx	16	5.03%
Trojan.MSIL.Agent.dnh	13	4.09%
Worm.MSIL.Arcdoor.ae	10	3.14%
Trojan.Win32.Buzus.aick	9	2.83%
others	47	14.78%

Table 11: 10 most frequent Malware Families found in Cracks and Keygens for Games