Dark Matter: Uncovering the DarkComet RAT Ecosystem

Brown Farinholt
University of California, San Diego

Mohammad Rezaeirad
George Mason University

Damon McCoy
New York University

Kirill Levchenko
University of Illinois Urbana-Champaign

ABSTRACT

Remote Access Trojans (RATs) are a persistent class of malware that give an attacker direct, interactive access to a victim's personal computer, allowing the attacker to steal private data, spy on the victim in real-time using the camera and microphone, and verbally harass the victim through the speaker. To date, the users and victims of this pernicious form of malware have been challenging to observe in the wild due to the unobtrusive nature of infections. In this work, we report the results of a longitudinal study of the DarkComet RAT ecosystem. Using a known method for collecting victim log databases from DarkComet controllers, we present novel techniques for tracking RAT controllers across hostname changes and improve on established techniques for filtering spurious victim records caused by scanners and sandboxed malware executions. We downloaded 6,620 DarkComet databases from 1,029 unique controllers spanning over 5 years of operation. Our analysis shows that there have been at least 57,805 victims of DarkComet over this period, with 69 new victims infected every day; many of whose keystrokes have been captured, actions recorded, and webcams monitored during this time. Our methodologies for more precisely identifying campaigns and victims could potentially be useful for improving the efficiency and efficacy of victim cleanup efforts and prioritization of law enforcement investigations.

CCS CONCEPTS

• **Security and privacy → Malware and its mitigation**.

KEYWORDS

security; malware; remote access trojan

ACM Reference Format:
Brown Farinholt, Mohammad Rezaeirad, Damon McCoy, and Kirill Levchenko. 2020. Dark Matter: Uncovering the DarkComet RAT Ecosystem. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3366423.3380277

1 INTRODUCTION

Traditional forms of malware generate revenue for a miscreant through large-scale illicit activity, be it spamming, click fraud, or ransom extortion. The direct victims of such malware experience the infection as a theft of CPU cycles, network bandwidth, or money. While costly in the aggregate, each user's loss is ultimately limited by an attacker's ability to extract value from such victims at scale. Remote Access Trojans (RATs) change this arrangement to one where an attacker interacts with each victim *individually*, scouring through the victim's file system, spying on the victim through the webcam and microphone, or harassing the victim using the computer's speakers and user interface.

In contrast to traditional malware, whose operators have made millions of dollars through illicit activity [41], the financial gains of RAT operators are necessarily limited by the small number of victims they can control. From the victim's point of view, however, a RAT infection may incur not only financial loss but also significant emotional distress due to blackmail and sextortion perpetrated by RAT operators [14, 19]. Thus, the apparent amateur nature of RAT operators and the negligible economic losses they cause belie the greater *individual* harm they beget. Unfortunately, aside from a handful of high-profile cases, little is known about the victims, as published studies of RATs have largely focused on the attackers, their behavior, practices, and business models [25, 42].

As opposed to most studies of malware, a main focus of this paper are the *victims* of RATs. A considerable challenge of studying RAT victims is our limited visibility into this population. Victims of RATs are difficult to identify: computers infected with RATs do not, as a rule, commit click fraud, send spam, participate in DDoS attacks, or otherwise stand out to an external observer. Thus, unlike botnets, even measuring the population of such victims poses a special challenge.

In this paper, we have created a framework for analyzing data collected from RAT operators that enables us to study the harms victims of RAT malware experience. It is commonplace for RAT controller software to maintain a database of each victim infected, along with data pertaining to that victim (e.g., logs of captured keystrokes). By treating the victim entries in these databases as a form of ancestry, we have developed techniques for tracking RAT controllers across hostname changes and for understanding their phylogeny with regards to the origin of their controller software. Further, we propose improvements to existing spurious victim records removal techniques [58] which are able to remove an additional 40% of the likely spurious victims using anonymized victim metadata from these databases. This allows us to determine, with high confidence, which records correspond to real victims.

A unique feature of one popular RAT called DarkComet provided us with an opportunity to collect these victim databases at scale. Whether added intentionally or by mistake, DarkComet makes it possible to download its victim database by issuing a specific command to the controller software over the command-and-control channel. We used this mechanism to download 6,620

WWW '20, April 20–24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380277

victim databases from 1,029 distinct controllers, which we discovered while monitoring a set of 69,227 domains from samples from MalwareConfig, Shodan, VirusTotal, and ReversingLabs.

Using the techniques we developed to track controllers and filter spurious victim records, and following a strictly-controlled methodology of anonymizing private data about victims, we report on a population of 57,805 victims infected by DarkComet controllers over a span of five years. While this study is not comprehensive due to the limitations of our data collection techniques, the sample set we observed allows us to understand the victims of the DarkComet ecosystem and the harms such as webcam and other forms of surveillance that they suffer. Our methodologies for more precisely identifying campaigns and victims could potential be useful for improving the efficiency and efficacy of victim cleanup efforts and prioritization of law enforcement investigations.

In summary, the major contributions of this paper are:

- We describe a methodology for tracking controllers of DarkComet, a popular commodity RAT, across hostname changes based on a phylogenetic analysis of their victims.
- We describe a methodology for identifying real DarkComet victims in the presence of honeypots, scanners, and VM execution of malware by researchers.
- We detail the process by which we collect information about victims of DarkComet at scale and present the results of our analysis of the victims in the studied ecosystem, the harms they incur, and their relationship with their attackers.

The rest of this paper is organized as follows. Section 2 provides the necessary background for the paper. Section 3 describes our data collection methodology; importantly, Section 3.4 discusses our **ethical and legal considerations**. Section 4 describes how we processed the collected data. Section 5 presents our results. Section 6 discusses our findings. Section 7 concludes the paper.

2 BACKGROUND

This work aims to report on the victims of DarkComet, a well-known RAT. In this section, we provide the necessary background on DarkComet for the rest of our study.

2.1 DarkComet RAT

DarkComet is the quintessential RAT, popular for its functionality, freely available for download online, and supported by hacking forum communities and a plethora of tutorial videos on YouTube [19]. It has been used broadly since 2011 by cybercriminals for sextortion [2], voyeurism [19], and, in rare cases, attacks by state actors [24, 42] and trade secret theft [38, 39, 64]. Illustrative of its diverse usage, DarkComet is most well-known for its uses by a sextortionist against Miss Teen USA [2, 3, 6, 18] and by the Syrian government against political dissidents in the Syrian Civil War [27, 42, 46, 52, 59, 61]. Marczak *et al.* [42] provided a particularly detailed examination of DarkComet's usage in the latter campaign.

Such high profile usage has naturally made DarkComet the focus of analyses by industry and academia alike. Malware researchers have studied individual DarkComet campaigns in depth [5, 8, 15, 37, 62], and have thoroughly analyzed its network protocol and behavior [9, 10, 17]. Denbow and Hertz [17] and Breen [7–10] performed the seminal reverse engineering of DarkComet's network protocol handshake and executable configuration, respectively. Most recently, Farinholt *et al.* [25] studied the behavior of DarkComet *operators* themselves in the wild, while Rezaeirad *et al.* [58] investigated the DarkComet ecosystem by sinkholing thousands of RAT-related domains.

We use the following terminology throughout this paper:

- **Operator:** Miscreant interactively controlling a victim's computer using a RAT.
- **Victim:** User whose computer is infected with a RAT stub, who may be a target of a controller's extortion attempts.
- **Controller:** Software used by an operator to configure and build a stub, and to control a victim's computer. Also, the host on which it is running.
- **Stub:** Malware on a victim's computer that communicates with a controller, giving an operator control of the computer.

2.1.1 *Downloading Victim Databases.*

DarkComet allows an operator to configure a stub to automatically download a file from the controller, for example, to download updates or secondary payloads from the controller. Denbow and Hertz [17] reverse-engineered the DarkComet network protocol and discovered that DarkComet allowed a stub connected to a controller to request and download *any* file from the controller, without operator notification. Further described in Section 3.2, we use this feature to glean information about the victims of DarkComet operations.

DarkComet stores information about every victim ever infected in an SQLite database file. Researchers have previously investigated the possibility of obtaining this database from controllers; in particular, Breen [8–10] proposed using DarkComet's arbitrary file download functionality to collect DarkComet controller databases for research purposes. Breen's `dc-toolkit` [7] provides a set of working Python scripts for downloading DarkComet databases, which was later incorporated into Metasploit as a module [11, 31]. Breen [8] also examines the contents of a sample database he downloaded with the `dc-toolkit`, highlighting some of its sensitive contents, such as the keylog table.

2.1.2 *Hack Pack Sharing.*

DarkComet was initially offered freely for download by its author, `DarkCoderSc`, from an official site [40]. However, its authors removed DarkComet from the official site following its widely publicized use by the Syrian government in a cyber-espionage campaign against dissidents at the onset of the Syrian Civil War [27, 46, 59]. The official site reads now states, "DarkComet-RAT development ceased indefinitely in July 2012. Since the [sic], we do not offer downloads, copies or support."

Despite this, DarkComet is available for download, packaged as what is known as *hacking packs* or *hack packs*, collections of RATs and other malware that are sold or freely distributed in hacking forums online. Many RAT hack packs are bundled and distributed by RAT operators hoping to improve their reputation in a hacking forum. These operators package the very RAT software they use personally for distribution. RAT controller executables, including `DarkComet.exe`, run from a directory that contains its supporting DLLs (e.g., `SQLite.dll`) that hack pack distributors simply compress and ship this entire directory. The same directory also contains the victim SQLite database, stored in a file called `comet.db`. Most hack packs *also* include this database file, which contains records of

victims infected by the hack pack creator. We exploit this phenomenon in Section 4.1.1 to understand the ancestry of victim databases we obtain from live controllers.

2.2 RAT Controller Discovery

BladeRunner [22] was the first scanning-based system to actively discover RAT controllers by emulating RAT victims. Since then, Shodan partnered with Recorded Future [30] to add active probing and banner identification for numerous RAT families, including DarkComet, to the Shodan Malware Hunter project [43]. Recorded Future [33] recently presented on its use of Shodan Malware Hunter to identify active RAT controllers and corporate infections over four years of operation, including publishing the IP addresses of detected controllers [32]. To the best of our knowledge, Shodan Malware Hunter represents state of the art RAT controller detection in industry. Marczak *et al.* [42] created a scanner that was able to detect stealthy APT controllers by triggering error conditions. Most recently, Farinholt *et al.* [25] presented a scanner that used ZMap [21], Shodan, and a custom port scanner to detect DarkComet controllers based on their initial handshake challenges. Our scanner is based on these systems.

2.3 Estimating Infected Population

The accuracy of malware infection size measurements and estimation has long been an issue broached by botnet-focused measurement studies. Ramachandran *et al.* [54] proposed a method for estimating botnet infection size based on the frequency of DNS lookups to C&C domains. A subsequent pair of botnet studies used DNS lookups [16] and IRC channel monitoring [1] as measurement vectors but arrived at different estimates due to churn [53]. Recently, Antonakakis *et al.* [4] used a variety of techniques to gauge the size and scope of the Mirai botnet, including active scanning and running a so-called *milker* to obtain attack commands.

2.4 Understanding Data Set Pollution

Part of our data processing methodology entails pre-processing our data to remove records introduced by interfering measurement and counter-offensive operations. In malware infection size estimation, this is a particularly significant obstacle due to the prevalence of such operations by security researchers and anti-malware vendors alike. A number of botnet measurement studies have expounded on the issue of data set pollution [23, 29, 47, 51, 60]. Particularly, Kanich *et al.* [34] showed that data set pollution caused by interfering measurement operations and other active participants in the network could magnify the measured size of the Storm botnet by 10 to 20 times when using a naive estimation approach. Another common source of measurement pollution is the sandboxed execution of malware. We attempt to counter this by submitting our own malware samples to Internet-connected sandbox services to obtain measurement artifacts about said sandboxes, mimicking part of the methodology showcased by Yokoyama *et al.* [63]. Specifically related to RAT malware, Rezaeirad *et al.* [58] recently investigated and enumerated the "stakeholders" in the RAT ecosystem via active scanning and sinkholing, tailoring their methodology to identify active participants polluting the RAT ecosystem. We adopt their techniques to identify and exclude such pollution in our data, as described in Section 4.2.

3 DATA COLLECTION

Figure 1 illustrates our methodology for collecting DarkComet victim databases, which we describe in this section.

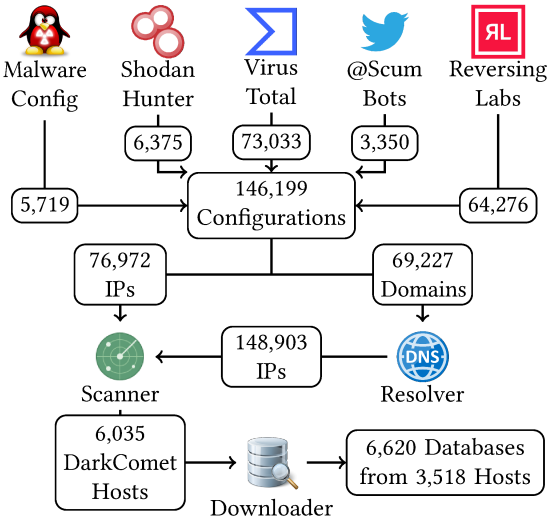

Figure 1: Illustration of our data collection methodology, from sourcing DarkComet configurations from threat feeds to downloading databases from detected hosts.

3.1 Controller Discovery

The DarkComet protocol is password-protected, so we cannot acquire targets from which to download victim databases by indiscriminately scanning the Internet for controllers. Instead, from December 1, 2016 to July 6, 2019, we collected 146,199 unique DarkComet controller configurations from the malware feeds in Figure 1. Many configurations use domain names to address their DarkComet controllers, so we resolve each suspected DarkComet domain name continuously to augment our list of suspected DarkComet host IP addresses. Of the sources in Figure 1, only Shodan does *not* provide domain names; however, all IP addresses provided by Shodan's feed were also either present in configurations from the other sources or discovered during domain resolution.

We implemented a custom Internet scanner to detect RAT controllers. Our scanner continuously probed this final list of 224,172 target hosts for DarkComet network signatures. From December 5, 2018 to July 6, 2019, we made contact with 6,035 live DarkComet hosts. From 3,518 of these hosts, we successfully downloaded databases, recording both their hostnames and resolved IP addresses. During this 213-day observational period, we monitored roughly 500 active DarkComet controllers every week.

3.1.1 Anonymizing Infrastructure Usage. 16.5% of scanned DarkComet hosts use known VPN or VPS services, mainly IPjetable and Relakks VPN.[1] Such a relatively small population of hosts using anonymizing infrastructure suggests that the DarkComet operators in our data set may lack even basic operational security measures.

[1]We use MaxMind and Recorded Future to compile a list of anonymized IP ranges.

Therefore, Internet-wide scanners like ZMap could potentially locate most of these operators' actual gateways.

3.1.2 Dynamic DNS Usage. As Dynamic DNS (DDNS) is a popular tool among DarkComet operators [58], we compare the domain names found in the RAT configurations in our data set against a list of 1,193 domains belonging to 119 prominent DDNS providers. We find that most of the domains used by DarkComet operators belong to one of two free DDNS providers, No-IP and DuckDNS. DarkComet's controller software explicitly interfaces with No-IP's update client, a likely source of its popularity in particular.

3.2 Victim Database Acquisition

The central focus of this study is the DarkComet database. In this section, we describe how we acquire a data set of DarkComet databases, and what information they contain. In Section 2.1.1, we described Denbow and Hertz's discovery of DarkComet's arbitrary file download functionality [17]. In summary, a network device impersonating a DarkComet stub can request arbitrary files from any controller to which it connects. Following this discovery, Breen released the `dc-toolkit` [7], a Python tool for blind file retrieval from DarkComet controllers. We use a modified version of this tool to collect *victim databases* and *DarkComet configuration files* from DarkComet controllers discovered by our scanner.

3.2.1 DarkComet Victim Database. On execution, the DarkComet controller executable (`DarkComet.exe`) creates or loads a file in its working directory named `comet.db`. This SQLite database manages victim connections, and is described thoroughly in the following Section 3.2.2. We downloaded this file from `DarkComet.exe`'s working directory. From December 5, 2018 to July 6, 2019, we downloaded 6,620 databases from 3,518 unique IP addresses. Each time we download a DarkComet database, we append a unique, tainted victim record (a *taint*) to its `dc_users` table (continue to Section 3.2.2 for more details on this table). This tainting happens automatically, as our downloader registers with the controller as a new victim each time it downloads a database; we simply taint the victim information we transmit such that we can identify our downloader's records uniquely in the `dc_users` table.

3.2.2 Victim Database Schema. DarkComet uses a SQLite database, stored in a file named `comet.db`, to manage victim connections and metadata. Table 1 depicts the schemas of each of its tables of importance, as well as provides examples of each.

dc_users. This table contains a single row for every unique victim that has connected to the controller. In Table 1, we observe the contents of a sample row in the dc_users table. As this table is append-only, the order of its contents indicates the order in which victims first connected; users whose IP addresses or operating systems change maintain their original row. Most items in this row are self-explanatory. `userGroup` references the `groupId` field in `dc_groups`. `UUID` is the victim machine's hardward profile ID, returned by the function `GetCurrentHwProfile`, sometimes appended with a random identifier. Since this table is likely to contain victims' personally identifiable information (PII), we hash the `userIP` and `userName` fields before storing them. Prior to hashing victim IP addresses, we resolve their geolocations against a local MaxMind GeoLite2 City database [45].

dc_keyloggers. This table stores victim keystrokes. Each row contains the keystokes logged from a victim, denoted by a `UUID` that references `dc_users`, on a given day. The `name` field refers to the daily file on the victim machine where keystrokes are logged. DarkComet caches victim keystrokes until connected to a controller, at which point all stored daily logs are uploaded at once. The `contents` field stores all captured victim keystrokes, delimited by the victim's active window as it changes. As this table is likely to contain PII, we only store the number of keystrokes captured, and, as of our methodology update on March 25, 2019, letter distributions and victim active window matches against 141 regular expressions for common applications and websites like the Alexa Top 100.

dc_groups. This table allows for attackers to sort and annotate victims into groups. Each row is an attacker-created group, complete with a title, subtitle, and footer. These groups tend to reveal an attacker's language and motivations.

3.2.3 Databases from Hack Packs. To supplement our data set of downloaded DarkComet databases, we downloaded DarkComet hack packs from a combination of hacking forums and VirusTotal. Recipients of hack packs often upload them to malware scanning sites like VirusTotal, as the software in hack packs is (ironically) frequently infected by the packager of said hack packs. From this source, we collected an additional 29 distinct DarkComet victim databases. We use these databases in Section 4.1.1 to describe the phylogeny of DarkComet controller software.

3.2.4 Database Download Failures. In the first month of operation, our downloader was disabled by a series of denial-of-service attacks. Since then, we have used `SOCKS5` proxying to anonymize our download requests, impacting our ability to successfully download databases consistently. In the course of the experiment, we attempted 8,775 database downloads, but 2,155 downloads failed. Network connectivity problems were the main cause of failure, due to `SOCKS5` proxying during large file downloads. Additionally, DarkComet allows the operator to cancel downloads while they are occurring, displaying a pop-up window during a file transfer offering the operator the ability to abort a file transfer in progress. Operators sometimes used this to prevent us from downloading databases. Overall, we failed to extract a single database from 802 controllers; for another 345 controllers, some downloads succeeded.

Of the 6,035 DarkComet hosts detected by our scanner, we only attempted to download databases from 4,320; the remaining 1,717 were never probed due to two factors. First, per the legal and ethical framework on which we based our data collection methodology (see Section 3.4), we do not attempt downloads from hosts which have active web or email servers running, an aggressive measure to avoid probing unaware, compromised hosts being used as intermediary infrastructure by DarkComet campaigns. Second, our downloader is network-constrained; there are some short-lived DarkComet hosts from which it never has a chance to download a database.

3.3 DarkComet Configuration File

DarkComet also uses an `INI` file named `config.ini` to manage configuration information internally. As such, we updated our methodology on March 25, 2019 to collect this file as well. From

Table	Column	Format	Example
dc_users	UUID	<HWID>-<Unique suffix>	{846ee340-7039-11de-9d20-806e6f6e6963-12345678}
	userIP	<WAN IP> / [<LAN IP>] : <Port>	8.8.8.8 / [10.0.0.5] : 1604
	userName	<Hostname> / <Username>	DESKTOP-432AHT11 / Administrator
	userOS	<OS> [<Build>] <Arch> (<Drive>)	Windows 7 Service Pack 1 [7601] 32 bit (C:\\ )
	userGroup	<dc_groups:groupId>	0
dc_keyloggers	UUID	<dc_users:UUID>	{846ee340-7039-11de-9d20-806e6f6e6963-12345678}
	name	<Date>-<Random integer>.dc	2015-12-10-5.dc
	content	<hexified victim keystrokes>	..
dc_groups	groupId	<Sequential integer>	0
	groupTitle	<Operator-created title>	Webcam

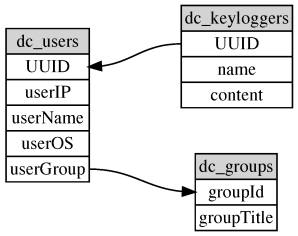

Table 1: The schemas of the tables of importance in the DarkComet database.

March 25, 2019 to July 6, 2019, we downloaded 2,963 configuration files from 2,345 unique IP addresses. This file encodes additional, valuable information about operator interactions with victims; among other things, this file encodes whether an operator has interacted with a specific victim via the inclusion of a configuration section header with the victim's database `UUID`. Further, these victim-specific sections contain keys indicating whether the operator has accessed or recorded the victim's webcam or screen. `config.ini` also contains automation information, listing the tasks the operator has configured stubs to execute upon connection, from controller hostname updates to DDoS targets. It also includes the operator's No-IP login information and DDNS hostname(s), as the controller can automatically issue IP address updates.

3.4 Ethical and Legal Considerations

Part of our data collection methodology emulates DarkComet victims and uses the well-documented DarkComet file download API to retrieve a copy of the victim database from controllers. Before employing this methodology, we consulted with the general legal counsel at our institution, who confirmed that our methodology was legal based partly on the fact that we were using existing functionality that is accessible to any DarkComet stub, and that we were thus not "exceeding authorized access," part of the Computer Fraud and Abuse Act (CFAA) U.S. legal statute. In addition to consulting with our legal general counsel, we also submitted our protocol to our Institutional Review Board (IRB). As part of our protocol, we remove or hash all fields from the database that might contain personally identifiable information (PII), such as the keystroke logs. We also hash other sensitive fields like the victim IP address or user name, which is often the computer owner's name. These redacted copies of the databases were stored on a server with strict access controls and an encrypted file system. Our IRB exempted our study since we neither store nor analyze PII.

Although acceptable from a legal perspective and exempted by our IRB, one might still argue the ethics of our data collection methodology. Here, we present our framework for data collection and analysis within the ethical guidelines described in the Menlo Report [20], which is in turn based on the 1979 Belmont Report [49], and is a cornerstone for computer and information security research. This framework is based on four principles: respect for persons, beneficence, justice, and respect for law and public interest. Our framework addresses each of these principles as follows.

Respect for persons. Since "participation" in this study is not voluntary and cannot be based on informed consent, we take great care not to analyze victim PII, as they form the most vulnerable party involved. We only compile aggregate statistics about victims.

Beneficence. We believe that our analysis does not create further harm. The method we use to collect our data has been well-publicized in prior public reports and talks [7, 9, 10, 17, 28]. We feel the benefits of a better understanding of RAT operators and victims outweigh the potential harms of publishing aggregate statistics.

Justice. The benefits of this work are distributed to the wider public, in terms of helping to reduce crime. The study particularly helps protect persons who are vulnerable to being victimized by RATs. We see no impact to persons from being included in the study.

Respect for law and public interest. We describe the legal framework for data collection and argue that it is in full compliance with U.S. laws. In addition, the researchers that participated in this project have obtained an exemption from their IRB. It is important to note that, while captured information may point to certain illegal conduct, establishing legal proof of criminal conduct is *not* the purpose of this study.

4 DATA ANALYSIS & PROCESSING

We downloaded 6,620 DarkComet databases from 3,518 unique IP addresses from December 5, 2018 to July 6, 2019. Whenever possible, we downloaded the database from a given controller multiple times – no more than once every 24 hours – allowing us to observe the acquisition of new victims over the course of the 213-day measurement period.

However, the raw data we have downloaded is far from ready for analysis. We assert that a single IP address is not synonymous with a single controller; indeed, most controllers use one or more domain names for addressing rather than hard-coded IP addresses. So in Section 4.1, we describe a novel technique for identifying databases from the same controller (potentially downloaded from different IP addresses) based on constructing a family tree of database inheritance. Our technique uncovers *unexpected operator behaviors*, also detailed in Section 4.1.

We further assert that some of the records in a given database may not be real victims. Rezaeirad *et al.* [58] demonstrated that numerous entities are running DarkComet malware samples in sandboxes or operating high-fidelity DarkComet network scanners. These operations pollute the databases we download with fake victim records. To filter this pollution from the data set, we first apply the technique described by Rezaeirad *et al.*, and then improve on their method with novel strategies based on additional metadata included in our data set. We describe this in Section 4.2.

4.1 Database Attribution

When we download a database from a controller, we record the hostname used to contact the controller, which may be either a

domain name or a raw IP address. This hostname is used to uniquely identify a particular controller over the course of the experiment, allowing us to track controllers identified by domain names across multiple IP addresses. Based on our corpus of DarkComet samples, we know that some controllers use more than one hostname. We consider any domain names and IP addresses that appear in the same DarkComet sample to belong to the same controller.

Using this initial technique of hostname-based consolidation, we condense the 3,518 DarkComet IP addresses from which we downloaded databases to **1,162** controllers. 667 of these controllers are identified by domain names, encompassing 86% of the 3,518 IP addresses (3,023) and 72% of the 6,620 downloaded databases (4,750). The remaining 495 IP addresses identify controllers with hard-coded IPs, to which the other 1,870 databases belong. Interestingly, only 15% of our DarkComet samples contain hard-coded IP addresses, compared to the 41% of active controllers identified by them.

Because a controller may produce stubs with multiple, disjoint configurations, there may be more than one controller hostname for each database in our data set. Therefore, 1,162 is an *overestimate* of the number of unique controllers we observed. To identify cases where the same controller's databases were contacted under a different hostname, we use the records in `dc_users` to construct an inheritance tree of DarkComet databases.

4.1.1 DarkComet Database Ancestry. The `dc_users` table in a DarkComet database is append-only, meaning that when a controller infects a new victim, the victim's metadata is appended to the `dc_users` table. Returning users are identified by their `UUIDs`, so duplicate records are never created for the same victim. Thus, the order of the records in `dc_users` describes the order in which the corresponding victims were infected. Each time we download a `dc_users` table from a controller, we expect it to have new victims appended to the end, so that the previously downloaded `dc_users` table is a prefix of the new one.

Furthermore, recall that we add a unique victim record, or *taint*, to the `dc_users` table each time we download it because the process of connecting to the controller generates a victim record. A controller's `dc_users` table should, therefore, not only contain a history of the victims the controller has infected in the order they were added, but also a special victim record corresponding to each time we downloaded the database.

Using the monotonic growth property of the `dc_users` table described above allows us to identify a controller by its database, even if we contact it at a different hostname and IP address. Applying this technique identified 78 controllers using 211 hostnames or hard-coded IP addresses, reducing the number of distinct controllers from 1,162 (identified by hostname only) to **1,029**. Thus, our ancestry-based technique reduced the controller count by over 70% from a naïve (but commonly reported) 3,518 IP addresses.

Having fully consolidated the controllers in our data set, we find that 71% of controllers used just one IP address; the remainder traversed multiple IP addresses during the window of observation. Further, 19% of controllers actively switched domain names during observation. In these cases, our methodology for controller tracking is necessary to accurately report on the observed controllers.

4.1.2 Database Divergence. If two controllers start with the same initial database and then go on to acquire distinct victims, the two databases will share a common prefix of user records from the initial database, followed by distinct sequences of victims acquired by each controller. This is precisely what happens when two or more operators start from a common hack pack (Sections 2.1.2 and 3.2.3): their `dc_users` tables will each contain the set of victims inherited from the hack pack, followed by each operator's own victims. We use the term *divergence* to describe cases where two or more databases have a common non-empty prefix of victim records and different non-empty suffixes of victim records in their `dc_users` tables.

If, for two divergent databases in our corpus, there is no database containing their common prefix, we *infer* such an ancestor database and add it to our data set. The collected and inferred databases can now be arranged into a forest of trees representing database inheritance. The nodes of the inheritance tree represent databases, with an edge from a parent to child if the `dc_users` table of the parent is a prefix of the `dc_users` table of the child, that is if the child is derived from the parent. (Note that there are never points of *convergence* in the DarkComet inheritance tree because there is no mechanism to combine the records from two databases into a new one, so the inheritance tree is indeed a well-formed tree.)

In addition to hack packs, databases may diverge when a controller *reverts* to an earlier version of the database. This happens when an operator runs the RAT controller software in a virtual machine and periodically restores the virtual machine state to an earlier snapshot. Unlike cases of database sharing (e.g., via hack packs), at most one database derived from a common ancestor by reversion will exist at any given point in time, while there may be multiple databases derived from the same hack pack active at a given point in time. In addition, databases related by reversion may be downloaded from a controller identified by the same hostname, while two different databases related by sharing should never appear on the same controller. Only 11% of controllers (116) exhibit this behavior; they reverted their databases 497 times in total during the observation period.

Figure 2 shows a fragment with two inheritance trees from our data set. Open circles represent databases downloaded in the course of the study. Inferred ancestral databases are shown shaded black: black circles denote inferred reversion databases and black squares denote inferred shared databases. Grey squares denote known hack packs (publicly shared databases). In all, the set of inheritance trees consists of 6,620 downloaded databases, 164 inferred ancestral databases related by reversion, 43 inferred shared databases that are not known hack packs and 17 known hack packs.

4.1.3 Hack Pack Prevalence. Of note is that 68% of controllers' databases are derived from an inferred hack pack, while 45% are based on one of the 17 hack packs we possess. This indicates both the prevalence of hack pack sharing in the DarkComet community, as well as the relatively few points of origin for DarkComet controller software downloads. We find that using a hack pack corresponds to both longer operational duration, as well as a higher number of victims; the median hack pack user accumulates 3 times as many victims and operates for 13 times as many days. All outlier attackers in Section 5 acquired DarkComet from hack packs.

4.1.4 Controller Attrition. We managed to download just a single database from about 44% of all controllers. We only downloaded

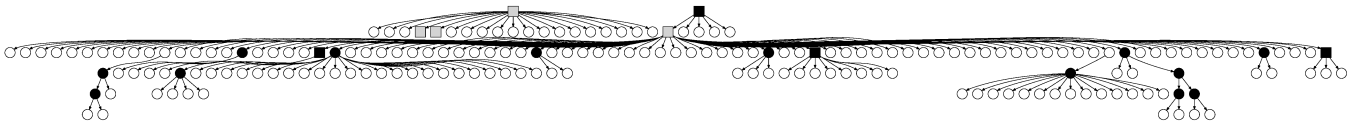

Figure 2: Fragment of the reconstructed DarkComet database inheritance tree. Open circles are sequences of databases downloaded from a single controller; grey rectangles are known hack packs; black rectangles are inferred hack packs not part of our corpus of hack packs; black circles are single-controller reversion points.

10 or more databases from 14% of controllers, and were able to download a database on each day of the measurement period from just 3%. Similarly, 55% of all controllers were observed for fewer than 5 days, calculated as the difference between the first and last download from a given controller, while just 25% were observed 30 or more days. (Recall that 44% of controllers were only seen once.)

Such high attrition led to sparse data collection from a large portion of controllers. We hypothesize that this phenomenon is a result of several factors. First, per Section 3.2.4, our downloader is visible to operators; we suspect that indication of discovery drives operators to abandon either DarkComet or their current infrastructure (e.g., domain names). Second, we believe that many DarkComet operations are inherently short-lived or even experimental, and that they thus expire quickly, regardless of our intervention; we explore this further in Section 5.1.

Description	Records		UUIDs	
Invalid UUID field	6,346,794	82.4%	345,373	72.4%
Matches known scanner (dc_toolkit)	5,267,538	68.4%	276,262	57.9%
Invalid userOS field	1,181,572	15.3%	81,093	17.0%
In hack pack †	453,536	5.9%	15,053	3.2%
Matches suspected scanner	256,821	3.3%	33,080	6.9%
Missing expected keystrokes †	161,572	2.1%	19,199	4.0%
Matches known scanner (ours)	56,612	0.7%	6,482	1.4%
Matches known sandbox	51,392	0.7%	3,755	0.8%
Invalid userIP field	15,729	0.2%	1,308	0.3%
Empty UUID field	1,086	<0.1%	1	<0.1%
Anomalous keystrokes †	745	<0.1%	54	<0.1%
Empty userOS field	178	<0.1%	2	<0.1%
Invalid userName field	92	<0.1%	25	<0.1%
Empty userName field	29	<0.1%	1	<0.1%
Empty userIP field	29	<0.1%	1	<0.1%
Original anomalous victims	6,582,326	85.4%	385,181	80.7%
New anomalous victims †	312,152	4.1%	23,983	5.0%
Total victims in hack packs †	303,701	3.9%	10,323	2.2%
Total unique victims	506,407	6.6%	57,805	12.1%
Total records	7,704,586	100.0%	477,292	100.0%

Table 2: Records and UUIDs filtered by our anomaly detection logic. Many records exhibit more than one anomaly.

4.2 Identifying Victim Pollution

Correctly identifying and enumerating victims is essential to understanding the scope and severity of the DarkComet campaigns under observation. DarkComet assigns each victim a universally unique identifier, or UUID. In Section 5, we will consider this UUID to be equivalent to a victim; however, since imposter victims can fabricate their UUIDs to the controller, we first describe our reduction efforts in terms of *records*, that is, rows in `dc_users` tables. In the 6,620 databases' `dc_users` tables, there are 7,704,586 total records corresponding to 477,292 distinct UUIDs.

4.2.1 Static Anomaly Detection. Rezaeirad *et al.* [58] indicated that there are "active participants" in the DarkComet ecosystem that impersonate victims, including malware sandboxes and network scanners (like ourselves). We applied the techniques they described to the victim records in our DarkComet databases' `dc_users` tables in order to detect and filter these entities. Table 2 lists the anomalies by which we first detect and filter imposter victims, as well as the number of records filtered by each rule. We find that Rezaeirad's rules filter 6,582,326 records corresponding to 385,181 UUIDs. In all, 85.4% of all records and 80.7% of all UUIDs are considered anomalous by these rules. Note that anomalies marked with a † are novel and will be discussed below.

4.2.2 Hack Pack Victim Detection. In Section 4.1.1 we described the process by which we inferred the existence of hack packs in addition to those we possess. We consider victims shared in hack packs separately from victims belonging to individual campaigns. The hack packs we possess contain 373 victim records, 311 of which do not have any anomalies. Our inferred hack packs contain 14,930 victim records, 10,247 of which are not anomalous. As there is some overlap, the total number of UUIDs across all hack packs is 15,053 with 10,323 apparent real victims.

4.2.3 Keylog Validation. Using metadata from the keylog table, we attempt to filter short-lived, sandboxed executions of DarkComet and provide a conservative bound on the victim population. The `dc_keyloggers` table, described in Section 3.2.2, contains a file per victim per day that keystrokes were logged. DarkComet's keylogging functionality cached keystrokes locally on the victim machine and dumps them, a file per day, to the controller's `dc_keyloggers` table when both machines are online simultaneously. Keylogging is enabled on all victims by default, so we expect a real victim to have numerous days of keylogs in the table, while a sandboxed execution might have one or few. However, this is complicated by the fact that the operator can configure keylogs to be sent to an offsite FTP server. However, if configured for offsite FTP, the database will contain a table for storing FTP credentials. While we do not collect this table, we do record its presence in the database schema. Therefore, we bound our victim population based on the following conditions:

(1) If a victim record has anomalous dates in the keylog table (e.g. year 2077), it is excluded (54 records).
(2) If the FTP table exists, the database's victim records are included in analysis (48,356 records).
(3) If the victim record has more than two days of keylogs, it is included in analysis (9,449 records).
(4) Otherwise, the victim record is excluded. (19,199 records).

Applying all validation steps (Sections 4.2.1, 4.2.2, and 4.2.3) leaves a conservative estimate of **57,805 victims**. Had we only

applied the rules described by Rezaeirad *et al.* [58], we would have erroneously considered an additional 34,306 victims to be real victims (10,323 that we determined to be present in hack packs, and 23,983 that we caught using keystroke-based rules). Our new rules reduced pollution by almost 40% in comparison.

5 OBSERVATIONS & APPLICATIONS

Having applied the previously described techniques to our data set, consolidating observed controllers and eliminating spurious victim records, our data set consists of 57,805 victims controlled by 1,029 controllers. With a better understanding of what the ecosystem under observation actually looks like, we now consider potential applications of our techniques and derived data set towards the mitigation of DarkComet and other RAT malware.

5.1 Operator Takedown

Law enforcement agencies worldwide have shown interest in RAT malware takedown efforts (e.g., the 2012 international takedown of Blackshades RAT [26, 55–57]) and in arresting authors and distributors of RAT malware [35, 36, 50]. Just last year, Ukrainian police arrested a DarkComet operator with over 2,000 victims, *an operator we had also been tracking*. In our data set, we find evidence of several operations we consider even more critical than this. For instance, 10 controllers we observed have more than 1,000 victims; 2 have over 9,000. Our techniques for DarkComet controller tracking could help law enforcement agencies prioritize investigations, whether based on controller longevity, infection rate, or magnitude.

Infection Rate. Over our 213-day measurement period, controllers added 14,420 new victims between the first database downloaded from each controller and the last, an aggregate rate of **69 new victims per day**, or a new victim every 20 minutes. Infection rates vary tremendously by controller. The average daily infection rate across all controllers is less than one victim a day. 90% of controllers infected just a single victim over the entire period of observation, perhaps an indication of a targeted attack. However, 3 controllers amassed more than 1,000 victims each during this same period, one of which infected 463 victims in just 23 days at a rate of *20 new victims per day*. Rapidly growing campaigns could be prioritized for law enforcement investigation while targeted attacks could be cataloged and used as a resource when investigating personal abuse like stalking or intimate partner violence cases.

Total Victims. The total number of victims infected per controller over all time, excluding victims from known or inferred hack packs, is also dominated by a handful of outliers. In fact, 2 controllers in our data set have *over 9,000* victims each; the next closest has 3,468. Just 10 controllers have more than 1,000 victims, and only 56 have more than 100. A full 325 controllers have no real victims at all (at least, per our conservative filtering), and the median number of victims per controller is just 2.

Campaign Longevity. We consider the operational age of a controller to be the time between when the first victim keylog was received by a controller and when its most recent database was download by our scanner. We only report on the 49% of controllers (507) that have keylogs, which allows us to effectively decouple our understanding of campaign longevity from our relatively limited window of observation. Figure 4 shows the cumulative distribution of controller operational longevity for controllers whose database was derived from a known hack pack and those whose database was not. The median longevity of controllers derived from hack packs, 262 days, is about twice that of controllers not using a hack pack (116 days). Over 40% of controllers in our data set have been operational for at least a year, and almost 17% have been functional for over 3 years. The longest lived controller in our data set has been infecting new victims and collecting keylogs for well over 5 years. The average operational duration was roughly 484 days, and the median was 228 days.

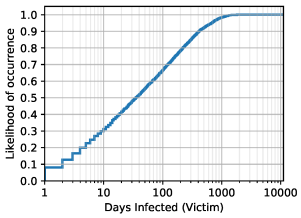

Figure 3: CDF of victim infection duration (n=6,354).

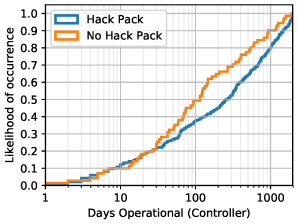

Figure 4: CDF of controller age (n=507).

These results are biased towards controllers that have victims and have thus collected keystrokes (recall that 32% have no victims). In Section 4.1.4, we previously remarked on the high rate of controller attrition, that is, controllers disappearing after a small number of downloads. It may be that our downloader scares off some operators, or that some observed operations are inherently short-lived or experimental, but we cannot determine this with a right- and left-censored window of observation. We nevertheless observed two populations of controllers, transient and long-lived, the latter of which could be investigated by law enforcement and potentially reduce the harm to victims from persistent attackers.

5.2 Infection Duration & Cleanup

We consider the time between a victim's first and last keylogs in a database to be its infection duration, excluding those victims without keylogs. Figure 3 plots the cumulative distribution of victim infection duration, showing the probability of a victim being infected after a given number of days. At the time of our observation, 53% of all victims had been infected for over a month, with a median infection duration of 36 days. However, there is a small population of long-lasting infections, the longest of which is over 5 years old.

Çetin *et al.* [12, 13] have demonstrated the effectiveness of ISP notification of known victims in malware cleanup efforts. Our initial IRB proposal did not include a notification component. This precluded us from considering victim notification as a mitigation to the observed infections. On October 6, 2019, our IRB approved our modification request to allow us to report the 57,805 victims in our data set to their ISPs for notification and cleanup. Our techniques for determining victim realism reduce wasted resource expenditure in this effort, while the data set we collected provides other indicators of victim identity that could assist in notification.

For instance, 126 of the 1,029 controllers' operators annotate their victims using group names, recorded in the `dc_groups` database table. 8,411 victims have one such label. Group labels often reveal operator intent. 77% of labels indicated voyeuristic intentions (e.g. webcam access). 19% suggest targeting specific individuals;

15% the deployment of hacking tools; and 6% credential theft. The voyeuristic motivations and propensity to target individuals (often by name) align with anecdotes of RAT usage for perverse acts such as sextortion [18, 19], but can also help identify specific victims of malware in cleanup efforts.

5.3 Operator and Victim Geography

Country	Controller				Victims	
	Count		Footprint			
Turkey	238	23.1%	7,451	12.9%	7,472	12.9%
Russia	210	20.4%	5,840	10.1%	7,854	13.6%
Ukraine	80	7.8%	362	0.6%	936	1.6%
France	28	2.7%	10,123	17.5%	1,358	2.3%
Brazil	26	2.5%	1,147	2.0%	2,278	3.9%
Italy	19	1.8%	2,678	4.6%	905	1.6%
United States	15	1.5%	256	0.4%	4,045	7.0%
Germany	14	1.4%	38	0.1%	2,936	5.1%
Ivory Coast	4	0.4%	11,306	19.6%	62	0.1%
India	3	0.3%	61	0.1%	1,803	3.1%
Netherlands	3	0.3%	11	0.0%	3,299	5.7%
Philippines	2	0.2%	2	0.0%	3,295	5.7%
Trinidad and Tobago	1	0.1%	3,468	6.0%	67	0.1%
Anonymous VPN	170	16.5%	8,868	15.3%	-	-
Other	216	21.0%	6,194	10.7%	21,495	37.2%
Total	1,029	100.0%	57,805	100.0%	57,805	100.0%

Table 3: Sample of controller locations and victim counts. Controller footprint is the total number of victims controlled by all controllers from a given country.

Our data set uniquely links victims and operators, allowing us to understand the geographic relationship between them.[2] This information reveals an interesting dichotomy in the behavior of DarkComet operators: most operators appear to infect one or a few local victims, while a small number of operators control hundreds to thousands of victims worldwide.

Table 3 shows the number of controllers in each country together with their footprint, which is the number of victims whose controller is in that country. Countries are listed in decreasing order of the number of controllers. About 17% of operators used a known anonymizing service like a VPN or VPS, making true geo-location impossible for these controllers; such controllers are counted separately in the *Anonymous VPN* row. Controllers whose location we could not determine are counted in the *Other* row. Figure 5 shows the number of victims for each combination of controller country and victim country. The vertical axis enumerates controller countries (ordered by the controller footprint), while the horizontal axis enumerates victim countries, in the same ordered as controller countries. The horizontal axis extends to include additional countries in order of decreasing number of victims.

Turkey and Russia appear to be hotbeds of DarkComet activity; however, while these two countries account for 44% of all controllers, their operators control only 23% of all victims. Rather, a handful of controllers in France and Ivory Coast control 37% of all victims. This illustrates an important detail: most controllers have just one or a few victims, while a small population of operators control most of the victims in our data set; indeed, 5 controllers control half of all victims.

Rezaeirad *et al.* [58] evince that attackers and victims in the commodity-grade RAT ecosystem are often co-located, that is, in the same country or region. The propensity for some RAT operators to target people they know likely influences this, as do shared language and culture. As Figure 5 shows, there is also a tendency for controllers to be in the same country as their victims, although the majority of victim hosts (77%) are *not* in the same country as their controller. The converse, however, is not true. Because Figure 5 is weighted by victims (each victim contributes 1 count to the numbers shown), it emphasizes the heavy-hitters with a global victim population. Viewed from the controller side, a full 74% of attackers are in the same country as the *majority* of their victims. Colocation of operator and victim *is* the norm for operators with fewer victims. Thus, we find in our data set evidence of two categories of DarkComet operator; the prolific spreader with many victims across the world, and the typical operator with one or a handful of victims, most of whom are geographically co-located.

We find DarkComet campaigns with over 1,000 victims likely located in jurisdictions where we can reasonably expect law enforcement to investigate them, such as France and Italy. We also find over 2,000 victims in the Netherlands, where their ISPs have previously launched effective notification campaigns [12, 13]. Just 16% of operators use VPNs, with the remaining unprotected operators controlling some 85% of victims. This suggests that law enforcement operations would likely be both successful *and* high-impact. The low rate of operator VPN usage (just over 16%) suggests that such operations would likely be successful.

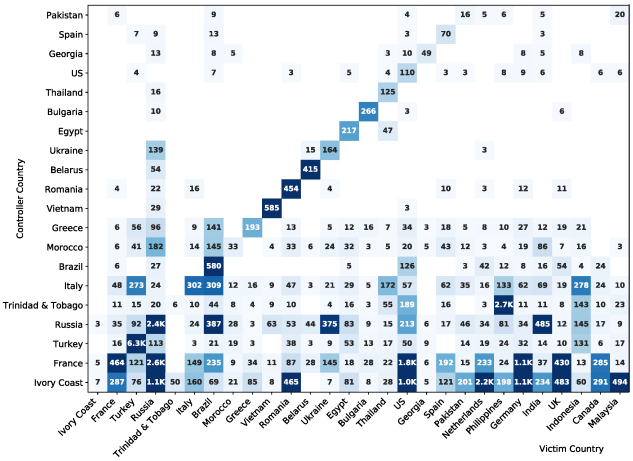

Figure 5: Number of victims for each combination of controller and victim country. Rows denote controller countries and columns victim countries, based on geo-located IP address (excluding VPN providers).

5.4 Observed Harm to Victims

Our efforts to notify victims are motivated by troubling abuses we detected in our data set. We *quantify* the harm incurred by the victims in our data set through several vectors: captured keystrokes, time monitored by attackers, non-consensual webcam accesses, and direct interaction or communication with attackers.

[2] We geo-locate IP addresses using MaxMind's Precision Insights service [44].

By default, DarkComet installs a keylogger on each victim's machine. As some RAT operators seek access to victim accounts and credentials [25], and many are plainly opportunistic [48], victim keylogs can offer insight into the potential harms victims incur from DarkComet campaigns. As described in Section 3.2.2, on March 25, 2019 we began collecting additional metadata from the keylog table; we were able to do so for 2,664 recent victims from 378 controllers. DarkComet demarcates and timestamps victim keystrokes by the victim's active window. For each active window in a victim's keylogs, we record the number of keystrokes collected while the victim interacted with it and the time the victim spent using it. We compare the active window name to a set of application names containing common applications and websites. We do not record the name of the active window, which may contain sensitive information.

The 2,664 victims in this sample set had **210,835,801 keystrokes** captured over 25,315 days, amounting to over **162,098 hours** of keystroke monitoring. On average, DarkComet collected 79,142 keystrokes and recorded 60 hours of activity over 9.6 days from each victim. The active windows from which keystrokes were stolen indicate that the DarkComet campaigns in our data set likely obtained victims' sensitive information like emails, transcripts of private conversations, login credentials, and credit card numbers, putting them at risk of blackmail or financial compromise.

Per Section 3.3, DarkComet's configuration file encodes whether an operator has issued commands to a victim or accessed the victim's webcam. We were able to download the `config.ini` file from 697 of the 1,029 controllers in our data set, encompassing 50,358 total victims. We find that operators accessed the webcams of 13,269 (26%) victims they actively controlled. Though webcam access suggests a voyeuristic motive, we do not know how much time the controller spent accessing the webcam and cannot differentiate between webcam access for machine vetting versus voyeurism. This suggests the potentially life altering personal harms that some RAT victims experience from targeted stalking and harassment.

6 DISCUSSION

Campaign Tracking. Our method of obtaining victim databases from RAT controllers combined with our lineage analysis technique enables us to identify distinct RAT campaigns across any number of IP address and domain name changes, assuming a controller represents a RAT campaign. This allows us to better understand the size and dynamics of RAT campaigns, including 2 campaigns with over 9,000 real victims. This type of information can help security researchers perform attack attribution, or law enforcement prioritize investigations. Given that law enforcement appears interested in combating RAT malware [26, 55–57]), the techniques presented in this paper could be particularly useful in prioritizing and tracking operations. As our techniques are also able to determine DarkComet controller points of origin (i.e. hack packs), they could also help law enforcement target the distributors of RAT malware [35, 36, 50].

Victim Identification. The pollution reduction heuristics we improved upon enable us to reduce our initial set of 477,292 potential victims by around 93%, leaving us with 57,805 likely real victims. Our original IRB protocol did not include plans for victim notification, thus we could not do notifications. However, we modified our IRB protocol to include ISP notification to victims based on our findings of longer lived infections and recent work indicating ISP notifications help speed up desktop victim cleanup [12, 13]. Our modified protocol which includes a plan to share victim's IP addresses with their ISP was approved on Oct 6, 2019. We plan to start notifying ISPs and monitor the efficacy based on continued data collection from DarkComet operators. Our improved victim identification will likely reduce the resources wasted on notifications to fake victims and allow ISPs to devote resources to assisting real victims clean up infections. As our data processing methodology is not dependent on our form of data collection, it could be used in other scenarios. For instance, law enforcement acting on search warrants could use this technique to expedite victim notification. Çetin *et al.* [12, 13] demonstrated the potential for victim notification by ISPs to mitigate malware infections. While our initial IRB proposal did not allow us to store victim IP addresses, we have since modified it so that we can begin engaging with ISPs to notify the victims of DarkComet found in our data set.

Understanding Victim Harm. Understanding the harms incurred by victims of low-volume malware infections is challenging, particularly in comparison to the large-scale malware campaigns waged by spambots and ransomware. Our system's data collection and automated analysis methods allow for quantifying these harms at scale, in terms of disquieting metrics like keystrokes stolen, hours monitored per application, and webcam accesses made.

Study Limitations & Extensibility. Our data collection methodology is currently limited to DarkComet; however, prior work indicates that a number of other RAT families expose the same arbitrary file read functionality [28], suggesting that our data collection methodology scales to other RATs. Further, like DarkComet, most RATs maintain databases of victim metadata. While they may not expose the same download capabilities, they are still often shared in hack packs. If we could scale our collection of hack packs (e.g., through more access to malware upload repositories), this could enable our analysis methodology to expand to additional RAT families. Further, as the data processing techniques we debuted in Section 4 are independent of our data collection technique, they can be applied to data obtained otherwise (e.g., by legal seizure).

7 CONCLUSION

In this work, we presented a broad study on the ecosystem of RAT malware. To carry out the study, we used a feature of the DarkComet RAT controller software that allows anyone to download the database of its victims. Using this capability, we collected 6,620 databases from 1,029 unique controllers. To arrive at our data set for analysis, we developed new methods for tracking controllers and improved existing methods for identifying real victims. Using this data, we presented the results of our analysis of controllers, victims, and the relationship between them. We propose to use our techniques in DarkComet and other RAT cleanup efforts, and are engaging ISPs to notify the 57,805 victims in this data set.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their helpful suggestions and ReversingLabs for their assistance. This work was supported by NSF grants 1237264, 1619620, 1629973, and 1717062, DHS/AFRL award FA8750-18-2-0087, and a gift from Google.

REFERENCES

[1] M. Abu Rajab, J. Zarfoss, F. Monrose, and A. Terzis. A multifaceted approach to understanding the botnet phenomenon. In *ACM Internet Measurement Conference (IMC)*, 2006.

[2] N. Anderson. How an omniscient internet "sextortionist" ruined the lives of teen girls. http://arstechnica.com/tech-policy/2011/09/how-an-omniscient-internet-sextortionist-ruined-lives/, September 2011.

[3] N. Anderson. How the fbi found miss teen usa's webcam spy. http://arstechnica.com/tech-policy/2013/09/miss-teen-usas-webcam-spy-called-himself-cutefuzzypuppy/, September 2013.

[4] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou. Understanding the Mirai Botnet. In *USENIX Security Symposium (USENIX)*, 2017.

[5] L. Aylward. Malware analysis - dark comet rat. http://www.contextis.com/resources/blog/malware-analysis-dark-comet-rat/, November 2011.

[6] BBC. Miss teen usa hacker pleads guilty to 'sextortion' threats. http://www.bbc.com/news/technology-24929916, November 2013.

[7] K. Breen. Dc toolkit. https://github.com/kevthehermit/dc-toolkit.

[8] K. Breen. Look inside a dark comet campaign. Technical report, 2014.

[9] K. Breen. DarkComet – Hacking The Hacker. https://techanarchy.net/2015/11/darkcomet-hacking-the-hacker/, November 2015.

[10] K. Breen. DarkComet: From Defense To Offense - Identify your Attacker. In *Security BSides London*, 2015.

[11] BuddhaLabs. Packetstorm-exploits. https://github.com/BuddhaLabs/PacketStorm-Exploits/blob/master/1606-exploits/darkcomet-download.rb.txt, 2016.

[12] O. Çetin, C. Gañán, L. Altena, T. Kasama, D. Inoue, K. Tamiya, Y. Tie, K. Yoshioka, and M. van Eeten. Cleaning up the internet of evil things: Real-world evidence on isp and consumer efforts to remove mirai.

[13] O. Cetin, C. Ganán, L. Altena, S. Tajalizadehkhoob, and M. van Eeten. Let me out! evaluating the effectiveness of quarantining compromised users in walled gardens. In *Fourteenth Symposium on Usable Privacy and Security (SOUPS 2018)*, pages 251–263, 2018.

[14] R. Chatterjee, P. Doerfler, H. Orgad, S. Havron, J. Palmer, D. Freed, K. Levy, N. Dell, D. McCoy, and T. Ristenpart. The spyware used in intimate partner violence. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 441–458. IEEE, 2018.

[15] F. Cybersecurity. Looking at the sky for a darkcomet. https://www.fidelissecurity.com/sites/default/files/FTA_1018_looking_at_the_sky_for_a_dark_comet.pdf, August 2015.

[16] D. Dagon, C. Zou, and W. Lee. Modeling Botnet Propagation Using Time Zones. In *Networked and Distributed System Security Symposium (NDSS)*, 2006.

[17] S. Denbow and J. Hertz. Pest Control: Taming the Rats. Technical report, 2012.

[18] Department of Justice, U.S. Attorney's Office, Central District of California. Temecula student sentenced to federal prison in 'sextortion' case. https://www.justice.gov/usao-cdca/pr/temecula-student-sentenced-federal-prison-sextortion-case, March 2014.

[19] Digital Citizens Alliance. Selling "Slaving" - Outing the principal enablers that profit from pushing malware and put your privacy at risk, 2015. https://media.gractions.com/314A5A5A9ABBBBC5E3BD824CF47C46EF4B9D3A76/07027202-8151-4903-9c40-b6a8503743aa.pdf.

[20] D. Dittrich and E. Kenneally. The menlo report: Ethical principles guiding information and communication technology research, 2012.

[21] Z. Durumeric, E. Wustrow, and J. A. Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *USENIX Security Symposium (USENIX)*, 2013.

[22] M. Eisenbarth and J. Jones. BladeRunner: Adventures in Tracking Botnets. In *Botnet Fighting Conference (Botconf)*, 2013.

[23] B. Enright, G. M. Voelker, S. Savage, C. Kanich, and K. Levchenko. Storm: When researchers collide. *USENIX ;login:*, 2008.

[24] R. Falcone and S. Conant. Projectm: Link found between pakistani actor and operation transparent tribe, 2016.

[25] B. Farinholt, M. Rezaeirad, P. Pearce, H. Dharmdasani, H. Yin, S. Le Blond, D. McCoy, and K. Levchenko. To catch a ratter: Monitoring the behavior of amateur darkcomet rat operators in the wild. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2017. https://www.ieee-security.org/TC/SP2017/papers/380.pdf.

[26] FBI. International blackshades malware takedown: Coordinated law enforcement actions announced. *FBI News*, 2014. https://www.fbi.gov/news/stories/international-blackshades-malware-takedown-1.

[27] E. Galperin and M. Marquis-Boiremay. Fake skype encryption tool targeted at syrian activists promises security, delivers spyware. *Electronic Frontier Foundation*, 2012.

[28] W. Grange. Digital vengeance: Exploiting the most notorious c&c toolkits. In *Black Hat USA*, 2017.

[29] J. B. Grizzard, V. Sharma, C. Nunnery, B. B. Kang, and D. Dagon. Peer-to-peer botnets: Overview and case study. In *USENIX Workshop on Hot Topics in Understanding Botnets (HotBots)*, 2007.

[30] L. Gundert. Proactive threat identification neutralizes remote access trojan efficacy. Technical report, Recorded Future, 2015.

[31] J. Hertz, S. Denbow, and J. Wetzels. Darkcomet server 3.2 remote file download. Technical report, 2016.

[32] Insikt Group. Talking to RATs - RAT IPs Dec2 Jan8 only. https://github.com/Insikt-Group/Research/blob/master/Talking%20to%20RATs/RAT_IPs_Dec2_Jan8_only, 2019.

[33] Insikt Group. Talking to rats: Assessing corporate risk by analyzing remote access trojan infections. Technical report, Recorded Future, 2019.

[34] C. Kanich, K. Levchenko, B. Enright, G. M. Voelker, and S. Savage. The heisenbot uncertainty problem: Challenges in separating bots from chaff. In *USENIX Conference on Large-scale Exploits and Emergent Threats (LEET)*, 2008.

[35] B. Krebs. 'luminositylink rat' author pleads guilty. *Krebs on Security*, 2018. https://krebsonsecurity.com/2018/07/luminositylink-rat-author-pleads-guilty/.

[36] B. Krebs. Canadian police raid 'orcus rat' author. *Krebs on Security*, 2019. https://krebsonsecurity.com/2019/04/canadian-police-raid-orcus-rat-author/.

[37] A. Kujawa. You dirty rat! part 1 – darkcomet. https://blog.malwarebytes.org/threat-analysis/2012/06/you-dirty-rat-part-1-darkcomet/, June 2012.

[38] S. Le Blond, C. Gilbert, U. Upadhyay, M. G. Rodriguez, and D. Choffness. A broad view of the ecosystem of socially engineered exploit documents. In *Network and Distributed System Security Symposium (NDSS)*, 2017.

[39] S. Le Blond, A. Uritesc, C. Gilbert, Z. L. Chua, P. Saxena, and E. Kirda. A look at targeted attacks through the lense of an ngo. In *USENIX Security Symposium (USENIX)*, 2014. https://seclab.ccs.neu.edu/static/publications/sec2014ngo.pdf.

[40] J.-P. Lesueur. Darkcomet remote administration tool. http://darkcomet-rat.com/.

[41] S. Malby, R. Mace, A. Holterhof, C. Brown, S. Kascherus, and E. Ignatuschtschenko. United Nations Office on Drugs and Crime: Comprehensive study on cybercrime. https://www.unodc.org/documents/commissions/CCPCJ/CCPCJ_Sessions/CCPCJ_22/_E-CN15-2013-CRP05/Comprehensive_study_on_cybercrime.pdf, 2013.

[42] W. R. Marczak, J. Scott-Railton, M. Marquis-Boire, and V. Paxson. When governments hack opponents: A look at actors and technology. In *USENIX Security Symposium (USENIX)*, 2014. https://www.usenix.org/system/files/conference/usenixsecurity14/sec14-paper-marczak.pdf.

[43] J. Matherly. Shodan - Malware Hunter. https://malware-hunter.shodan.io/.

[44] MaxMind. Geoip2 precision insights service. https://www.maxmind.com/en/geoip2-precision-insights, 2012.

[45] MaxMind. Geolite2 downloadable databases. https://dev.maxmind.com/geoip/geoip2/geolite2/, 2012.

[46] R. McMillan. How the boy next door accidentally built a syrian spy tool. *Wired*, 2012. https://www.wired.com/2012/07/dark-comet-syrian-spy-tool/.

[47] Y. Nadji, M. Antonakakis, R. Perdisci, D. Dagon, and W. Lee. Beheading hydras: performing effective botnet takedowns. In *ACM Conference on Computer and Communications Security (CCS)*, 2013.

[48] National Cyber Crime Unit / Prevent Team. Pathways into cyber crime. http://www.nationalcrimeagency.gov.uk/publications/791-pathways-into-cyber-crime/, 2017.

[49] Office of the Secretary, Department of Health, Education, and Welfare. The belmont report: Ethical principles and guidelines for the protection of human subjects of research, 1979.

[50] S. Øyvann. Rat trap: Norway police nab five in remote-access trojan europol swoop. Technical report, ZDNet, 2018.

[51] P. Porras, H. Saidi, and V. Yegneswaran. A multi-perspective analysis of the storm (peacomm) worm. Technical report, Computer Science Laboratory, SRI International, 2007.

[52] Quequero. Darkcomet analysis – understanding the trojan used in syrian uprising. http://resources.infosecinstitute.com/darkcomet-analysis-syria/, March 2012.

[53] M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis. My Botnet is Bigger Than Yours (Maybe, Better Than Yours): Why Size Estimates Remain Challenging. In *USENIX Workshop on Hot Topics in Understanding Botnets (HotBots)*, 2007.

[54] A. Ramachandran, N. Feamster, and D. Dagon. Revealing Botnet Membership Using DNSBL Counter-intelligence. In *Steps to Reducing Unwanted Traffic on the Internet - Volume 2*, 2006.

[55] S. S. Response. W32.shadesrat (blackshades) author arrested? Technical report, Symantec, 2012.

[56] S. S. Response. Blackshades rat usage on the rise despite author's alleged arrest. Technical report, Symantec, 2013.

[57] S. S. Response. Blackshades - coordinated takedown leads to multiple arrests. Technical report, Symantec, 2014.

[58] M. Rezaeirad, B. Farinholt, H. Dharmdasani, P. Pearce, K. Levchenko, and D. McCoy. Schrödinger's RAT: Profiling the stakeholders in the remote access trojan ecosystem. In *USENIX Security Symposium (USENIX)*. USENIX Association, 2018. https://www.usenix.org/system/files/conference/usenixsecurity18/sec18-rezaeirad.pdf.

[59] K. Stevens and N. Villeneuve. Darkcomet surfaced in the targeted attacks in syrian conflict. *Trend Micro: TrendLabs Security Intelligence Blog*, 2012. https://blog.trendmicro.com/trendlabs-security-intelligence/darkcomet-surfaced-in-the-targeted-attacks-in-syrian-conflict/.

[60] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna. Your Botnet is My Botnet: Analysis of a Botnet Takeover. In *ACM Conference on Computer and Communications Security (CCS)*, 2009.

[61] N. Villeneuve and M. Scott. Crimeware or apt malware: Fifty shades of grey. https://www.fireeye.com/blog/threat-research/2014/04/crimeware-or-apt-malwares-fifty-shades-of-grey.html, 2014.

[62] C. Wilson. Exterminating the rat part i: Dissecting dark comet campaigns. https://www.arbornetworks.com/blog/asert/exterminating-the-rat-part-i-dissecting-dark-comet-campaigns/, July 2012.

[63] A. Yokoyama, K. Ishii, R. Tanabe, Y. Papa, K. Yoshioka, T. Matsumoto, T. Kasama, D. Inoue, M. Brengel, M. Backes, and C. Rossow. SandPrint: Fingerprinting Malware Sandboxes to Provide Intelligence for Sandbox Evasion. In *Research in Attacks, Intrusions, and Defenses*, 2016.

[64] A. Zaharia. Security alert: Infamous darkcomet rat used in spear phishing campaigns. Technical report, Heimdal Security, 2015.