To Catch a Ratter: Monitoring the Behavior of Amateur DarkComet RAT Operators in the Wild

Brown Farinholt[†], Mohammad Rezaeirad[‡], Paul Pearce[§], Hitesh Dharmdasani[¶], Haikuo Yin[†]
Stevens Le Blond[‖], Damon McCoy[††], Kirill Levchenko[†]

[†]University of California, San Diego [‡]George Mason University [§]University of California, Berkeley
[¶]Informant Networks [‖]EPFL and MPI-SWS [††]New York University

***Abstract*—Remote Access Trojans (RATs) give remote attackers interactive control over a compromised machine. Unlike large-scale malware such as botnets, a RAT is controlled individually by a human operator interacting with the compromised machine remotely. The versatility of RATs makes them attractive to actors of all levels of sophistication: they've been used for espionage, information theft, voyeurism and extortion. Despite their increasing use, there are still major gaps in our understanding of RATs and their operators, including motives, intentions, procedures, and weak points where defenses might be most effective.**

In this work we study the use of DarkComet, a popular commercial RAT. We collected 19,109 samples of DarkComet malware found in the wild, and in the course of two, several-week-long experiments, ran as many samples as possible in our honeypot environment. By monitoring a sample's behavior in our system, we are able to reconstruct the sequence of operator actions, giving us a unique view into operator behavior. We report on the results of 2,747 interactive sessions captured in the course of the experiment. During these sessions operators frequently attempted to interact with victims via remote desktop, to capture video, audio, and keystrokes, and to exfiltrate files and credentials. To our knowledge, we are the first large-scale systematic study of RAT use.

I. Introduction

Recent events indicate that malware usage has started to shift from large-scale threats like botnets to lower-volume threats designed to spy on specific users or systems (e.g. [10], [20], [35], [51]). In a botnet, each machine is an indistinguishable bundle of resources. Though imposing in size, the use of botnets was banal—spamming, click fraud, and the like. Low-volume threats, on the other hand, aim to extract something of greater value from each infection. In this regime, the preferred tool for exploiting individual infections is a RAT (Remote Access Trojan or Remote Administration Tool), malware that gives a *human user* interactive remote access to an infected machine.

Because of their great flexibility, RATs have been used by a broad range of actors. For example, intelligence agencies and governments use RATs to spy on dissidents, journalists, and other governments [10], [35], [51]; while voyeurs use these tools to spy on victims, collecting pictures stored on the computer, capturing live webcam images, and recording audio [17], [20]. The latter's intentions range from pure voyeurism to extortion and blackmail, with victims from celebrities like Miss Teen USA [20] to countless unnamed users worldwide.

The subject of this work is the behavior of amateur RAT operators in newly infected machines. Though use of RATs is well documented, knowledge of *how* they are used by these actors is limited. Indeed, studying RATs presents its unique challenges. The first of these is procuring fresh malware samples that are still in operation. In practice, low-volume malware typically does not have the same broad distribution as, say, a botnet executable. An attacker often infects the intended victims by sending them an email message with a malicious attachment or luring them to a Web page that exploits a browser vulnerability. To obtain such samples, researchers must obtain them from victims or a vantage point between attackers and victims, and before they cease operating.

We obtain our samples from VirusTotal [24], an online virus scanner. VirusTotal is often used to check a suspicious file or URL received via email, social media, etc., and thus provides a unique vantage point for studying low-volume malware. For example, recent related work leveraged VirusTotal to measure and analyze malicious documents employed in targeted attacks against two ethnic groups and 12 countries spanning three continents [32]. In this work, we show that we can rely on VirusTotal to collect fresh RAT samples and leverage controllers' aliveness to monitor them while they are in operation.

The second challenge in studying RATs is monitoring what the attacker does when connected. Attackers expect a successful infection to give them access to a victim's computer. Preliminary experiments showed that executing a RAT in a typical VM used to study malware may lure the attacker to connect, but will quickly give away the setup when examined more closely. To elicit natural behavior, we disguised our machines as real users' PCs, suitably personalized, though not linked to a real user. Finally, we need to capture the activity of the RAT operators to reliably reconstruct their behaviors.

In this work, we obtained 19,109 samples of DarkComet, a popular RAT used by threat actors of all levels of sophistication. In most cases, our sample was the result of running a dropper executable that may have been dropped itself. Based on the primary infection vectors, at least some of the instances appear to be genuine attempts to infect a victim. We took each sample we obtained, ran it in a Cuckoo sandbox [25], and recorded all commands issued to the RAT by the controller. We then used the data collected to reconstruct the behavior of the operator in our system and carry out our analysis.

In particular, we analyzed operator behavior in order to infer the operator's purpose for infecting the machine. Though some actions like searching for password files were common to most sessions, others gave us insight into operator goals. In 61% and 26% of the cases, respectively, operators attempted to monitor the user through the webcam and microphone. More niche groups of operators stole credentials and bitcoin wallets, or dropped malware and hacking tools to use the host as a staging point for further infection.

The contributions of this paper are as follows:

- We describe a system for automatically executing RAT samples in a high-interaction honeypot intended to faithfully resemble a real user, and thus elicit genuine operator behavior.
- We develop a means of scanning the Internet for DarkComet instances. We use this technique to measure the number of DarkComet controllers online over time.
- We describe the results of a measurement study of DarkComet operator behavior. We executed 1,165 unique samples of DarkComet over two separate two-week periods, resulting in 785 interactive sessions with live operators, totaling 52.9 hours of engaged operator interaction with our honeypots.
- We describe the use of RAT honeypots as a defensive measure, both as a tarpit defense, drawing attacker attention and resources from legitimate targets, and as a threat intelligence sensor. We use our experiments to assess the viability of each.

The rest of the paper is organized as follows. Section II provides background information on RATs and DarkComet in particular. Section III describes our honeypot system and measurement methodology, and Section IV describes our results. Section V discusses our results and examines possible applications of honeypots. Section VI concludes the paper.

II. Background

A. Remote Access Trojans

Remote Access Trojans (RATs) are a type of malware that give a remote attacker total interactive access to a victim machine. Most RATs allow an attacker to capture audio and video from an attached webcam and microphone, log keyboard input, browse files on the machine, edit the machine's Windows registry, and so on.

Traditional malware is built for automatically extracting value from compromised hosts at scale, whether as anonymous members of a pool of resources (e.g. botnets, Bitcoin-mining Trojans) or sources of credentials and cookies (e.g. banking Trojans). RATs, alternatively, require hands-on operator interaction with *each* compromised host in exchange for features like webcam access and audio recording, making them the tool of choice for targeted or personal attacks. RATs are used by actors of varying degrees of sophistication, for activities ranging from voyeurism and sextortion [17], [2], [6], [15], [16] to nation-state surveillance [19], [34], [40], [28], [53].

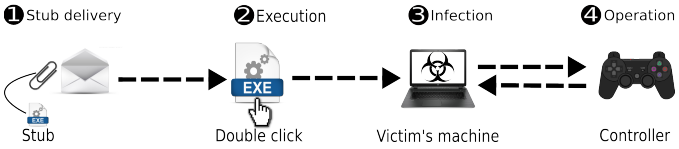


Fig. 1: RAT infection process.

Figure 1 illustrates the RAT infection process. A RAT is made up of two pieces of software: a *stub* residing on the victim's machine, and a *controller* on the attacker's machine. A RAT infection process starts with the stub being delivered to the victim, for example, as an email attachment (❶). For the infection to be successful, the victim must execute the stub (❷). (The related work showed that stubs often masquerade as images or documents via the manipulation of their icons an/or extensions [33].) Every stub is configured with its controller address, which is either a hardcoded IP address or a domain name for resolution at the time of infection. Upon infection, the stub beacons to the controller on a preconfigured port until it establishes a connection with a controller (❸). Once connected, the stub executes commands sent to it by its controller, which serves as both the command and control server for the infected machine and the RAT operator's interactive interface to victims' machines (❹). A RAT operator, colloquially known as a *ratter*, interacts with the victim's machine via a GUI that allows even unsophisticated attackers to operate a RAT.

B. DarkComet

In this work, we study DarkComet, a common, off-the-shelf RAT. It has been used by a wide variety of actors [3], [21], [19], [34], [48], [43], and exhibits an architecture and communication protocol typical of most RATs.

Once installed on the victim's machine, the DarkComet stub begins by opening a TCP connection to the pre-configured address of its controller. After a connection is established, the stub and controller complete a handshake in which the stub sends identifying information to the controller about itself (e.g. campaign ID) and the infected host. For more detail, see Appendix Section A. From this point on, communication between the controller and stub consists of manual commands issued by the operator. All communication including the handshake is RC4-encrypted using a static key concatenated to a password configured by the operator. The static key and password are embedded in the stub and can be recovered (see Section III-B), allowing us to decrypt all DarkComet communications.

C. Related Work

The closest academic works on low-volume attacks have primarily focused on the reconnaissance phases occurring *before* infection [27], [32], [33], [34]. Although Marczak et al. looked at the possible real-world consequences of these attacks, their conclusions were based on conjectures instead of live compromise monitoring [34]. In addition, the use of DarkComet in the wild has been well documented by the security community [5], [9], [21], [14], [31], [43], [53]. To our knowledge, however, we are the first to systematically study *RAT operator behavior*. We do this by executing DarkComet

samples in a sandboxed environment, an established malware analysis technique [7], [29], [30], [42], [50], [52].

D. Observational Biases

The challenges involved in measuring low-volume attacks often result in observational biases. For example, the first studies on targeted attacks were performed in direct collaboration with the Tibetans [27], Uyghurs [33], and political dissidents in the Middle East [34]. As a result, each of them is fundamentally biased towards the threats targeting its respective community. Recent work mitigated that issue by leveraging VirusTotal for data collection and by scaling analysis to hundreds of thousands of samples uploaded by tens of thousands of users [32]; however, VirusTotal uploaders are probably not representative of all victims. In particular, the authors acknowledged that VirusTotal offered "a partial coverage of attacks where individuals and NGOs [were] likely over-represented" [32]. Our measurement of RAT operators' behavior similarly exhibits several main observational biases.

DarkComet. As we focus exclusively on DarkComet, our analysis is limited to operators using this malware family. We chose DarkComet for a variety of reasons. In continuous development since 2008, DarkComet has near comprehensive functionality compared to other RAT families. Additionally, or perhaps consequently, it has been employed by the wide range of actors cited in Section II-B and maintains long-standing popularity in hacking forums like Hackforums [26]. DarkComet is still in widespread use at the time of this paper's publication, being a top-five RAT family according to VirusTotal submissions.

However, there remains the possibility that DarkComet is not representative of all other RATs, and that we could be missing the behaviors of entire classes of actors that eschew DarkComet for whatever reason. As this study is focused on *amateur* RAT operator behavior, a class of operator with whom DarkComet is known to be popular, we accept this risk while acknowledging the potential for bias.

VirusTotal. Our collection of DarkComet samples is limited to those samples uploaded to VirusTotal. While this did not prevent large-scale analysis - indeed, we collected 19,109 unique DarkComet samples during the course of our experiments - it certainly introduces biases. As stated above, certain populations are more likely to upload samples to VirusTotal than others, if at all. It is therefore possible that our sample set is not equally representative of the entire RAT ecosystem, but is instead biased towards more indiscriminate operators attempting to spread infection via public channels (a known behavior [17]), for example.

Targeted Attacks. RATs are often the tool of choice in attacks targeting specific individuals. While this class of behavior is very interesting, it is *not* the intended subject of this study. Our methodology is not designed to emulate any particular target, nor would it be feasible to do so at scale. An operator who is expecting a specific target will encounter any number of indicators that we are not this target, from IP address to system language to username. As such, our study is biased against operators conducting targeted attacks; however, it is conceivable that we receive such samples from VirusTotal, and it would be hard to exclude them from our study before execution. Sections IV-J and IV-K refer to this bias, and how it affects our experimental results.

Infection Longevity. Another class of behavior which our study does not capture is that of "return visits," operators that maintain control of an infected machine over numerous interactive sessions. Anecdotally, voyeurs and sextortionists will monitor their victims for extended periods of time, even trading or selling long-standing infections to other members of the community [17]. Our methodology is not designed to offer RAT operators extended control of a machine; rather, we observe the initial behavior of an operator gaining access to a newly-infected machine. As RAT infections are hardly stable, we expect operators' initial behavior to be deliberate and revelatory of motivation, but acknowledge that our results are biased against more patient operators.

III. Methodology

Recall that the goal of this work is to understand how one particular RAT, DarkComet, is used in the wild. Our study has two parts. The first part is concerned with the global population of DarkComet RAT operators. We collected data for this part of the study by scanning the IPv4 address space for DarkComet controllers. We describe our global scanning methodology and dataset in Section III-C.

The second part of our study is concerned with operator behavior. We collected data for this experiment by executing samples of DarkComet in a contained environment and monitoring operator behavior. We carried out two rounds of experiments, executing 1,165 samples over 31 days. Sections III-A through III-F describe these experiments, which also rely on the scanning data collected in Section III-C. We conclude the section with a discussion of limitations of this work (Section III-G) and ethical considerations (Section III-H).

A. Sample Collection

We obtain samples of DarkComet by regularly querying VirusTotal Intelligence using a set of `YARA` rules, collecting fresh samples from the service's newest submissions [1], [24]. We use an up-to-date, open-source set of DarkComet `YARA` rules [13]. On average we acquire 10 new, unique DarkComet samples per hour. As per Figure 2, we obtained 19,109 total unique DarkComet samples over the course of the study.

VirusTotal enables us to retrieve the geolocation of the IP addresses used to upload these samples. The two most popular sources of our samples are Russia and Turkey. In Appendix Section B, we see that DarkComet samples tend to be uploaded and controlled from the same countries.

B. Configuration Extraction

Malware is often packed to evade antivirus detection. DarkComet offers two runtime packing options, UPX and MPress [39], [41]. Of the 19,109 samples we collected, 18% were packed with one of these tools. 74% were not

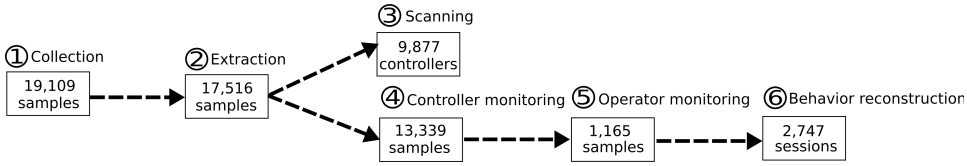


Fig. 2: Our data collection and processing workflow. Each box displays the number of corresponding entries after each step.

	Domain Names		*Campaign IDs*		*Filenames*	
Source	*Cnt*	*Pct*	*Cnt*	*Pct*	*Cnt*	*Pct*
English	321	24%	236	48%	381	38%
Turkish	56	4%	32	6%	38	3%
German	49	3%	3	-	6	-
Spanish	22	1%	7	1%	9	-
Vietnamese	18	1%	5	1%	12	1%
Other	269	20%	93	19%	131	13%
Undetermined	555	43%	110	22%	401	41%
Total	1,290		486		978	

TABLE I: Languages used in DarkComet stub configurations for each unique and alphabetic domain, campaign name, and submitted filename (sanitized for more accurate detection). *Other* is any other spoken or written language.

packed at all, and the remaining 8% were malformed. In total, we were able to automatically unpack 17,516 samples, as per Figure 2. For each unpacked sample, we extract its configuration information using an open-source RAT decoder [8]. From this information, we obtain the following:

- The **password** used to encrypt network communication to the controller.
- The **version** of DarkComet, also used in network communication encryption.
- The **campaign ID** assigned to the stub by the operator, used to manage multiple campaigns.
- A list of addresses of the stub's **controller(s)**: domain names and/or IP addresses, plus ports.

In particular, we store the stub's version and password for use in decrypting the controller's network traffic in Section III-F. We feed all controller-related information - domain names, IP addresses, ports - into our controller monitoring infrastructure (Section III-D).

Further, we perform automated language analysis of each sample's campaign ID, domain name(s), and submitted filename(s) using Google's language detection API [23]. Table I lists the results. English is the top language in all categories, while Turkish is the second most common, also across all categories. Nothing prevents operators from using languages other than their native language; however, DarkComet appears to be quite popular in Turkey currently, as Section IV will explore further.

C. Scanning

In order to gauge and monitor the overall number of DarkComet operators online, we first must discover them. Per the DarkComet protocol described in Appendix Section A, an infected host establishes contact with a controller by opening a TCP connection to it, after which the controller will indicate that it is indeed a DarkComet controller by sending a specific banner to the host. As such, it is trivial to determine whether the DarkComet controller service is running on a target machine: simply open a socket and await a properly formatted banner. We used two tools to conduct Internet-wide scans for DarkComet controllers: ZMap and Shodan.

ZMap. ZMap is an open-source tool that conducts rapid, Internet-wide network scans [18]. We scan IPv4 space on the default DarkComet port (1604) for new controllers twice daily, at our scanner machine's capacity. As Table III will show, scanning on the default port is not a *comprehensive* detection strategy, but it is sufficient to establish a baseline for the live controller population.

Shodan. Shodan is a search engine for Internet-connected devices that continuously scans the Internet for various services, including DarkComet [36]. We parse Shodan's daily response logs for new controllers.

D. Controller Monitoring

We continuously probe every known DarkComet controller to determine if the service is online or not. Our list of known controllers comes from the aforementioned ZMap and Shodan scans, as well as the addressing information extracted from our VirusTotal samples in Section III-B. Of the 17,516 samples from which we were able to extract configuration information, 13,339 were configured with valid addressing information - domain name(s) or IP address(es).

DNS Resolution. Each extracted domain name is resolved hourly. This allows us to track controllers that use dynamic DNS services to frequently change IP addresses, a very common trend among RAT operators.

Targeted Scanning. Our targeted scanner banner-grabs each suspected DarkComet socket on the Internet every 30 minutes, a concession between total coverage and covertness. It probes every address found in configuration extraction (Section III-B), plus those collected by Zmap, Shodan, and DNS resolution.

The primary purpose of this monitoring is to determine which samples are most likely to result in a connection during live analysis, as our sample submission strategy favors samples that have a controller online at the time of submission. A byproduct of our monitoring is that we collect metrics on DarkComet controller aliveness across the Internet, including uptime, longevity, volume, and geolocation.

Over the course of the experiment, we discovered and monitored 9,877 unique DarkComet controllers across the Internet, the results of which are discussed in Section IV. Table II breaks down the sources of our controllers. The high

	Total		*Unique*	
Source	*Cnt*	*Pct*	*Cnt*	*Pct*
ZMap	5,451	56%	4,417	45%
Domain Resolution	4,385	45%	3,604	37%
Shodan	810	8%	456	5%
Hard-coded IP Address	226	2%	125	1%

TABLE II: Percentages are reported as percentage of the total number of controllers monitored (n=9,877). The sum of the controllers in *Total* exceeds n, as some controllers were discovered by multiple sources. Likewise, the sum of the controllers in *Unique* falls short of n, for the same reason.

numbers of unique controllers indicate that each source largely contributes its own population to the scanner; for instance, those controllers found by ZMap did not tend to overlap with the controllers configured in our samples.

E. Live Operator Monitoring

We conducted two separate experiments, each lasting two weeks. The first began on 2016-05-04, the second on 2016-10-16. The purpose of these experiments was to monitor the behavior of live DarkComet operators in realistic machines by executing the samples obtained in Section III-A. Over the course of the two experiments, the executions of 1,165 unique samples resulted in 2,747 total runs for analysis.

Sample Selection. Samples whose controllers have responded to our scanner more recently are ranked higher in our sample submission priority queue. We submit the top $\mathcal{N}$ samples that: (a) have not previously been submitted for live analysis more than $\mathcal{M}$ times, (b) have responded to a probe in the past $\mathcal{T}$ seconds, and (c) have disjoint controller addresses. $\mathcal{N}$ is the total number of honeypots available, while $\mathcal{T}$ and $\mathcal{M}$ were determined experimentally to be 3,600 and 3, respectively.

Honeypot Design. In Experiment 1, we ran 20 honeypots concurrently. The honeypots were identical, save minimal cosmetic differences, and did not ellicit radically different responses from operators. In Experiment 2, we ran only 8 honeypots concurrently (per new system limitations), but each with a carefully designed, unique persona:

- Control: Unmodified Windows installation.
- PC gamer (male).
- Medical doctor (male).
- U.S. political figure (female).
- Academic researcher (male).
- Bitcoin miner.
- College student (female).
- Bank teller.

The honeypots are all Windows 7 VMs deployed on VMware vSphere Hypervisor [49]. All installed browsers have full access to the Internet for realism.

The results of each experiment are discussed in Section IV, while a comparison of the personas in Experiment 2 is shown in Figure 6b and discussed in corresponding Section IV-J.

Containment & Monitoring. We use the open-source Cuckoo Sandbox as our analysis platform [25]. Cuckoo includes a variety of malware-monitoring capabilities. We leverage these capabilities to (1) store all files created, deleted and downloaded during execution, (2) capture network traffic, (3) take screenshots upon screen buffer change, (4) counter anti-sandbox and anti-VM techniques, and (5) interface with many virtualization platforms (e.g. VMware vSphere).

We defined and enforced a set of containment policies to minimize the possibility of our honeypots being used to launch attacks against systems not under our control. We implemented a restrictive policy in which our honeypots' gateways filtered through a transparent SSL interception proxy. The proxy allowed HTTP(S) traffic only if the user-agent string matched one of our installed browsers. We only allowed outbound traffic over HTTP(S), and rate-limited burst traffic. We also employed a firewall, dynamically whitelisting the controller IP addresses of the samples under analysis. Experiment 1 was conducted under this set of restrictions.

We updated our containment policy for Experiment 2, allowing out all TCP and UDP traffic and removing the transparent SSL interception proxy. Instead, we deployed an IPS with DDoS prevention, SSL Blacklist rules on outbound traffic [46], and burst and bandwidth limit enforcement.

	Global Scanning		*Sample Configs*	
Port	*Cnt*	*Pct*	*Cnt*	*Pct*
Standard Port	7,928	80%	8,971	67%
Non-standard Ports	1,949†	20%	6,016‡	45%
Total	9,877		13,339	

TABLE III: A comparison of the fractions of monitored controllers and collected samples that operate on the standard DarkComet port (1604) vs. non-standard ports. Information about collected samples comes from the controller configurations extracted from them in Section III-B. Samples are sometimes configured with multiple controller addresses and ports, explaining why the sum of samples in the *Cnt* column exceeds the total. †: Comprised of 246 *unique* ports. ‡: Comprised of 1,209 *unique* ports.

F. Behavioral Reconstruction

The behavior of each RAT operator is reconstructed from the `.pcap` of the related trial. We decrypt the DarkComet traffic present in the file, and then use our signature engine to decode DarkComet's protocol and extract the sequence of operator commands and relevant metadata.

Network Decrypter. Given a `.pcap`, our decrypter reassembles each TCP flow and then decrypts any DarkComet traffic using the encryption key extracted in Section III-B. DarkComet uses RC4 for both network encryption and resource storage (e.g. RAT configurations). The encryption key is a combination of version number, a constant (depending on version), and an optional password set by the RAT operator.

Signature Engine. We developed a comprehensive set of DarkComet network signatures for versions 5.0+ using a combination of static analysis and exhaustive testing of DarkComet commands. The signature engine takes its input from the network decoder. If a network signature matches the decoded traffic, the engine returns the action performed (e.g. `Uploads`

`& Executes Binary File`), the source of the action (controller or victim), the mode of the action performed (automatic or manual), and other tags for metadata extraction.

Remote Desktop (RDP). DarkComet includes RDP functionality, allowing the operator to interact with the victim's OS directly. DarkComet RDP works by periodically sending the operator a JPEG-encoded image of the victim's screen, while the operator can issue mouse click and keyboard events to the victim machine. Our signature engine can decode operator interaction during RDP (click coordinates and keystrokes), but at such fine granularity cannot infer the operator's actual actions on the victim machine. Rather, to fully reconstruct RDP sessions, we configured Cuckoo to capture and timestamp the screen any time it changed. We then manually annotated these images in order to reconstruct operator RDP sessions, using detected keystrokes and clicks as a filter on the screenshots.

G. Limitations

ZMap Coverage. Our ZMap scans, intended to discover the global DarkComet controller population, only run on the default DarkComet port, 1604. Table III demonstrates the limitations of this strategy; while 80% of our discovered DarkComet controllers run on the default port, a full 45% of collected samples are configured with at least one controller using a non-default port. However, these samples were configured with 1,209 unique non-default ports; scanning globally on that many ports was simply beyond our capacity.

RDP Inspection. As we will see, 83% of connected operators use DarkComet's remote desktop functionality (RDP) to control the victim machine. Our methodology, manually inspecting screenshots of RDP sessions to build complete behavioral profiles of operators, is not a scalable solution.

Victim User Data Feeds. Much of the observed operator behavior involved attempted interaction with the victim user. 61% of operators attempted to access the victims's webcam, and 26% the microphone. Further, 8% of operators attempted to chat with the victim via DarkComet's chat client. Our honeypot provides neither a webcam nor an audio feed, nor do we simulate a user that can respond to the operators' chat attempts. This is a major limitation which we will discuss in our lessons learned (Section V-C).

Prior Knowledge. Sample unpacking and decoding is the most thorough method for obtaining RAT configuration information from our samples; however, its applicability is limited. We encountered little packer diversity in our sample set, with unpackers being freely available for the two types we did mainly encounter. However, we were still unable to unpack and decode 8% of obtained samples. Likewise, the RAT decoders we used - both the network command decoder and the sample configuration extractor - have finite coverage; were we to expand our study to include multiple RAT families (of which there are hundreds), the effort required to produce a general decoder would be immense.

Narrow Scope. Our network decoder does not process non-DarkComet actions. Were a DarkComet operator to upload and execute a tool to use through remote desktop, for instance, we would not be able to decode operator interaction with said tool. As shown in Table VIII, 23 operators did just this. Further, an operator could upload a new configuration (e.g. a new password), or even another type of RAT. Any interaction with this new malware would be un-decipherable. During our trials, operators did, in fact, upload 35 unique RAT executables (see Section IV-E), 13 of which are new DarkComet configurations.

Sandbox Evasion. RATs (and operators) can also detect and abort execution in virtualized environments. More specifically, there have been efforts to detect if a binary is running inside a Cuckoo sandbox [44]. Cuckoo implements some countermeasures to these techniques [11], but they are not comprehensive.

H. Ethics

Unlike most types of malware analysis, this work entails interaction with human subjects: the RAT operators. Research involving human subjects imposes ethical obligations on us as researchers and requires additional institutional oversight to ensure that those obligations are met. We sought and obtained approval from the UC San Diego Institutional Review Board (IRB) for this study.

In our experiments, the biggest concern is that exposure of Personally Identifying Information (PII) about the operator may cause harm to the operator. To minimize the risk of harm, the raw data was only analyzed by members of the research team that were part of the institution whose IRB approved the study, and was stored encrypted. All IP addresses and network traces collected were hashed after geo-location was performed using a local copy of the MaxMind GeoLiteCity database [38]. All further analysis was performed on the hashed IP addresses and sanitized network and API traces. The raw screenshots were transcribed and all PII was removed before performing additional analysis. No PII is included in this paper and after publication all raw data will be deleted, with anonymized versions of the data saved for 2 years. All RAT operators voluntarily interacted with our honeypots and we made no attempt to obtain PII information from them or to actively identify individuals that connected to our honeypots.

The second concern is that our honeypots might be used as stepping stones for attacks targeting other systems not under our control. In order to mitigate this risk, our monitoring honeypots were configured with a conservative firewall containment policy that only allowed outbound connection to the RAT's controller and any other server that initiated a connection with the honeypot first. The one exception is that we allowed outbound HTTP and HTTPS (man-in-the-middled to check headers) traffic from our honeypots through our HTTP proxy server only if the user-agent exactly matched one of the installed browsers. The user-agent could be spoofed by an attacker, but we found no evidence of an attacker doing this in our dataset. We actively analyzed the network activity from our honeypots to detect if attacks were evading our containment and did not witness any attacks that were not blocked by our firewall containment. We feel that our containment methods and protocols for analyzing data minimized potential harms while allowing us to perform measurements that will benefit

the security community with increased understanding of the behavior of manual attackers.

During the second round of RAT executions we deviated slightly from the approved IRB protocol in terms of our containment implementation. Instead of implementing a strict set of firewall rules, we implemented an IDS that was integrated with our firewall and would block any malicious messages detected by the IDS. We have notified our IRB of this slight deviation and requested an IRB protocol amendment that would approve this new containment policy. Our analysis of the network traffic generated by operators indicated that this new containment implementation likely did not allow any harmful messages to be sent using our honeypots.

IV. Analysis

A. Global Operator Analysis

As noted in our methodology, we developed a method for scanning the Internet for DarkComet controllers. This allows us to poll all discovered controllers on the Internet at any given time. Note that this includes controllers for which we do not have a sample. Figure 4 shows the results of the continuous scan, with a series for all monitored controllers as well as a series for just those controllers that connected to our honeypots during live trials. At any given time, we are monitoring about 175 online DarkComet controllers. As our scanning is not comprehensive, these numbers only provide a baseline for the actual number of DarkComet controllers on the Internet.

The cumulative number of unique controllers discovered during scanning increased essentially linearly over the course of the measurement. Many controller domain names from our samples used a dynamic DNS service. We suspect that this steady growth is at least partly due to IP address churn. Indeed, 45% of monitored controllers were discovered by continuous resolution of DarkComet-associated domain names.

	Global Scanning		*Live Trials*	
Country	*Cnt*	*Pct*	*Cnt*	*Pct*
Turkey	3,680	37%	222	25%
Russian Federation	1,495	15%	188	21%
United States	319	3%	36	4%
Brazil	306	3%	40	4%
France	283	2%	22	2%
Ukraine	282	2%	52	5%
Other	3,512	36%	307	35%
Total	9,877		867	

TABLE IV: Countries of the IP addresses of a) the global population of scanned DarkComet controllers, and b) the controllers to which our live trials connected, as resolved by MaxMind's GeoLiteCity database [38]. Addresses without resolution are omitted.

	Global Scanning		*Live Trials*	
Country	*Cnt*	*Pct*	*Cnt*	*Pct*
Residential	8,830	89%	779	90%
Hosting	704	7%	54	6%
Cellular	288	3%	24	3%
Other	47	-	6	-
Undetermined	8	-	4	-
Total	9,877		867	

TABLE V: User-types of the IP addresses of a) the global population of scanned DarkComet controllers, and b) the controllers to which our live trials connected, as resolved by MaxMind's GeoIP2 Insight service [37]. Addresses without resolution are omitted.

Table IV shows the geographic distribution of controller IP addresses that we monitored globally and that connected to our honeypots. Russia and Turkey are the most prevalent, but operators may be using a VPN service. If this is the case, then the IP location will indicate the location of the VPN rather than the actual operator location. Therefore, Table IV should be interpreted with caution.

Figure 3b shows the average number of controllers online, binned by day of the week. Note that controller aliveness trends upwards towards the weekend, with about 20% more controllers online on Sunday than Monday. We suspect that this is due to our focus on "casual" RAT operators who may be online only during weekends. Figure 3a shows the same data by hour of the day. If we assume that the geolocation data in Table IV represents true operator location, then the peak between 16:00 and 17:00 UTC falls in the early evening in Eastern Europe, while the trough at 2:00 UTC falls in the very early morning in Eastern Europe. This again suggests that at least some of the monitored RAT operators are "casual" ratters. Nevertheless, even at the lowest point, there are over 100 controllers online.

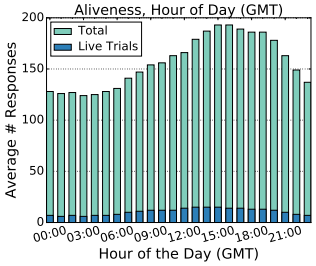
Figure 8 gives us reason to suspect that the data in Table IV is indicative of the actual operator geolocation, so we produce the previous graphs for just the controllers that geolocate to Turkey, adjusted to EET (Turkey's timezone). The results, illustrated in Figures 3c and 3d, clearly support the expected "casual operator" trends: weekend activity is significantly higher than weekday activity, and early evening activity dwarfs early morning activity.

The user-type distribution reported in Table V further supports this trend. Almost 90% of the discovered controllers operate behind IP addresses with residential user-types. This suggests that the majority of controllers are run on residential networks, likely with little of the operational security often seen in botnet proxies.

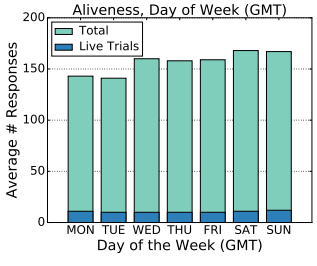
B. Operator Behavior Analysis: Overview

While measuring the global population of DarkComet controllers yields interesting results, it is, in fact, a secondary contribution; our primary contribution is to understand DarkComet operator behavior in the wild.

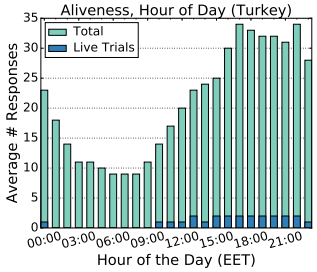
To do so, we ran 1,165 unique DarkComet samples over the course of two, several-week-long experiments following the methodology described in Section III. Overall, our experiments ran for nearly 2,400 combined hours, divided approximately equally between the honeypots executing in parallel. In all, the experiment accumulated 52.9 hours connected to a DarkComet controller. The average DarkComet session lasted about 4 minutes, while the average DarkComet session with RDP lasted about 7 minutes. In this section, we report what operators did during these sessions. In some cases, operator actions give us a clear indication of *motive* and *process*. In addition, we examine



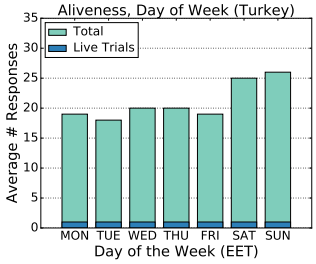
(a) Average number of DarkComet controller nodes online per hour of the day.



(b) Average number of DarkComet controller nodes online per day of the week.



(c) Average number of DarkComet controller nodes online per hour of the day in Turkey.



(d) Average number of DarkComet controller nodes online per day of the week in Turkey.

Fig. 3: Scanning breakdowns by day and hour.

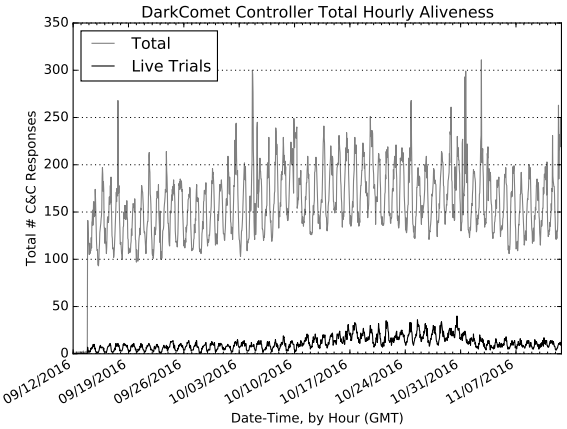


Fig. 4: Number of DarkComet controller nodes online each hour, binned by hour. We detected 9,877 unique controllers in total. Counts are from an Internet-wide scan for active DarkComet controllers, and include controllers for which we do not have a RAT sample.

	Experiment 1		*Experiment 2*	
	Total	*Daily*	*Total*	*Daily*
Unique Samples	793	49.6	478	34.1
Unique Controllers	432	27.0	439	31.4
Total Runs	1,725	107.8	1,022	73.0
Total Connections	830	51.9	461	32.9
Total Interactions	531	33.2	254	18.1
Start Date	2016-05-04		2016-10-16	
End Date	2016-05-20		2016-10-31	

TABLE VI: Summary statistics of the two experiments in the dataset. *Unique Samples* is the total number of unique samples executed; *Unique Controllers* is the total number of unique controllers to which the samples connected; *Total Runs* is the total number of individual executions we performed; *Total Connections* is the total number of trials in which the sample connected to a live controller; *Total Interactions* is the total number of trials in which an operator interacted with a honeypot; *Start Date* and *End Date* are the date range over which the experiment was conducted.

the actionable information we can glean about operators, and whether there are elements of our setup that hindered our ability to observe DarkComet operators. This section presents an analysis of those executions.

Table VI provides a summary of the two experiments, broken down by unique subject and trial result. We monitored 2,747 interactive sessions overall. Of those sessions, 785 resulted in direct operator engagement with our environment. Almost 75% of samples that resulted in operator interaction were executed within a month of submission to VirusTotal, and a full 95% within four months. For more detail about the relative age of our samples, see Figure 9 in Appendix Section C. Additionally, for information about the schedule by which we executed samples, see Appendix Section D.

DarkComet provides two different ways of interacting with the victim machine: directly using RDP or indirectly using DarkComet commands. Table VII reports observed commands separately by sessions where there was RDP activity (47% of sessions), where RDP was started but there was no interaction (36% of sessions), and where RDP was not started (17% of sessions) in columns *RDP Act.*, *RDP Pass.*, and *RDP None*, respectively. The rest of this section outlines various facets of operator behavior, from actions performed to engagement.

C. Common Actions

Table VII shows detailed information about all operator activity broken down by RDP status and activity category. The most common actions we observed across all three session types were webcam monitoring, password theft and file exfiltration. Operators attempted to access our webcam in 61% of all trials. Stored passwords were grabbed in 43% of all trials. The victim filesystem was explored in 40% of all trials. Other types of user monitoring were prevalent as well, with operators attempting audio capture and keylogging in 26% and 31% of trials, respectively. The prevalence of actions attempting to collect physical information about the user as well as their files suggests that surveillance is a dominant use of DarkComet, and, in fact, its intended use.

Some operator actions tended to occur more frequently with or without RDP. For example, webcam capture occurred in 76% of active RDP runs, compared to only 16% of non-RDP runs. Conversely, operators attempted to uninstall the

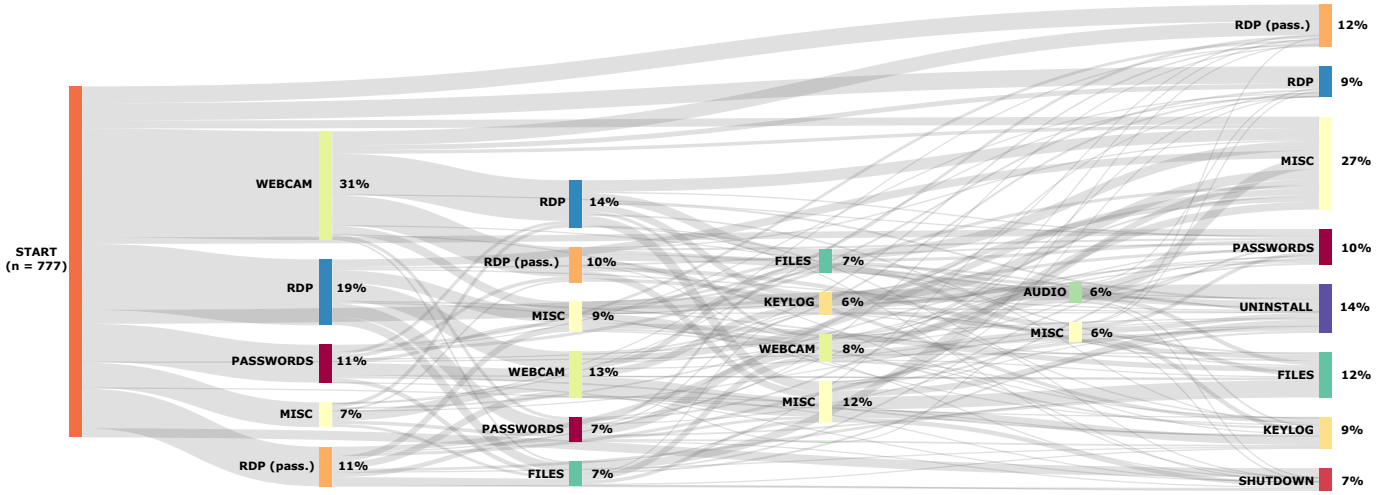


Fig. 5: Composite flowchart of prevalent operator behaviors and sequences broken down by RAT interaction phase and category. Individual paths are labeled with the percentage of all executions that traversed that edge. Sequences occurring in fewer than 5% of engaged executions are omitted. The figure shows an operator preference for engaging with remote desktop or surveillance first in a majority of trials. See Section IV-C for additional details.

	RDP			
Category	*Act.*	*Pass.*	*None*	*Total*
AUDIO CAPTURE				
Record Microphone Sound	43%	15%	2%	26%
AUTOMATIC				
URL Download	–	–	1%	–
URL Visit	–	2%	1%	1%
IP Scanner	–	–	1%	–
Uninstall Server	–	–	1%	–
COMPUTER POWER				
Restart	5%	1%	1%	3%
Shutdown	4%	2%	4%	3%
Poweroff	1%	–	1%	1%
Logoff	1%	–	1%	–
FILE MANAGER				
Explore Files	52%	36%	15%	40%
Upload File	14%	5%	2%	9%
Download File	12%	7%	–	8%
Delete File	7%	3%	–	5%
Modify File	4%	2%	–	3%
Search for Files	2%	–	–	1%
FUN FUNCTIONS				
Fun Manager	19%	3%	2%	10%
Chat	15%	1%	2%	8%
Message Box	11%	1%	–	6%
Piano	11%	2%	1%	6%
Microsoft Reader	4%	–	1%	2%
MISC FUNCTIONS				
Clipboard	2%	1%	–	2%
Print Manager	1%	–	–	–
MSN FUNCTIONS				
MSN Control	2%	–	–	1%
MSN Contacts	1%	1%	–	1%
NETWORK FUNCTIONS				
Browse Page	7%	1%	–	3%
LAN Computers	5%	3%	–	4%
WIFI Access Points	5%	2%	–	3%
URL Download	4%	1%	1%	3%
Active Ports	3%	1%	–	2%
URL Visit	3%	1%	–	2%
IP Scanner	2%	1%	1%	2%
Net Gateway	2%	1%	–	2%
Network Shares	1%	1%	–	1%
DDOS Attack	1%	1%	1%	1%
SOCKS5 Proxy	1%	–	–	–

	RDP			
Category	*Act.*	*Pass.*	*None*	*Total*
PASSWORDS DATA				
Stored Passwords	51%	36%	31%	43%
uTorrent Downloads	9%	1%	–	5%
REMOTE MSCONFIG				
Services Startup	4%	2%	–	2%
Registry Startup	1%	1%	–	1%
REMOTE SCRIPTING				
HTML or VB Scripting	4%	2%	1%	3%
Binary	3%	1%	10%	4%
Batch Scripting	1%	–	–	1%
URL	1%	–	1%	–
RESTART SOCKET				
Server	–	–	–	–
SERVER ACTIONS				
Remote Edit Server	2%	2%	1%	2%
Restart Server	2%	2%	–	1%
Lock Computer	1%	–	–	–
Upload and Execute	–	–	–	–
SERVER REMOVAL				
Uninstall Server	7%	17%	27%	14%
SERVER SHUTDOWN				
Close Server	5%	6%	16%	7%
SPY FUNCTIONS				
Keylogger	41%	27%	13%	31%
SYSTEM FUNCTIONS				
Process Manager	20%	7%	4%	12%
Remote Shell Start	9%	3%	1%	6%
Remote Shell Stop	9%	3%	1%	6%
Windows List	8%	1%	–	4%
Remote Shell Command	7%	1%	1%	4%
Uninstall Applications	5%	1%	–	3%
Remote Registry	5%	1%	–	3%
System Privileges	4%	1%	–	3%
Hosts File	2%	1%	–	1%
SYSTEM INFO				
Computer Info	11%	5%	–	7%
Trace Map	5%	1%	–	3%
UPDATE SERVER				
From File	1%	1%	2%	1%
From URL	–	–	–	–
WEBCAM CAPTURE				
Attempt Webcam Access	76%	64%	16%	61%

TABLE VII: Categories of network signatures used in the experiment. These categories abstract the 151 network signatures present in the dataset. Prevalence of each category broken down by trials that contained remote desktop activity (367 trials), those that contained passive remote desktop sessions (287 trials), those without remote desktop (134 trials), and trials overall (788 trials). Remote desktop activity is defined as any remote desktop I/O event.

malware stub in 7% of RDP runs compared to 27% of non-RDP runs. Operators checked the running processes of the victim in 20% of RDP runs when compared to only 4% of non-RDP runs. Also, a binary was dropped in only 3% of the RDP runs as compared to 10% of non-RDP runs, showing us that DarkComet is also used in an automated fashion to drop malware. These differences continue to lesser degrees across many other actions. The contrast in actions between RDP and non-RDP runs may indicate a variety of uses of DarkComet depending on operator goals.

While Table VII shows aggregate statistics of operator actions, it says nothing about the *order* in which the actions take place. Figure 5 is a Sankey diagram showing the sequences of the actions most reported in Table VII. Each node shows the percentage of total trials that perform the corresponding action at that point in their sequence. For example, 14% of trials initiate an active RDP session as their second action. All percentages are based on the total trials.

One of the main takeaways from this diagram is that most trials are comprised of a subset of these very common actions: webcam access, remote desktop, password theft, file exploration, and audio capture and keylogging. This is perhaps one of the most prominent indicators of our operators' motivations, which we discuss further in Section IV-L.

Other findings: A non-trivial number of trials interact with the victim machine solely through RDP. A full 14% of operators uninstall the stub as their final action, discussed more in Section IV-K. 31% of operators attempt to access the webcam before any other action, despite the fact that a webcam light could expose them to an online victim; the desire to view the victim environment is that great, whatever the motivation.

D. Data Collection

An operator conducting reconnaissance against a newly infected system has near-total access to the system through the DarkComet GUI, including full filesystem and registry access; lists of open windows, startup services, processes, and applications; and a plethora of network interrogation tools.

As we observe in Table VII, 40% of live trials involve filesystem exploration, roughly the same amount as all other forms of reconnaissance combined. Further, 8% of trials involve downloading files from the honeypot to the operator's machine, across 600 distinct events. Operators that downloaded files exhibited certain patterns. 34% collected files from the desktop. 8% collected the 'My Documents' folder. And in instances where it was available, 33% stole a bitcoin wallet.

DarkComet also offers a variety of other data exfiltration tools beyond downloading files, such as automated functionality for collecting and downloading passwords, user account information, keylogger logs, and clipboard text. Table VII shows us that, although only 8% of trials involve the downloading files or directories, a full 43% do password collection, and 31% attempt to collect user keystrokes using the keylogger.

After Experiment 1 indicated such high levels of password collection, we decided to provide actual account credentials in Experiment 2 to monitor access attempts. We created a number of accounts for each persona at popular websites, saving the credentials in the browser. We protected each with two-factor authentication, both to prevent account abuse and to record each access attempt, the language of the machine attempting access, and the exact time of the attempt. Over the course of the experiment, we observed 13 access attempts: nine attempts for two popular email services, three for two different gaming-related services, and one for a social media site. Of these attempts, three were made from machines configured for Turkish, two for Russian, and the remainder English.

E. Dropped files

In 28 of our trials, we observed files being dropped by the operator. We consider only files dropped explicitly by the operator during interaction, not embedded files automatically dropped during sample execution. In 29% of these trials, an operator performed only the single action of dropping a file, akin to a traditional malware dropper. In 18% of trials, we observe that an operator dropped a file after checking the webcam of the victim and deploying a keylogger. It is interesting to note that almost an equal number of trials start with webcam as the first action followed by dropping files; this suggests RAT operators are interested in person-to-person interaction as much as dropping more malicious files.

Over the course of the study, operators dropped 48 unique files. 34 (71%) were executables, 19 of which were not previously seen by VirusTotal. 35 were RAT stubs, 13 being DarkComet stubs with new configurations, 5 njRAT, 3 NetWire, and the remainder custom RATs and worms. Operators also dropped 5 distinct scripts - `HTML` executables, Batch files - for executing destructive actions on the host machine.

F. Visiting URLs

DarkComet operators visited 123 non-unique URLs from 33 unique domains during the course of our experiment. Out of these URLs, 26 contained adult content, 13 gaming, seven personal blogs, six streaming, and five internet pranks. The remaining URLs were VPN, search, banking, social, education, IP address tools, and file storage websites. 23 URLs generated a 404 error and one had an unregistered domain. 20 of these URLs were written in a language different from English: Nine in Russian, six in Turkish, and five in Japanese. Finally, operators performed 18 visits to eight unique IP addresses (five of them in the same \24 subnet).

G. Command Line Activity

DarkComet provides a command line interface, allowing operators to interact directly with the victim machine. Our signature engine extracts all commands issued by the operator from the command shell buffer. Inspecting the 92 commands across all trials, we classify the operators' intentions as such:
Reconnaissance. 60% of commands pulled information on system configuration, file system, and network configuration.
Manipulation. 26% launched or terminated processes.
Destruction. 10% damaged the host machine or filesystem.
Concealment. 3% hid operator actions, e.g.:

```
ECHO OFF DEL *.*/Q DEL:VIRUS.BAT
```

H. Operator Interaction with the User

RAT infections are personal affairs. Not only does a human operator interact individually with each victim machine, but they often also monitor the machine's user as well. Indeed, the image most commonly associated with a RAT infection is that of an attacker watching a victim through their machine's webcam, an image well supported by historical anecdotes [4], [16]. There are a variety of motives that would compel a ratter to interact with their victim (passively or actively), and DarkComet provides an array of capabilities to do them all.

Surveillance. Victim surveillance is, perhaps, the most notorious activity associated with RAT infections. DarkComet offers access to live feeds from the infected machine's webcam and microphone. Though our honeypots offer neither, we do detect attempts to access both. As is shown in Table VII, attempts to access these are prevalent, with 61% of sessions attempting to access the webcam, and 26% the microphone.

Operators which engaged in remote desktop tended to use webcam and microphone monitoring more than their counterparts. 76% of remote desktop users accessed the webcam, compared to 16% of non-remote desktop users. Likewise, 43% of remote desktop users accessed the microphone, compared to 2% of non-remote desktop users. This correlation further indicates that operators using remote desktop are more engaged in the RAT session overall, as user monitoring and remote desktop are both time-intensive activities.

Direct Communication. Beyond passive surveillance, DarkComet provides operators several means to communicate with the victim directly. These include opening a two-way chat client on the victim's screen, displaying a pop-up alert message, sending text to the victim's printer, and reading messages aloud using Microsoft Reader. We observed operators attempting to communicate with the victim through each medium, as per the "Fun Functions" section of Table VII. We classify operator communications with four categories:

Harassment. 53% of communications were intended to harass the victim. Threats, attempts to induce fear, and heavy sexual innuendo comprise the majority of victim harassment, e.g.:

```
YOU ARE NOT ALONE, I SEE AND HEAR YOU
```

Extortion. 2% of communications were intended to exact a ransom from the user in return for control of the machine. As our victims could not respond to the operators, none of these demands persisted. Further, we suspect that some portion of harassment leads to extortion in scenarios where a victim is actively replying. Example:

```
HI <redact> WANNA PAY ME $50 TO NOT SHARE ALL
UR IMFORMATION TO THE WORLD.... EVERYTHING U EVER
BEEN ON UR PC
```

Misdirection. 16% of communications were legitimate-looking message boxes, either to cover the installation of the RAT (in the case of error messages), or to gain a victim's trust and "phish" them, e.g.:

```
THIS IS MICROSOFT (TM) or YOU MAY BE AT RISK OF
A VIRUS PLEASE LOG OUT OF ALL GAMES AND BANKING
SITES ON INTERNET
```

Recognition. A full 9% of communications were operators posting their hacker groups and tags, taking responsibility for the "hack." Though juvenile, this provides potentially useful attribution information, e.g.:

```
HACKED BY #ZONED OUT XDDDDDD
```

We used Google's language detection API on our operators' communications, but ethereal text like IM chat generates mostly misclassifications. See Appendix Section E for results.

Visibility. This level of interaction with the victim highlights an interesting phenomenon: a significant portion of our operators choose to be blatantly visible to the victim, either through direct communication or via actions that manipulate the victim's screen. We can categorize operators by their degree of visibility; specifically, whether they are *intentionally* visible or not. Visible actions comprise any action the operator initiates that is visible to the victim (e.g. opening a chat window), and exclude more discreet actions that could still be detected (e.g. modifying a file's attributes).

We observe that, of the 785 total engaged trials, 62% are visible to the victim. Lack of discretion is often an indication of unsophisticated operators, and we have some confidence that this is the case. However, lack of discretion does not imply harmlessness. Of communications directed at the victim, 2% were extortion demands and 16% were attempts at deception. These operators used visibility intentionally to intimidate or deceive the victim to some goal.

I. Remote Desktop Sessions

One of DarkComet's most heavily used features is remote desktop (RDP), with 654 total trials (83% overall) containing remote desktop sessions. RDP offers the operator immersive control of the victim, particularly the ability to interact with programs that require GUI interaction; however, the tradeoff is that the operator's actions are completely visible to the victim.

Though visible to the user, remote desktop functionality has been abused by criminals. In the 2016 TeamViewer compromise, attackers hijacked TeamViewer clients (which *only* provide remote desktop) and made purchases with victims' eBay and PayPal accounts late at night to avoid being seen [22]. Our operators' tendencies toward remote desktop usage for user data theft should not be discounted.

Using the methods described in Section III-F, we found that 44% of RDP sessions had no operator input. In this section, we report on the 367 sessions that had *active* RDP engagement; that is, where the operator actually interacted with the victim via RDP. We manually inspected each RDP session to determine the operator's actions, the results of which are summarized in Table VIII. These findings are complementary to the data in Table VII.

Hacking Tools. The deployment of additional hacking tools occured in 4% of our trials. We suspect these trials use remote desktop out of necessity, as the tools deployed were GUI-based. We observed the following: Android mobile phone RAT builders & infectors; webcam and Skype screen recording tools; spam bots; YouTube view-fraud bots; DLL injectors; and the Havij SQL injector. Some of these tools were bundled

with the DarkComet stub. We were surprised to find operators using these tools in plain view of the victim, indicating that the operators felt there was no need for secrecy.

Masquerading as Legitimate Programs. A pattern emerged in the remote desktop screenshots which followed pre-existing RAT literature: the bundling of DarkComet with legitimate tools. We found that 5% of remote desktop users pack their DarkComet samples with legitimate software programs.

We also found a number of samples impersonating software programs like Adobe Photoshop and WinRAR, as well as utilities like .NET Framework Setup Cleanup Utility and even Avast Antivirus. Cracked versions of these programs were distributed bundled with DarkComet, suggesting that the initial DarkComet infection vector was pirated software distribution.

The programs our samples most commonly impersonated are gaming tools: ping checkers, cracked versions of popular games (`HackFifa.exe`), game enhancements (Enhanced Minecraft Shaders), and "cheat" tools to gain an unfair advantage in online games. We discuss campaigns against the gaming community further in Section V-B.

User Data Collection. Table VII indicates that our operators tend to target user data, with 43% of trials collecting passwords, 31% using a keylogger, and 41% exploring the filesystem in depth. It comes as no surprise that operators use remote desktop to the same end, with 63% of active remote desktop sessions being used for user data collection.

Operators leveraged built-in browsers to access user accounts, browsing history, and download history. We suspect this is a combination of data collection and gauging level of interest in the victim machine. Most operators checked Chrome and Firefox, and some investigated Internet Explorer. Some operators navigated to popular Web sites. We strongly suspect this is to determine if the user has an active session, and, if so, to hijack the session. The websites most commonly targeted were social media sites: Facebook.com and VK.com (a popular Russian social media platform).

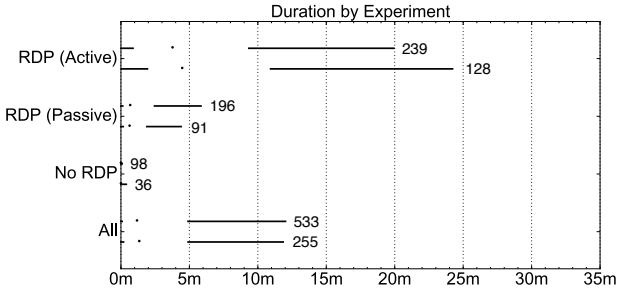
Operators also accessed installed applications, Dropbox and Steam in particular. In fact, 33 of the 48 trials that accessed applications targeted Steam, lending more credibility toward the gaming campaigns discussed in Section V-B.

Finally, operators inspected system information, typically exhibiting one of two behaviors: checking WiFi and network connectivity settings, or using the DirectX diagnostic tool. Both behaviors appear to be system vetting, especially DirectX, which even reveals the presence of virtualization.

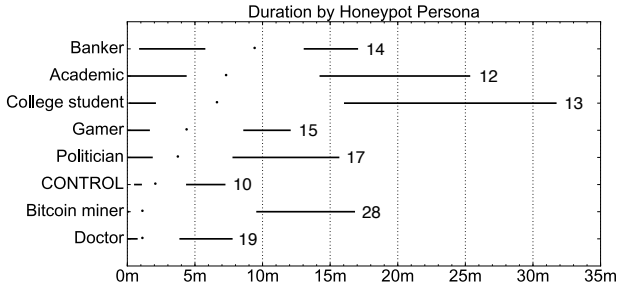
Display of Illicit Material. We noticed that 4% of active remote desktop sessions involved the display of online video pornography, clearly visible to the victim user. There are multiple motivations behind this action. A subset of operators use pornography to enhance their chat interactions with the victim, threatening to continue displaying porn until demands are met or intimidating the victim. We speculate another subset simply use pornography to harass the victim, given the high rates of harassment and "Fun Function" usage in Table VII. The display of pornography likely indicates an operator's short-term motivations, given its striking visibility.

Behavior	*Count*	*Percentage*
User Data Collection	360	63%
System information	170	29%
Browser data	72	12%
Application and Online account data	64	11%
Filesystem exploration	54	9%
Targeting of Gaming-related Data	52	9%
Alternative Communication Methods	37	6%
Masquerading as Legitimate Programs	30	5%
Illicit Material	23	4%
Deployment of Hacking Tools	23	4%
Suspected YouTube View Fraud	20	3%
Personally Identifiable Information	6	1%

TABLE VIII: Common patterns of behavior exhibited by operators employing active remote desktop sessions.



(a) Comparison of the time spent in the honeypots by operators in the first and second experiments, in minutes. Operators are distinguished by their use of RDP, as per Table VII.



(b) Comparison of the time spent between honeypot personas (in only the second experiment), in minutes. Only sessions with active RDP are reported, as these are representative of the whole. Operators are distinguised by the honeypot persona with which they interacted.

Fig. 6: These box plots represent the time spent in our honeypots by operators. Note that the number of trials is reported next to each plot.

Alternative Communication Methods. As a final note of interest, we observed several alternative methods of communication with the victim specific to remote desktop. Operators communicated using Notepad, Paint, the command line, Word documents, and uploaded images. Overall, 6% of remote desktop trials involved some form of alternate communication.

J. Operator Time Engagement

Most of DarkComet's numerous functions must be manually executed by the operator. From the timestamps associated with these actions, we can develop an understanding of how long an operator engages with each infection, and how that varies by their own actions.

Metric for Determining Operator Engagement Duration. From enumerating all possible DarkComet actions and exploring their functionality, we identified all actions that require manual operator intervention. From our execution traces, we identified all such actions and calculated the time difference between any two adjacent manual actions. If the time difference between these two actions was less than 60 seconds, we considered this to be a single period of engagement. Using this metric we summed the complete engagement time for each trial, and report this as operator engagement.

Engagement Results. Figure 6 presents an overview of operator engagement throughout our dataset. Figure 6a compares the overall operator engagement times between the two experiments, while Figure 6b breaks down engagement time in Experiment 2 by honeypot persona (for active RDP users).

In Figure 6a, we observe the operators' engagement time with the honeypots, as determined by our metric. The two series represent the two experiments, each of which is broken down by RDP usage, as in Table VII. The average overall engagement time in both experiments is 4 minutes.

The extent of operator engagement varies significantly between RDP category. Trials with active RDP had at least 60% more actions and engagement than average across both experiments, while trials with no RDP activity at all showed almost no activity - an average engagement below 20 seconds for both experiments. These differences indicate that the primary way operators manually engage with victims is via RDP, despite using many non-RDP features as well. As such, RDP usage appears to be an indicator of operator engagement and interest in the victim.

In all RDP categories, operator engagement also saw high variance, with standard deviations exceeding the mean in all cases. All categories also saw several large duration runs skew the average significantly. This indicates a wide variety of operator interests, ranging from cursory to extreme. This non-trivial level of operator engagement has implications for RAT defense, discussed in Section V-A.

Influence of Honeypot Persona. Piqued by the results of Experiment 1, we decided to measure the affect of honeypot persona on operator engagement. Figure 6b provides a comparison of operator time engagement across the 8 honeypot personas in Experiment 2. We display only the results of active RDP sessions, which constitute truly engaged operators.

The honeypots (recall Section III-E) are displayed in order of increasing median: doctor, bitcoin miner, *control*, politician, gamer, college student, academic, banker. The results are quite distinct. The three personas that exceed the mean (college student, academic, banker) show engagement nearly three times greater than the control. It is difficult to say exactly what factors specifically contributed to the engagement difference, but one factor that appears to play a prominent role is *file system depth*. An analysis of actions by persona indicates that file system exploration is more prevalent in the three more engaged personas. Further, these personas were designed with more detailed file systems than the *control* or bitcoin miner, for instance. Overall, it seems that the design and depth of the victim machine's filesystem directly affects operator engagement time, though a more regimented study focused on just this aspect is necessary to confirm our hypothesis.

Category	*RDP*			
	Act. (n=367)	*Pass. (n=287)*	*None (n=131)*	*All (n=785)*
REMOTE DESKTOP, ACTIVE: Remote Desktop I/O	32%	-	-	15%
SERVER REMOVAL: Uninstall Server	7%	16%	27%	14%
REMOTE DESKTOP, PASSIVE: Remote Desktop Start	-	30%	-	11%
FILE MANAGER: Explore Files	7%	10%	8%	8%
REMOTE DESKTOP, ACTIVE: Remote Desktop Start	15%	-	-	7%
REMOTE DESKTOP, PASSIVE: Remote Desktop Stop	-	16%	-	6%
REMOTE DESKTOP, ACTIVE: Remote Desktop Stop	12%	-	-	6%
PASSWORDS DATA: Stored Passwords	2%	6%	15%	6%
SPY FUNCTIONS: Keylogger	3%	6%	9%	5%
WEBCAM CAPTURE: Attempt Webcam Access	3%	5%	6%	4%
Session termination was only manual action	-	-	17%	3%
AUDIO CAPTURE: Record Microphone Sound	3%	1%	2%	2%
REMOTE SCRIPTING: Binary	-	-	9%	2%
FUN FUNCTIONS: Fun Manager	2%	1%	-	1%
FILE MANAGER: Download File	-	2%	-	1%

TABLE IX: Last operator actions before session termination for trials with manual interaction. Actions that appear in less than 1% of trials are omitted.

Category	*RDP*			
	Act. (n=25)	*Pass. (n=45)*	*None (n=36)*	*All (n=106)*
Session termination was only manual action	-	-	83%	28%
REMOTE DESKTOP, PASSIVE: Remote Desktop Start	-	47%	-	20%
FILE MANAGER: Explore Files	4%	18%	3%	9%
REMOTE DESKTOP, ACTIVE: Remote Desktop I/O	32%	-	-	8%
PASSWORDS DATA: Stored Passwords	4%	9%	6%	7%
WEBCAM CAPTURE: Attempt Webcam Access	8%	4%	6%	6%
REMOTE DESKTOP, PASSIVE: Remote Desktop Stop	-	13%	-	6%
SPY FUNCTIONS: Keylogger	4%	4%	-	3%
REMOTE DESKTOP, ACTIVE: Remote Desktop Stop	12%	-	-	3%
SERVER SHUTDOWN: Close Server	8%	-	-	2%
REMOTE DESKTOP, ACTIVE: Remote Desktop Start	8%	-	-	2%

TABLE X: Last operator actions before uninstalling malware for trials with manual interaction. Actions that appear in less than 1% of trials are omitted.

K. Operator Final Action

In Sections II-D and III-G, we noted that a number of factors would very likely affect our results; in particular, the engagement results just reported in Section IV-J. One prominent factor was our lack of realistic user data streams: webcam, audio, and user chat engagement. Another was our inability to exclude targeted operators from our sample source, operators whose targets we had no way of emulating properly.

In an attempt to determine what factors ultimately did cause the operator to leave our honeypot, we analyze the last actions performed by the operator before disconnection. These actions may be indicative of the factor(s) that dissuaded the operators from continuing the infection, or of their motivation provided they accomplished their goal prior to disconnection.

Table IX details the last actions performed by operators before session termination. Unexpectedly, a full 14% of operators actually *uninstall* the malware prior to disconnection. Thus we also provide Table X, which details the last actions, prior to uninstallation, of those operators that uninstalled the malware.

Remote Desktop. 45% of all operators were engaged in RDP prior to leaving. Recall from Section IV-I that 63% and 29% of RDP activities are data collection and system information collection, respectively. As these operators do *not* uninstall the malware before leaving, we suspect they have deemed the honeypot at least somewhat viable as a long-term victim;

however, drawing conclusions from RDP sessions is difficult given our manual inspection limitations.

System & User Information. More interesting are the operators that did not engage in remote desktop. 17% of all operators disconnect directly after attempting to access user information: passwords, keylogger, webcam, and microphone. We suspect the absense of user presence in these cases dissuades them. Another 9% disconnect after file system exploration, though only 1% actually download anything; once again, this is indicative of loss of interest due to banal file system contents. While none of these operators uninstalled the malware, (as we will discuss next), it seems probable that our honeypots did not present the stimuli they were seeking.

Malware Uninstallation. A shocking 14% of all operators uninstall the malware before disconnection. Table X elucidates this. 28% of operators that uninstall the malware do nothing else; it is their sole action. This may be an indication of targeted attacks, to which our machine is a nuisance, or of sinkhole operations that automatically uninstall infections. Other operators that uninstall follow the previous trends: 39% after RDP, 9% after file exploration, 16% after user investigation (webcam, passwords, keylogger). Apparently motivated by access to interesting user data - credentials, files, or user data streams - these operators either gathered all interesting data in a single session, or (far more likely) were unimpressed by the lack of realism in the honeypot. Our discussion of lessons learned in Section V-C features these results.

L. Operator Motives

One of our motivating questions in carrying out this work has been to determine *why* an attacker would compromise a machine and install a RAT on it and *what* an attacker might do after compromise. Although motive is impossible to infer with perfect accuracy, certain operator actions betray attacker intent. Here we consider three potential attacker motives. These motives are not mutually exclusive, and an attacker may be driven by more than one of these.

User Reconnaissance. RATs are unique among malware in that they allow the attacker to interact with a user, and we suspect that this is the primary motivation for many operators. Among the sessions we observed, 18% attempted to harass or extort the user. Furthermore, in 41% of interactive sessions, the operator attempted to access personal information about the user in the form of pictures or documents. (We exclude attempts to collect user credentials like password or cookies from this count.) Together, these two categories indicate that **at least 45% of sessions were motivated by access to a human user.** We note that 63% of sessions attempted to access a webcam or microphone. Unfortunately, our honeypots were equipped with neither, so we cannot determine whether an operator was accessing these devices because he wanted to see and hear the user or simply to gather information about the machine or confirm the identity of the victim.

Credentials. Probably the most easily monetized resources on a compromised PC are user credentials. In 50% of observed sessions, the operator attempted to access files containing credentials, and in 31% of sessions the operator installed a keystroke logger. There were attempts to steal bitcoin wallets and to grab data from an installed Steam account, and we recorded several attempts to access stolen accounts (Section IV-D). This leads us to conclude that **at least 58% of RAT operators were motived by access to user credentials.**

Vantage Point. In many cases, a RAT can serve as a valuable vantage point for an attacker to launch other attacks or spread laterally through an organization. We consider an attacker to be motivated by the vantage point of the victim if he attempts to perform any network actions beyond testing network connectivity. These include scanning the network (Section IV-G), attempting to launch network attacks (DDoS in Table VII), deploying hacking tools (Section IV-I), and perpetrating view fraud (Table VIII). In all, **16% of sessions exhibited some behavior that exploited the victim's vantage point.**

V. Discussion

DarkComet is a versatile tool, giving the operator a rich menu of actions to carry out on a remote machine. We were surprised, therefore, to find 47% of sessions involved RDP use, which reveals the presence of the operator to the victim. We expected even amateur operators to try to stay undetected on the machine in an attempt to obtain as much information as possible over time. The large number of RDP sessions and the actions we observed indicate that most operators are not trying to be stealthy. To the contrary, many actively sought to harass the user. This suggests that a substantial portion of operators are using DarkComet either for immediate amusement or with the hope of eventually extorting the user.

A. Honeypot as Tarpit Defense

The interactive nature of RATs means that attacks are limited by the number of operators available and the time spent interacting with a victim machine, so operator time may be a bottleneck for such attacks. We consider whether deploying multiple, *sufficiently realistic* honeypots could be used to draw operator attention from real targets and potentially even deter operators. We found that in our experiments, sessions with RDP activity lasted an average of about 4 minutes, whereas sessions without RDP lasted on average less than 20 seconds (Figure 6a). Across all machines during both several-week-long experiments, we accumulated just over 52.9 hours of operator engagement, despite a total uptime of about 10,080 machine-hours. This means that operators were trapped in our "tarpit" for a dismal 0.5% of its total lifetime.

It is clear that our experiment itself did not have an appreciable impact on RAT operators' capacity for wrongdoing. According to the results in Section IV-K and the general breakdown of actions in Table VII, it would appear that a major inhibitor to our honeypots' success as a tarpit is its realism. A more realistic target might attract operators for longer periods; however, creating such targets also burdens the defender. The findings in Figure 6b suggest this is true, but additional experiments are necessary to determine the effect of

realism on operator engagement, and to conduct a cost-benefit analysis of using realistic honeypots as a defensive tarpit.

B. Honeypot as Threat Intelligence Sensor

Threat intelligence, broadly speaking, is any information about threats that might be operationally useful for a security practitioner. Like a conventional honeypot aimed at extracting information from a malware sample, a RAT honeypot can be used to extract information like command and control IP addresses and samples of additional malware dropped by the RAT. The hands-on nature of RATs allows us to observe an attacker at work. This includes information about what files are searched manually and what tools an attacker installs once he/she determines that the machine is a real victim. A realistic honeypot can potentially extract information from an attacker beyond that extracted from the malware sample alone.

Though the amateur operators we monitored did not provide particularly fascinating intel, this study validates the capacity for honeypots to act as threat intelligence sensors. Our honeypots give us some insight into aggregate RAT operator demographics (with Russia and Turkey being well represented). We also obtained several malware samples not previously seen on VirusTotal, though they appeared to be updates and RATs of similar quality rather than more sophisticated malware. We witnessed the deployment of other hacking tools, and numerous indicators of the prevalence of attacks against the gaming community. Further, we observed the theft and attempted use of credentials belonging to a number of popular online services. Overall, our experiments illuminated a vibrant gamut of RAT operators engaged in hands-on exploitation of compromised machines.

C. High-Interaction Honeypots: Lessons Learned

Achieving realism in a high-interaction honeypot is difficult. Ideally, a honeypot would be completely indistinguishable from a real user's workstation; however, in practice this is exceptionally challenging to accomplish. Our experiments have shed light on a number of factors critical to future studies involving honeypots of a similar nature.

Cosmetic Appearence. Most RAT operators use RDP, whether to just inspect the desktop or to actively control the system. Given the overwhelming usage of RDP, the importance of providing a cosmetically-realistic honeypot cannot be overstated.

User Presence. This study confirms the anecdotal supposition that amateur RAT operators are often motivated by access to a live victim user, whether purely for recreational trolling or for more insidious means like blackmail, voyeurism, and sextortion. Indeed, some of the most common actions performed by our operators are accessing the webcam, recording audio with the microphone, monitoring user keystrokes with the keylogger, and even initiating chat with the victim. No future study should neglect to provide video and audio feeds to the honeypot. Implementing some means of responding to chat communication should also be considered. We suspect that the former would result in higher operator engagement times, while the latter may prompt operators to unveil their motives in the form of chat-based threats and demands.

File System Depth. File system exploration was one of the primary actions across all operators, and represents the second most time consuming action behind RDP. In Section IV-J we saw that personas with more detailed file systems occupied operators longer. Despite its importance, providing a realistic file system is one of the most unscalable challenges in creating realistic honeypots [45], and has been since the very inception of the live operator honeypot [47].

Credentials. 10% of operators began by searching for stored passwords, and 43% did so overall. Given that credential theft is the goal of so many operators, providing credentials for them to steal provides multiple benefits. The visceral success of password theft adds realism to the honeypot and may keep operators engrossed. But more interestingly, this allows the seeding of "honey-credentials" to gather more information about attackers. Our trial with honey-credential seeding yielded numerous recorded access attempts.

VI. Conclusion

In this paper, we presented the results of our study focusing on understanding the actions of DarkComet operators. We developed a technique to scan for DarkComet operators active on the Internet. We combine these scans with a collection of DarkComet instances found in the wild that we ran in a realistic environment simulating a real victim machine. From this, we were able to determine how attackers interact with a victim, including time spent on the machine, user data collected, and so on. We found that the most common uses of DarkComet are as a means of accessing a *human* victim for surveillance, harassment, or extortion; stealing user data and account credentials; and abusing the victim machine's vantage point to deploy hacking tools and other malware, probe lateral machines, and launch attacks.

We find that honeypots are a promising tool to monitor the manual actions of DarkComet operators, which enables us to understand the motivations and techniques of these operators. In addition, we demonstrate that honeypot environments show promise as potential tarpit defenses. It is our hope that this initial exploration of the manual attacker ecosystem will spur further investigation.

Acknowledgments

This work was supported by the National Science Foundation through grants NSF-1237264 and NSF-1619620, by the Max Planck Society and the European Research Council under the imPACT Synergy Grant No. 610150, by the AXA Research Fund, and by a gift from Google. We would also like to thank the following: VirusTotal, for providing us with the Intelligence account from which we sourced our malware; John Matherly of Shodan, for providing us with daily global controller scans; and Dr. Andreas Haeberlen, our shepherd, for his invaluable assistance guiding this paper to its final form.

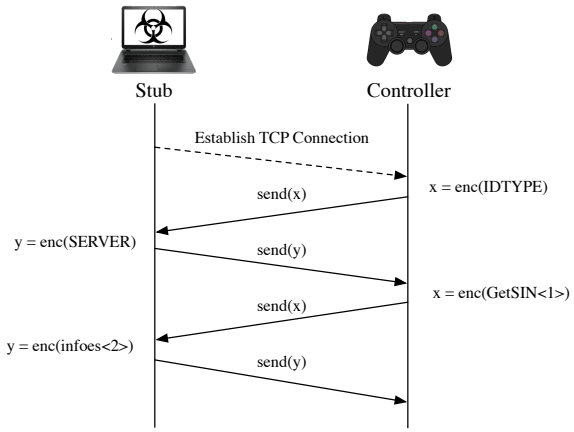


Fig. 7: A diagram of the DarkComet handshake. ⟨1⟩ consists of victim IP address and an integer nonce based on the system clock. ⟨2⟩ consists of the following: campaign ID, victim local IP, victim global IP, infection port, hostname, username, the same integer nonce, uptime, operating system, location and language, hardware ID, RAM usage, current time, version, and active window. The function $enc()$ RC4-encrypts (and then base64-encodes) its argument message using a key consisting of a static string based on DarkComet version concatenated to a user-chosen password.

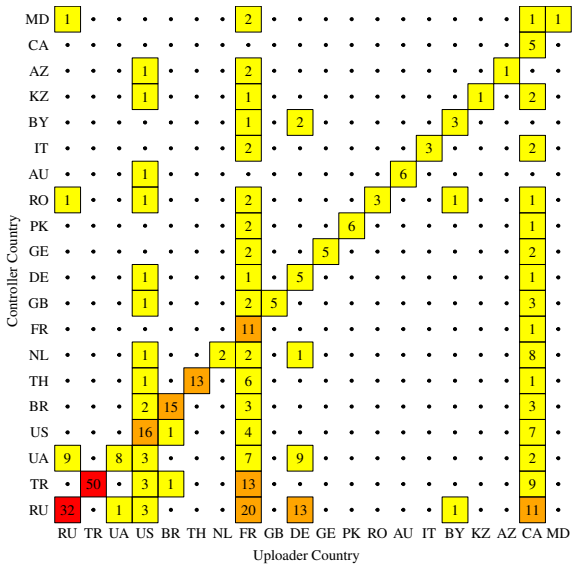


Fig. 8: Correlation between the geolocated countries of the VirusTotal uploaders and those of the controllers accessing our honeypots. Countries are sorted by decreasing number of controllers.

APPENDIX

A. DarkComet Protocol Handshake

Figure 7 describes the DarkComet protocol handshake in detail. Note that this is a custom protocol over TCP; many RATs eschew HTTP and other common protocols for custom command and control protocols. The stub establishes the TCP connection, as is customary with RAT infections, but then listens for the controller to identify itself - a behavior we classify as "passive." Many RATs exhibit "active" protocols wherein the stub will both establish the connection *and* send the first identifying packet.

The handshake itself is simple enough; the controller identifies itself, after which the stub does the same. The controller then asks for information, to which the stub replies with identifying information about itself (e.g. version number, campaign ID) and the victim machine (e.g. username, hostname).

Note that all of this communication is RC4-encrypted with a pre-shared key. The stub will not respond to a DarkComet controller's first message if it is not encrypted with the correct key; thus, by choosing a unique password, operators are able to protect their stubs from being controlled by another operator or sinkhole, as well as prevent fake stubs from flooding their control panels with bogus greetings.

Our analysis of the DarkComet protocol draws heavily on work by Denbow and Hertz [14].

B. VirusTotal Sample Geolocation Matrix

VirusTotal enables us to retrieve the geolocation of the IP addresses used to upload samples. In Figure 8, we show the correlation between the countries of sample uploaders and the countries of controllers that connected to our honeypots. For clarity, we discard the DarkComet samples uploaded or controlled from multiple countries, comprising 5 and 260 samples, respectively. We see in Figure 8 that the two most popular countries for controllers are Russia and Turkey with a clear diagonal indicating that DarkComet samples tend to be uploaded and controlled from the same countries. For example, 34% and 52% of all samples uploaded from Russia and Turkey, respectively, were controlled from the same country. The vertical lines for the US, France, and Canada are indicative of users uploading DarkComet samples in bulk. As these samples were likely acquired from users residing in different countries than these uploaders, the correlation between uploader and controller countries is weaker in those cases.

C. Sample Age

Knowning the relative age of the samples we executed, meaning the difference in time between when they were received by VirusTotal and when we executed them, is important to interpret the measurements in this study. Figure 9 shows the ages of all samples which resulted in manual interaction with our honeypots.

D. Sample Execution Schedule

Figure 10 shows the total number of trials that were run by hour of the day and day of the week, respectively. Note that we only ran samples whose controllers were determined to be online; because of this, Figure 10a matches our scanning data in Section IV-A quite nicely. Figure 10b bucks this trend, instead likely demonstrating the rate at which new samples arrive at, or are made available by, VirusTotal.

E. Dynamic Language Analysis

We attempted to use Google's language detection API on our operators' communications (and other metadata), the results of which are shown in Table XI. Though English is

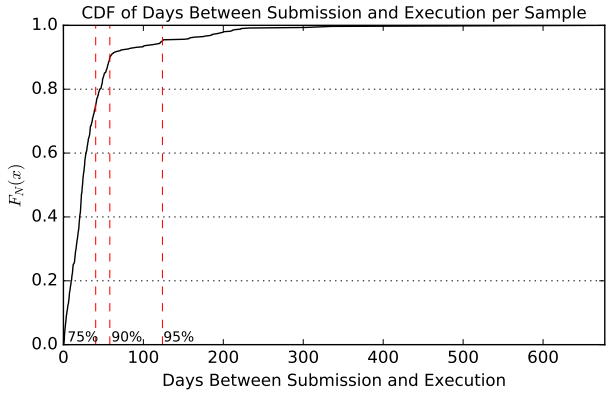
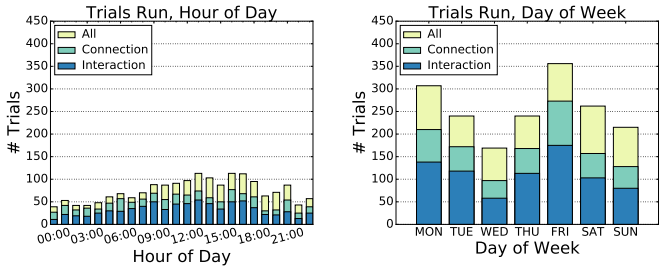


Fig. 9: A CDF showing the relative age of our samples, where age is the number of days between submission to VirusTotal and execution in our sandbox. We report on 596 unique samples for which (1) we have submission information, and (2) resulted in manual operator interaction. 75% are executed within 40 days of submission, 90% in 58 days, and 95% in 124 days. The oldest sample that resulted in manual interaction was an astonishing 677 days old; the newest, 12 hours.



(a) Number of samples run on the particular hour of the day.

(b) Number of samples run on the particular day of the week.

Fig. 10: Malware sample submissions by day and by hour. Colors from colorbrewer2.org [12].

understandably the top language, the presence of languages like Igbo indicates the shortcomings of applying language detection to ethereal text like IM chat messages. For reference, Igbo is a language spoken in Nigeria, with which the language detection API appears to frequently confuse Turkish slang.

	Chat		*RDP Keystrokes*		*Dropped Files*	
Source	*Cnt*	*Pct*	*Cnt*	*Pct*	*Cnt*	*Pct*
English	265	42%	27	21%	19	13%
Igbo	22	3%	0	-	0	-
Hawaiian	18	2%	0	-	0	-
Turkish	15	2%	5	3%	0	-
Corsican	15	2%	1	-	0	-
Other	31	5%	19	15%	2	1%
Undetermined	252	40%	74	58%	123	85%
Total	618		126		144	

TABLE XI: Languages of metadata obtained during live trials, including chat messages, remote desktop keystrokes, and filenames of dropped files. *Other* is any other spoken or written language.

REFERENCES

[1] V. M. Alvarez. YARA: The pattern matching swiss knife for malware researchers (and everyone else). http://virustotal.github.io/yara/.

[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] N. Anderson. Digital voyeur spied on women's webcams 5-12 hours a day. http://arstechnica.com/tech-policy/2015/10/digital-voyeur-spied-on-womens-webcams-5-12-hours-a-day/, October 2015.

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

[6] C. Baraniuk. Webcam hacker spied on sex acts with blackshades malware. http://www.bbc.com/news/technology-34475151, October 2015.

[7] U. Bayer, C. Kruegel, and E. Kirda. TTAnalyze: A tool for analyzing malware. In *European Institute for Computer Antivirus Research (EICAR)*, 2006.

[8] K. Breen. Rat decoders. https://github.com/kevthehermit/RATDecoders, April 2014.

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

[10] Citizen Lab. Packrat: Seven Years of a South American Threat Actor. https://citizenlab.org/2015/12/packrat-report, Dec. 2015.

[11] N. Correia and A. Chevalier. zer0m0n driver for cuckoo sandbox. https://github.com/conix-security/zer0m0n/blob/master/bin/update_cuckoo.sh, 2015.

[12] Cynthia A. Brewer, Geography, Pennsylvania State University. Colorbrewer2. http://colorbrewer2.org/.

[13] DarkComet YARA rules. https://github.com/Yara-Rules/rules/blob/master/malware/RAT_DarkComet.yar.

[14] S. Denbow and J. Hertz. Pest control: taming the rats. Technical report, 2012.

[15] Department of Justice, U.S. Attorney's Office, Central District of California. Fbi arrests glendale man in 'sextortion' case. https://www.justice.gov/usao-cdca/pr/fbi-arrests-glendale-man-sextortion-case, January 2013.

[16] 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.

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

[18] Z. Durumeric, E. Wustrow, and J. A. Halderman. Zmap: The internet scanner. https://zmap.io/.

[19] R. Falcone and S. Conant. Projectm: Link found between pakistani actor and operation transparent tribe. http://researchcenter.paloaltonetworks.com/2016/03/unit42-projectm-link-found-between-pakistani-actor-and-operation-transparent-tribe/, 2016.

[20] C. Farivar. Sextortionist who hacked Miss Teen USA's computer sentenced to 18 months. http://arstechnica.com/tech-policy/2014/03/sextortionist-who-hacked-miss-teen-usas-computer-sentenced-to-18-months, Mar. 2014.

[21] Fidelis 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.

[22] D. Goodin. Teamviewer users are being hacked in bulk, and we still don't know how. https://arstechnica.com/security/2016/06/teamviewer-users-are-being-hacked-in-bulk-and-we-still-dont-know-how/, June 2016.

[23] Google. Google translation api. https://cloud.google.com/translate/v2/detecting-language-with-rest.

[24] Google. Virustotal intelligence. https://www.virustotal.com/intelligence.

[25] C. Guarnieri, A. Tanasi, J. Bremer, and M. Schloesser. Cuckoo sandbox. https://cuckoosandbox.org/, 2010.

[26] Hack Forums. https://hackforums.net/.

[27] S. Hardy, M. Crete-Nishihata, K. Kleemola, A. Senft, B. Sonne, G. Wiseman, and P. Gill. Targeted threat index: Characterizing and quantifying

politically-motivated targeted malware. In *USENIX Security Symposium*. USENIX, August 2014.

[28] T. Haruyama and H. Suzuki. I know you want me - unplugging plugx. https://www.blackhat.com/docs/asia-14/materials/Haruyama/Asia-14-Haruyama-I-Know-You-Want-Me-Unplugging-PlugX.pdf, March 2014.

[29] C. Kanich, C. Kreibich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage. Spamalytics: An empirical analysis of spam marketing conversion. In *ACM Conference on Computer and Communications Security (CCS)*. ACM, 2008.

[30] C. Kreibich, N. Weaver, C. Kanich, W. Cui, and V. Paxson. Gq: Practical containment for measuring modern malware systems. In *ACM SIGCOMM Conference on Internet Measurement Conference (IMC)*. ACM, 2011.

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

[32] 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.

[33] 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.

[34] 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.

[35] M. Marquis-Boire and E. Galperin. A brief history of governments hacking human rights organizations. https://www.amnesty.org/en/latest/campaigns/2016/01/brief-history-of-government-hacking-human-rights-organizations, Jan. 2016.

[36] J. Matherly. Shodan - the search engine for the internet of things. https://www.shodan.io/.

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

[38] MaxMind. Geolite legacy downloadable databases. https://dev.maxmind.com/geoip/legacy/geolite/, 2012.

[39] MPRESS. https://autohotkey.com/mpress/mpress_web.htm, 1997.

[40] I. Muller, A. Serper, and A. Frazer. Dissecting the hacking team's operation methods: What security professionals need to know. http://go.cybereason.com/rs/996-YZT-709/images/Cybereason-Labs-Reasearch-Analysis-Hacking-Team.pdf, 2015.

[41] M. Oberhumer, L. Molnar, and J. Reiser. Upx: Ultimate packer for executables. http://upx.sourceforge.net/, 1996.

[42] N. Provos. A virtual honeypot framework. In *USENIX Security Symposium*. USENIX, 2004.

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

[44] D. Reguera Garcia. Anti cuckoo. https://github.com/David-Reguera-Garcia-Dreg/anticuckoo, 2015.

[45] N. C. Rowe. Measuring the effectiveness of honeypot counter-counterdeception. In *Hawaii International Conference on System Sciences (HICSS)*, volume 6. IEEE, 2006.

[46] SSLBL: SSL blacklist. https://sslbl.abuse.ch/.

[47] C. Stoll. *The cuckoo's egg: tracking a spy through the maze of computer espionage*. Simon and Schuster, 2005.

[48] N. Villeneuve and M. Scott. Crimeware or apt malware: Fifty shades of grey, 2014.

[49] VMware. vsphere platform. https://www.vmware.com/products/vsphere.

[50] M. Vrable, J. Ma, J. Chen, D. Moore, E. Vandekieft, A. Snoeren, G. Voelker, and S. Savage. Scalability, Fidelity and Containment in the Potemkin Virtual Honeyfarm. In *ACM Symposium on Operating System Principles (SOSP)*, 2005.

[51] WikiLeaks. SpyFiles 4. https://wikileaks.org/spyfiles4, Sept. 2014.

[52] C. Willems, T. Holz, and F. Freiling. Toward automated dynamic malware analysis using cwsandbox. *IEEE Security and Privacy (S&P)*, Mar. 2007.

[53] 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.