Schrödinger’s RAT: Profiling the Stakeholders in the Remote Access Trojan Ecosystem

Mohammad Rezaeirad, George Mason University; Brown Farinholt, University of California, San Diego; Hitesh Dharmdasani, Informant Networks; Paul Pearce, University of California, Berkeley; Kirill Levchenko, University of California, San Diego; Damon McCoy, New York University

https://www.usenix.org/conference/usenixsecurity18/presentation/rezaeirad

This paper is included in the Proceedings of the 27th USENIX Security Symposium.
August 15–17, 2018 • Baltimore, MD, USA

Open access to the Proceedings of the 27th USENIX Security Symposium is sponsored by USENIX.
Schrödinger’s RAT: Profiling the Stakeholders in the Remote Access Trojan Ecosystem

Mohammad Rezaeirad,† Brown Farinholt,⋄ Hitesh Dharmdasani,‡ Paul Pearce†† Kirill Levchenko,⋄ Damon McCoy⋆
†GMU, ⋄ UC San Diego, ‡Informant Networks, †† UC Berkeley, ⋆NYU

Abstract
Remote Access Trojans (RATs) are a class of malware that give an attacker direct, interactive access to a victim’s personal computer, allowing the attacker to steal private data from the computer, spy on the victim in real-time using the camera and microphone, and interact directly with the victim via a dialog box. RATs are used for surveillance, information theft, and extortion of victims.

In this work, we report on the attackers and victims for two popular RATs, njRAT and DarkComet. Using the malware repository VirusTotal, we find all instances of these RATs and identify the domain names of their controllers. We then register those domains that have expired and direct them to our measurement infrastructure, allowing us to determine the victims of these campaigns. We investigate several techniques for excluding network scanners and sandbox executions of malware samples in order to filter apparent infections that are not real victims of the campaign. Our results show that over 99% of the 828,137 IP addresses that connected to our sinkhole are likely not real victims. We report on the number of victims, how long RAT campaigns remain active, and the geographic relationship between victims and attackers.

1 Introduction
Remote Access Trojans (RATs) are an emerging class of manually operated malware designed to give human operators direct interactive access to a victim’s computer. Unlike automated malware (i.e., spam and DDoS), RATs are predicated on the unique value of each infection, allowing an attacker to extort a human victim or otherwise benefit from access to a victim’s private data. RATs are sold and traded in underground communities as tools for voyeurism and blackmailing [11, 18]. RATs have also been reported to be used for state-sponsored espionage and surveillance, and have been used to spy on journalists [46], dissidents [30], and corporations [27].

While the unique danger posed by this new class of malware has received considerable attention, the relationship between the RAT operator and victim is poorly understood. In this work, we bring to light the behavior of operators and victims of two popular RAT families, njRAT and DarkComet. Our primary aim is to determine who is attacking whom, the size of the victim and attacker population, and how long victims remain vulnerable after a campaign ends.

Figure 1: Intelligence pollution obfuscates the stakeholders in the RAT Ecosystem.

One of the pervasive challenges of studying populations of victims and attackers is the difficulty of accurately determining the population. Network hosts behaving as victims may be security researchers scanning for command-and-control servers [15, 17], while potential command-and-control servers may be vigilantes operating sinkholes [12, 39, 49]. The central challenge of conducting a study of the RAT ecosystem, then, is to be able to correctly determine who is really a victim or operator, and who is pretending to be one.

In this paper, we design, implement, and deploy improved methodologies for accurately measuring real victims that connect to our sinkhole, RAT-Hole, and identifying RAT controllers using our scanner, RAT-Scan. The task of identifying victims at scale is made difficult by the amount of pollution sinkholes receive from increasingly high-fidelity scanners and sandboxes. Differential-
ing between real controllers and sinkholes is also a non-
trivial undertaking due to higher fidelity sinkholes. This
increasing fidelity in RAT scanners that emulate more
of a victim’s behavior and sinkholes that emulate more
of a real RAT controller’s protocol has likely created an
arms-race between entangled threat intelligence opera-
tions which we call Intelligence Pollution. This leads to
inaccurate measurements and wasted notification efforts,
wherein researchers and security vendors may confuse
beneficent sinkholes for malicious controllers, or scan-
ners and sandboxes for actual victims.

In order to mitigate this pollution, we have created
RAT-Hole which implements the handshake protocol
and error triggering tests for two common RATs, Dark-
Comet and njRAT. Based on extensive testing, we de-
veloped a set of heuristics to accurately differentiate
sandboxes, scanners, and victims. We leverage the pheno-
momenon that operators of commodity-grade RATs fre-
quently configure their RATs to use free dynamic DNS
(DDNS) services [17] domains which expire after one
month. We conduct an experiment where we sinkholed
6,897 RAT controller domains associated with Dark-
Comet and njRAT samples.

Based on our classification methodology we found
that only 6,710 (0.8%) of the over 800k Source IP ad-
dresses that connected to our RAT-Hole were likely vic-
tims. If we filter out the lower fidelity scanners, we find
that only 3,231 (69%) of the RAT fingerprints that com-
pleted a RAT handshake are likely victims. Our analy-
sis also found that telemetry from a /32 and /24 inter-
et telescope could filter less than 1% of the higher fi-
delity scanners and sandboxes. We also received several
repeated manual notifications based on scanners misclas-
sifying our RAT-Hole deployment as a large-scale RAT
controller hosting operation.

As part of our study we also created RAT-Scan, which
is able to differentiate some sinkholes, including our high
fidelity RAT-Hole, from actual RAT controllers by emu-
lating DarkComet and njRAT victims. We deployed
RAT-Scan to scan the entire IPv4 address space and
found 6,401 IP addresses hosting suspected RAT con-
trollers. Our efforts to accurately differentiate sinkholes
from real RAT controllers were complicated by VPN
proxy services that were highly dynamic and appear to
host both sinkholes and real RAT controllers. In particu-
lar, we found that IPJetable, a free VPN service, hosted
over 40% of the suspected RAT controllers we found.

We propose that our more accurate identification of
controllers and victims could reduce wasted notification
effort. In addition, we propose some potential interven-
tions involving the free DDNS and VPN proxy services
that provide support infrastructure for RAT operators.
This could be in the form of these services voluntarily
assisting in blocking RATs from their infrastructure. The
other potential intervention is for law enforcement enti-
ties to more closely monitor these services to better iden-
tify attackers and victims.

The primary contributions of our study include:
- Proposal and evaluation of methods for classifying
  RAT sandboxes, scanners, and likely victims based
  on connection to a sinkhole.
- Conducted a large-scale measurement study based
  on sinkholing 6,897 RAT controller domains. We
  found that only 6,710 (0.81%) of the over 800k
  Source IP addresses that connected to our sinkhole
  were likely victims.
- Proposed and deployed a RAT controller scanner that
  is able to differentiate some sinkholes from real RAT
  controllers. Based on our analysis we found that IP-
  Jetable, a free VPN service, hosted over 40% of the
  suspected RAT controllers that we found.
- Identified potentially improved interventions that
  could mitigate the threat of RATs.

2 Background

The subject of this work is the relationship between vic-
tims and operators of two commodity RATs (Remote Ac-
cess Trojans), njRAT and DarkComet. In this section, we
provide the necessary background on RATs for the rest
of the paper.

2.1 RAT Components

Most RATs are made up of three parts: builder, stub,
and controller. At the start of a malware campaign, the
attacker runs the builder program, creating a new in-
stance of the stub for installation on a victim’s com-
puter. The newly built stub contains the code that will
run on the victim’s computer with parameters such as the
host name of the command-and-control server to con-
tact upon infection. During the campaign, the attacker
runs the controller software on the command-and-control
server to interact with the victims. In most cases (e.g., for
njRAT and DarkComet), the controller provides a graphi-
cal user interface and runs directly on the attacker’s com-
puter. The attacker, also called the RAT operator, inter-
acts with the victim via the controller interface.

2.2 RAT Command and Control Protocol

For the RATs studied in this paper, communication be-
tween stub and controller begins with the stub opening a
TCP connection to the controller host name hard-coded
in the stub. The attacker provides this host name to the
builder program which produces the stub. Once the stub
establishes this connection, RATs can be divided into
two groups. In RATs where the application-layer hand-
shake is controller-initiated, the controller speaks first
by sending a banner to the stub immediately after ac-
cepting the stub’s connection. DarkComet is controller-
initiated. In contrast, in a protocol where the handshak
is **victim-initiated**, the stub sends the first message immediately upon connecting (receiving the SYN-ACK from the controller). njRAT is victim-initiated.

Whether a protocol handshake is victim-initiated or controller-initiated determines how we scan for controllers and sinkhole stubs, as described below. Additionally, many RAT protocols support symmetric encryption to obfuscate the command stream and as a form of access control to the stub. In these cases, the encryption key or password is embedded in the stub’s configuration.

The initial message sent by a stub contains both information configured by the builder (e.g., password, campaign ID) as well as information unique to the victim machine (e.g., username, hostname, operating system, active window). This information allows the operator to manage multiple campaigns and also to obtain a summary of the victim. Some of the information sent by the stub is potentially Personally Identifiable Information (PII), which introduces ethical challenges to researching RAT sinkholes that we discuss in our ethical framework.

### 2.3 Sinkholing

**Sinkholing** is a term used to indicate the redirection of infected machines’ connections from their intended destinations (e.g., attackers’ command & control servers [36]) to the sinkhole owner. Local sinkholing efforts, implemented by organizations or individual ISPs, often involve reconfiguring DNS servers and routers to block communication with malicious domains or IP addresses. Larger, coordinated sinkhole operations are often part of broader takedown efforts, requiring cooperation between domain registrars and international authorities. [29, 49]

A prior study found that RAT operators often utilize Dynamic DNS (DDNS) services [17], which allow their controllers to migrate between IP addresses without disruption of operation. Services like No-IP [35] offer free DDNS hostname registrations that expire after 30 days. As we will show, operators often allow their hostnames to expire and this provides a large pool of RAT domain names that can be claimed and sinkholed.

Ideally, a DNS sinkhole operation would be able to identify all victims associated with its acquired domains and to accurately measure victims. Unfortunately, scanners and sandboxes introduce a significant amount of intelligence pollution, as we will show in our study.

### 2.4 Scanning

Internet-wide scanning is a popular technique for Internet measurement, particularly in the field of security. It was recently leveraged to measure the Mirai botnet [3], and is likely used by many academic groups and security vendors. Open-source tools such as ZMap [14] make rapid scanning of IPv4 space accessible to researchers. There are also services such as Censys [13] based on ZMap and Shodan [32] that uses a custom scanner.

Scanning for RAT controllers presents a similar set of challenges to sinkholing. RATs often use victim-initiated handshake protocols to communicate, so simple port scanning or banner grabbing is often not sufficient to confirm the existence of a RAT controller. One must also implement the RAT’s handshake, which can be complicated by the inclusion of encryption and custom passwords. Proxies may also conceal multiple controllers behind the same address, while a single controller may reside behind ever-changing addresses (using DDNS, for example). Finally, many academic groups and security vendors operate sinkholes which can be challenging to differentiate from real RAT controllers.

### 2.5 Ethical Framework

Our methodology was approved by our institution’s Institutional Review Boards (IRB) and general legal counsel. The ethical framework that we operated under is that we only completed the protocol handshake with peers that contacted us and controllers that are publicly reachable. We did not attempt to execute any commands on infected peers. During the handshake there is some potentially Personal Identifiable Information (PII) that the peer sends to us, such as the PC name (often the name of the victim) or full website URLs a person is visiting if the active window is a browser. In order to mitigate the potential harm caused by our study, we immediately encrypted any fields that might contain PII and did not ever store an unencrypted version of these fields (PII listed at Table 3). Our IRB takes the position that IP addresses are not personally identifiable. In no cases did we attempt to tie our measurements to an actual identity.

### 3 RAT-Hole Methodology and Dataset

Our system consists of two primary components: a high-fidelity sinkhole (**RAT-Hole**) that imitates RAT controllers, and a high-fidelity scanner (**RAT-Scan**) that imitates RAT victims. We present the details of our RAT-Scan system in Section 5. Figure 3 shows a timeline of when each part of our methodology was deployed.

![Figure 2: The major components of our operation and their interactions with the subjects of our study.](image)

Figure 2 depicts the system’s operation. 1 An attacker registers a hostname with a DDNS provider like No-IP, creates malware binaries configured with this hostname, and spreads the binaries to victims in the wild. 2 Some of
the malware is detected and uploaded to VirusTotal [52].

Our VirusTotal-deployed YARA rules [54] find these malware samples, from which we extract configurations (including controller IP addresses, hostnames, and passwords). Many hostnames belong to No-IP. Our DDNS Claimer registers any expired, No-IP-owned hostnames and configures them to resolve to RAT-Hole’s IP range. RAT-Hole then receives all connection attempts to these hostnames. Simultaneously, RAT-Scan continuously probes all extracted IP addresses and hostnames for controller activity in the wild.

We limit the scope of our study to two RAT families, DarkComet and njRAT, since reverse-engineering and implementing parts of their protocols in RAT-Hole and RAT-Scan is a time-consuming manual effort. These two were chosen because they are the most popular RAT families for which we were able to obtain source code for multiple versions, and there is existing documentation of their protocols to assist with the reverse-engineering process [6, 10, 20]. As an approximate measure of prevalence, we count the number of unique binaries associated with RAT families uploaded to VirusTotal by using up-to-date YARA rules for sample hunting. We found that njRAT and DarkComet were the third and fifth most popular RATs on VirusTotal when we began our study.

### 3.2 Domain Claiming

**Domain dataset.** We performed an analysis of the domain names found in the RAT configurations. Table 2 shows that most of the domains we found that were used by DarkComet and njRAT are associated with free Dynamic DNS (DDNS) providers, and that No-IP managed 60% of all discovered domains (77% of DDNS).

<table>
<thead>
<tr>
<th>Controller Type</th>
<th># Domain</th>
<th>% Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>No-IP</td>
<td>8,564</td>
<td>60.0</td>
</tr>
<tr>
<td>DuckDNS</td>
<td>2,459</td>
<td>17.2</td>
</tr>
<tr>
<td>FreeDNS</td>
<td>92</td>
<td>&lt;0.1</td>
</tr>
<tr>
<td>DynDNS</td>
<td>38</td>
<td>&lt;0.1</td>
</tr>
<tr>
<td>Total Dynamic DNS</td>
<td>11,153</td>
<td>78.1</td>
</tr>
<tr>
<td>Unknown</td>
<td>3,120</td>
<td>21.9</td>
</tr>
<tr>
<td>Total</td>
<td>14,273</td>
<td>100.0</td>
</tr>
</tbody>
</table>

Table 2: Breakdown of C&C domains in our RAT sample population by Dynamic DNS provider. Unknown encompasses all domains unrelated to a known DDNS provider.

**DDNS Claimer.** We developed a web automation toolkit to automate the process of identifying expired DarkComet and njRAT domains controlled by No-IP and claiming them. We only claimed domains from No-IP, since there is manual effort involved in building the web automation toolkit and No-IP was by far the most popular free DDNS provider used by DarkComet and njRAT. We limit our study to only expired domains that we can claim freely; we do not attempt to seize owned domains, in order to avoid additional disruption to the ecosystem that we were measuring. As future work we plan to expand our claiming infrastructure to other DDNS providers and actual registered domains to understand if this affects the results of our analysis.

Our DDNS domain claiming operation, which started on 2017-07-15 and ended on 2017-11-17, claimed 6,897, or 81%, of the 8,564 DDNS domains managed by No-IP.

### 3.3 RAT-Hole Operation

**Architecture.** RAT-Hole is a socket server application that utilizes epoll in order to handle a large number of connections, simultaneously.
A RAT-Hole node has three sets of interfaces. (1) Management: all the management traffic, such as data backup and database iterations, are performed using this interface. (2) Sinkholing: This set of virtual interfaces (206 IP addresses) are assigned to the sinkholed domains by our domain claiming system. We randomly claimed 200 expired DarkComet and njRAT from the set of No-IP domains that we discovered. These domains were monitored for one hour. After that one hour we released these domains, waited for 5 minutes for the old domain names to expire from DNS caches, and then claimed a new set of 200 randomly selected domains. (3) Decoy: We intentionally did not use 11 of the IP addresses in our subset as sink-holing interfaces so that we could identify connection attempts to them that indicate scanning behavior. We randomly selected IP addresses to be decoy interfaces.

RAT-Hole includes a Deep Packet Inspection (DPI) pattern-matching based state machine that maintains the state of each active TCP connection from the sink-hole and decoy interfaces. These states are:

**Incoming.** RAT-Hole allocates a data structure indexed by connection ID for any TCP connection that completes a TCP three-way handshake with the RAT-Hole. Here, connection ID is a tuple of source IP address, destination IP address and TCP port. This ID is used as an index by RAT-Hole to locate the previous states of the connection and to update the connection’s state.

**Protocol detection.** Once a TCP connection is established RAT-Hole waits three seconds to receive data. After three seconds it will probe the peer by sending a DarkComet C&C banner to determine if it is an njRAT stub and update the state of the connection.

**Victim-initiated RAT protocol detection.** When RAT-Hole receives an initial message from a completed TCP connection before the three second timer expires, it will examine all of our regular expression-based RAT family detection signatures over the payload to identify the incoming peer’s protocol. We have created 16 signatures that are able to detect the initial message sent by common versions of njRAT, Xtream-RAT, ShadowTech, NetWire, H-Worm, LuminosityLink, BlackWorm, and KJWorm. However, we implement the complete njRAT handshake to determine if it is an actual njRAT stub or intelligence gathering operation. Although we did not claim any domains used by these variants, we did see initial messages for these other RAT families.

**Controller-initiated RAT protocol detection.** If the three second timer expires, then RAT-Hole will probe the peer by sending a DarkComet banner. For DarkComet (refer to Appendix D for more details on DarkComet protocol behavior), we send a series of 125 banners. After each banner is sent the connection is terminated and the stub will attempt to connect again if the stub conforms to the DarkComet protocol. This ordered set of banners is comprised of: (1) a randomly generated banner (this is to test if the stub deviates from the protocol by accepting any banner), (2) a set of 20 default DarkComet banners (this is to see if the stub will incorrectly accept the default banner), (3) a complete set of valid banners for the current set of domain(s) being sinkholed by that interface (this is to see if the stub responds correctly to a valid banner), and (4) the remainder of the 125 banners are randomly generated (based on our reverse engineering we found that after 124 banners are received by a stub it will lock up and stop attempting further connections until the RAT process is restarted).

**Handshake test.** We implemented a handshake test for both DarkComet and njRAT which implements each full RAT handshake protocol. In addition, it includes specifically malformed messages in order to trigger error handling and identify likely RAT stub execution as opposed to a scanner that has reimplemented the protocol. This methodology is similar to that of Marczak et al. to fingerprint malware C&C servers [30].

We send a malformed command at the end of the njRAT handshake (see Appendix C) and if the expected error handling response is received, we mark the connecting peer as conforming to the standard implementation of the protocol. In the same vein, for DarkComet (see Appendix D) we observe and enumerate the execution pattern. For instance, we expect a true DarkComet infection to stop operating after connecting to RAT-Hole as a result of 124 attempts or more. Note that our handshake test does not distinguish between stub execution in sandbox and victim machine. However, it does perform well at identifying RAT scanners specifically those that do not implement the complete RAT protocol handshake and true execution behavior of the RAT protocol as standardly implemented.

After handshake probing is completed, RAT-Hole closes the connection and removes the state for that connection ID. RAT-Hole logs the final state of the connection, which can be one of three states: (1) no valid banner was received; (2) a valid initial banner was received, but the handshake was not completed; (3) the handshake was completed successfully.

**Handshake metadata.** During the RAT handshake, the stub includes a number of fields in the handshake that we parse and store in the database. A list of the fields that we parse and store is shown in Table 3. Using AES we encrypt any fields that might contain potential PII before storing them in our database. From these fields, we create a fingerprint that is largely unique for each infection by combining the PC-NAME, USERNAME, and HWID. VSN and UUID comprise the HWID for njRAT and Dark-
Comet, respectively. (See Appendix C, D.) This fingerprint enables us to persistently identify and thus enumerate unique RAT peers despite victim IP address churn.

**RAT-Hole classification.** Based on the data, we are accurately able to classify peers that connected to RAT-Hole into one of five categories:

- **Unknown:** These peers send a payload that is not known to RAT-Hole. Internet wide scanners (e.g., zmap) and other custom IP intelligence operations are examples of these peers. We have also found that these unknown payloads could be a RAT family that RAT-Hole does not support, since some of the sinkholed domains are used as the C&C for multiple RAT infection campaigns.
- **Low Fidelity (LF) Scanner:** These peers do not complete the RAT handshake. Based on our analysis they often send many fingerprints, connect from many Src-IP addresses, connect to our decoy interfaces, and their Src-IP address might be included in known scanner list(s).
- **High Fidelity (HF) Scanner:** These peers complete the RAT handshake and present one or more highly anomalous characteristics described in Table 4.
- **Sandbox Execution:** These peers complete the RAT handshake, but exhibit one or more of the characteristics commonly associated with a sandbox execution, such as a short execution duration which we defined as slightly more than the longest default execution duration value (600 seconds) of the major sandboxes we analyzed. Table 4 includes a complete list of these heuristics.
- **Victim:** These are likely real infections.

### 3.4 Operation Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>PII</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVE_WINDOW</td>
<td>Title and content of currently open window</td>
<td>✓</td>
</tr>
<tr>
<td>CAMPAIGN_ID</td>
<td>Stub’s identity which operator defines</td>
<td>✓</td>
</tr>
<tr>
<td>COUNTRY</td>
<td>Geo-Location of victim’s machine</td>
<td>✓</td>
</tr>
<tr>
<td>HWID</td>
<td>Hardware identity of victim’s machine</td>
<td>✓</td>
</tr>
<tr>
<td>INSTALL_DATE</td>
<td>First day on which stub was executed</td>
<td></td>
</tr>
<tr>
<td>LANG_IPDC</td>
<td>Private IP address of victim’s machine</td>
<td></td>
</tr>
<tr>
<td>LANGUAGEDC</td>
<td>Language setting of victim’s machine</td>
<td></td>
</tr>
<tr>
<td>OS</td>
<td>Operating system name of victim’s machine</td>
<td></td>
</tr>
<tr>
<td>PC_NAME</td>
<td>PC name of victim’s machine</td>
<td>✓</td>
</tr>
<tr>
<td>USERNAME</td>
<td>Username of victim’s machine</td>
<td>✓</td>
</tr>
<tr>
<td>PORTDC</td>
<td>Port number of stub</td>
<td></td>
</tr>
<tr>
<td>VERSION</td>
<td>Version of RAT</td>
<td></td>
</tr>
<tr>
<td>URL_IPDC</td>
<td>Public IP address of victim’s machine</td>
<td></td>
</tr>
<tr>
<td>WEBCAM_FLAG</td>
<td>Webcam capture is supported</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Fields extracted from handshakes for DarkComet and njRAT families. PII indicates whether we consider the field to be potential PII of the victim, and determines whether we AES encrypt the value. (DC) identifies DarkComet specific fields.

Over the 31 days that RAT-Hole was deployed, it was in possession of domains for 23.1 total days - an average of 17.7 hours per domain, distributed randomly. During this time, it received 153,100,000 TCP connections. Table 5 provides a high-level view of these connections, broken down by determined peer type.

We performed an analysis of the “Unknown” peer type from Table 5 which composed 815,455 (98.5%) of all IP addresses that completed a three-way TCP handshake connection to RAT-Hole, but were not classified by RAT-Hole as peers related to either njRAT or DarkComet. We suspected that some of these connections might be other RAT families when an operator reuses the same DDNS domain for other RAT campaigns. In order to provide some measurements of this phenomenon, RAT-Hole implements a simple payload parser for the first message of the handshake for 19 other popular RAT families in addition to the complete RAT handshake protocol for njRAT and DarkComet RAT families. Of these peers, 73.6% sent no additional TCP messages after the handshake, 31.9% sent unknown payloads, and 1,463 (<1%) were detected as other types of RAT families. The small degree of overlap indicates that some IP addresses presented multiple behaviors; see Table 17 in Appendix E.

We also wanted to understand if this pollution from likely sandboxes and scanners could be filtered using data from IP telescopes (unused IP address subnets that act as large sinkholes). To evaluate this possibility, we looked for overlap in IP addresses during our deployment period from a /24 sized (256 IP addresses) IP telescope located in India. We find that there is not much overlap. Only 31,014 (3.8%) of the IP addresses we classified as Unknown appear in our telescope data, and less than 0.01% overlap with any other category of IP addresses. This suggests that most of RAT-Hole’s pollution is targeted and thus not filterable. See Appendix B for details.

## 4 RAT-Hole Validation

This section describes our efforts to validate our methodology for differentiating RAT scanners, sandboxes, and victims. Validating our methodology is challenging since we have limited ground truth, except in some instances where we could create it (e.g. Section 4.3).

In this section, we describe our method of building up a high confidence set of RAT scanners, sandboxes, and victims based on additional heuristics for DarkComet and njRAT families.

### 4.1 Low Fidelity (LF) Scanners

Recall that low fidelity scanners are peers that sent valid initial handshake messages, but did not complete the handshake process. In Table 6, we separate 1,421 IP ad-
connections to one of our decoy IP addresses. This behavior. The (\text{1}) cluster are source IP addresses that attempted to connect to one of our decoy IP addresses. This is a fairly strong indication of broader IP address scanning being performed by this source IP address and we are confident that these are scanners. As a point of reference no high fidelity scanner, sandbox, or victim connected to one of our decoy IP addresses.

For cluster (\text{2}), 7,607 (99.7\%) of the fingerprints only attempted to establish one connection. This is a strong indication of a scanner that is randomizing its fingerprint. Another two fingerprints had multiple unique \texttt{INSTALL\_DATE} fields, indicating the possibility that they are sandboxes. For 19 of the fingerprints, we did not detect any anomalies. These 19 peers could be real victims that speak a version of the protocol that is incompatible with RAT-Hole, or that have persistent connectivity issues that prevented them from completing a handshake. We conservatively label these peers low fidelity scanners.

Our anomaly analysis for fingerprints in cluster (\text{3}) shows that 140 (53.6\%) of the fingerprints have multiple unique \texttt{INSTALL\_DATE} fields, likely indicative of scanners that update \texttt{INSTALL\_DATE} based on the current time. Another 24 (9.2\%) had an incorrectly formatted \texttt{HWID}, indicating scanners with protocol formatting errors. The remaining 97 (37.2\%) had no anomalies, but again we conservatively label them low fidelity scanners.

Table 6: Breakdown of LF (Low Fidelity) Scanners

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Decoy Interface</td>
<td>4,105,659</td>
<td>99.8</td>
<td>28</td>
<td>2.0</td>
</tr>
<tr>
<td>Many FP Per Src-IP</td>
<td>7,628</td>
<td>0.2</td>
<td>39</td>
<td>2.7</td>
</tr>
<tr>
<td>Many Src-IPs Per FP</td>
<td>261</td>
<td>&lt;0.1</td>
<td>827</td>
<td>58.2</td>
</tr>
<tr>
<td>Many FPs, Many Src-IPs</td>
<td>6</td>
<td>&lt;0.1</td>
<td>17</td>
<td>1.2</td>
</tr>
<tr>
<td>Single FP, Single Src-IP</td>
<td>510</td>
<td>&lt;0.1</td>
<td>510</td>
<td>35.9</td>
</tr>
<tr>
<td>Total</td>
<td>4,114,064</td>
<td>100.0</td>
<td>1,421</td>
<td>100.0</td>
</tr>
</tbody>
</table>

Table 4: Anomaly, Anomaly type and their descriptions used by RAT-Hole peer classifier.

(1) Peers are identified by Fingerprint. (2) Session = FP + Src-IP + Dst-IP + Dst-Port. (3) Global: All the sessions belonging to a fingerprint. (4) Condition in Row 2 is checked first and Row 3 is followed. (5) We account for domain rotation where a domain is registered under the different RAT-Hole interfaces. (\text{DC}) DarkComet specific rule. (\text{NJ}) njRAT specific rule.

Table 5: Summary of connections received by RAT-Hole, grouped by peer type, fingerprint, Src-IP, ASN, and country. The first three rows (Victims, Sandboxes, and HF Scanners) are detailed in Table 8, while LF Scanners are described in Table 6. †Note that ASN and country show a significant amount of overlap across peer types.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Victim</td>
<td>5,320,297</td>
<td>3.5</td>
<td>6,710</td>
<td>0.8</td>
<td>3,231</td>
<td>0.1</td>
<td>1,079</td>
<td>10.1</td>
</tr>
<tr>
<td>Sandbox</td>
<td>372,883</td>
<td>0.2</td>
<td>1,181</td>
<td>0.1</td>
<td>877</td>
<td>&lt;0.1</td>
<td>418</td>
<td>3.9</td>
</tr>
<tr>
<td>HF Scanner</td>
<td>563,019</td>
<td>0.4</td>
<td>1,349</td>
<td>0.2</td>
<td>589</td>
<td>&lt;0.1</td>
<td>347</td>
<td>3.2</td>
</tr>
<tr>
<td>LF Scanner</td>
<td>17,746,010</td>
<td>11.6</td>
<td>1,421</td>
<td>0.2</td>
<td>4,114,064</td>
<td>99.9</td>
<td>390</td>
<td>3.6</td>
</tr>
<tr>
<td>Unknown</td>
<td>129,097,791</td>
<td>84.3</td>
<td>815,455</td>
<td>98.5</td>
<td>N/A</td>
<td>N/A</td>
<td>10,418</td>
<td>97.2</td>
</tr>
<tr>
<td>Total</td>
<td>153,100,000</td>
<td>100.0</td>
<td>828,137</td>
<td>100.0</td>
<td>4,118,761</td>
<td>100.0</td>
<td>10,722</td>
<td>100.0</td>
</tr>
</tbody>
</table>

Table 6: Breakdown of LF (Low Fidelity) Scanners

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Victim</td>
<td>5,320,297</td>
<td>3.5</td>
<td>6,710</td>
<td>0.8</td>
<td>3,231</td>
<td>0.1</td>
<td>1,079</td>
<td>10.1</td>
</tr>
<tr>
<td>Sandbox</td>
<td>372,883</td>
<td>0.2</td>
<td>1,181</td>
<td>0.1</td>
<td>877</td>
<td>&lt;0.1</td>
<td>418</td>
<td>3.9</td>
</tr>
<tr>
<td>HF Scanner</td>
<td>563,019</td>
<td>0.4</td>
<td>1,349</td>
<td>0.2</td>
<td>589</td>
<td>&lt;0.1</td>
<td>347</td>
<td>3.2</td>
</tr>
<tr>
<td>LF Scanner</td>
<td>17,746,010</td>
<td>11.6</td>
<td>1,421</td>
<td>0.2</td>
<td>4,114,064</td>
<td>99.9</td>
<td>390</td>
<td>3.6</td>
</tr>
<tr>
<td>Unknown</td>
<td>129,097,791</td>
<td>84.3</td>
<td>815,455</td>
<td>98.5</td>
<td>N/A</td>
<td>N/A</td>
<td>10,418</td>
<td>97.2</td>
</tr>
<tr>
<td>Total</td>
<td>153,100,000</td>
<td>100.0</td>
<td>828,137</td>
<td>100.0</td>
<td>4,118,761</td>
<td>100.0</td>
<td>10,722</td>
<td>100.0</td>
</tr>
</tbody>
</table>
Three of the six fingerprints in Cluster (4) had multiple unique INSTALL_DATE fields, again indicating likely scanners that update INSTALL_DATE based on the current time. The remaining three did not have anomalies, but we conservatively label these peers as low fidelity scanners.

### Table 7: Breakdown of Anomalies by Field Format and Peer Behavior

<table>
<thead>
<tr>
<th>Anomaly Type</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Remainder</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Total</td>
<td>✓ ✓ ✓ ✓ ✓ ✓</td>
</tr>
</tbody>
</table>

We did not observe any RAT protocol violations from 387 out of 510 fingerprints (remainders at Table 7) belonging to the peers that had one fingerprint and one IP address (3). On the average peers in this cluster failed 3,000 (90%) attempted connections and a minimum of 100 (2%) attempted connections. Thus it is unlikely that intermittent connectivity issues prevented the completion of the handshake at least once. It is unclear if these are victims that implemented a version of the protocol that is incompatible with our RAT-Hole, persistent connectivity issues, or if they are low fidelity scanners that did not implement the entire protocol. We conservatively label these peers as low fidelity scanners.

### 4.2 Victims, Sandboxes, & High Fidelity (HF) Scanners

We classify a peer as a high fidelity scanner if it is able to complete the handshake, but it violates the field formatting, exhibits peer behavior, or protocol behavior that indicates it is likely a scanner that is reimplementing the njRAT or DarkComet stub instead of an actual stub execution. A peer is conservatively classified as a sandbox if it exhibits peer or protocol behavior that indicates it is likely a sandbox. Finally, if a peer does not violate the protocol or exhibit any anomalous behavior we classify it as a likely victim. Table 8 shows that 69% (3,231) of all peers that complete the handshake with our RAT-Hole are classified as victims. This indicates the significant degree to which high-fidelity scanners and sandboxes will pollute sinkhole results if the sinkhole eschews a deeper analysis of the peers similar to RAT-Hole.

### Table 8: Breakdown of Anomalies by Field Format and Peer Behavior

<table>
<thead>
<tr>
<th>Anomaly Type</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Peer Type</td>
<td>RAT Family</td>
</tr>
<tr>
<td>HF Scanner</td>
<td>DarkComet</td>
</tr>
<tr>
<td>HF Scanner</td>
<td>DarkComet</td>
</tr>
<tr>
<td>HF Scanner</td>
<td>DarkComet</td>
</tr>
<tr>
<td>HF Scanner</td>
<td>DarkComet</td>
</tr>
<tr>
<td>HF Scanner</td>
<td>DarkComet</td>
</tr>
<tr>
<td>HF Scanner</td>
<td>DarkComet</td>
</tr>
<tr>
<td>Sandbox</td>
<td>DarkComet</td>
</tr>
<tr>
<td>Sandbox</td>
<td>DarkComet</td>
</tr>
<tr>
<td>Sandbox</td>
<td>DarkComet</td>
</tr>
<tr>
<td>Sandbox</td>
<td>njRAT</td>
</tr>
<tr>
<td>Sandbox</td>
<td>njRAT</td>
</tr>
<tr>
<td>Sandbox</td>
<td>njRAT</td>
</tr>
<tr>
<td>Victim</td>
<td>DarkComet</td>
</tr>
<tr>
<td>Victim</td>
<td>njRAT</td>
</tr>
<tr>
<td>Total</td>
<td></td>
</tr>
</tbody>
</table>

### 4.3 Honey Sample Seeding

In order to evaluate our classification in a setting where we have ground truth, we conducted an experiment where we uploaded DarkComet and njRAT samples to malware analysis services. Our expectation for this experiment is that all of the connections will be from scanners or sandboxes, which will enable us to validate our classification methodology. Finally, what we classify as victims are the fingerprints that do not exhibit any anomalous behavior and are likely to be actual victims.

Using our automated RAT Seeder, we generated 84 DarkComet and 84 njRAT. Each of these samples has a unique Campaign-id, IP Address, and TCP Port configuration that directed the sample to connect to one of our RAT-Hole IP addresses on a different network segment, which we only used for this experiment. We uploaded 4 DarkComet and 4 njRAT samples to 21 different malware analysis services, of which only 9 of the services initiated a connection for at least one of our samples. A full list of these services and the ones that initi-
In order to maintain an updated list of potential C&C addresses, we resolved each of the 14,273 domains we extracted from our malware samples hourly, beginning on 2017-04-21 and ending on 2017-11-26. Over this period, we recorded 67,023 resolutions to unique IP addresses. We augmented these with passive DNS records dating back to 2010 for each domain using feeds from Farsight [19], VirusTotal, and PassiveTotal [43].

### 5.1 Controller Tracking

In order to maintain an updated list of potential C&C addresses, we resolved each of the 14,273 domains we extracted from our malware samples hourly, beginning on 2017-04-21 and ending on 2017-11-26. Over this period, we recorded 67,023 resolutions to unique IP addresses. We augmented these with passive DNS records dating back to 2010 for each domain using feeds from Farsight [19], VirusTotal, and PassiveTotal [43].

### 5.2 Active Scanning

We continuously probed each of these 67,023 IP addresses hourly for evidence of RAT controller software. We checked for services running on any port configured in any sample related to the IP address or related to a domain that resolved to said IP address at any time.

RAT-Scan probes for controllers of both DarkComet and njRAT, emulating a newly-infected victim contacting the controller for the first time. RAT-Scan first approaches every connection passively, waiting to receive an initial DarkComet handshake banner. If it does not receive a banner before a three second timeout, it restarts the connection and treats it actively, sending the initial njRAT handshake banner. Regardless of which handshake proceeds, the scanner completes the entire handshake with the controller if possible.

**Sinkhole identification.** RAT-Scan can, to some extent, distinguish between legitimate controllers and sinkhole operations like our own RAT-Hole. If a controller begins a handshake but does not complete it, it is labeled as a sinkhole. Additionally, after successfully completing a handshake with a controller, our scanner attempts to elicit an improper response to a second handshake with a different configuration (e.g., different password). Any response is cause for sinkhole classification.

### 5.3 Detected Service Classification

The actors that our scanner probed during its operation fall under one of the following classes: **controller** completes an njRAT or DarkComet handshake flawlessly. Does not respond to solicitation for improper behavior; and **sinkhole** either makes an error during a RAT handshake, or accepts an improper second handshake after the first (like RAT-Hole).

**Important caveats.** RAT-Hole and RAT-Scan have a significant disparity in the confidence of their classifications. RAT-Hole makes use of several protocol artifacts in the DarkComet and njRAT handshakes to detect imposter victims. Because RAT victims are intentionally loquacious during the handshake, this is possible; however, RAT controllers are oppositely taciturn, revealing practically nothing to RAT-Scan during the handshake. DarkComet controllers acknowledge a victims’ correct password and njRAT controllers do not acknowledge this. Therefore, when we classify a host as a DarkComet sinkhole we are fairly confident, but when we label a host a controller it is possible that it is a high-fidelity sinkhole or sandboxed controller.

**Attempted validation.** The joint investigation by Recorded Future and Shodan [22] in 2015 that resulted in Malware Hunter reported 696 IP addresses as suspected RAT controllers, 10 of which appear in our dataset. However, Malware Hunter has since flagged RAT-Hole as a RAT controller and high-priority threat, so we question

---

Table 9: Breakdown of Anomalies for Different Peer Types for Honey Sample Seeding Experiment

<table>
<thead>
<tr>
<th>Peer Type</th>
<th>RAT Family</th>
<th>Field Format</th>
<th>Peer Behavior</th>
<th>Protocol Behavior</th>
<th>Count</th>
<th>Pct.</th>
</tr>
</thead>
<tbody>
<tr>
<td>HF-Scanner</td>
<td>njRAT</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>4</td>
<td>66.7</td>
</tr>
<tr>
<td>HF-Scanner</td>
<td>njRAT</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>2</td>
<td>33.3</td>
</tr>
<tr>
<td>Subtotal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>100.0</td>
</tr>
<tr>
<td>Sandbox</td>
<td>DarkComet</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>55</td>
<td>98.3</td>
</tr>
<tr>
<td>Sandbox</td>
<td>DarkComet</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>1</td>
<td>1.7</td>
</tr>
<tr>
<td>Subtotal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>56</td>
<td>100.0</td>
</tr>
<tr>
<td>Sandbox</td>
<td>njRAT</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>48</td>
<td>96.0</td>
</tr>
<tr>
<td>Sandbox</td>
<td>njRAT</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>2</td>
<td>4.0</td>
</tr>
<tr>
<td>Subtotal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td>100.0</td>
</tr>
<tr>
<td>Victim</td>
<td>njRAT</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>2</td>
<td>100.0</td>
</tr>
<tr>
<td>Subtotal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>100.0</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>117</td>
<td></td>
</tr>
</tbody>
</table>

Note that we used the same classification methodology as for our in the wild sinkholing experiments and only incorrectly classified 2 out of 117 (1.7%) fingerprints as victims. We inspected the active windows for these two fingerprints and found that both appear to be manually reverse-engineering the samples using executable debugging and network protocol analysis tools. Recall that for this experiment we did not encrypt the active windows since we did not expect any real victims. We also classified some njRAT peers as high fidelity scanners. We can confirm that when we tested these samples before submitting them they did not have any protocol violations. This gives us further confidence that our classification methodology is fairly accurate.

### 5. RAT-Scan Operation

#### 5.1 Controller Tracking

In order to maintain an updated list of potential C&C addresses, we resolved each of the 14,273 domains we extracted from our malware samples hourly, beginning on 2017-04-21 and ending on 2017-11-26. Over this period, we recorded 67,023 resolutions to unique IP addresses. We augmented these with passive DNS records dating back to 2010 for each domain using feeds from Farsight [19], VirusTotal, and PassiveTotal [43].
the value of any such threat intelligence feed as proper ground truth. We leave developing a method for improved validation of our scanning results as future work.

5.4 Operation Summary

<table>
<thead>
<tr>
<th>Controller Type</th>
<th># IP</th>
<th>% IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>njRAT</td>
<td>4,584</td>
<td>71.6</td>
</tr>
<tr>
<td>DarkComet</td>
<td>2,032</td>
<td>31.7</td>
</tr>
<tr>
<td>DarkComet (Unknown Password)</td>
<td>11</td>
<td>0.2</td>
</tr>
<tr>
<td>Total</td>
<td>6,401</td>
<td>100.0</td>
</tr>
</tbody>
</table>

Table 10: Breakdown of RAT controllers detected on IP addresses responsive to RAT-Scan. Some IP addresses hosted multiple types of RAT controller.

Our scanning operation began on 2017-05-11 and ended on 2017-11-25, for a total of 198 days. During this period, we established 86,694 connections to 6,401 IP addresses exhibiting behavior indicative of RAT controller software; 2,032 DarkComet controllers and 4,584 njRAT controllers, with some IPs hosting both. Table 10 provides a summary of our scanning operation.

Other than on RAT-Hole itself, our sinkhole detection methods did not trigger during this study. We are led to believe that all controllers reported here are either legitimate instances of the controller software, or services that have implemented the handshake properly and maintain a single configuration. We suspect that such services exist; however, we currently have no way of distinguishing them from legitimate controllers. Further, we have no reason to believe that we encountered any high-fidelity sinkholes similar to RAT-Hole.

6 Measurements and Analysis

6.1 Victim Analysis

IP address churn. We find that most victims do not change their IP address, with 60% of victim using one IP addresses and an additional 20% of all victims use a total of two IP addresses. Note that we might not observe all of the victims’ IP address changes due to our periodic sinkholing of domains.

Webcam availability. As part of the handshake, DarkComet and njRAT victims report if they have a camera device. We found that 1,725 (53.4%) of victims have a camera, making them susceptible to visual monitoring unless they have physically covered the camera.

Infected servers. 21 njRAT victims reported a server version of Windows (i.e., Windows Server 2012) running on the peer. We manually investigated the Autonomous System Numbers for the IP addresses used by these peers and confirmed that they were located on corporate networks or cloud hosting providers. This suggests that some higher profile peers associated with companies are infected with njRAT, providing the operator with an entry point into their systems.

6.2 Attacker Campaign Analysis

Infection longevity. Our methodology is predicated on victims remaining after the command-and-control dynamic DNS domain used by the attacker expires, which occurs 30 days after registration with No-IP. Figure 4 shows the fraction of domains still receiving legitimate victim connections as a function of time since the dynamic DNS domain expired. Because our sinkholing period does not span our full domain monitoring period (31 days from 2017-08-15 to 2017-09-16, and 220 days from 2017-04-21 to 2017-11-26; see Figure 3), we do not necessarily known victim availability immediately starting from when the domain expired. Figure 4 shows an upper and lower bound curve; the upper bound corresponds to the case that at least one victim connection occurred during the period when the command-and-control domain was not monitored, and the lower bound corresponds to the case that no victim connections occurred during the same period. Thus, 120 days after the command-and-control domain expired, 10% of domains were still receiving connections from legitimate victims.

In all, 975 domains received victims, 14% of the 6,897 we sinkholed. 1,686 of these domains had no known historic resolution from any of our sources, including threat intelligence feeds and our own resolver.

In all, 975 of the domains we sinkholed yielded victim connections, yet they received connections from 3,231 unique victims. In Figure 5, we examine the number of unique victims any one domain received. 43% of domains received only a single victim; 90% received at most 20 unique victims; 95%, 41 or less. Three outlier domains received over 100 victims. This disparity suggests that some attackers are distributing their malware more widely, or are more proficient at compromising their targets, than others.
We also find that 45% of victims connected to just one domain. 90% of victims connected to four or less different domains, while 95% connected to five or less. These victims connecting to multiple different domains, and domains receiving tens of unique victim connections, suggest a number of phenomena. Attackers may be using sets of domains interchangeably, or victims may be infected by multiple different attackers.

To investigate the former, we examine the samples which we downloaded from VirusTotal. Our 975 domains are found in 1,429 unique samples. Once again, there is bidirectional overlap. Only eight samples contain more than one domain in their configurations; 1,421 have a single domain. Multiple domains being in a single sample is our strongest indicator of them being shared by an attacker. Oppositely, 246 domains are found in more than one sample’s configurations (the remaining 729 domains are each present in just one sample). Some domains are shared by many samples - one being found in 24 unique DarkComet samples. Unfortunately, these domains further muddy our analysis. In the case of the domain shared by 24 samples, only two of those samples clearly belong to the same actor (based on other shared configuration parameters).

Our methodology cannot definitively answer whether attackers use rotating domains, or whether victims are multiply infected by different campaigns. Based on our evidence, both appear probable, and confound our attempts at differentiating attackers and their victims.

### 6.3 Geographic Analysis

All IP-based geolocations were performed using MaxMind’s GeoIP2 Precision Insights service.

**Proxies.** MaxMind provides information regarding the likelihood that an IP address is a proxy, as well as IP ownership (which can be used to manually determine proxies). We use this information to separate proxies from non-proxies, as in Tables 11 and 12. A large portion of the controllers in our data set appear to be utilizing proxies from certain countries like France, Sweden, and the U.S. We manually investigate the largest

We manually investigate the largest in Appendix A. In short, we find two VPN providers (IPjetable [24] and Relakks [42]) account for 40% and 3% of all actively-probed controllers, respectively, while prominent VPS services like Amazon AWS, Microsoft Azure, and Digital Ocean are also frequently abused.

As the geolocation results of the proxies only serve to muddle the geospatial relationships between victims and attackers, we filter them from the following analyses. We report only on those results in the Other columns of the geolocation tables.

**Controller geography.** Tables 11 and 12 show the geolocations of historic and actively-probed controller IP addresses, respectively. We find both to have heavy presences in North Africa and the Middle East. Outliers include Brazil and Russia, both of which tend to correspond with victims in bordering nations.

**Victim geography.** Exploring Table 13, we find that virtually every country has some RAT victims with Brazil being the top location for victims of both DarkComet and njRAT, as shown in Table 13. We find what appears to be
geographic concentrations of DarkComet and njRAT victims in South America and North Africa / Middle East, including some bordering countries. We also find that DarkComet is used to infect a larger percentage of victims in Russia and bordering countries. Note that these measurements might be biased by our methodology of acquiring RAT samples and sinkholing DDNS domains.

<table>
<thead>
<tr>
<th>Country</th>
<th>#Src-IP</th>
<th>#FP</th>
<th>Country</th>
<th>#Src-IP</th>
<th>#FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brazil (BR)</td>
<td>2,416</td>
<td>1,070</td>
<td>Brazil (BR)</td>
<td>318</td>
<td>178</td>
</tr>
<tr>
<td>Egypt (EG)</td>
<td>331</td>
<td>94</td>
<td>Turkey (TR)</td>
<td>188</td>
<td>130</td>
</tr>
<tr>
<td>Iraq (IQ)</td>
<td>207</td>
<td>82</td>
<td>Russia (RU)</td>
<td>184</td>
<td>127</td>
</tr>
<tr>
<td>Argentina (AR)</td>
<td>138</td>
<td>62</td>
<td>Ukraine (UA)</td>
<td>44</td>
<td>38</td>
</tr>
<tr>
<td>Algeria (DZ)</td>
<td>149</td>
<td>60</td>
<td>Egypt (EG)</td>
<td>74</td>
<td>36</td>
</tr>
<tr>
<td>Peru (PE)</td>
<td>131</td>
<td>55</td>
<td>Poland (PL)</td>
<td>28</td>
<td>26</td>
</tr>
<tr>
<td>Vietnam (VN)</td>
<td>117</td>
<td>53</td>
<td>Philippines (PH)</td>
<td>22</td>
<td>21</td>
</tr>
<tr>
<td>United States (US)</td>
<td>54</td>
<td>47</td>
<td>Thailand (TH)</td>
<td>35</td>
<td>17</td>
</tr>
<tr>
<td>Venezuela (VE)</td>
<td>105</td>
<td>47</td>
<td>Vietnam (VN)</td>
<td>16</td>
<td>14</td>
</tr>
<tr>
<td>India (IN)</td>
<td>88</td>
<td>46</td>
<td>Algeria (DZ)</td>
<td>21</td>
<td>13</td>
</tr>
<tr>
<td>Turkey (TR)</td>
<td>93</td>
<td>40</td>
<td>Bosnia (BA)</td>
<td>17</td>
<td>13</td>
</tr>
<tr>
<td>Thailand (TH)</td>
<td>189</td>
<td>38</td>
<td>Indonesia (ID)</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>Mexico (MX)</td>
<td>66</td>
<td>37</td>
<td>India (IN)</td>
<td>11</td>
<td>10</td>
</tr>
<tr>
<td>Other</td>
<td>1,401</td>
<td>659</td>
<td>265</td>
<td>207</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>5,485</td>
<td>2,390</td>
<td>1,235</td>
<td>841</td>
<td></td>
</tr>
</tbody>
</table>

Table 13: Geolocations of victim IP addresses

Controller-victim geography: Recall that during the sinkholing portion of the experiment, we registered the command-and-control domain, directing all potential victims to our server. During this period, we were able to observe all victims that attempted to connect to the controller. Prior to the sinkholing period, controller domains may have been held by the original controller or may have been sinkholed by researchers or vigilantes. In addition, for four and a half months prior to the sinkholing experiment, we resolved all controller domains to determine whether they were registered, and, if registered whether they had an associated A record, and whether the corresponding hosts behaved correctly (as a controller). Thus, for each domain, we have the IP addresses of all controllers that held the domain, as well as of all victims that attempted to connect to the domain during the sinkholing period. (Note that two periods are necessarily disjoint: both we and the original controller cannot hold the same domain at the same time.) Figure 6 shows the geographic relationship between responsive controllers and the victims, using the geolocation methodology above. Each cell of the matrix shows the number of distinct campaigns (domains) associated with the given country pair. In cases where a domain resolved to more than one country or where victims located in more than one country, the domain contributed a fractional weight to each cell in proportion to the number of controller-victim pairs of the domain from the country pair, so that the total contribution of each domain was 1. Figure 6 shows only the top 25 countries, ordered by the greater of the number of victims and controllers in the country. The dominant feature of the data is the controller and victim being located in the same country, visible as a concentration around the diagonal in the matrix. In addition, there were 5 campaigns with a controller in Ukraine (UA) and victims in Russia (RU). This may be due to a common infection vector, as Ukraine has a large Russian-speaking population and its users may frequent the same Russian-language sites. The incidence of controllers and Russia and victims in Brazil (BR) is more puzzling; although both Russia and Brazil have large victim and controller populations, there is no obvious reason why controllers in Russia might target victims in Brazil specifically. Another possibility is that the controllers were using a proxy in Russia that was missed by our filtering.

Figure 6: Relational matrix comparing geolocations of actively-probed controller IP addresses to received victim IP addresses, per sinkholed domain. Proxy IP addresses are filtered.

Figure 7 shows the same type of data, but for all controllers using the historic controller dataset. Note that this data spans the period 2010 to 2017 and includes name resolution from passive DNS sources (see Section 5.1), where we did not verify the correct behavior.
of the controller. As such, this data should be interpreted with caution. Figure 7 exhibits the same concentration around the diagonal as Figure 6, indicating campaigns where both controller and victim are in the same country. As the results of Table 13 suggest, Brazil has by far the largest concentration of victims across both RATs. Moreover, Brazil appears to be victimized indiscriminately. We also note some language clustering, where countries that speak the same language or are geographically proximate are more likely to be paired; e.g., Russia on Ukraine (13), Ukraine on Russia (18), Ukraine on Kazakhstan (4), Ukraine on Belarus (5), Morocco (MA) on Algeria (DZ) (9), Algeria on Morocco (8).

7 Discussion

7.1 Limitations

Adversarial robustness. Our classification methods that RAT-Hole uses to distinguish sandboxes, scanners, and victims are not robust to an adversarial actor that is actively trying to evade our classification. Based on our validation it appears that there are many detectable sandboxes and scanners. As future work, we will explore potentially more robust features, such as inter-arrival timing of connection attempts in order to detect scanners and analysis of active window patterns to detect sandboxes.

Manual reverse engineering. We have not developed a method for automatically decrypting RAT Protocols or parsing out information from fields in the protocol. This caused us to limit our analysis to two common RAT family types. It is unclear what biases might have been introduced into our results due to limiting the number of RAT families and DDNS services included in our study. As future work we will explore how well existing methods for automated protocol reverse-engineering [7,8] and decrypting of messages [47] perform at our task.

Family-specific classification heuristics. Our methodology for building up a set of classification heuristics was again a manual process and in some instances, such as triggering error conditions, was RAT family specific. As future work, we will explore more automated methods, such as semi-supervised machine learning based approaches using inter-arrival timing of connections to differentiate scanners from execution of the actual malware. We will also explore methods based on victim behavior to identify sandboxes. We hypothesize that it will be difficult for a sandbox to mimic the patterns of a real victim.

Validating scanning results. We have little ground truth to evaluate methods for distinguishing between legitimate RAT controllers and sinkhole operations, other than our own sinkhole. As future work we will explore additional methods of ethically probing controllers, such as calling rarely used API functions that are unlikely to be implemented by sinkholes.

7.2 Protecting Victims

Our results show that expired RAT domains still have likely victims attempting to connect to them. The 3,231 likely victims we detected could be further victimized by an adversary that claimed these domains. We are in the process of working with some free DDNS providers to understand if they would be willing to permanently block domains associated with RAT controllers.

8 Related Work

Our work is influenced heavily by research projects from industry and academia. We discuss works that informed our study’s primary aspects: sinkholing and scanning.

Sinkholing and infection enumeration. A number of early botnet measurement studies mused on its challenges. A Trend Micro industry report from 2001 [29] qualitatively discussed the problems with sinkholing botnet domains, like receiving PII. The ethical issue of victim PII receipt is universal to infection enumeration efforts; Han et al. [23] built a system for sandboxing phishing kits explicitly designed to protect victim privacy.

Always prominent has been the issue of accurate infection size estimation. Ramachandran et al. [41] proposed a method of estimating botnet infection size based on frequency of DNS lookups to C&C domains. A subsequent pair of botnet size estimation studies used DNS lookups [9] and IRC channel monitoring [1], but arrived at different estimates due to errors caused by churn [40].

A number of studies explored how to estimate the size of the Storm botnet [16, 21, 38], while Stone-Gross et al. [49] actually sinkholed the Torpig botnet, live, and created unique fingerprints for each infection to address infection measurement difficulties, as do we in this study. A follow-up study by Kanich et al. [25] showed that pollution caused by interfering measurement operations had inflated the measured size of the botnet. Nadji et al. [34] discuss the same issue of measurement pollution while running a domain sinkhole performing botnet takedowns.

Novel approaches for detecting and filtering scanners exist. For instance, Rossow et al. [44] proposed a method for detecting sensors based on detecting crawlers injecting themselves into large numbers of points in a P2P network. Successful methods for detecting scanners tend to be highly tailored, as was ours.

Our methodology exploits the fact that DDNS domains used as C&C’s will ultimately expire, though victims are still contacting them. This is one of the premises behind work by Starov et al. [48]; though they focus on web shells rather than more traditional RATs, their goal of measuring the ecosystem of attackers and victims is similar to ours. lever et al. [28] measure the adversarial possibilities behind re-registering an expired domain.

Part of our methodology focuses on the challenge of detecting malware samples being executed in sandboxes,
which we found to be a source of intelligence pollution. Most prior studies on sandbox detection focus on malware sandbox evasion techniques [4, 26, 33, 37, 45, 53]. A more recent study demonstrated that intentionally-designed binary submissions to antivirus companies can exfiltrate sandbox fingerprints [55]. Our approach furthers these efforts to identify Internet-connected sandboxes, using unmodified malware binary submissions and leveraging artifacts of the execution process like short execution duration to inform our detection.

In a 2014 report, researchers at FireEye enumerated infections for an XtremeRAT campaign by sinkholing the controller domain [51]. This study notes the challenges of victim IP address churn, which our work also encountered. We designed our methodology to explicitly handle the challenges this study uncovered, as well as to filter intelligence pollution from scanners and sandboxes, such that we could accurately and ethically enumerate RAT infections based on sinkhole data.

Scanning and controller discovery. BladeRunner [15] was the first scanning-based system to actively discover RAT controllers by emulating RAT victims. Since then, Shodan [31] has added active probing and banner identification for numerous RAT families including DarkComet and njRAT. Marczak et al. [30] created a scanner that was able to detect stealthy APT controllers by triggering error conditions. Most recently, Farinholt et al. [17] presented a scanner that used ZMap [14], Shodan, and a custom port scanner to detect DarkComet controllers based only on their initial handshake challenges. RAT-Scan’s design is based on these systems.

RAT-Scan also contains logic to (attempt to) address the issue of sinkholes polluting controller measurements. The most closely related work is SinkMiner, a system which proposed a method to detect sinkholes based on historic DNS data [39]. Though SinkMiner uses passive DNS to detect sinkholed domains, its research goals - measuring the effective lifetime of a C&C domain and avoiding enumerating fellow security vendors’ infrastructure - matched ours. We consider RAT-Scan complimentary to SinkMiner in this regard.

9 Conclusion
We presented the results of our study of attacker and victim populations of two major RAT families, njRAT and DarkComet. One of the challenges of studying both operators (attackers) and victims is the noisy nature of the signal. To distinguish real operators and victims, we develop a set of techniques for testing the behavior of a suspected victim to determine if it is a genuine infection or not. Using a similar set of tests, we identify genuine controllers, excluding sinkholes and controllers using VPNs.

Using our collected data, we then report on the population of victims and controllers, their geographic relationship, and periods of activity. Our results show that the RATs we studied are used primarily by operators and victims located in the same country, with the bulk of the population in Russia, Brazil, and Turkey. We also found that victims remain vulnerable long after the controller abandons the campaign, presenting an opportunity for third-party intervention by sinkholing the domains.

Acknowledgments
This work was supported by the National Science Foundation through grants CNS-1237264, CNS-1619620, and CNS-1717062, and by gifts from Comcast, Farsight Security, and Google. We would also like to thank the following: VirusTotal, for the invaluable Intelligence account from which we sourced malware; Richard Harper of DuckDNS, for generous access to a Duck Max account; Matthew Jonkman of EmergingThreats, for generous access to an unlimited Threat Intelligence account; and finally, our reviewers, for their invaluable feedback.

References


[46] Satter, R., J Donn, E., and Vasilyeva, N.


YARA: The pattern matching swiss knife for malware researchers (and everyone else). http://virustotal.github.io/yara/


A VPN/VPS Provider Abuse

IPjetable VPN: 141.255.144/20. Of the 6,401 IP addresses RAT-Scan successfully probed, a full 2,635 (or 40.2%) came from this address space. Further, these IP addresses accounted for over 40% of all connections made during the six months of active scanner operation, exhibiting abnormal longevity compared to other controllers. This space is owned by IPjetable, a French company that provides free VPN services and is likewise recommended by RAT instruction videos [50] and HackForums members.

VPS providers. In addition to using VPN’s, we found the use of VPS instances from prominent services like Amazon AWS, Microsoft Azure, and Digital Ocean, as well as less reputable providers like OVH.

B Telescope Data

<table>
<thead>
<tr>
<th>Peer Type</th>
<th>Overlapping /32</th>
<th>% Overlap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Victim</td>
<td>5</td>
<td>&lt;0.1</td>
</tr>
<tr>
<td>Sandbox</td>
<td>8</td>
<td>&lt;0.1</td>
</tr>
<tr>
<td>LF Scanner</td>
<td>38</td>
<td>&lt;0.1</td>
</tr>
<tr>
<td>HF Scanner</td>
<td>1</td>
<td>&lt;0.1</td>
</tr>
<tr>
<td>Unknown</td>
<td>31,014</td>
<td>3.8</td>
</tr>
<tr>
<td>Total</td>
<td>828,137</td>
<td>100.0</td>
</tr>
</tbody>
</table>

Table 14: Breakdown of the Src-IPs (/32) of our defined peer types that overlap with IP addresses from our telescope dataset.

Table 14 compares RAT-Hole’s connection dataset with a prominent network telescope’s connection dataset, showing the overlapping connecting source IP addresses. Of import is the lack of overlap between the datasets. This refutes our initial hypothesis that network telescope data could be used to filter most indiscriminate scanning operations from a sinkhole’s dataset.

![Image](a) ![Image](b)

Figure 8: The standard handshake protocol for (a): njRAT (b): DarkComet

C njRAT Network Protocol

njRAT speaks a custom application-layer network protocol over TCP. In Figure 8a, we provide a diagram of the njRAT handshake, the initial exchange of messages between the stub and controller before the command-response cycle begins. We note that, as njRAT is a victim-initiated RAT, its stub sends the first payload of the handshake after establishing a TCP connection to the controller.

njRAT has many variants (or sub-species). The three most commonly found in the wild are main (the original version), KilerRAT, and Coringa-RAT. We reverse engineered the protocols of each of these three variants, and as such RAT-Hole fully supports connections from all three.

Each of the messages exchanged in the njRAT handshake from Figure 8a is now further detailed individually.
The document contains a detailed explanation of the njRAT malware, focusing on its communication protocol and behaviors. It includes a table that categorizes various fields and their descriptions, along with examples of message patterns. The text is technical and covers topics such as malware version, victim operating system, and communication protocols. ThenjRAT Victim Info Message is described in detail, including payload headers, size, active window, and other parameters. The DarkComet Network Protocol is also discussed, highlighting the differences in communication methods between njRAT and DarkComet RATs.
null