Defending Dissidents from Targeted Digital Surveillance

William Marczak

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2016-213
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-213.html

December 16, 2016

Defending Dissidents from Targeted Digital Surveillance

by

William R Marczak

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Vern Paxson, Chair
Professor Eric Brewer
Associate Professor Deirdre Mulligan
Professor Ronald Deibert

Fall 2016

Defending Dissidents from Targeted Digital Surveillance

Abstract

Defending Dissidents from Targeted Digital Surveillance

by

William R Marczak

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Vern Paxson, Chair

Computer security research devotes extensive efforts to protecting individuals against indiscriminate, large-scale attacks such as those used by cybercriminals. Recently, the problem of protecting institutions against targeted attacks conducted by nation-states (so-called "Advanced Persistent Threats") has likewise elicited significant research interest. Where these two problem domains intersect, however—targeted cyber attacks by nation-states against *individuals*—has received little significant, methodical research attention. This new problem space poses challenges that are both technically complex and of significant real-world importance.

In this thesis, we undertake to characterize the emergent problem space of nation-state Internet attacks against individuals engaged in pro-democracy or opposition movements. We first present several years of research we have conducted into cases from two Middle Eastern countries, in the aftermath of the Arab Spring. Leveraging our connections in Bahrain and the United Arab Emirates, we encouraged potential targets to send us any "suspicious" electronic communications they received. Dissidents forward us messages with malicious attachments, links, and other content designed to deanonymize them and break into their computers and phones. Strong circumstantial evidence ties some of these messages to specific nation-state attackers. We frame the nature of these attacks, and the technology and infrastructure used to conduct them, in the context of their impacts on real people.

Building on our understanding of attacks targeting dissidents, we engaged with 30 potential targets of Middle Eastern and Horn of Africa-based governments, in order to better understand subjects' perceptions of the risks associated with their online activity. We interviewed subjects, and examined settings and software on their computers and phones. Our data illuminate the ways that dissidents are vulnerable to the types of attacks employed by nation-states.

Informed by our fieldwork, we developed *Himaya*, a defensive approach that readily integrates with targets' workflow to provide near real-time scanning of email messages to check for threats. Our prototype implementation of *Himaya* currently protects 36 subjects,

and has found a number of attacks both from scans of past message archives and in live activity.

For my parents

Contents

List of Figures

List of Tables

Acknowledgments

I've always wanted to work on big things.

When I came to Berkeley in 2009, I was fortunate to be working with my advisor Joe Hellerstein and colleagues Peter Alvaro and Neil Conway in re-imagining computer science for a "disorderly" big data future. Sequentiality was out, and commutativity, associativity, and idempotence were in. Thinking of programming as logic highlighted the complexity inherent in distributed environments: problems were only as hard as the amount of "coordination" (quantification) needed to specify them. I was inspired by the magnitude and potential implications of the endeavor. Despite my despair at large corporations' monopoly on big data, I found hope in Peter's desire to "bring programming back to the streets."

In 2011, as protesters flooded into the streets as part of the Arab Spring, my focus shifted to Bahrain, the country where I attended high school. As the government mobilized an unprecedented campaign to attack demonstrators on the ground, in the media, and online, building the future of computer programming took a back seat to more immediate questions. Who was supplying police with weapons to attack civilians? How was the government shaping the media narrative? What could I do to help?

In early 2012, two activists and I formed an NGO, Bahrain Watch, to rigorously investigate these questions, and advocate based on our research. Our first project, *Arms Watch*, traced armored vehicles, tear gas, and birdshot used by police to its manufacturers, based on crowdsourced images of police weapons. Our second project, *PR Watch*, identified how Bahrain spent tens of millions of dollars on the services of western PR companies, to try to muzzle protesters in the international media through propaganda and legal threats. Our work attracted the attention of Bahrain's government. In mid-2012, one of my colleagues at Bahrain Watch began receiving suspicious emails, promising details of secret dialogues, and images leaked from the Ministry of Interior.

With little experience in computer security or malware analysis, I performed a very crude investigation on the emails to determine that they contained spyware. I recalled seeing a recent news story about the use of spyware against Syrian activists, and got in touch with the researchers who published it at Citizen Lab: John Scott-Railton and Morgan Marquis-Boire. Morgan helped introduce me to the ins and outs of malware analysis. We discovered that the spyware was FinSpy, a type of malware sold exclusively to governments, ostensibly for the purposes of "lawful interception." Morgan and I published a report together, and got incredible media coverage, including our pictures on the front page of the New York Times. As a result of the publicity, interest in our work snowballed.

In early 2013, I found myself spending more time on surveillance research than on my schoolwork. It was time for a change. Joe graciously allowed me to switch research directions and advisors. Vern Paxson very kindly took me on as his student, and my surveillance work became my PhD research.

In October 2013, Bahrain Watch received word of an upcoming tear gas shipment to Bahrain. The Ministry of Interior was attempting to buy in excess of three million tear gas canisters, approximately three canisters for every man, woman, and child in the country.

We heard that Bahrain had taken delivery of one million tear gas canisters in 2012, and saw police indiscriminately launch hundreds of canisters every night in residential areas, to punish entire communities where some had protested earlier. Unsurprisingly, this policy resulted in dozens of deaths, as tear gas canisters discharged inside poorly ventilated homes, sometimes igniting fires, and metal tear gas projectiles hit civilians in the head. We had to stop Bahrain from buying more tear gas, and we had an opportunity to act.

We had proof of the upcoming shipment: a leaked tender document from Bahrain's Ministry of Interior, the first one we had ever seen. I recall staying up all night to design our campaign, *#StopTheShipment*, around the leaked tender. I built a website allowing people around the world to call and send emails to governments and companies we knew were supplying Bahrain with tear gas. I believed we could be successful if we hit hard and fast. Our campaign flooded tear gas manufacturers' inboxes, some of whom responded to us, pointing the finger at a Korean company, Dae Kwang Chemical Corporation.

Protesters converged on Korean embassies in Washington DC and London, as people from around the world phoned flabbergasted officials in the Korean government through our website: "How did you get this number? Why do you care about Bahrain? You're calling from *where*?" A coalition of dozens of Korean NGOs marched on the Ministry of Foreign Affairs in Seoul. As momentum built, Bahrain Watch filed lawsuits, UN complaints, and an OECD action. "How did Bahrain's opposition reach the six continents?" wondered a pro-government Bahraini newspaper columnist. Another newspaper slammed us for "trying to deny protection" to police. The government blocked our campaign website inside Bahrain. But it was too late.

In January 2014, the Financial Times reported that South Korea's Dae Kwang Chemical Corporation had been denied a government export license to ship three million canisters of tear gas to Bahrain in a deal valued at $50m. The Korean Charge d'Affairs in Bahrain told a local newspaper that the government was shocked by the huge amount of international attention generated by our campaign, and felt they had no choice but to deny the license. We had changed the world! Three months later, two Bahrain Watch colleagues and I travelled to Korea, upon the invitation of Korean NGOs. We met high ranking officials in the Korea National Police Agency, the Ministry of Defense, and the Ministry of Foreign Affairs. They apologized to Bahrainis for the death and destruction caused by their products.

I wasn't thinking of my dissertation while working on *#StopTheShipment*, but it turned out that my activism helped deepen trust and connections with contacts who ultimately became subjects of my research studies.

As the years passed, I had opportunities to work on ever more exciting projects, including the discovery of the first ever weaponized iPhone zero-day remote jailbreak in 2016.

I would like to acknowledge everyone that has helped me complete this dissertation, and everyone who has facilitated my work on big things for the past 8 years, especially my advisor Vern Paxson, my Bahrain Watch colleagues, my Citizen Lab colleagues, especially John Scott-Railton, and members of the press who have covered my work, including Vernon Silver from Bloomberg, Nicole Perlroth from the New York Times, Raphael Satter from the Associated Press, Lorenzo Franceschi-Bicchierai from Motherboard, and Bryan Burrough

from Vanity Fair.

Also a special shout-out to some of my academic colleagues: Zakir Durumeric and Nick Weaver. Zakir's brilliant *zmap* tool proved indispensable to my research. Even though Zakir is an academic rockstar, he always made time to assist me whenever I had a question about *zmap*. Thanks, Zakir! I am also grateful to Nick Weaver, for his constant inspiration and upbeat attitude.

Chapter 1

Introduction

In recent years, bolstered by the popularity of social networking, online communication is increasingly mediated by *platforms*, such as those of Facebook, Twitter, and Google. These platforms often employ *encryption* to obscure communications in transit between the platform and its users, and support pseudonymous communication between users.

The encryption and pseudonymity provided by these platforms frustrates long-standing efforts by repressive nation-states to control information online. Traditional *passive* surveillance alone can no longer reliably uncover the content of these encrypted communications, or even the identity of their communicators, and censors examining Internet traffic can often no longer distinguish undesirable information from benign content.

In response, some states have augmented their existing monitoring apparatuses to include the use of targeted electronic attacks against dissidents' *devices*. These attacks typically involve the use of malicious links or e-mail attachments, designed to unmask a pseudonymous user, or break into a target's computer or phone to extract private information. As an illustration of this phenomenon, consider the following anecdote, pieced together from public reports and court documents.

At dawn on March 12, 2013, police raided the house of 17-year-old Ali Al-Shofa (Section 2.3.2), confiscated his laptop and phone, and took him into custody. He was charged with referring to Bahrain's King as a "dictator" (**الطاغية**) and "fallen one" (**الساقط**) on a pseudonymous Twitter account, `@alkawarahnews`. According to court documents, Bahrain's Cyber Crime Unit had linked an IP address registered in his father's name to the account on December 9, 2012. Operators of `@alkawarahnews` later forwarded a suspicious private message to us. The message was received on December 8, 2012, on a Facebook account linked to the Twitter handle, and contained a link to a protest video, purportedly sent by an anti-government individual. The link redirected through `iplogger.org`, a service that records the IP address of anyone who clicks. Analytics for the link indicate that it had been clicked once from inside Bahrain. On June 25, 2013, Ali was sentenced to one year in prison.

Ali's case is an example of the larger phenomenon we study throughout this thesis: attacks against activists, dissidents, trade unionists, human rights campaigners, journalists, and members of NGOs in the Middle East. The attacks we have documented usually involve

the use of malicious links or e-mail attachments, designed to obtain information from a device.

Attack tools and technology

On the one hand, we have observed attacks using a wide range of off-the-shelf spyware (Section 2.4.2), as well as publicly available third-party services, like `iplogger.org` (Section 2.3.2). On the other hand, some attacks use so-called "lawful intercept" trojans and related equipment (Sections 2.3.1, 2.4.1, 2.4.5), purportedly sold exclusively to governments by companies like Gamma International, NSO Group, and Hacking Team. The latter advertises that governments need its technology to "look through their target's eyes" rather than rely solely on "passive monitoring" [35]. Overall, the attacks we document are rarely technically novel. In fact, we suspect that the majority of attacks could be substantially limited via well-known security practices, settings, and software updates. Yet, the attacks are noteworthy for their careful social engineering, their links to governments, and their real-world impact.

Where possible, we empirically characterize the attacks and technology we have observed. We map out government use of commercial hacking tools by conducting Internet scanning, using fingerprints for command-and-control (C&C) servers derived from our spyware analysis. We systematically identify government users of two such tools (Sections 3.2, 3.3), and in other cases, we discover some *bait content* themes (Section 3.5) used to convince targets to interact with malicious messages (e.g., attacks posing as "unsubscribe" links, or service messages from a target's ISP), and even enumerate the bait content itself (Section 3.4).

Target perceptions and defenses

With information about attackers, tools and tactics established, we next explore potential targets' perceptions of the risks they face, and their security behaviors, with an eye towards devising effective surveillance defenses for targeted groups. We conducted interviews and examined the computers and phones of 30 subjects on the ground in two Middle Eastern countries, as well as activists originally from the Middle East or Horn of Africa but now residing abroad (Chapter 4). Our survey illuminates several important differences between these subjects and ordinary users, including subjects' perceptions of risk (e.g., surveillance resulting in government punishment was feared by more than half of on-the-ground activists), as well as their security behaviors (e.g., using out-of-country human "password managers" to preserve account security in case of arrest).

Despite their heightened awareness of risk and steps taken in response to it, on the whole however our results indicate that activists, NGOs, and civil society remain vulnerable to attacks involving social engineering. Even subjects who report positive security behavior can come up short in implementing it correctly.

Informed by the results from our interviews, to address the threat of social engineering to such targets we developed a cloud-based message scanning tool, *Himaya*, to help potential

surveillance targets better decide whether messages they receive may be malicious. Subjects authorize *Himaya* to access their online accounts, enabling the system to check message elements against a variety of both static and behavioral signatures in order to alert users to instances of malicious messages. In the long term, we hope that *Himaya* will provide deeper insight into how activists are targeted, and fundamentally advance the surveillance arms race by rendering an entire class of attacks likely to be detected by targets.

Chapter 2

Ecosystem Studies

2.1 Introduction

In this chapter we analyze a large collection of suspicious files and links targeting activists, opposition members, and non-governmental organizations in the Middle East between 2012 and 2016. We provide extensive detail from both technical and operational perspectives as seen in two countries: Bahrain and the United Arab Emirates (UAE). Since early 2011, both countries have been shaped by the Arab Spring. We frame the nature of these attacks, and the technology and infrastructure used to conduct them, in the context of their impacts on real people. We view such characterizations as the fundamental first step necessary for the rigorous, scientific pursuit of a new problem space. (Additional details about attacks in a third country, Syria, are available in [129].)

We obtained the majority of our artifacts by encouraging individuals who might be targeted by governments to provide us with suspicious files and unsolicited links, especially from unfamiliar senders. Some of these individuals have been systematic in their efforts to identify potential threats, while in other cases we received samples of attacks simply because they caught someone's attention.

In some cases, we find that the attackers employed expensive and government-exclusive malware (Sections 2.3.1, 2.4.1, 2.4.5), while in other cases, attackers used cheap and readily available Remote Access Trojans (RATs), or publicly available "IP logging" services (Sections 2.4.2, 2.3.2).

Across these cases we find that clever social engineering often plays a central role, which is strong evidence of a well-informed adversary. We also, however, frequently find technical and operational errors by the attackers. These errors sometimes allow us to link different attacks to each other, and develop strong evidence tying attacks to government sponsors and corporate suppliers, countering denials, sometimes energetic and sometimes indirect, of such involvement [6, 175, 99, 18]. Despite these denials, data leaked from two vendors of commercial surveillance tools confirms our conclusions [27, 70].

2.2 Related Work

In the past decades, a rich body of academic work has grown to document and understand government Internet censorship, including nationwide censorship campaigns like the Great Firewall of China [24, 28, 206]. Research on governmental Internet surveillance and activities like law-enforcement interception is a comparatively smaller area [170]. Some academic work looks at government use of devices to enable censorship, such as keyword blacklists for Chinese chat clients [29], or the Green Dam censorware that was to be deployed on all new computers sold in China [204]. We are aware of only limited previous work looking at advanced threat actors targeting activists with hacking, though this work has not always been able to establish evidence of government connections [115].

Platforms used by potential targets, such as Gmail [36], Twitter [122], and Facebook [25] increasingly make transport-layer encryption the default, obscuring communications from most network surveillance. This use of encryption, along with the global nature of many social movements, and the role of diaspora groups, likely makes hacking increasingly attractive, especially to states who are unable to request or compel content from these platforms. Indeed, the increasing use of encryption and the global nature of targets have both been cited by purveyors of "lawful intercept" trojans in their marketing materials [35, 53]. In one notable case in 2009, UAE telecom firm Etisalat distributed a system update to its then 145,000 BlackBerry subscribers that contained spyware to read encrypted BlackBerry e-mail from the device. The spyware was discovered when the update drastically slowed users' phones [14]. In contrast to country-scale distribution, our work looks at this kind of pro-government and government-linked surveillance through highly *targeted* attacks.

The term APT (Advanced Persistent Threat) refers to a sophisticated cyber-attacker who persistently attempts to target an individual or group [124]. Work outside the academic community tracking government cyberattacks typically falls under this umbrella. There has been significant work on APT outside the academic community, especially among security professionals, threat intelligence companies, and human rights groups. Much of this work has focused on suspected government-on-government or government-on-corporation cyber attacks [123, 50].

Meanwhile, a small but growing body of this research deals with attacks carried out by governments against opposition and activist groups operating within, as well as outside their borders. One of the most notable cases is GhostNet, a large-scale cyber espionage campaign against the Tibetan independence movement [37, 146]. Other work avoids drawing conclusions about the attackers [178].

2.3 Bahrain

Bahrain's government has been facing an uprising since February 2011, when protesters inspired by the Arab Spring occupied a main square in Bahrain's capital. The government

began a wide-ranging crackdown, both offline and online, against institutions and segments of society that were seen to have supported the protests.

We analyze two attack campaigns in the context of Bahrain. The first involved malicious e-mails containing *FinSpy*, a "lawful intercept" trojan sold exclusively to governments. The second involved specially crafted *IP spy* links and e-mails designed to reveal the IP addresses of operators of pseudonymous accounts. Some individuals who apparently clicked on these links were later arrested, including Ali Al-Shofa (Chapter 1), whose click appears to have been used against him in court. While both campaigns point back to the government, we have not as yet identified overlap between the campaigns. We examine each campaign in turn.

2.3.1 FinSpy Campaign

Beginning in April 2012, the authors received 5 suspicious e-mails from US and UK-based activists and journalists working on Bahrain. In April 2012, Bahraini activists began forwarding us e-mails containing `.rar` attachments, such as the one in Figure 2.1. We received five distinct e-mails from four activists: a UK-based Bahraini journalist, two UK-based Bahraini activists, and the director of the advocacy organization *Americans for Democracy and Human Rights in Bahrain*. Some of the emails appeared to be poorly customized; for instance, the email in Figure 2.1 mentions a nonexistent organization, "Human Rights Bahrain." In reality, Zainab's sister, Maryam, was Acting President of the *Bahrain Center for Human Rights*.

From: Melissa Chan ⟨melissa.aljazeera@gmail.com⟩
Subject: Torture reports on Nabeel Rajab

Acting president Zainab Al Khawaja for Human Rights Bahrain reports on torture on Mr. Nabeel Rajab after his recent arrest.

Please check the attached detailed report along with torture images.

[**Attachment: Rajab.rar**]

Figure 2.1: One of the malicious email messages sent to Bahraini activists.

We found that some of the attachments contained a PE (`.exe`) file designed to appear as an image. These executables used the default image icon for Windows Vista and 7, and their filenames contained a Unicode *right-to-left override* (RLO) character, causing Windows to render a filename such as `gpj.1bajaR.exe` instead as `exe.Rajab1.jpg`. (Figure 2.2).

Another other `.rar` files contained a Word document with an embedded ASCII-encoded PE file containing a custom macro set to automatically run upon document startup. Under

Figure 2.2: A fake image file sent to an activist, as rendered by Windows Vista.

default security settings, Office disables all unsigned macros, so that a user who opens the document will see an informational message that the macro has been disabled, alongside a button to enable macros in the present document.

Identification as FinSpy

We ran the samples in a virtual machine (VM) and found the following string in memory:

```
y:\lsvn_branches\finspyv4.01\finspyv2\
```

This string suggests FinSpy, a product of FinFisher GmbH [57]. The samples were also structurally similar to a demonstration copy of FinSpy [133] found on VirusTotal [198].

We found that the spyware has a modular design, and can download additional modules from a command & control (C&C) server, including password capture (from over 20 applications) and recording of screenshots, Skype chat, file transfers, and input from the computer's microphone and webcam. In one of the modules we found a string that apparently enumerates a range of additional functionality including the ability to recover deleted and printed files, and perform "Forensics Recording." We did not observe these modules in action.

To exfiltrate data back to the C&C server, a module encrypts and writes it to disk in a special folder. The spyware periodically probes this folder for files that match a certain naming convention, then sends them to the C&C server. It then overwrites the files, renames them several times, and deletes them, in an apparent effort to frustrate retrospective forensic analysis.

Analysis of encryption

Because the malware employed myriad known anti-debugging and anti-analysis techniques, it thwarted our attempts to attach debuggers. Since it did not include any anti-VM code, we ran it in TEMU, an x86 emulator designed for malware analysis [187]. TEMU captures instruction-level execution traces and provides support for taint-tracking.

We found that when a module wants to write a new item of data to disk, FinSpy instantiates a *record file* that stores the data in an encrypted, proprietary TLV-based format. At a high level, record files comprise data used to derive a symmetric encryption key, followed by

a number of encrypted *records* containing surveillance data gathered by the module. When FinSpy generates a new record file, it first gathers 256 bytes of data by repeatedly reading memory address `0x7ffe0014` 64 times in a tight loop; in Windows this address contains the low-order-4-bytes of the Windows clock, which contains the number of 100 nanoseconds since January 1, 1601. It uses this to generate an 256-bit key and IV to encrypt the records using AES-256-CBC. FinSpy takes the first 32 bytes of the system clock readings as the AES key, and the subsequent 16 bytes as the IV. The key and IV only consume 48 bytes; we did not identify a purpose for the remainder. The spyware encrypts all 256 bytes using an embedded RSA-2048 public key. Even though we did not find a corresponding private key—it presumably only resides on the C&C server—the weak AES keys make decryption of record files straightforward based on brute-forcing different possible clock readings. We wrote a program that generally can find the key in under an hour.

In our analysis, FinSpy used AES keys like the following, which consists of the same four byte sequence repeated:

```
7031bdcc7031bdcc7031bdcc7031bdcc
7031bdcc7031bdcc7031bdcc7031bdcc
```

We also saw AES keys where the difference between consecutive 4-byte blocks was 0x2625A, the default quantum for updating the system clock [186] (0x2625A hundred-nanoseconds is about 16ms):

```
26e92360804b2660daad286034102b60
8e722d60e8d42f60423732609c993460
```

In addition, we found that FinSpy's AES code failed to encrypt the last block of data if less than the AES block size of 128 bits, leaving trailing plaintext. Finally, FinSpy's wire protocol for C&C communication uses the same type of encryption, and thus is subject to the same brute force attack on AES keys.

While we suspect FinSpy's cryptographic deficiencies reflect bugs, it is also conceivable that the cryptography was deliberately weakened to facilitate one government monitoring the surveillance of others. Security firm Sophos found that the Android mobile version of FinSpy effectively used four-byte AES keys [131].

C&C server

The FinSpy samples communicated with `77.69.140.194`, which belongs to a subscriber of Batelco, Bahrain's main ISP. Ongoing monitoring showed that the server took on three other addresses inside Batelco: `89.148.15.15`, `77.69.181.162`, and `217.17.237.133`. All addresses are registered to Batelco, Bahrain's major Internet Service Provider.

Analyzing network traffic between our infected VM and the C&C server revealed that the server used a *global IPID*, which allowed us to infer server activity by examining gaps in the IPID sequence between subsequent packets, as follows.

The IPID is a field in every IP packet that is used to identify fragments of a packet. As a packet flows across the Internet, it may be fragmented into smaller packets if it is too big to traverse a link between its source and destination all at once. Each packet fragment has the same IPID as the original packet, allowing the fragments to be reassembled into the original packet at the destination. The sender generally does not know which packets will be fragmented *a priori*, so must assign an IPID to almost every packet. Many older operating systems assign IPIDs to every packet using a global sequential method: the first packet sent to any destination has IPID 0, followed by 1, 2, and so on. We call this a *global IPID*. Modern OS versions assign IPIDs randomly.

For our purposes, if a server has a global IPID, then we can use it as a counter for the number of packets that the server has sent to anyone. Furthermore, anyone can probe the server for this value by sending a request (e.g., TCP SYN) to the server, and looking at the IPID value in the response (e.g., SYN/ACK). By probing the IPID value twice, once at time t_1 and once at t_2, one can see if the server sent any packets between t_1 and t_2.

In response to our preliminary work [133], an executive at Gamma told the press that Bahrain's FinSpy server was merely a proxy and the real server could have been anywhere, as part of a claim that the Bahrain FinSpy deployment could have been associated with another government [175]. However, a proxy would show gaps in a global IPID as it forwarded traffic; our frequent observation of strictly consecutive IPIDs at the C&C during communications from our infected VM thus contradicts this statement.

Exploitation of captured data

Since we suspected the spyware operator would likely seek to exploit captured credentials, particularly those associated with Bahraini activist organizations, we created a fake login page on the *Bahrain Watch* website, `bahrainwatch.org`. From a clean VM, we typed a username and password into this page, saving the password in Mozilla Firefox. We then infected the VM with one of the Bahraini FinSpy samples and allowed it to connect to the Bahrain C&C server. Bahrain Watch's website logs revealed a subsequent hit from `89.148.0.41`—made however to the site's homepage, rather than its login page—coming shortly after we had infected the VM. Decrypting packet captures of the spyware's activity, we found that our VM sent the password to the server exactly one minute earlier:

```
INDEX,URL,USERNAME,PASSWORD,USERNAME FIELD,PASSWORD FIELD,FILE,HTTP
1,http://bahrainwatch.org,bhwatch1,watchba7rain,username,password,signons.
sqlite,,Very Strong,3.5/4.x
```

The URL provided to the server did not include the path to the login page, which was inaccessible from the homepage. This omission reflects the fact that the Firefox password

database stores only domain names, not full login page URLs, for each password. Repeating the experiment again yielded a hit from the same IP address within a minute. We inspected the logs for `bahrainwatch.org`, which showed no subsequent (or previous) activity from that address, nor any instances of the same User Agent string.

Subsequent leak

In August 2014, a persona called "Phineas Phisher" released 40GB of data purportedly taken from FinFisher [27]. The data included what appeared to be a C&C server log file sent from a FinFisher client in Bahrain to technical support. The server log file contained details on 77 computers (e.g., computer names, and user account names) that had been successfully infected by the FinFisher client in Bahrain between November 2010 and February 2012. We contacted individuals who we believed were mentioned on the list, and obtained hard drives from three computers listed. In each case, our forensic analysis established a past FinSpy infection consistent with the log file.

We believe we know the identity of the owner of 26 computers listed in the logs. The infected computers appear to belong to 11 lawyers, including 9 lawyers at a single firm, several officials in three opposition political parties, Iran's semi-official Fars news agency, and a prominent Lebanese businessman. We also believe two infected computers belonged to members of the Bahrain Independent Commission of Inquiry [13], its Head Investigator, and its Chief Administrative and Financial Officer. The independent commission was invited by Bahrain's King to write an impartial report into the government's response to the protests.

The identity of the FinFisher client in Bahrain is not mentioned in the leaked data. However, a 2015 email leaked from a FinFisher competitor [70] mentioned that Bahrain's National Security Agency (BNSA) [119] was still experiencing "internal issues with the publicity that the organization used hacking tools on opposition people," suggesting the BNSA as FinFisher's Bahrain customer [200].

2.3.2 IP spy campaign

In an *IP spy* attack, the attacker aims to discover the IP address of a victim, who is typically the operator of a pseudonymous social media or e-mail account.

The attacker sends the pseudonymous account a link to a webpage or an e-mail containing an embedded remote image, using one of many freely-available services. Services we have seen used include `iplogger.org`, `ip-spy.com`, `ipspy.ru`, `whatstheirip.com`, `fuglekos.com/ip-grabber`, `shivampatel.net/trace`, `myiptest.com`, `readnotify.com`, `whoreadme.com`, and `pointofmail.com`.

When a victim clicks on the link or opens the e-mail, their IP address (and browser headers) are revealed to the attacker. Several webmail providers and e-mail clients take limited steps to automatically block loading this content, but e-mails spoofed to come from a trusted sender sometimes bypass these defenses. Once the attacker has the victim's IP, they may then discover the victim's identity from their ISP. In one case we identified legal

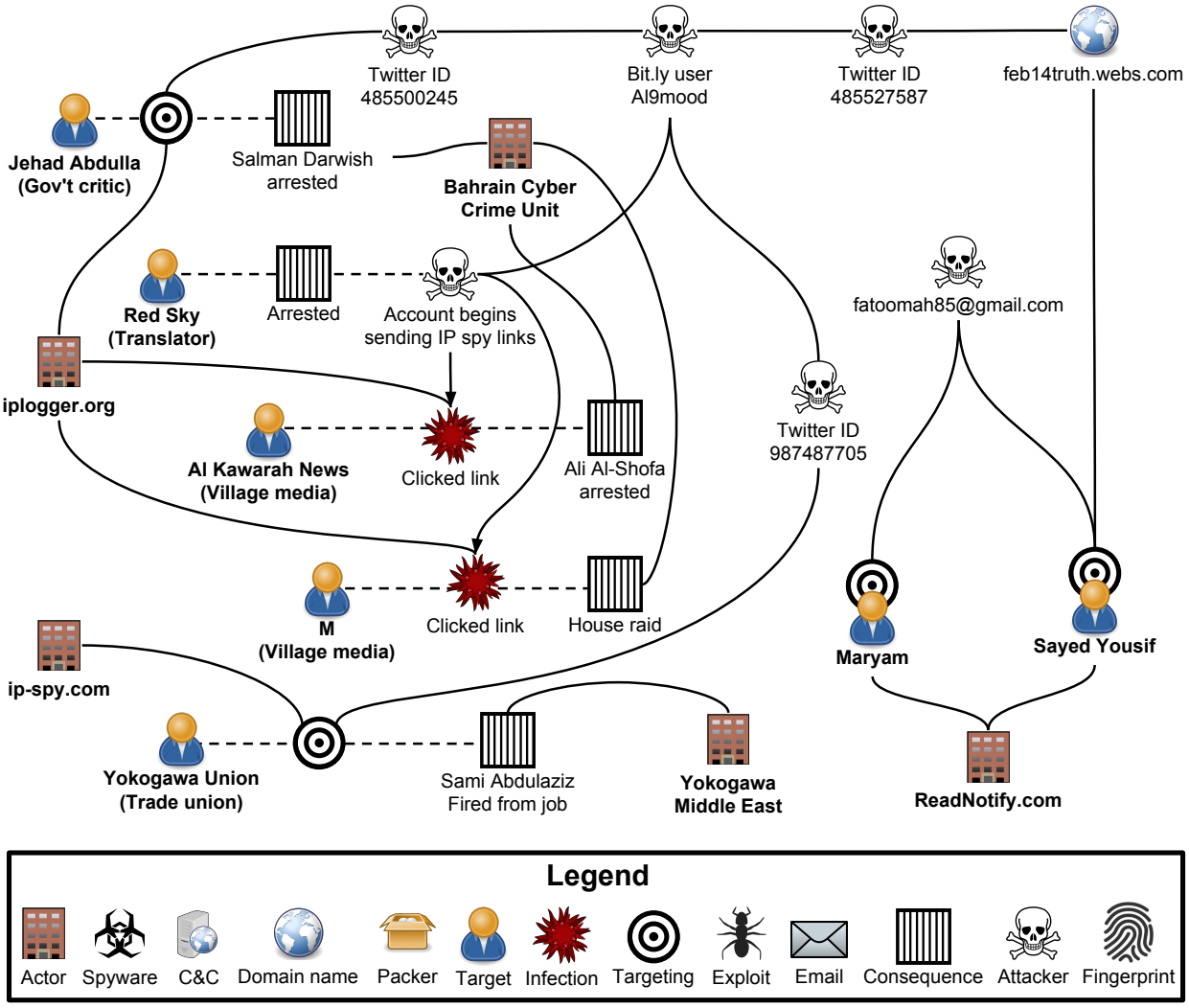


Figure 2.3: The ecosystem of Bahrain "IP spy" attacks.

documents that provided a circumstantial link between such a spy link and a subsequent arrest.

We have also occasionally seen *IP spy* emails sent as a prelude to breaking into a target's account. We speculate that this may be a way for the attacker to get information about a device and location from which a target logs in, so as to make their malicious account access seem plausible if discovered by the target.

Figure 2.3 illustrates the larger ecosystem of these attacks. The attackers appear to represent a single entity, as the activity all connects back to accounts that sent links shortened using a particular user account *al9mood*[1] on the `bit.ly` URL shortening service.

[1]A Romanization of the Arabic word for "steadfastness."

Recall Ali Al-Shofa (discussed in Chapter 1), who was accused of sending insulting tweets from an account `@alkawarahnews` (**Al Kawarah News** in Figure 2.3). An operator of the account forwarded us a suspicious private message (Figure 2.4) sent to the Al Kawarah News Facebook account from **Red Sky**. Red Sky was purportedly arrested on October 17, 2012, was convicted of insulting the King on his Twitter account `@RedSky446`, and was sentenced to four months prison, according to information we received from two Twitter users, one of whom claimed to have met Red Sky in prison; another to be a colleague. When released, Red Sky found that the passwords for his Twitter, Facebook, and email accounts had been changed, and did not know how to recover his accounts.

Figure 2.4: Message that Al Kawarah News Facebook account received from an arrested activist's account.

The message that Red Sky's account sent to Al Kawarah News included a link shortened using Google's `goo.gl` service. We used the `goo.gl` API to access analytics for the link, finding that it unshortened to `iplogger.org/25SX` and was created on:

```
2012-12-08T19:05:36.019+00:00
```

The link had received only one click, which came from Bahrain with the referrer `www.facebook.com`.

Ali's case files contained a request from the Public Prosecution for information on an IP address that it had linked to Al Kawarah News about 22 hours after the link was created (Figure 2.5). Court documents indicate that ISP data linked the IP address to Ali, and on this basis he was sentenced to one year in prison. for insulting tweets sent from the account.

Red Sky also targeted **M** in Figure 2.3. M recalled clicking on a link from Red Sky (Figure 2.6) while using an Internet connection from one of the houses in M's village. The house was raided by police on March 12, 2013, who were looking for the subscriber of the

Bahrain
ECUTION

مملكة البحرين
النيابة العامة

رقم الإشارة:ن ع/ أ4/ ٢٠٥٧/ 2013
التاريخ: 2013/2/20

السيد الفاضل / مدير الإدارة العامة لمكافحة الفساد و الأمن الاقتصادي و الالكتروني المحترم،،،

تحية طيبة وبعد،،،

الموضوع:

طلب الكشف عن مستخدم البرتوكول

@alkawarahnews

عنوان البروتكول	التاريخ	الساعة
89.148[redacted]	2012/12/9	19:57:18

نأذن للملازم اول / فواز حسن الصميم أو لمن يندبه أو يعاونه من مخاطبة شركة مينا تيلكوم البحرين والمزودة بخدمة الانترنت للكشف عن اسم وعنوان صاحب البرتوكول المذكور أعلاه من اجل استكمال التحريات التي تجريها إدارتكم للتوصل للفاعل ومن ثم عمل المحاضر اللازمة وتعرض علينا في حينه لأتخاذ اللازم.

وتفضلوا سعادتكم بقبول وافر التحية والاحترام،،،

محمد صلاح
وكيل النيابة
القائم باعمال رئيس نيابة محافظة العاصمة

Figure 2.5: Court document from Ali's case files showing a request for information about an IP address 22 hours after the account was sent an IP spy link.

house's Internet connection. Police questioning revolved around Tweets that referred to Bahrain's King as a "cursed one." Red Sky had earlier targeted other users with IP spy links shortened using the *al9mood* `bit.ly` account.

Figure 2.6: Message that M received from an arrested activist's account.

The attack on **Jehad Abdulla** (Figure 2.3) is noteworthy, as the account's activity aligned with communities typically critical of Bahrain's opposition. However, the account also directly criticized the King on occasion, in one case referring to him as "weak" and "stingy."

An account linked to *al9mood* sent Jehad Abdulla an IP spy link on October 2, 2012, in a public message. On October 16, 2012, Salman Darwish was arrested for insulting the King using the Jehad Abdulla account. He was sentenced to one month in prison, partly on the basis of his confession. Salman's father claims that police extracted a confession from Salman by denying him food, drink, and medical care.

Another account linked to *al9mood* targeted `@YLUBH`, the Twitter account of **Yokogawa Union**, a trade union at the Bahraini branch of a Japanese company. `@YLUBH` received at least three IP spy links in late 2012, sent via public Twitter messages. Yokogawa fired the leader of the trade union, Sami Abdulaziz Hassan, on March 23, 2013 [161]. It later emerged that Sami was indeed the operator of the `@YLUBH` account, and that the police had called him in for questioning in relation to its tweets [191]. Yokogawa says that the reason for his sacking is that he failed to inform the company of a police investigation into him and other members of the union, for allegedly defaming the company. The Ministry of Interior says that it launched an investigation in relation to a complaint filed by Yokogawa [192].

Use of embedded remote images

We identified several targets who received spoofed e-mails containing embedded remote images. Figure 2.3 shows two such cases, **Maryam** and **Sayed Yousif**. The attacker sent the e-mails using `readnotify.com`, which records the user's IP address upon their mail client downloading the remote image. The iPhone mail client automatically loaded these remote images when we checked. We found that Yahoo Mail loaded the images if the email was spoofed to come from a sender trusted by the target.

While `readnotify.com` forbids spoofing in their TOS, the service has a vulnerability known to the attackers (and which we confirmed) that allows spoofing the `From` address by directly setting the parameters on a submission form on their website. We have not found evidence suggesting this vulnerability is publicly known, but it appears clear that the attacker exploited it, as the web form adds a `X-Mai1er: RNwebmail` header not added

Figure 2.7: Public messages from an IP spy attacker posing as an ex-employee of Yokogawa.

when sending through `ReadNotify.com`'s other supported methods. The header appeared in each e-mail the targets forwarded to us.

When spoofing using this method, the original sender address still appears in `X-Sender` and other headers. According to these, the e-mails received by the targets all came from `fatoomah85@gmail.com`. A link sent in one of these e-mails was connected to the *al9mood* `bit.ly` account.

Other IP spy link attacks

In monitoring accounts connected to *al9mood*, we counted more than 200 IP spy links in Twitter messages and public Facebook posts. Attackers often used (1) accounts of prominent or trusted but jailed individuals like "Red Sky," (2) fake personas (e.g., attractive women or fake job seekers when targeting a labor union), or (3) impersonations of legitimate accounts. In one particularly clever tactic, attackers exploited Twitter's default font, for example substituting a lowercase "`l`" with an uppercase "`I`" or switching vowels (e.g. from "`a`" to an "`e`") to create at-a-glance identical usernames. In addition, malicious accounts tended to quickly delete IP spy tweets sent via (public) mentions, and frequently change profile names. We were able to track these accounts across renaming by the attacker because renaming does

not alter an account's numerical user ID.

We observed several interesting tactics in public IP spy targeting. One tactic involved impersonating a participant in a public Twitter conversation. In Figure 2.8, `@saudi44` and `AlBinSanad` are having a public Twitter conversation, however the final message shown is from an impersonation account `AIBinSanad`—note the capital `I` (underlined) substituted for the lowercase `L`—and contains an IP spy link.

Figure 2.8: An IP spy account impersonates one member of a public Twitter conversation.

Another tactic involved sending two public Twitter messages, each mentioning several different users (so as not to appear targeted), with only a single (target) user in common between both messages. Assuming the target clicked on both links, the IP address in common between both links would likely be the target's IP. We show an example in Figure 2.9; the links to `richtweets.com` both had invisible embedded IP spy images.

Attribution to Bahrain Cyber Crime Unit

One IP spy account `@sabreeena30`, had an eponymous associated Facebook account. The Facebook account shared [5] the first and only post on a blog entitled "Cyber Crime Unit." The publication timestamp on the blog post was exactly the same time that `sabreeena30` shared the post on Facebook, suggesting that the author of the blog is also the operator of `sabreeena30`. The blog post was published by a user called "Usman Fazal," whose Blogger profile claimed he was a "Computer Forensic Examiner" working for the "military." A LinkedIn profile under the same name said that he worked at the Ministry of Interior's Cyber Crime Unit.

Figure 2.9: IP spy accounts send apparently nontargeted links to discover a target's IP address.

2.4 UAE

Unlike Bahrain, the UAE has experienced no recent uprising or political unrest. However, the government has nevertheless cracked down on dissent, concurrent with the Arab Spring.

The attacks we first identified in the UAE context involved *Remote Control System (RCS)*, another "lawful intercept" spyware system, sold by Italian company Hacking Team [75]. Over time, we stopped receiving RCS samples from UAE targets, and instead observed a shift to the use of off-the-shelf RATs, and possible involvement of cyber-mercenary groups. However, poor attacker operational security allowed us to link most observed attacks together.

2.4.1 Hacking Team RCS

UAE activist Ahmed **Mansoor** (lower-left of Figure 2.10), imprisoned from April to November 2011 after signing an online pro-democracy petition [4], received an e-mail purportedly from "Arabic Wikileaks" in July 2012 (red elements in Figure 2.10). Believing that a Microsoft Word document could not infect him, He opened the associated attachment, *veryimportant.doc*, and saw what he described as "scrambled letters". He forwarded us the e-mail

for investigation. Mansoor is an internationally recognized human rights defender, based in the United Arab Emirates (UAE), and recipient of the Martin Ennals Award [12].

The attachment exploited CVE-2010-3333 [32], an RTF parsing vulnerability in Microsoft Office. The document did not contain any bait content, and part of the malformed RTF that triggered the exploit was displayed in the document. The exploit loaded shellcode that downloaded a second shellcode stage from `ar-24.com`, which in turn downloaded a spyware payload from the same website. We denote this combination as the **HT Exploit Kit** in Figure 2.10.

The C&C server for the spyware also ran on `ar-24.com`. When we obtained the sample in July 2012, `ar-24.com` resolved to an IP address on Linode, a hosting provider. Three months later, it resolved to a UAE address belonging to the Royal Group [164], an organization linked to the UAE government; it is chaired by Sheikh Tahnoon bin Zayed Al-Nayhan, a member of the UAE ruling family and a son of the founder of the UAE.

Identification as RCS

Strings in memory of a computer infected with the RCS sample sent to Mansoor matched those in a Symantec analysis [101] of RCS (also known as *DaVinci* or *Crisis*), a product of the Italian company Hacking Team [75]. A structurally similar Word document on VirusTotal used the same exploit and attempted to download a second stage from `rcs-demo.hackingteam.it`, which was unavailable at the time of testing [132].

Hacking Team files subsequently leaked in July 2015 [70] revealed that the C&C server `ar-24.com` belonged to a Hacking Team client in the UAE alternately referred to as the *UAE Air Force* [66] and *UAE Intelligence* [181]. The technical point of contact for the client appears to have been PAL Group, itself a member of Royal Group.

Analysis

RCS has a suite of functionality largely similar to FinSpy. We observed some differences in installation vectors, namely the use of exploits. While FinFisher claims to provide an *exploit portal* [53], we have not seen a sample of FinSpy distributed in this way. The RCS sample sent to Mansoor used a Word Document exploit; we located other samples embedded in `.jar` files that install an OS-appropriate version of RCS (Windows or OSX), optionally using an exploit. Other samples used Adobe Flash exploits embedded in Microsoft Office documents.

We also noticed some architectural differences between the FinSpy and RCS payloads and communications protocols. In contrast with FinSpy's proprietary TLV-based binary protocol, RCS used HTTP to communicate with the C&C server. While FinSpy's initial payload was designed to download further modules, the first versions of RCS we analyzed appeared to be *monolithic*. Later versions of RCS we analyzed had reduced functionality in the initial payload.

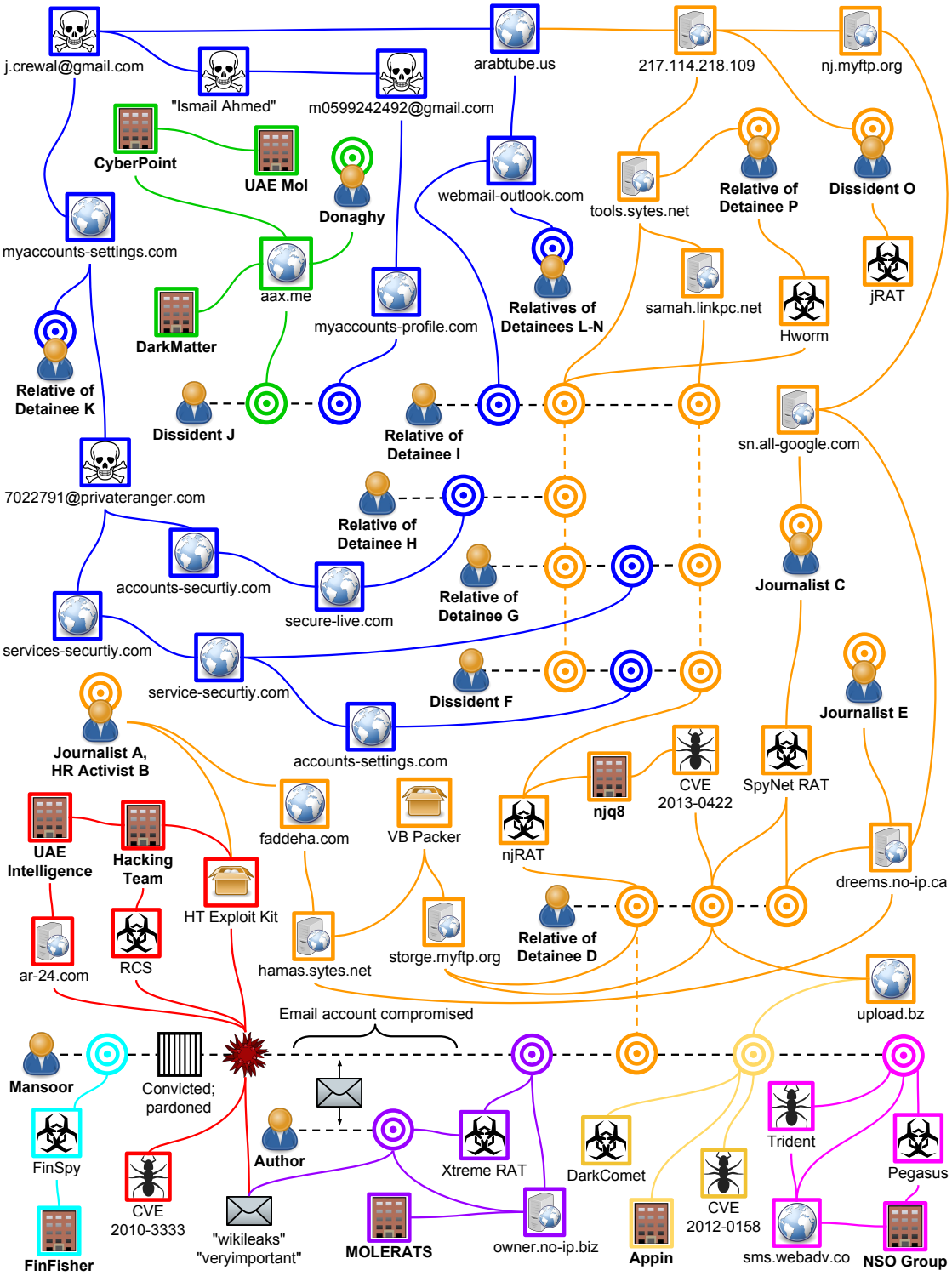


Figure 2.10: Ecosystem of UAE surveillance attacks.

Exploitation of captured data

When Mansoor received the RCS document, he opened it, infecting his computer (Figure 2.10). Mansoor subsequently noted several suspicious accesses to his GMail account using IMAP. Even after he changed his password, the accesses continued. While corresponding with Mansoor on his compromised account, an author of this paper discovered that the attackers had installed an *application-specific password* [69] in Mansoor's GMail account, a secondary password that they apparently used to access his account even after he changed his main password. The suspicious accesses stopped after removal of the application-specific password.

Two weeks after this correspondence with Mansoor, we (**Author** in Figure 2.10) received a targeted e-mail with a link to a file hosted on Google Docs containing a commercial off-the-shelf RAT, Xtreme RAT (purple elements in Figure 2.10). The e-mail was sent from the UAE's timezone (as well as of other countries) and contained the terms "veryimportant" and "wikileaks," just like in the e-mail received by Ahmed.

The instance of Xtreme RAT sent to **Author** used `owner.no-ip.biz` for its C&C, one of the domains mentioned in a report published by Norman about a year-long campaign of cyberattacks on Israeli and Palestinian targets carried out by a group that Norman called **MOLERATS**. They were unable to identify who was behind the group [49]. Three months after **Author** was targeted, Mansoor received an e-mail containing an attachment with Xtreme RAT that communicated with the same C&C server (Figure 2.10), suggesting that the attackers who infected Ahmed with RCS may have provided a list of interesting e-mail addresses to another group for further targeting.

Possible consequences

Shortly after Mansoor was targeted, he says he was physically assaulted twice by an attacker who appeared able to track Mansoor's location [176]. He also reports that his car was stolen, a large sum of money disappeared from his bank account, and his passport was confiscated [85]. He believes these consequences are part of a government intimidation campaign against him, but we did not uncover any direct links to his infection.

2.4.2 Further attacks by a related adversary?

As UAE dissidents continued to forward us suspicious messages they received, we noticed a shift away from Hacking Team, to off-the-shelf RATs (orange elements in Figure 2.10).

In October 2012, UAE **Journalist A** and **H**uman **R**ights **activist B** (per Figure 2.10) forwarded us suspicious e-mails they had received that contained a Word document corresponding to the first stage of the **HT Exploit Kit**. The attachment contained an embedded Flash file that exploited a vulnerability fixed in Adobe Flash 11.4, loading shell code to download a second stage from `faddeha.com`. We were unable to obtain the second stage or the ultimate payload, as the website was unavailable at the time of testing. However, the ex-

ploit kit appears indicative of Hacking Team involvement. A page on `faddeha.com` found in Google's cache contained an embedded `.jar` with the same applet class "WebEnhancer" as those observed in other `.jar` files that we found to contain RCS.

Off-the-shelf RATs

We found a file that VirusTotal had downloaded from `faddeha.com`, which appeared to be a RAT known as *SpyNet*, available for general purchase for 50 Euros [179]. The SpyNet sample communicated with the C&C `hamas.sytes.net`.

We found another instance of the first stage of the **HT Exploit Kit** on VirusTotal. The exploit downloaded a second stage, which in turn downloaded a sample of SpyNet from `maile-s.com`. This sample of SpyNet communicated with the same C&C `hamas.sytes.net`. In one leaked email from Hacking Team, the company's Operations Manager says that the UAE client that hacked Mansoor also uses SpyNet and Xtreme RAT for some operations [140].

The SpyNet sample we found was packed using *ASProtect* [11]. When run, the sample unpacks a compiled Visual Basic project that loads, via the *RunPE* technique [193], an executable packed with *UPX* [190]. Finally, this executable unpacks SpyNet. SpyNet's GUI only offers an option to pack with UPX, suggesting that the attackers specially added the other layers of packing. In some cases, the Visual Basic project bears the name *NoWayTech*, which appears to be an underground RunPE tool, while others are named *SpyVisual*, which we have been unable to trace to any public underground tools, and thus also may reflect customization by the attacker. The *SpyVisual* projects contain the following string:

```
c:\Users\Zain\AppData\Local\Temp\OLE1EmbedStrm.wav
```

We used this string as the fingerprint **VB Packer** in Figure 2.10.

The same **VB Packer** was used in an attack on **Relative of Detainee D** and Mansoor (Figure 2.10). These individuals received e-mails containing a link to a web page hosted on `cedarkeyrv.com` impersonating YouTube. Loading the page greeted the target with "*Video loading please wait ...*" The page redirected to a YouTube video a few seconds later, but first loaded a Java exploit, CVE-2013-0422 [33]—a known vulnerability with no patch at the time that the e-mails were sent. Oracle released a patch 12 hours after activists began receiving these links.

The `cedarkeyrv.com` domain is associated with an RV park in Cedar Key, Florida. The website's hosting company told us that the site had apparently suffered a compromise, but did not have further details.

The exact CVE-2013-0422 exploit used in the attack appears to have been originally posted by a Kuwaiti user, *njq8*, on an Arabic-language exploit sharing site [148]. We contacted *njq8*, who told us that he had obtained the exploit elsewhere and modified it prior to posting. The attack used this exploit to download an instance of SpyNet from `isteeler.com`

(which from our inspection did not appear to have any legitimate content), which used the C&C `storge.myftp.org`. This same C&C occurred in another attack (Figure 2.10) targeting **Relative of Detainee D**; in that case, the payload was a freely-available RAT known as *njRAT*, written by the same *njq8* as the exploit-poster discussed above. However, we did not find any other evidence suggesting *njq8*'s involvement in either attack.

The domain `hamas.sytes.net`, which we previously saw used by two SpyNet samples, resolved to `67.205.79.177`. Historically, `dreems.no-ip.ca` also resolved to this address. An unidentified dropper using this C&C targeted **Journalist E**; a SpyNet attack on **Relative of Detainee D** also used this C&C. In that latter case, the sample arrived via e-mail in a `.rar` attachment that contained an `.scr` file disguised as a Word document. The `.scr` file was a self-extracting archive that decompressed and ran both the bait document and the payload. The SMTP source of the e-mail was `webmail.upload.bz`.

In early 2013, Mansoor forwarded numerous documents he received that included a particular flavor of CVE-2012-0158 exploit for Microsoft Word. In all, these totaled 17 distinct hashes of documents, and 10 distinct hashes of payloads (some documents that differed in their hash downloaded the same payload). The exploits primarily downloaded instances of SpyNet from `upload.bz`, which for the most part communicated with C&C at `sn.all-google.com`. This domain, which shared the same address as `dreems.no-ip.ca`, was also used for C&C in other attacks, including that on **Journalist C**.

Two of the other CVE-2012-0158 exploits received by Mansoor downloaded DarkComet from `www.getmedia.us` and `www.technopenta.com` after posting system information to `random123.site11.com`. All three domains match those used by an Indian cyber-mercenary group said to be linked to Appin Security Group [8] (yellow elements in Figure 2.10), profiled in the Hangover report [50]. The former two domains hosted content other than spyware (i.e., they may have been compromised).

The domain `sn.all-google.com` shared an IP address with `nj.myftp.org`, which also resolved to `217.114.218.109`. **Dissident O** was targeted several times in one day with a sample of *jRAT*, a Java-based RAT available for $29 [100], that communicated with this address. In one instance, **Dissident O** received the spyware as a `.jar` file attached to an email from "Dr. Barry Stevenson," who claimed to be looking for a "trustworthy and reliable person" with whom to start a business venture. Bizarrely, the name "Dr. Barry Stevenson," and the text from the email appeared to be modified from an example of an email scam published as part of an anti-spam awareness campaign by New Zealand's Electronic Messaging Compliance Unit one year earlier [92].

Relative of Detainee P was targeted with a sample of *Hworm* [77], a RAT written in Visual Basic Script, communicating with `tools.sytes.net`, which also resolved to `217.114.218.109`. **Relative of Detainee I** also received an Hworm sample, using the same C&C server. The same attack also targeted **Relative of Detainee H** and **G**, as well as **Dissident F**. **Relative of Detainee I**, and **G**, and **Dissident F**, additionally received a sample of njRAT using the C&C server `samah.linkpc.net`.

Phishing Attacks

We identified 9 individuals targeted with credential phishing, apparently by the same attackers. The phishing attacks we saw (blue elements in Figure 2.10) involved attackers registering potentially misleading domain names—often with typos—and sending emails designed to appear to be alerts about suspicious activity from Facebook, Google, and Hotmail. The emails contained links to the misleading domains, which returned pages designed to appear identical to the legitimate login pages for Facebook, Google, and Hotmail. Attackers received any credentials supplied to the phishing pages. Several of these misleading domains were registered by an "Ismail Ahmed," in the UAE.

Attackers used one such domain, `webmail-outlook.com`, to target **Relatives of Detainees L-N**, and **I**, with a Hotmail phishing page. This domain used `ns1.arabtube.us` as its name server; we found ten spyware samples hosted on `arabtube.us`, all of which communicated with a C&C server we previously identified, `217.114.218.109`. The `SOA` record of `arabtube.us` listed the email address `j.crewal@gmail.com`.

The `SOA` record of `myaccounts-settings.com` listed the same email; attackers used that domain to target another detainee's relative, **Relative of Detainee K**, who received an SMS message apparently from Google with a login verification code. The target then received the subsequent SMS in Figure 2.11, containing a link to `myaccounts-settings.com`.

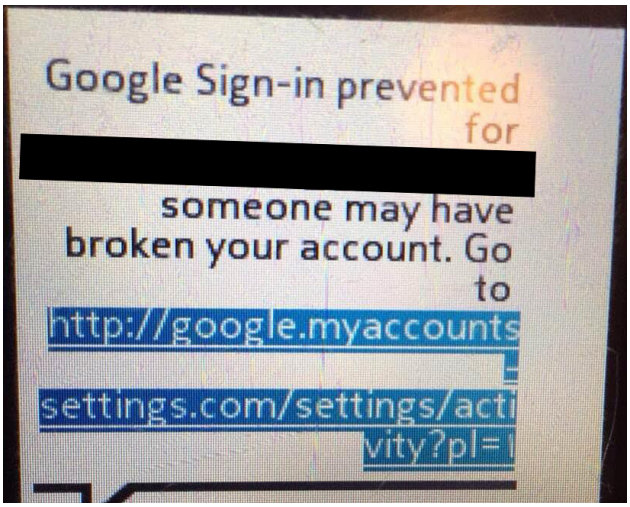


Figure 2.11: UAE phishing SMS.

Relative of Detainee H, **G**, and **Dissident F** were also targeted with phishing emails, involving the domains `secure-live.com`, `service-securtiy.com`, and `account-settings.com` respectively.

We observed a connection between the domains received by **Relative of Detainee G** and **Dissident F**: `account-settings.com` redirected to `service-securtiy.com`.[2] An-

[2]The typo "securtiy" appears in the domain name.

other domain name, `security-settings.com`, registered using the pseudonymous email `7022791@privateranger.com`, also redirected to the **Dissident F** domain. This same email address appeared as the registrant for `myaccounts-settings.com` (linked to `j.crewal@gmail.com` through its SOA record). Similarly, another domain with registrant `7022791@privateranger.com` redirected to `secure-live.com`, the phishing received by **Relative of Detainee H**.

The most recent phishing attack we observed occurred against **Dissident J** in September 2016, and involved the domain `myaccounts-profile.com`. The domain's SOA record contained the email `m0599242492@gmail.com`. We found a public account on Arabic crowdsourcing website `mostaql.com` for the alias `m0599242492`, listing the individual's name as "Ismail Ahmed," based in the UAE, which matched registrant information associated with the email `j.crewal@gmail.com`.

2.4.3 Stealth Falcon

While all of the UAE attacks discussed thus far appear to have been conducted by closely-linked attackers, we also identified a different campaign of attacks which showed more stealth, more advanced operational security, and spyware custom-written by the attackers. These attacks centered around a fake URL shortener `aax.me`, run by a group we call *Stealth Falcon* (green elements in Figure 2.10).

Both Rori **Donaghy** and **Dissident J** were among the dozens of journalists, activists, and dissidents we identified as targeted by Stealth Falcon between 2012 and 2016. We discovered this campaign when an individual purporting to be from an apparently fictitious organization called "The Right to Fight" contacted Rori Donaghy. Donaghy, a UK-based journalist and founder of the Emirates Center for Human Rights, received a spyware-laden email in November 2015, purporting to offer him a position on a human rights panel. Donaghy has written critically of the United Arab Emirates (UAE) government in the past [39], and has published a series of articles based on leaked emails involving members of the UAE government [42, 41, 43].

Circumstantial evidence suggests a link between Stealth Falcon and the UAE government. We traced digital artifacts used in this campaign to links sent from an activist's Twitter account in December 2012, a period when it appears to have been under government control. We also identified other bait content employed by this threat actor. We found 31 public tweets sent by Stealth Falcon, 30 of which were directly targeted at one of 27 victims. Of the 27 targets, 24 were obviously linked to the UAE, based on their profile information (e.g., photos, "UAE" in account name, location), and at least six targets appeared to be operated by people who were arrested, sought for arrest, or convicted in absentia by the UAE government, in relation to their Twitter activity.

The attack on Donaghy — and the Twitter attacks — involved a malicious URL shortening site. When a user clicks on a URL shortened by Stealth Falcon operators, the site profiles the software on a user's computer, perhaps for future exploitation, before redirecting the user to a benign website containing bait content. In Section 3.4 we describe our efforts

to characterize the scope of Stealth Falcon, by exhaustively querying the URL shortener, and scanning for the C&C servers used in the attack.

The November 2015 Attack: An "Invitation"

We describe the attack on Donaghy in detail, as it involved an unusual feature: at our direction, Donaghy communicated with the attacker and received further malicious content.

As mentioned above, in November 2015, Donaghy received an email message (Figure 2.12), purportedly offering him a position on a panel of human rights experts.

From: the_right_to_fight@openmailbox.org
Subject: Current Situation of Human Rights in the Middle East

Mr. Donaghy,

We are currently organizing a panel of experts on Human Rights in the Middle East.

We would like to formally invite you to apply to be a member of the panel by responding to this email.

You should include your thoughts and opinions in response to the following article about what more David Cameron can be doing to help aid the Middle East.

`http://aax.me/d0dde`

Thank you.

We look forward to hearing back from you,

Human Rights: The Right to Fight

Figure 2.12: Email message sent to Donaghy, purportedly offering him a position on a panel of human rights experts.

Donaghy was suspicious of the email, and forwarded it to us for analysis. We found that the link in the email (`http://aax.me/d0dde`) loaded a page containing a redirect to the website of Al Jazeera. Before completing the redirect, it invoked JavaScript to profile the target's computer.

We were unfamiliar with the website `aax.me`, so we investigated it further. We found that the main page of aax.me (Figure 2.13) purported to be a public URL shortening service, powered by YOURLS [172], an open source PHP framework allowing anyone to set up their

own URL shortening service. We are unable to ascertain whether the site actually uses any YOURLS code. We also noted that the homepage contains a typo ("Shortend [sic] URL").

Figure 2.13: Homepage of aax.me.

We shortened a long URL using the homepage, but found that clicking on the shortened URL redirected to our long URL without triggering the loading of `http://aax.me/redirect.php`. We also did not find the code for redirect.php or redirect.js in the public code repository for YOURLS [174]. Thus, we deduced that this code was likely specially written by the operators, and the link sent to Donaghy was likely created by someone with administrator access to aax.me.

On our advice, Donaghy responded to the initial email he received, asking for further information. The attackers responded (Figure 2.14).

By chance, the attachment sent by attackers was identified as malicious and blocked by a program running in Donaghy's email account. We advised Donaghy to request that the attackers forward the attachment via another method, and he received the reply in Figure 2.15.

This second link (`http://aax.me/a6faa`) redirects to `https://cloud.openmailbox.org/index.php/s/ujDNWMmg8pdG3AL/authenticate`. This is a password-protected link to a file shared on an ownCloud [153] instance. We obtained this file (Figure 2.16), and found it to be a Microsoft Word document.

The Malicious Document

When opened, the target is greeted with an image (Figure 2.17), purporting to be a message from "proofpoint," a legitimate provider of security solutions for Office 365 [156]. The image claims that "This Document Is Secured" and requests that the user "Please enable macros to continue."

If the target enables macros, the document changes to reveal text (Figure 2.18). The document purports to be from an organization called "The Right To Fight," and asks the target Donaghy to open the link in the original email he received (the email containing the profiling URL). We believe that "The Right To Fight" is a fictitious organization, as

From: the_right_to_fight@openmailbox.org
Subject: RE: Current Situation of Human Rights in the Middle East

Mr. Donaghy,

Thank you for getting back to us. We are very interested in you joining our panel.

The information you requested is in the attached document.

In order to protect the content of the attachment we had to add macro enabled security.

Please enable macros in order to read the provided information about our organization.

We hope you will consider joining us.

Thank you.

We look forward to hearing back from you,

Human Rights: The Right to Fight

Figure 2.14: Email message sent to Donaghy, in response to his request for further information.

their logo appears to be copied from an exhibition about "African American Experiences in WWII" [188]. Further, "The Right to Fight" has no discernable web presence.

Profiling: The document attempts to execute code on the recipient's computer, using a macro. The macro gathers system information via Windows Management Instrumentation (WMI), and attempts to determine the installed version of .NET.

Communication and Obtaining a Shell: The macro returns gathered information to `http://adhostingcache.com/ehhe/eh4g4/adcache.txt`, and executes the server's response as a PowerShell command. At the time, `adhostingcache.com` resolved to `95.215.44.37`. Attackers apparently deleted the domain on November 30, 2015 (Donaghy received the malicious Word Document on November 24, 2015), and on December 3rd, registered a new domain (`adhostingcaches.com`) that resolved to the same IP. The deletion of `adhostingcache.com` may reflect attacker suspicion that the file received by Donaghy had been sent to security researchers.

The server's response decodes and materializes an invocation of a Base64-encoded PowerShell command to disk as `IEWebCache.vbs`, and creates a scheduled task entitled "IE Web Cache" that executes the file hourly.

`IEWebCache.vbs` runs a Base64-encoded PowerShell command, which periodically and

From: the_right_to_fight@openmailbox.org
Subject: RE: Current Situation of Human Rights in the Middle East

Mr. Donaghy,

We apologize for having problems with our attachment.

Please follow this link to download our organizational information.

`http://aax.me/a6faa`

The link has been password protected. The password is: right2fight

In order to protect the content of the attachment we also had to add macro enabled security.

Please enable macros in order to read the provided information about our organization.

We hope you will consider joining us.

Thank you.

We look forward to hearing back from you,

Human Rights: The Right to Fight

Figure 2.15: Email message sent to Donaghy, in response to his statement about problems with the earlier attachment.

```
Filename: right2fight.docm
MD5:      80e8ef78b9e28015cde4205aaa65da97
SHA1:     f25466e4820404c817eaf75818b7177891735886
SHA256:   5a372b45285fe6f3df3ba277ee2de55d4a30fc8ef05de729cf464103632db40f
```

Figure 2.16: Malicious document sent to Donaghy.

Figure 2.17: Fake Proofpoint image in the malicious document sent to Donaghy. The image includes an Arabic translation below the English text.

via an `HTTPS POST`, sends a unique identifier to `https://incapsulawebcache.com/cache/cache.nfo`. The script executes server responses as PowerShell commands, responding back to the server with the exit status of, output of, or any exceptions generated by the commands. This gives the attacker control over the victim's computer, and allows the attacker to install additional spyware or perform other activities.

We were not able to obtain any additional commands in Donaghy's case. However, a structurally similar document was uploaded to VirusTotal several months later in May 2016. In that case, we received a bundle of 7 commands from the stage2 C&C server. The commands perform the following actions:

1. Gathers system information from WMI.
2. Gathers the ARP table.
3. Gathers a list of running processes.
4. Materializes a file `OracleJavaUpdater.ps1` to disk. This file gathers passwords and web browser data from a variety of sources: Windows Credential Vault, Internet Explorer, Firefox, Chrome, Outlook. In general, the file appears to custom code written

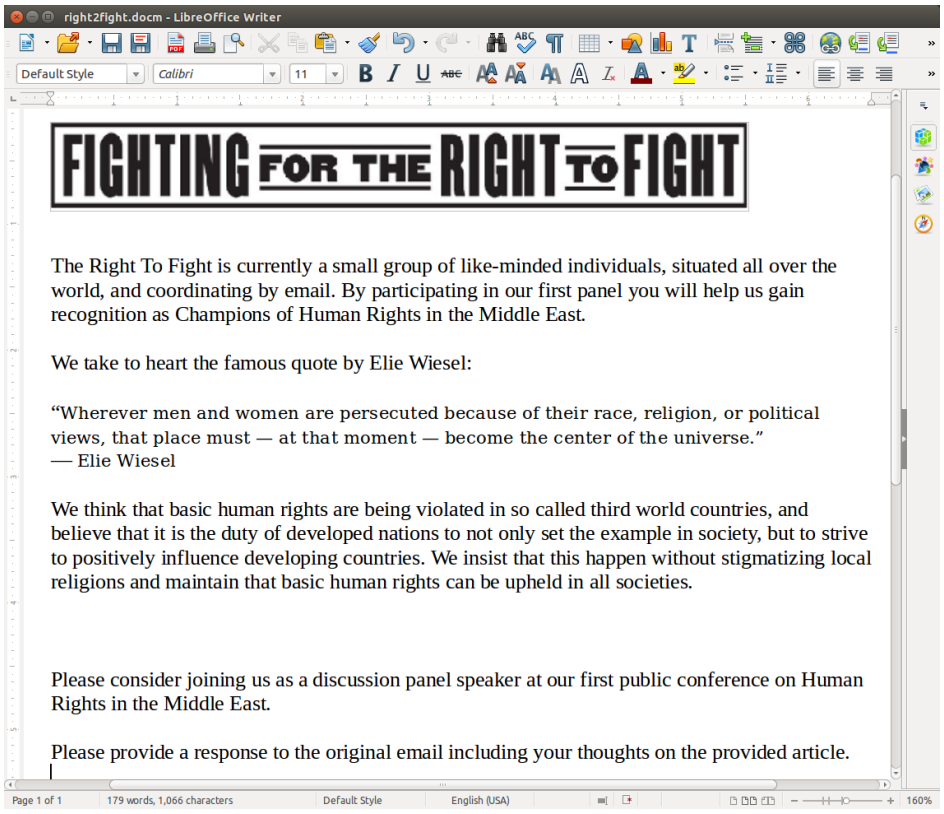


Figure 2.18: Document that Donaghy would have seen, had he enabled macros.

by attackers, though some routines are copied from other sources (e.g., some Internet Explorer password gathering code appears to be lifted from the GPLv3-licensed QuasarRAT [157]).

5. Executes `OracleJavaUpdater.ps1`.

6. Deletes `OracleJavaUpdater.ps1`.

7. Gathers a list of running processes again.

Technical Analysis: `aax.me` Browser Profiling

While `aax.me` has a public interface where anyone may shorten a link, `aax.me` only conducts browser profiling of individuals who click on links that are specially shortened by Stealth Falcon attackers.

When we accessed the link in the first email that Donaghy received, `http://aax.me/d0dde`, the server responded with the following page:

The page is apparently designed to redirect to an Al Jazeera op-ed [80] after twenty seconds. However, the URL is incorrect: the last character of the filename should be a “1”

```
<iframe src='redirect.php' height='1' width='1' border='0' scrolling='no'
frameborder='0' unselectable='yes' marginheight='0' marginwidth='0' onload='
setTimeout("document.location =\"http://www.aljazeera.com/indepth/opinion
/2015/11/british-pm-middle-east-human-rights-151103070038237.html\"", 20000)
'></iframe><br><br><br><center><img src='loading.gif'><br>Loading the website
:<br><b>http://www.aljazeera.com/indepth/opinion/2015/11/british-pm-middle-
east-human-rights-151103070038237.html</b>.<br>This may take a few seconds.</
center>
```

Figure 2.19: Redirect code returned by aax.me profiling page.

instead of a "7." Therefore, an Al Jazeera 404 page is returned instead of the op-ed. It is possible that the use of "7" instead of "1" represents a transcription error on the part of the attackers. When we accessed this same `aax.me` URL in March 2016, it redirected directly to the Al Jazeera URL (with typo) via an HTTP 302 redirect.

The iframe (Figure 2.19), `http://aax.me/redirect.php`, reloads itself with a parameter `inFr` in its query string, to indicate whether the page has been opened up inside a frame (Figure 2.20).

```
<html><body><script type="text/javascript">if(window!=window.top){inFr="1"}
else{inFr="0"}document.location=document.location+"?inFr="+inFr;</script><
noscript><img src="?jSc=0"/></noscript></body></html>
```

Figure 2.20: The redirect.php file on aax.me.

If the page is opened inside a frame (`inFr=1`), as is the case here, then the page in Figure 2.21 returned.[3]

```
<html><head></head><body><div id='display' height='1' style='display:none;'></
div><form id='statsPost' action='?stats=1' method='POST'><input type="hidden"
name="PHPSESSID" value="" /><input id='theData' name='theData' type='hidden'
value='' /></form><script type='text/javascript' src='redirect.js'></script></
body></html>
```

Figure 2.21: The redirect.php file on aax.me when loaded with inFr=1.

We examined the referenced JavaScript file, `http://aax.me/redirect.js`. The file is designed to profile a user's system, perhaps to gather intelligence about potentially exploitable vulnerabilities. The file has apparently not been updated since 7 May 2013,[4] rendering some of the probing obsolete.

[3] We ommit the `value` parameter of `PHPSESSID`.

[4] Based on the *last-modified* HTTP header.

The profiling script does the following:

- For Internet Explorer, it attempts to create several instances of ActiveXObject to get the versions of Flash, Shockwave, Java, RealPlayer, Windows Media Player, and Microsoft Office (classified as either 2003, 2007, or 2010).
- For non-Internet Explorer browsers, it attempts to get a list of enabled plugins from navigator.mimeTypes.
- For all browsers, it captures the user agent, whether cookies are enabled, the OS, the size of the browser window, and the timezone. It classifies browsers into different versions, denoted by letters, based on the existence and behavior of certain JavaScript methods.
- The script attempts to exploit an information leak in older versions of Tor Browser.
- For Windows browsers (except Opera, and versions of Internet Explorer before IE9), it sends a series of XMLHttpRequests to 127.0.0.1, which we believe are designed to deduce if the computer is running any one of several specific antivirus programs. The code for this appears to be borrowed from the JS-Recon port scanning tool [110]. The creator of JS-Recon presented the tool at BlackHat Abu Dhabi in 2010 [107].

Technical Analysis: aax.me Tor Deanonymization Attempt

The aax.me site appears to attempt to deanonymize users of Tor Browser. While the technique the attackers used was out-of-date at the time we observed the attack, the attempted Tor deanonymization speaks to their motivations and potential targets.

The script first detects Tor Browsers by checking whether `navigator.buildID` is set to zero. Versions of Tor Browser before 2.3.25-12 (released on 13 August 2013) had their buildID set to zero.[5] This behavior was originally introduced in TorButton, a Firefox extension to be used in conjunction with Tor, before the introduction of Tor Browser, in support of the goal of making Tor users appear homogeneous. Newer Tor Browser versions have navigator.buildID set to a different distinctive value, `20000101000000`.

When the script detects a Tor Browser, it attempts to deduce the version of Tor Browser by checking for the existence and behavior of certain JavaScript methods. Once a browser is determined to be *older* than a certain version of Tor Browser, the script exploits a now-fixed bug to get the disk path of the browser installation [10]. The disk path may contain the target's username, which may include the target's real name.

The bug in Tor Browser was first disclosed at Defcon 17, which took place in August 2009 [58]. The bug was first fixed on on 25 May 2012 in Tor Browser release 2.2.35-13. [168] The bug was, however, later reintroduced into Tor Browser on 18 December 2013 with the release of Tor Browser 3.5, and subsequently fixed again in Tor Browser 3.6 on 29 April

[5] All of our testing was conducted on English, Windows builds of Tor Browser.

2014. [139] However, unfortunately for the attackers, they failed to update their profiling script to reflect Tor Browser's `navigator.buildID` change (before the bug was reintroduced). Thus, the profiling script did not detect Tor Browsers with the reintroduced bug as Tor Browsers, so it did not try to exploit them.

The version of Tor Browser (as determined by JavaScript checks) is submitted back to the server, along with the value of `navigator.oscpu` which is set to "Windows NT 6.1" in the latest Tor browser, `navigator.vendor` (which appears blank in the latest Tor Browser), and any data gathered about the installation path.

Technical Analysis: aax.me Antivirus Profiling

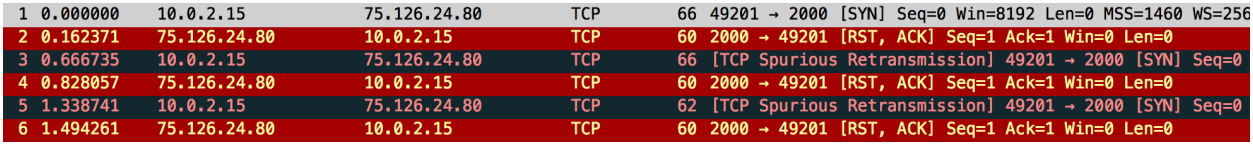
Interestingly, aax.me also attempts to determine the presence of various antivirus products on a target's machine from within the target's web browser. The technique appears to work on any modern version of Windows, with the latest versions of Chrome, Firefox, and IE/Edge. Specifically, the script conducts `GET` XMLHttpRequests (one at a time) to `http://127.0.0.1/` on the following ports: 12993, 44080, 24961, 1110, 6646, 6999, 30606. The script stops conducting these requests if it finds one request whose `readyState` is set to 4 less than 20ms after the request was initiated (200ms for port 6646), and submits the number of this port to the server.

The latest versions of Internet Explorer/Edge, Chrome, and Firefox (except Tor Browser) will all perform these XMLHttpRequests to 127.0.0.1 on behalf of any site. Of course, the result of such a request will most likely not be available to the script, due to the same-origin policy, and likely absence of a CORS [201] header in the response. Indeed, the script does not attempt to read the results of its requests. Rather, it leverages the fact that the web browser makes the status of the request sent available, via the `readyState` parameter of an XMLHttpRequest instance (1 approximately represents TCP SYN sent, and 4 represents HTTP response received or TCP connection terminated). For a closed port, Windows will issue an RST/ACK for each SYN sent. However, it appears (Figure 2.22) that Windows' TCP stack will not consider an outgoing connection it is initiating to be terminated until it has sent 3 SYNs, and received three corresponding RST/ACKs (or timeouts).

1	0.000000	127.0.0.1	127.0.0.1	TCP	66 49868 → 2000 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=2
2	0.000014	127.0.0.1	127.0.0.1	TCP	54 2000 → 49868 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
5	0.508806	127.0.0.1	127.0.0.1	TCP	66 [TCP Spurious Retransmission] 49868 → 2000 [SYN] Seq=0
6	0.508823	127.0.0.1	127.0.0.1	TCP	54 2000 → 49868 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
9	1.008816	127.0.0.1	127.0.0.1	TCP	62 [TCP Spurious Retransmission] 49868 → 2000 [SYN] Seq=0
10	1.008832	127.0.0.1	127.0.0.1	TCP	54 2000 → 49868 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Figure 2.22: Three RST/ACKs required until Windows considers outgoing TCP connection terminated.

When testing with a TCP connection from Windows to a remote host, we can clearly see (Figure 2.23) that Windows transmits the second SYN 500ms after the first RST/ACK, and the third SYN 500ms after the second RST/ACK.



1	0.000000	10.0.2.15	75.126.24.80	TCP	66 49201 → 2000 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256
2	0.162371	75.126.24.80	10.0.2.15	TCP	60 2000 → 49201 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
3	0.666735	10.0.2.15	75.126.24.80	TCP	66 [TCP Spurious Retransmission] 49201 → 2000 [SYN] Seq=0
4	0.828057	75.126.24.80	10.0.2.15	TCP	60 2000 → 49201 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
5	1.338741	10.0.2.15	75.126.24.80	TCP	62 [TCP Spurious Retransmission] 49201 → 2000 [SYN] Seq=0
6	1.494261	75.126.24.80	10.0.2.15	TCP	60 2000 → 49201 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Figure 2.23: Windows sends the next SYN 500ms after the latest RST/ACK.

Thus, the `readyState` value for a request to a closed port on 127.0.0.1 will not be set equal to 4 until approximately 1000ms after the request is issued. In summary, one can use this technique to distinguish between a closed port (`readyState` set to 4 at around 1000ms), an open port (`readyState` set to 4 before 1000ms), and a filtered port (`readyState` set to 4 typically long after 1000ms, based on the default behavior described in [185]).

This script was apparently designed to detect the presence of certain components of Avast, Avira, ESET, Kaspersky, and Trend Micro antivirus products. We were not able to determine which program the probing of port 24961 was designed to detect. We verified that the latest version of Avast can be detected by this script, as it opens TCP port 12993, which is associated with its Mail Shield component for scanning email traffic; port 6999 is opened by Trend Micro's *tmproxy* [138], which scans web and email traffic; port 1110 is used by Kaspersky [109] to scan web and email traffic; it appears that Avira's Web Protection component for scanning web traffic used to open port 44080 [154], though we observed it opening 44081 instead; port 30606 appears to have been used by ESET to scan web and email traffic [117], but we did not observe this port open while testing the latest version of ESET; port 6646 may be used by McAfee [15], though we did not test this.

The code for the port scanning appears to be adapted from the *JS-Recon* port scanning tool [110]. *JS-Recon* is a generic browser-based tool that enumerates all open ports on `127.0.0.1` in a range; it does not specifically target anti-virus programs. The `scan_xhr` and `check_ps_xhr` functions in the aax.me profiling script are similar to the `scan_ports_xhr` and `check_ps_xhr` functions in *JS-Recon*. The creator of *JS-Recon* seems to have first presented the tool at BlackHat Abu Dhabi in 2010 [107].

Note that this technique can be generalized to any remote content timing side channel (e.g, the `onerror` event for an `Image`). Additionally, one can identify the presence of an open port on `127.0.0.1` that speaks HTTP without using timing information, and thus without the Windows TCP behavior assumption (e.g., by handling the `onerror` and `oncomplete` events of certain types of `link` elements).

The Case of the Fake Journalist

While investigating Stealth Falcon, we enrolled Donaghy in *Himaya* (Chapter 5). We found evidence that he had been targeted by Stealth Falcon in 2013, using a fictitious journalist persona that we connected to other possible attacks against UAE figures.

In December 2013, Donaghy received the following message purporting to be from a UK journalist named Andrew Dwight (Figure 2.24).

From: andrew.dwight389@outlook.com
Subject: FW: Correspondence Request

Greetings Mr. Donaghy,

I have been trying to reach you for comment and I am hoping that this e-mail reaches the intended recipient. My name is Andrew Dwight and I am currently writing a book about my experiences in the Middle East. My focus is on human factors and rights issues in seemingly non-authoritarian regimes (that are, in reality, anything but). I was hoping that I might correspond with you and reference some of your work, specifically this piece (`http://goo.gl/60HAqJ`), for the book. I'm quite impressed with the way you articulate this complex issue for the masses, and hope to have a similar impact with my book.

Happy New Year,

Andrew

Figure 2.24: Email message sent to Donaghy by Andrew Dwight.

The link in the email, `http://goo.gl/60HAqJ`, redirects to `http://aax.me/0b152`, which, as of December 2015, redirected to a 2013 Huffington Post blog post authored by Donaghy [40]. We did not observe any `redirect.php` behavior with this link; as of December 2015, the aax.me link directly served an HTTP 302 redirect to the Huffington Post. However, it is possible that the link previously invoked the profiling script on `redirect.php`.

We found that Donaghy had responded to this message shortly after receiving it, offering to meet in-person with Andrew in the UK. Andrew responded several weeks later with a lengthy email explaining how he had been detained in upstate New York by adverse weather while on a ski holiday. The email did not appear to contain any malicious content.

While attempting to determine whether "Andrew Dwight" was a real person, we found a Twitter profile, `@Dwight389` for the same persona, and that includes the same email address from which Donaghy received the email. We found that this account messaged three UAE dissident accounts via Twitter mentions, indicating that Dwight had established, or was trying to establish, contact with them. We were unable to determine whether `@Dwight389` successfully attacked any of these individuals.

One individual messaged by Dwight on April 24, 2013 was Obaid Yousef Al-Zaabi, who was later arrested on July 2, 2013 [17] for Tweeting about the UAE94 detainees (94 defendants prosecuted in a mass trial on charges of attempting to overthrow the government) [21] on his `@bukhaledobaid` account [81]. Al-Zaabi remains in prison. Dwight also contacted Professor Abdullah Al-Shamsi, Vice Chancellor of the British University in Dubai, and one of the 133 signatories to the same petition that Ahmed Mansoor was arrested for signing. Dwight also contacted the account `@northsniper`. In May 2015, five Qataris were sentenced

(one present in the UAE to 10 years in prison, and four in absentia to life in prison) for posting allegedly offensive pictures of the UAE Royal Family on three Twitter accounts and two Instagram accounts [102], including `@northsniper` [106].

Connections to UAE government

In October 2016, The Intercept reported that technology used by Stealth Falcon was developed by a US-based defense contractor, **CyberPoint**, and may have later been transferred to a UAE-based cybersecurity company, **DarkMatter** [136]. Hacking Team sold its RCS spyware to the UAE Ministry of Interior through CyberPoint [79]. According to leaked Hacking Team data, the UAE Ministry of Interior was a second UAE client [180], not the one that targeted Mansoor in 2012.

Before The Intercept's reporting, we had also linked Stealth Falcon to the UAE government. In December 2012, we were contacted by an Emirati activist, who reported that an account, `@WeldBudhabi`, had sent him a link on 14 December 2012 via Twitter direct message that took him to a page on `a7rarelemarat.com`. A report by BBC notes that UAE authorities on 14 December 2012 arrested an individual who they believed to be associated with `@WeldBudhabi`, and that the account was "reportedly hacked by the authorities" on the same day [114]. The Emirati activist told us that he later contacted `@WeldBudhabi`, who reported that he did not send the link.

`a7rarelemarat.com` is a now-defunct website that purported to be an opposition web forum for discussing Emirati issues, and providing proxy tools for "hiding from the thugs" (presumably a reference to the UAE authorities). We found four instances where the site's Twitter account, `@a7rarelemarat`, posted two unique links redirecting through `goo.gl` and `aax.me`. The tweets were posted less than 2 minutes after `goo.gl` API statistics indicate that the shortened links were created.

The use of both `goo.gl` and `aax.me` in these cases suggests that the `goo.gl` link may have been designed to conceal the `aax.me` domain. Also, the proximity in creation time between the Tweet and the `goo.gl` link suggests that the person who posted the Tweet through `@a7rarelemarat` was likely the same person who created the `goo.gl` link. We suspect that the `aax.me` attacker had some control over `@a7rarelemarat` at the time, and may have had control of `a7rarelemarat.com` as well.

2.4.4 Earlier FinSpy attacks

We later identified an older UAE-linked attack involving *FinSpy* (teal elements in Figure 2.10), when we enrolled Ahmed Mansoor in *Himaya* (Chapter 5). In March 2011, Mansoor received an email with an attached `.rar` file containing an `.exe` file disguised as a `.pdf`, using the same Unicode RLO trick as we saw used in Bahrain to distribute FinSpy (Section2.3.1). When run, the `.exe` file opened a `.pdf` file that contained a copy of an online pro-democracy petition that Mansoor and 132 other individuals had signed earlier that month. Mansoor had replied to the attacker's email, indicating that he knew the file was

actually an `.exe` file, and would not open it. The attackers responded with the embedded, benign, `.pdf` file.

The tag in the FinSpy file was `EKHWAN_AHMED_MN9OOR`, suggesting that it was targeted both at Ahmed Mansoor,[6] and members of the UAE's Islamist *al-Islah* organization who signed the same petition. The government contends *al-Islah* is linked to the "Ekhwan" (Muslim Brotherhood).

2.4.5 NSO Group Pegasus and Trident

On August 10, 2016, Mansoor received an SMS text message that promised "new secrets" about detainees tortured in UAE prisons, and contained a hyperlink to an unfamiliar website. The next day he received a second, similar text. The messages arrived on Mansoor's stock iPhone 6 running iOS 9.3.3. Mansoor was suspicious of these text messages (Figure 2.25), and quickly forwarded them to us for investigation.

Figure 2.25: SMS messages received by Ahmed Mansoor designed to install NSO Pegasus spyware on his phone. The Arabic text says "new secrets about torture of Emiratis in State Prisons."

When Mansoor's messages reached us, we recognized the links: the domain name `webadv.co` belongs to a network of domains that we believed to be part of an exploit infrastructure provided by the spyware company NSO Group (Section 3.5). We had first come across the NSO Group infrastructure during the course of our earlier research into *Stealth Falcon* (Section 3.4).

[6] "Mn9oor" is a Romanization of Mansoor's Arabic surname.

When we first found the infrastructure and connected it to NSO Group, we hypothesized that operators of the NSO Group spyware would target a user by sending them an infection link containing one of the exploit infrastructure domain names. Though we had previously found several public occurrences of links involving these domains on Twitter, none of the links we found seemed to be active (i.e., none produced an infection when we tested them). In other exploit infrastructures with which we are familiar (e.g., Hacking Team's exploit infrastructure [73]), we had noted that attackers prefer to deactivate such links after a single click, or after a short period of time, perhaps in order to prevent the disclosure of the exploit to security researchers.

Device infection

We planned to access the links Mansoor received from an iPhone, while monitoring its Internet traffic, in an attempt to capture any exploit or malicious payload. We obtained a stock factory-reset iPhone 5 (Mansoor had an iPhone 6) with iOS 9.3.3 (the same version as Mansoor). We connected our iPhone to the Internet through an experiment laptop configured to act as a Wi-Fi hotspot. As the links Mansoor received used `https`, we installed `Wireshark` [203] and `mitmproxy` [143] on our experiment laptop in order to facilitate decryption of `https` traffic.

We accessed one link by opening Safari on our iPhone, and manually transcribing the link from the screenshot that Mansoor sent. After about ten seconds of navigating to the URL, which displayed a blank page, the Safari window closed, and we observed no further visual activity on the iPhone's screen. Meanwhile, we saw on our experiment laptop that the phone was served what appeared to be a Safari exploit, followed by intermediate files (`final111`), and a final payload (`test111.tar`). We show the Wireshark output in Figure 2.26.

The final payload contained an iPhone application, and was requested with a non-Safari user agent. This suggested that the link contained a zero-day iPhone remote jailbreak: a chain of heretofore unknown exploits used to remotely circumvent iPhone security measures to install an unauthorized application. During exploitation, the phone appeared to report back to the server `sms.webadv.co` about the progress of the jailbreak and exploit (the `HTTP GET` requests to `html` pages in Figure 2.26), perhaps to facilitate troubleshooting if the exploit failed.

Analysis of the attack

We provided samples of the attack to Lookout [116], a mobile security company, and collaborated on an analysis [90].

The link Mansoor received loads an initial stage, which we call *stage1*, containing obfuscated JavaScript that determines whether the phone is 32-bit (iPhone 5 and earlier) or 64-bit (iPhone 5s and later), and requests the appropriate *stage2* via `XMLHttpRequest`. In our case, we received `final111`, the 32-bit stage2. The 64-bit stage2 was named `final222`.

```
GET /9573305s/ HTTP/1.1
HTTP/1.1 200 OK  (text/html)
GET /9573305s/ntf_xps.html&nocache=1470937418505 HTTP/1.1
HTTP/1.1 200 OK
GET /9573305s/ntf_gog.html?a=568_320_2_SGX543&b=1&nocache=1470937420684 HTTP/1.1
GET /9573305s//final111?&nocache=1470937420687 HTTP/1.1
HTTP/1.1 200 OK
GET /9573305s/ntf_gog.html?a=568_320_2_SGX543&b=2&nocache=1470937420686 HTTP/1.1
HTTP/1.1 200 OK
GET /9573305s/ntf_gog.html?a=568_320_2_SGX543&b=210161856&nocache=1470937420582 HTTP/1.1
HTTP/1.1 200 OK  (application/octet-stream)
GET /9573305s/ntf_gog.html?a=568_320_2_SGX543&b=3&nocache=1470937436700 HTTP/1.1
HTTP/1.1 200 OK
HTTP/1.1 200 OK
GET /9573305s/ntf_gog.html?a=568_320_2_SGX543&b=4&nocache=1470937443385 HTTP/1.1
GET /9573305s/ntf_xpe.html&nocache=1470937443412 HTTP/1.1
HTTP/1.1 200 OK
HTTP/1.1 200 OK
GET /9573305s/ntf_bed.html?s=10040&d= HTTP/1.1
HTTP/1.1 200 OK
GET /9573305s/ntf_brc.html?m=0 HTTP/1.1
HTTP/1.1 200 OK
GET /9573305s/ntf_bed.html?s=10041&d=Tring%20to%20download%20bundle%28try%3A0%29 HTTP/1.1
HTTP/1.1 200 OK
GET /9573305s/test111.tar HTTP/1.1
```

Figure 2.26: Wireshark output of NSO iPhone exploit process.

Stage1 employs a previously undocumented memory corruption vulnerability in WebKit to execute this code within the context of the Safari browser (CVE-2016-4657 [89]).

Stage2 exploits a function that returns a kernel memory address, from which the base address of the iOS kernel can be mapped (CVE-2016-4655 [89]). Stage2 then exploits a memory corruption vulnerability in the kernel (CVE-2016-4656 [89]). This last vulnerability is used to disable code signing enforcement, allowing the running of unsigned binaries. Stage2 downloads and installs *stage3*, which is the spyware payload. Together, we refer to these three exploits as the **Trident** (magenta elements in Figure 2.10, lower-right).

The Trident is re-run locally on the phone at each boot, using the `JavaScriptCore` binary. To facilitate persistence, the spyware disables Apple's automatic updates, and detects and removes other jailbreaks.

The attack payload includes a renamed copy of Cydia Substrate [34], a third-party app developer framework, which it uses to help facilitate recording of messages and phone calls from targeted apps. To record WhatsApp and Viber calls, the spyware loads code into the WhatsApp and Viber processes using the Cydia Substrate. The loaded code hooks various call status methods, and sends system-wide notifications when call events occur; the spyware listens for these notifications and starts or stops recording as appropriate. It appears that

the payload can spy on apps including: iMessage, Gmail, Viber, Facebook, WhatsApp, Telegram, Skype, Line, KakaoTalk, WeChat, Surespot, Imo.im, Mail.Ru, Tango, VK, and Odnoklassniki.

The spyware also exfiltrates calendar and contact data, as well as passwords saved in the phone's keychain, including Wi-Fi passwords and networks. The attack payload beacons back, via HTTPS, to C&C servers delivered in stage2. The C&C servers are encoded in stage2 to appear like a text message from Google containing a login verification code:

```
Your Google verification code is:5678429
http://gmail.com/?z=FEcCAA==&i=
MTphYWxhYW4udHY6NDQzLDE6bWFub3Jhb25saW5lLm5ldDo0NDM=&s=zpvzPSYS674=
```

Legitimate Google messages of this type do not contain a link, and contain one fewer digit in the verification code. Base64-decoding the "`i`" parameter of the URL yields the C&C servers used by stage3:

```
1:aalaan.tv:443,1:manoraonline.net:443
```

A similar obfuscation appears to be used for exchange of information over SMS between an infected phone and the spyware C&C server. In case the spyware's C&C is disabled or unreachable, an operator may deliver updated C&C servers to an infection using and SMS in this format.

The final payload that we identified, `test111.tar`, contained several files, including `libaudio.dylib`, which appeared to be the base library for call recording, `libimo.dylib`, which appeared to be the library for recording chat messages from apps, and two libraries for WhatsApp and Viber call recording: `libvbcalls.dylib`, and `libwacalls.dylib`. In each file, we found several hundred strings containing the text `_kPegasusProtocol`, which contains the name of NSO Group's "Pegasus" spyware:

```
_kPegasusProtocolAgentControlElement_iv
_kPegasusProtocolAgentControlElement_key
_kPegasusProtocolAgentControlElement_ciphertext
```

Pegasus architecture

We published a report on our discovery of Pegasus in August 2016 [126]. Before our report, there was little reliable information available about NSO Group or its Pegasus product. Much of the publicly available information about Pegasus seemed to be rumor, conjecture, or unverifiable claims made to media about capabilities (e.g., [207]). However, when we examined the leaked Hacking Team emails (Section 2.4.1), we found several instances of Hacking Team

clients or resellers sharing what appeared to be NSO Group's product documentation and sales pitches.

In one December 2014 email [155], a reseller of surveillance technologies to the Mexican government forwarded a `.pdf` document containing detailed technical specifications of NSO Group's Pegasus system to Hacking Team. The document's metadata indicated it was created in December 2013 by Guy Molho, who is listed on LinkedIn as the Director of Product Management at NSO Group [74].

The leaked technical specifications mention that NSO Group offers two remote installation vectors for spyware onto a target's device: a one-click vector and a zero-click vector. The one-click vector involves sending the target a normal SMS text message with a link to a malicious website, like in Mansoor's case. The malicious website contains an exploit for the web browser on the target's device, and any other required exploits to implant the spyware.

To use NSO Group's zero-click vector, an attacker instead sends the same link via a special type of SMS message, like a WAP Push Service Loading (SL) message [20]. A WAP Push SL message causes a phone to automatically open a link in a web browser instance, eliminating the need for a user to click on the link to become infected. Many newer models of phones have started ignoring or restricting WAP Push messages. Mobile network providers may also decide to block these messages.

The documentation (Figure 2.27) refers to a malicious website employed in installation of the spyware ("Agent") as an **Anonymizer**, which communicates with a **Pegasus Installation Server** located on the attacker's premises. When a target visits a malicious link from their device, the Anonymizer forwards the request to the Pegasus Installation Server, which examines the target device's `User-Agent` header to determine if Pegasus has an exploit chain, such as the Trident, that supports the device.

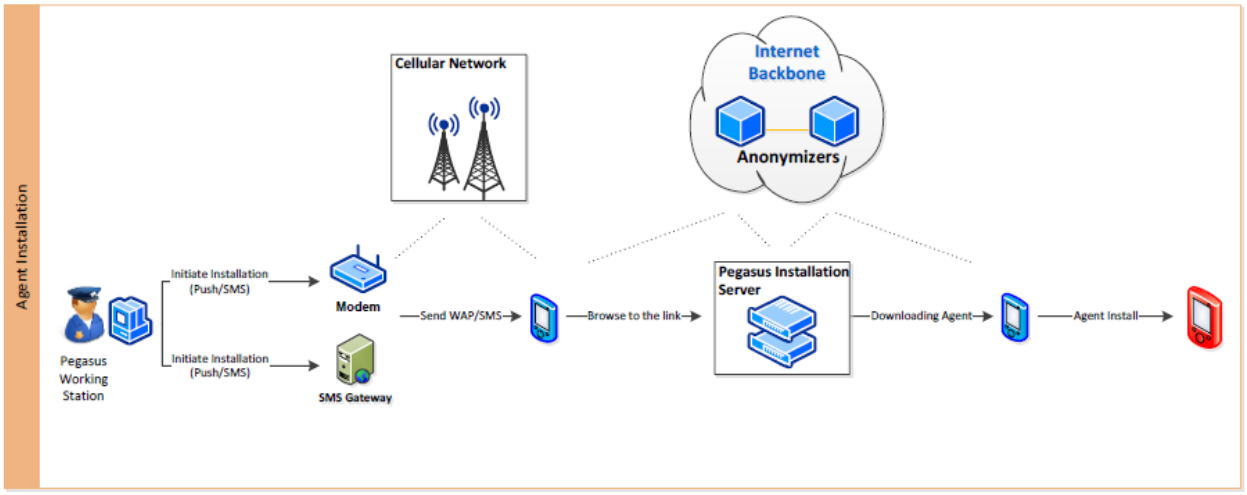


Figure 2.27: Diagram from purported Pegasus documentation showing the sequence through which the spyware agent is installed on a target's mobile device.

If the device is supported, the Pegasus Installation Server returns the appropriate exploit to the target device through the Anonymizer and attempts an infection. If infection fails for any reason, the target's web browser will redirect to a legitimate website specified by the Pegasus operator, in order to avoid arousing the target's suspicion.

In the operation targeting Mansoor, the one-click vector was apparently used with the anonymizer `sms.webadv.co`.

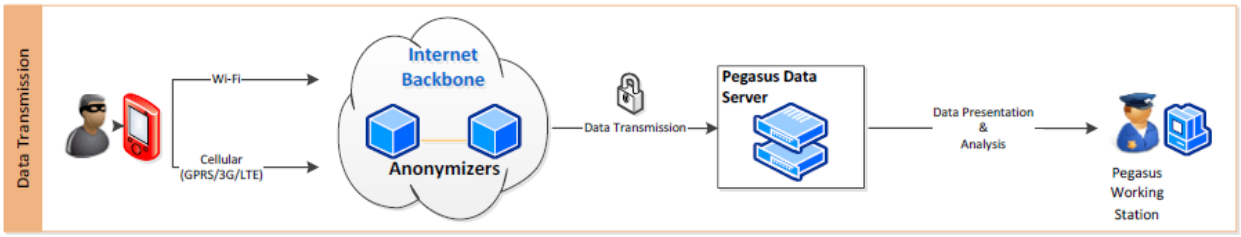


Figure 2.28: Diagram from purported Pegasus documentation showing how the spyware agent, once installed exfiltrates data.

According to the purported NSO Group documentation (Figure 2.28), an infected device transmits collected information back to a **Pegasus Data Server** at the operator's premises, via the **PATN (Pegasus Anonymizing Transmission Network)**. Once the collected information arrives on the Pegasus Data Server, an operator may visualize the information on a Pegasus Working Station.

The implant in the attack targeting Mansoor communicated with two PATN nodes: `aalaan.tv` and `manoraonline.net`.

2.5 Conclusion

Targeted surveillance of individuals conducted by nation-states poses an exceptionally challenging security problem, given the great imbalance of resources and expertise between the victims and the attackers. We have sketched the nature of this problem space as reported to us by targeted individuals in two Middle Eastern countries. The attacks include spyware for ongoing monitoring and the use of "IP spy" links to deanonymize those who voice dissent.

The attacks, while sometimes incorporating sophisticated social engineering, typically involve "just enough" technical sophistication, e.g., using older, known exploits rather than zero-day exploits (Section 2.4.1), or even no exploits at all (Sections 2.3.1, 2.4.3). Some attacks eschew spyware altogether for technically simple phishing attacks (Sections 2.3.2, 2.4.2).

However, when targeting devices such as iPhones, where software installation and updates are tightly controlled, attackers must use complex zero-day exploits in order to facilitate remote spyware infection (Section 2.4.5).

In many cases, attackers use prepackaged tools developed by commercial vendors or acquired from the cybercrime underground. This technology sometimes suffers from what

strike us as amateurish mistakes (multiple serious errors implementing cryptography, broken protocol messages), as does the attackers' employment of it (identifying-information embedded in binaries, clusters of attack accounts tied by common activity). Some of these errors assisted our efforts to assemble strong circumstantial evidence of governmental origins.

Our analysis also provided a key opportunity to answer questions about the extent of use and proliferation of the tools we uncovered, which we address in Chapter 3.

Chapter 3

Empirical Characterization

3.1 Introduction

In each case where we identified a sample of commercial government-exclusive or custom-written spyware (Chapter 2), we attempted to empirically characterize the prevalence of the spyware to better understand attacker behavior, and the nature of the market for spyware.

The three types of commercial government-exclusive spyware we identified: FinSpy (Section 2.3.1), RCS (Section 2.4.1), and Pegasus (Section 2.4.5), all have a similar architecture, according to their leaked technical manuals. Devices infected by attackers report back to a C&C server installed on the attackers' premises, by way of a series of *proxies*, typically hosted on cloud-based VPS providers. In Hacking Team's RCS spyware, the proxies are called *anonymizers* [183]; for FinFisher's FinSpy, the proxies are called *relays* [93], and for NSO Group's Pegasus, the proxies are referred to as a *Pegasus Anonymizing Transmission Network (PATN)* [155]. A spyware infection communicates directly with a proxy, without knowing the IP address of the C&C server.

The purpose of the proxies is to prevent a target (or researcher) from discovering the IP address of an operator's C&C server, and identifying the operator on that basis. FinFisher documentation describes that relays make it "**practically impossible**" (their emphasis) for a researcher to discover "the location and country of the Headquarter [sic]" [93]. Hacking Team documentation says that the purpose of anonymizers is to "protect[] the server against external attacks and permit[] anonymity during investigations" [183]. NSO Group documentation [155] says that the goal of the PATN is "to assure that trace back to the operating organization is impossible."

We also noted that FinFisher, NSO, and Hacking Team all appeared to design their proxies to use some form of *decoy page*. When a user typed an address of a proxy into their web browser, they were directed to this decoy page, which was typically either designed to impersonate a popular web server error page (usually *Apache httpd*), or to redirect the user to a popular website (usually `google.com`). Because the decoy pages were custom code implemented individually by each vendor, and were often imperfect copies of the originals, we

were able to design fingerprints for these decoy pages. Our fingerprinting was also aided by the fact that proxies sometimes responded to certain HTTP requests with improper headers.

In other cases, we were able to develop fingerprints for proxies based on elements of the server's TLS handshake, or fingerprints that tested whether a server recognized the spyware's initial handshake. In some cases, we were able to search for our fingerprints in public HTTP and TLS scanning datasets. In particular, we made use of the *2012 Internet Census* [94], *Shodan* [171], *Censys* [22], *critical.io* [144], data from *Project Sonar* [159], and TLS scan results from a University of Michigan study [46] and TU Munich [82]. When no existing dataset was applicable, we configured *zmap* [45] to scan the Internet using our fingerprint, or wrote our own custom scanning tools.

In the cases of Hacking Team's RCS and FinFisher's FinSpy, we found further deficiencies in the proxy implementations that allowed us to identify the IP addresses of C&C servers behind proxies in some cases. Some C&C servers were located in IP address ranges that were directly registered to government clients, according to WHOIS information. In other cases, we were able to attribute C&C servers based on the fact that other services, such as an email server, were running on the same subnet, or based on the fact that some outgoing Internet traffic from the operator's premises originated from the same IP address as the C&C, and this address was found in outgoing email headers in the leaked Hacking Team emails [70].

Our efforts identified 42 nation-state users of these tools. In several cases, we engaged with dissidents in these countries to try to establish evidence of abusive surveillance. Our engagement led to the discovery of attacks on Ethiopian dissidents with FinSpy, RCS, and other spyware [128, 127].

In the case of *Stealth Falcon* (Section 2.4.3), which operated a malicious URL shortener in addition to spyware, we had an opportunity to characterize not only prevalence of their spyware, but also attackers' choice of bait content.

While this chapter presents many of the fingerprints we used, we choose to keep some confidential, in order to facilitate our continued visibility into operators' infrastructure, and to avoid compromise of what may be legitimate government uses of these spyware tools. Indeed, our empirical characterization led us to several samples that appeared to be targeted at violent extremists, or used in police investigations.

3.2 FinFisher FinSpy

Immediately after our report on the use of FinFisher's FinSpy to target Bahraini dissidents [133], in which we published the unredacted IP address of the FinSpy C&C server in Bahrain, independent researcher Guarnieri noticed that sending an HTTP `GET /` request to this server yielded a bizarre response: "`Hallo Steffi`" [72]. Guarnieri searched Internet scanning results recently compiled by the *critical.io* project [144], locating 11 additional servers in 10 countries [72]. We refer to this fingerprint as ϕ_1 (Figure 3.1). We show an example of a FinSpy server response matching ϕ_1 in Figure 3.2. Note also the "`Hallo Steffi`"

message in the body of the response, as well as the missing space before the value of the `Content-Length:` header.

```
/^HTTP\/1.1 200 OK\r\nContent-Type: text\/html; charset=UTF-8\r\nContent-
Length:12\r\n\r\nHallo Steffi$/
```

Figure 3.1: Fingerprint ϕ_1.

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Content-Length:12

Hallo Steffi
```

Figure 3.2: An example of a FinSpy server response matching ϕ_1.

Our analysis of the FinSpy executables sent to Bahraini activists revealed that these samples used a custom TLV (type-length-value) binary protocol for communication with the C&C server, on common ports such as 22, 53, 80, and 443. We did not find any indications that HTTP was ordinarily used for communications with the C&C. We assume that "`Hallo Steffi`" was an anomaly that perhaps accidentally made it into the production version of FinSpy.

Concurrent with Guarnieri's scanning effort, we devised our own fingerprint that tested whether a server recognized FinSpy's custom TLV protocol. We noticed that the FinSpy sample we obtained from Bahrain sent the following bytes after establishing a TCP connection with the spyware server, where `WWXXYYZZ` are the last four octets of the infected computer's MAC address:

```
0C00000040017300WWXXYYZZ
```

In response to these bytes, the C&C server sent a TCP ACK, and waited for the client to send additional data. We further noticed that when we replaced bytes 4-8 of the message with `FFFFFFFF`, a FinSpy server would not send a TCP ACK, but would instead close the TCP connection. Thus, we devised fingerprint ψ_1 (Algorithm 1) to test whether a server recognized the FinSpy protocol.[1] For a given IP and port, we sent three valid and three invalid FinSpy protocol requests, and made sure that in each case, the IP accepted the valid data by not closing the TCP connection, while rejecting the invalid data by closing the TCP connection.

[1]For FinSpy servers, fingerprints ϕ denote HTTP fingerprints, and fingerprints ψ denote fingerprints based on FinSpy's custom TLV protocol.

We conducted targeted scanning of several countries' IP space using ψ_1, implemented as a Python script, and found eight additional servers.[2]

Algorithm 1 FinSpy fingerprint ψ_1.

```
1: procedure FinSpyTest(ip)
2:     for port ∈ {22, 50, 80, 443} do                    ▷ Test common ports
3:         successes ← 0
4:         for try ∈ {1, 2, 3} do                         ▷ Run three tests on each port
5:             T_0 ← connect(ip, port)                    ▷ TCP connection
6:             send(T_0,0x0C000000FFFFFFFF01020304)
7:             if T_0 closed by remote host then
8:                 T_1 ← connect(ip, port)
9:                 send(T_1,0x0C00000040017300010203040)
10:                if T_1 times out on read then          ▷ Server ACK with no data
11:                    successes ← successes + 1
12:                    if successes == 3 then             ▷ All three tests must succeed
13:                        return γ_1
14:                    end if
15:                end if
16:            end if
17:        end for
18:    end for
19: end procedure
```

We were able to confirm that all servers detected by Guarnieri using ϕ_1 matched our fingerprint ψ_1, for those servers still reachable after he published his findings. We did not publicize this fingerprint until February 2015 [125], though we noticed a slight change in FinSpy server behavior in late 2012, which broke our fingerprint ψ_1. We devised a similar fingerprint ψ_2, which we do not disclose here, that continues to work as of November 2016. We wrote a *zmap* scan module to scan for ψ_2.

3.2.1 Evolution of decoy pages

Subsequent to our July 2012 report on the Bahrain FinSpy samples [133], FinSpy servers were apparently reconfigured so that they closed the TCP connection in response to a `GET /` or `HEAD /` request. Thus, fingerprint ϕ_1 no longer could detect FinSpy servers. However, the servers continued to behave consistently with ψ_1.

In October 2012 we noted that the servers stopped responding to ψ_1, in that they failed to close the TCP connection when we sent certain types of invalid data. However, many of

[2]We were constrained in our scanning by the performance of our Python script; unfortunately `zmap` was not yet available.

the servers exhibited new behavior when sent a `GET /` request: the server's HTTP response claimed it was an Apache server, but the `Date` HTTP header returned by the server used the string "`UTC`" rather than "`GMT`". This is a violation of RFC 2616, which mandates the use of the string "`GMT`" [51]. We fingerprinted this error as ϕ_2 (Figure 3.3), and we show an example of such a response in Figure 3.4.

```
/UTC\r\nServer: Apache\r\nVary: Accept-Encoding\r\nContent-Length: 204\r\
nContent-Type: text\/html; charset=iso-8859-1\r\n\r\n/
```

Figure 3.3: Fingerprint ϕ_2.

```
HTTP/1.1 403 Forbidden
Date: Sun, 03 Feb 2013 00:30:27 UTC
Server: Apache
Vary: Accept-Encoding
Content-Length: 204
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access this server.</p>
</body></html>
```

Figure 3.4: An example of a FinSpy server response matching ϕ_2.

In November 2012, we noticed that some FinSpy servers began to exhibit a different distinctive behavior in response to `GET /` requests. Specifically, the servers returned a short response without any body content, despite the presence of a `Content-Type` header. We fingerprinted this behavior as ϕ_3 (Figure 3.5), and we show an example of such a response in Figure 3.6.

```
/^HTTP\/1.1 200 OK\r\nContent-Type: text\/html; charset=UTF-8\r\n\r\n$/
```

Figure 3.5: Fingerprint ϕ_3.

Around the end of February 2013, we noticed that the behavior of FinSpy servers in response to `GET /` requests changed yet again. The servers began returning a page that appeared visually identical to the default Apache "It works!" page (Figure 3.8), though the

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
```

Figure 3.6: An example of a FinSpy server response matching ϕ_3. Note that there is no response body, even though there is a `Content-Type` header.

HTML code was substantially different than the actual Apache page [62] (Figure 3.9). We devised a fingerprint ϕ_4 to capture this behavior (Figure 3.7)

```
/^HTTP\/1.1 200 OK\r\nServer: Apache\r\nDate: [^\\]+\r\nContent-Type: text\/
html; charset=iso-8859-1\r\nContent-Length: 140\r\n\r\n/
```

Figure 3.7: Fingerprint ϕ_4.

```
HTTP/1.1 200 OK
Server: Apache
Date: Sun, 03 Feb 2013 00:30:27 GMT
Content-Type: text/html;
charset=iso-8859-1
Content-Length: 140

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>200 OK</title>
</head><body>
<h1>It works!</h1>
</body></html>
```

Figure 3.8: An example of a FinSpy server response matching ϕ_4.

```
<html><body><h1>It works!</h1></body></html>
```

Figure 3.9: HTML code for the actual Apache httpd "It works!" page.

We found an exact copy of the fake Apache "It works!" page (the body of the HTTP response in Figure 3.8) in FinSpy relay code that was leaked in September 2014 [27].

3.2.2 Deanonymizing FinSpy C&C servers

In Spring 2013, a change to FinSpy servers allowed us the opportunity to identify the IP address of the C&C servers behind some relays. We scanned the Internet using *zmap* and our fingerprint ψ_2, and examined the behavior of the servers we found in response to `GET / ` requests. We noticed that some FinSpy servers were returning pages that appeared to be copies of `https://google.com`, `https://yahoo.com`, or other webpages. However, the pages contained localization data (e.g., request IP address or localized weather) that did not match the IP address or location of the FinSpy server we queried. Several of the embedded IP addresses appeared to belong to various government agencies. This suggested to us that when we queried a FinSpy relay server (Figure 3.10), it would return to us a page fetched by the FinSpy master server, thus revealing details about the location of the government client's premises.

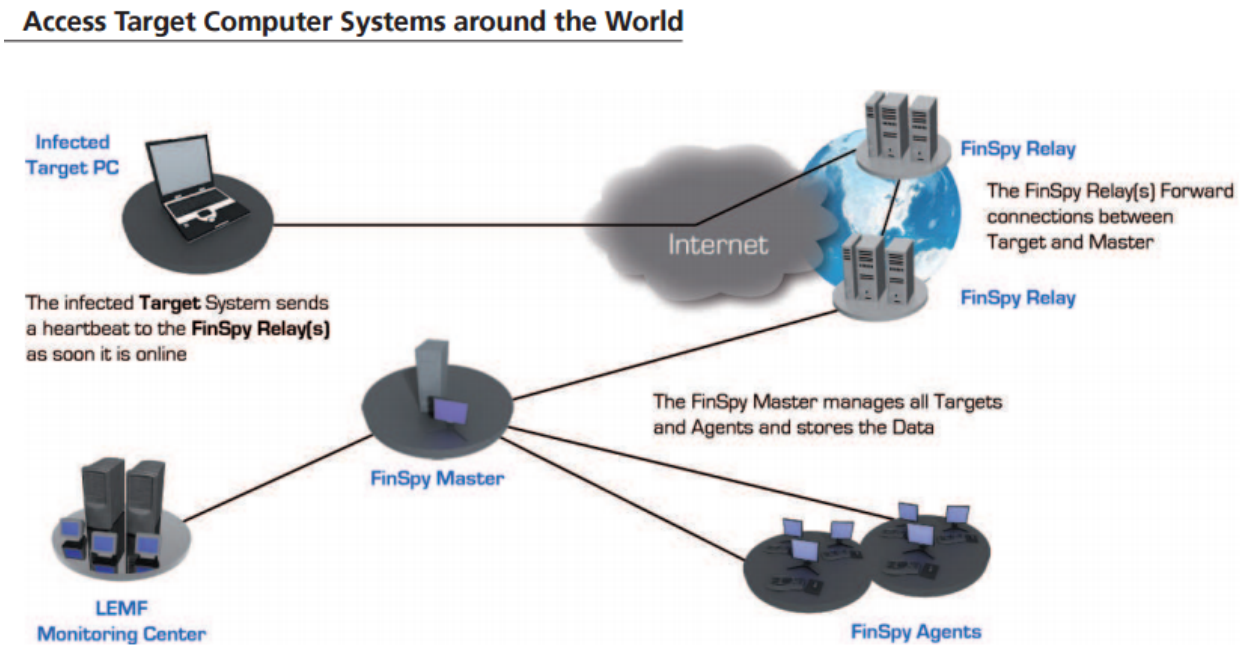


Figure 3.10: FinSpy relay architecture, from leaked FinFisher documentation.

FinSpy servers returning Google

The most popular page returned by FinSpy servers was Google's homepage, `google.com`. We noticed that when we sent one of these FinSpy servers a Google search query to ask for our IP address, e.g., "my ip address", the FinSpy server returned a Google page including an IP address that was neither ours, nor the FinSpy server's. For instance, some FinSpy servers in

the United States returned IP addresses in Indonesia (Figure 3.11). We sent queries similar to the one in Figure 3.12, where `$IP` represents the IP address of the FinSpy server.[3]

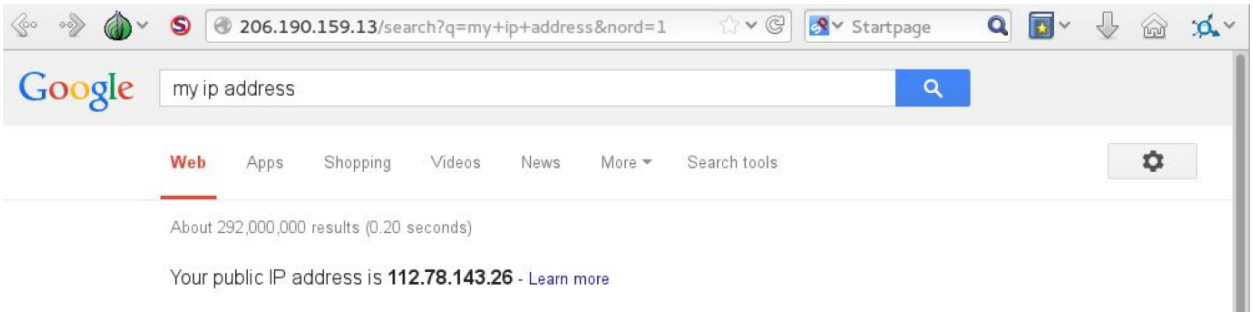


Figure 3.11: A US-based FinSpy server returns an IP address in Indonesia.

```
GET /search?q=my+ip+address&nord=1 HTTP/1.1
Host: $IP
User-Agent: Mozilla/5.0 (Windows NT 6.1; rv:38.0) Gecko/20100101 Firefox/38.0
```

Figure 3.12: Google query we sent to FinSpy relays to find IP addresses of C&C servers.

The Indonesian IP address we obtained (Figure 3.11) matched a FinSpy server IP address originally detected by Guarnieri using ϕ_1, and confirmed by us in 2012 using ψ_1.

That FinFisher proxies can apparently reveal the IP of the master is quite peculiar. We illustrate how we believe a query like "my IP address" is routed through FinSpy Relays to the FinSpy Master in Figure 3.13.

FinSpy servers returning Yahoo

The second most popular page we found returned by FinSpy servers was Yahoo's homepage. Unfortunately, we were unable to coax `yahoo.com` to return the FinSpy Master's IP address in the body. However, we were able to extract the FinSpy Master's rough localization data from the weather and news widgets on the Yahoo homepage.

We sent queries like the one in Figure 3.15 to FinSpy servers that returned Yahoo pages. In some cases we needed to substitute the correct localized version of Yahoo into the GET request and `Host` header (either `espanol.yahoo.com` or `maktoob.yahoo.com`) in order to elicit a server response.[4]

[3]We include the `User-Agent` header because we found that its exclusion resulted in a response without an IP address in the body. We included the `nord=1` parameter ("no redirect") in the query string of our request to disable Google's SSL redirection; if we omitted this parameter, the server returned us a 302 redirect to a localized version of `https://google.com`.

[4]Like our Google FinSpy server query, we include the `User-Agent` header. We circumvent Yahoo's SSL redirection by including the `https` URI schema in the `GET` request.

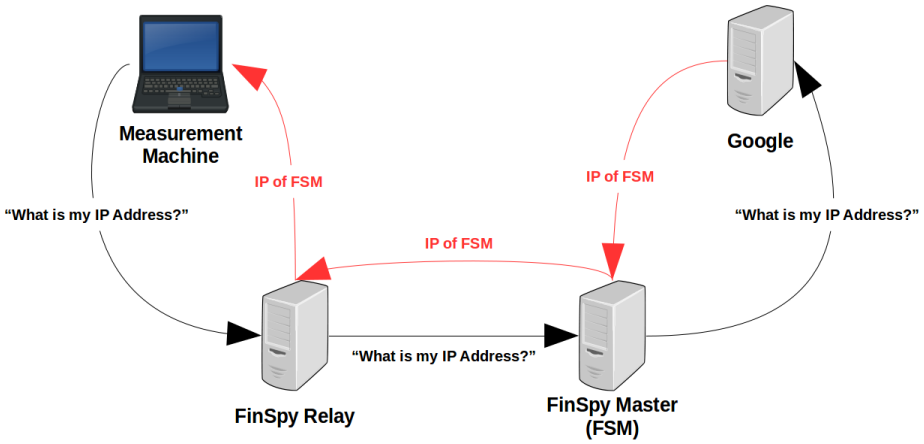


Figure 3.13: How we think a Google search query is routed through FinSpy Relays to a FinSpy Master.

Figure 3.14: Weather conditions in Caracas, Venezuela returned by a FinSpy server in Lithuania.

```
GET https://www.yahoo.com/ HTTP/1.1
Host: www.yahoo.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; rv:38.0) Gecko/20100101 Firefox/38.0
```

Figure 3.15: Yahoo query we sent to FinSpy relays to get location data regarding C&C servers.

Other decoys

While the majority of FinSpy servers we detected used either Google or Yahoo as a decoy page, we identified a number of other servers whose operators had apparently configured the servers to return a different page.

One FinSpy server used the Italian news source `libero.it` as a decoy. We noted that `libero.it` sets the `Libero` cookie, which contains the IP address of the computer that visited the `libero.it` website. We found an Italian IP address in a `Set-Cookie` HTTP header returned by the FinSpy relay, which was not in Italy. Servers that we traced to Macedonia used Macedonian newsmagazine `time.mk` as a decoy. Servers we traced to Taiwan used Taiwanese web portal `pchome.com.tw` as a decoy. We were unable to trace other servers which used file download site `filehippo.com` as a decoy. A handful of other untraceable servers returned custom HTML code as a decoy (e.g., a webpage with a `meta` redirect to `www.google.com`).

3.2.3 Results

We found 123 IP addresses from our *zmap* scans for ψ_2 between December 2014 and June 2015. Of these, 75 (61%) returned a page in response to a `GET /` request, and the remaining 48 (39%) did not. Of the servers that returned decoy pages, 44 (59%) returned `google.com` or its localized version, 18 (24%) returned `yahoo.com` or its localized version, and 13 (17%), returned a different page.

We managed to obtain an IP address from each of the 44 Google FinSpy servers, location data from all but two of the 18 Yahoo FinSpy servers, and an IP address from one of the 13 servers that returned a different page. In total, we found 26 IP addresses that we believe are FinSpy master servers in 15 countries. From these 26 IP addresses, we were able to identify 10 specific government entities that we believe are using FinSpy, based on IP address data. We identified an additional 17 countries where we believe a FinSpy master exists using location information embedded in responses, for a total of 32 countries.

A complete list of countries in which we detected FinSpy masters appears in Table 3.1 on page 60 (**FinSpy** column).

Attribution to specific government entities

We found a FinSpy master in the same `/30` as the mail server for the **Bangladesh Directorate General of Forces Intelligence (DGFI)** [59], `[redacted].dgfi.gov.bd`.

We found a FinSpy master in a `/28` assigned to Belgacom, which contained two other IP addresses pointed to subdomains of `raspol.be`, a domain name registered to the **Belgium Federal Police**.

We found a FinSpy master that matched an IP in the `Received` header of an email in the leaked Hacking Team emails [163]; the email was sent by a Hacking Team employee on June 25, 2015, when he was scheduled to be performing delivery [165] in Egypt for Hacking

Team customer TREVOR, identified as the **Egypt Technology Research Department (TRD)** [166, 120].

We found two FinSpy masters in the same Indonesian `/28`. Another IP address in this same `/28` was included in the `Received` header of an email [205] in the leaked Hacking Team emails; the email was sent by a Hacking Team employee on February 6, 2013, when he was in Indonesia [121] performing a demo for the **Indonesia National Encryption Body (Lembaga Sandi Negara)** [189].

We found a FinSpy master in a range of IP addresses registered to a Kenyan user named "National Security Intelligence" The **Kenya National Intelligence Service (NIS)** was formerly known as the National Security Intelligence Service (NSIS) [150].

We found a FinSpy master in a range of IP addresses registered to a Lebanese user named "General_Security." We assume that "General_Security" is a reference to the **Lebanon General Directorate of General Security**.

We found a FinSpy master at a Lebanese IP address that was formerly pointed to by what was apparently a mail server with domain `[redacted].intelligence.isf.gov.lb` in 2012. We assume that the IP still belongs to the **Lebanon Internal Security Forces (ISF)** [60].

We found a FinSpy master in a range of IP addresses registered to a Moroccan user named "Conseil Superieur De La Defense Nationale." We assume that this is a reference to the eponymous agency, the **Morocco Supreme Council of National Defense (CSDN)**.

We found a FinSpy master at a Mongolian IP address in the same `/28` as an IP address pointed to by the domain `td.sssd.mn`. Individuals with email addresses on this domain were said to be from the **Mongolia State Special Security Department (SSSD)**, according to a leaked Hacking Team email [195].

3.3 Hacking Team RCS

We devised fingerprints for the behavior of RCS C&C servers by analyzing copies of RCS along with related malware known as FSBSpy. We started with the UAE RCS sample from Ahmed Mansoor (Section 2.4.1), and additional samples obtained from VirusTotal by searching for AV results containing the strings "DaVinci" and "RCS." At the time, several AV vendors had added detection for RCS based on a sample analyzed by Russian AV company Dr. Web [31] and the UAE RCS sample sent to Ahmed. We also similarly obtained and analyzed samples of FSBSpy [67], a piece of malware that can report system information, upload screenshots, and drop and execute more malware. At the time of our initial analysis, FSBSpy was believed to be related to Hacking Team, because samples of FSBSpy were seen signed by the same code-signing certificate that was used to sign RCS samples. It was later revealed that FSBSpy was the first-stage implant of a newer version of Hacking Team's RCS spyware.

3.3.1 Initial scanning

We probed the C&C servers of the RCS and FSBSpy samples, and looked at their historical scanning responses. We found that servers responded in odd ways to HTTP `GET /` requests. Broadly, the HTTP responses appeared to be either error pages, or `meta` redirects to other pages. We noticed some variation in the body of HTTP responses, and less variation in the headers, so we attempted to develop our fingerprints based on response headers. Some servers responded as in Figure 3.16. We developed a fingerprint ρ_1 (Figure 3.17) based on the ordering of the headers in the response. We required that the HTTP response match both regular expressions specified in Figure 3.17. While the response in Figure 3.16 shows a `meta` redirect to Google, we identified several other URLs that were used by operators, including homepages for a number of US corporations.

```
HTTP/1.1 200 OK
Connection: close
Content-Type: text/html
Content-length: 83

<html> <head><meta http-equiv="refresh" content="0;url=http://www.google.com
"></head> </html>
```

Figure 3.16: An example of an RCS server response matching ρ_1.

```
/HTTP\/1.1 200 OK\r\n(Connection: close\r\n)?Content-Type: text\/html\r\
nContent-[lL]ength: [0-9]+\r\n(Connection: close\r\n)?(Server: Apache.*\r\n)?\
r\n/

/Connection: close\r\n/
```

Figure 3.17: Fingerprint ρ_1.

Other servers responded as in Figure 3.18, which seemed to be designed to appear as an Apache HTTP error page, though containing a missing space in the HTTP response "`HTTP/1.1 404 NotFound`" instead of "`HTTP/1.1 404 Not Found`". We developed a fingerprint ρ_2 based on this response (Figure 3.19).

We also noted that many of the severs that matched ρ_1 or ρ_2 returned one of several distinctive TLS certificates. We devised three TLS fingerprints for RCS servers, σ_1, σ_2, and σ_3 (Algorithm 2). Fingerprint σ_1 was striking, as servers identified themselves with the name of the product "`RCS`" and the name of the company "`HT srl`" (Hacking Team srl).[5] Fingerprint σ_3 was noteworthy, as the issuer and subject's common name appeared to be

[5]SRL is the Italian designation for a Limited Liability Company.

```
HTTP/1.1 404 NotFound
Connection: close
Content-Type: text/html
Content-length: 276
Server: Apache/2.4.4 (Unix) OpenSSL/1.0.0g

<!DOCTYPE HTML PUBLIC \"-//IETF//DTD HTML 2.0//EN\">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL / was not found on this server.</p>
<hr>
<address>Apache/2.4.4 (Unix) OpenSSL/1.0.0g Server at $HOST Port 80</address>
</body></html>
```

Figure 3.18: An example of an RCS server response matching ρ_2.

```
/HTTP\/1.1 404 NotFound\r\n(Connection: close\r\n)?Content-Type: text\/html\r\
nContent-[lL]ength: [0-9]+\r\n(Connection: close\r\n)?(Server: Apache.*\r\n)?\
r\n/

/Connection: close\r\n/
```

Figure 3.19: Fingerprint ρ_2.

random 11-character strings, and the certificate was marked as a *TLS client authentication* certificate (i.e., a certificate that a client would present to a TLS server to authenticate the client). TLS servers are expected to return certificates marked for use in *TLS server authentication*.

All active servers matching one of our signatures also responded peculiarly when queried with particular ill-formed HTTP requests (Figure 3.20). We located a project on GitHub, `em-http-server` [151], that returned this same peculiar response. The project's author is listed as Alberto Ornaghi, a software architect at Hacking Team. We suspected that the Hacking Team C&C server code may incorporate code from this project. We later confirmed this suspicion by examining the leaked source code from Hacking Team [184].

3.3.2 Deanonymizing RCS servers

We identified many cases where several servers hosted by different providers, and in different countries, returned identical TLS certificates matching our fingerprints σ_1, σ_2, and σ_3.

Algorithm 2 TLS fingerprints for RCS.

```
1: procedure RcsTls(cert)
2:     if cert.issuer matches /^\/CN=RCS Certification Authority \/O=HT srl$/ then
3:         return σ1
4:     end if
5:     if cert.issuer matches /^\/CN=default$/ then
6:         if cert.subject matches /^\/CN=server$/ then
7:             return σ2
8:         end if
9:     end if
10:    if cert.issuer matches /^\/CN=.{11}$/ then
11:        if cert.subject matches /^\/CN=.{11}$/ then
12:            if cert.serial == 1 then
13:                if cert.extendedKeyUsage == "TLS Web Client Authentication" then
14:                    if cert.authorityKeyIdentifier contains cert.issuer then
15:                        return σ3
16:                    end if
17:                end if
18:            end if
19:        end if
20:    end if
21: end procedure
```

```
HTTP1/1 400 Bad request
Connection: close
Content-type: text/plain

Detected error: HTTP code 400
```

Figure 3.20: An odd response from RCS servers.

We also observed 38 servers matching our HTTP fingerprints ρ_1 and ρ_2, which had a *global IPID* (Section 2.3.1).

The RCS servers we found with global IPIDs appeared to be located in countries that we saw less frequently in our scan results, whereas servers with SSL certificates appeared to be in countries that we saw more frequently. Based on this observation, we hypothesized that the servers returning SSL certificates were forwarding traffic to the servers with global IPIDs.

In order to establish whether a server, X, forwards traffic to another server with a global IPID, Y, we first waited until Y was idle (i.e., not sending any packets, as measured by repeated IPID probes). We then sent an IPID probe to Y, followed by a `HTTP HEAD /` request to X, followed by another IPID probe to Y. We looked for a gap (a difference greater than one) between the first and second IPID values; a gap would imply that Y responded to, and thus processed and received, our request to X. If this is the case, we say that X is a proxy for Y. We repeated the experiment more than ten times to be sure, checking in each case that Y was idle. We summarize this procedure in Algorithm 3.

If we concluded that X was a proxy for Y, we next asked whether Y was an *endpoint*, or whether it was proxying traffic to some further node Z. We noticed that when we sent an `HTTP HEAD` or `HTTP GET` request to some Y, the IPIDs of all IP packets returned by Y were consecutive. In other words, Y could not have been communicating with any other server by creating new packets, in responding to our request. However, Y could have been forwarding all existing packets of our request by rewriting destination addresses (e.g., Network Address Translation). This would not induce the creation of any new packets, so it would be consistent with observed consecutive IPIDs. However, this type of forwarding would likely still be measurable in latency (round trip time) differences between Y and other IP addresses in the same block. In order to determine whether this was the case, we compared the latency between our measurement machine and Y (measured using `hping` [83] in both TCP and ICMP modes) with the latency between our measurement machine and other IP addresses in the same block as Y.

3.3.3 Results

We identified 555 IP addresses in 48 countries matching our fingerprints, between May 2012 and February 2014. Of these, 38 IPs were endpoints, located in 21 countries, which we suspected were government clients of Hacking Team's spyware. We compared our results against the leaked Hacking Team data published in July 2015. The leaked data contained a spreadsheet [180] that listed Hacking Team's current and former customers, along with the month and year they first became a customer, and the month and year that their support license was set to expire (or had expired). The spreadsheet identified customers in 38 countries. Seven of these customers were not clients until after we published our scan results in February 2014. Thus, our scans detected 21/31 (68%) of governments using the spyware, and missed 10 governments (32%) using RCS, perhaps reflecting the fact that some govern-

Algorithm 3 Testing whether RCS server X forwards to Y.

```
1: procedure CheckProxy(X,Y)
2:     if Y has a global IPID then
3:         for try ∈ Range(0,10) do                    ▷ Run ten tests
4:             do                                      ▷ Wait for Y to be idle
5:                 i ← GetIPID(Y)
6:                 Sleep(2)                            ▷ Sleep for two seconds
7:                 j ← GetIPID(Y)
8:             while j ≠ i + 1
9:
10:            HTTPGet(X)
11:            i ← GetIPID(Y)
12:            if i == j + 1 then                      ▷ If Y did not respond to X
13:                return false
14:            end if
15:        end for
16:        return true                                 ▷ If all ten tests succeeded
17:    end if
18: end procedure
```

ments had firewalled their endpoint servers to accept connections only from anonymizers, as recommended by Hacking Team.

Table 3.1 summarizes our results in column **RCS**. We denote Hacking Team clients that our scans failed to detect with "Y*", and we denote governments that became Hacking Team clients after our scanning concluded in February 2014 with "Y†". Governments that were Hacking Team clients, which we detected, we denote with "Y". All countries our scans identified as likely Hacking Team clients were actually clients, according to the leaked spreadsheet.

We also provide each country's ranking in the 2015 *Economist Intelligence Unit (EIU)* Democracy Index [38] (Column **EIU**), which ranks 167 countries on a range of democratic properties such as their electoral processes, and civil liberties. We believe that countries without democratic systems are less likely to have surveillance oversight, which may lead to abusive surveillance.

The EIU values of FinFisher and Hacking Team customers range from 6 (Switzerland) to 162 (Turkmenistan). The median EIU value is 84, which is in the range of values defined by The Economist as *semi-authoritarian*. Our results suggest that these type of tools are likely used for both legitimate and abusive surveillance, and point to other countries where the type of engagement we undertook with dissidents in Chapter 2 may reveal abusive surveillance.

Country	FinSpy	RCS	EIU
Angola	Y		131
Azerbaijan		Y	149
Bahrain	Y	Y*	146
Bangladesh	Y		86
Belgium	Y		26
Bosnia-Herzegovina	Y		104
Brazil		Y†	51
Chile		Y†	30
Colombia		Y	62
Cyprus		Y*	39
Czech Rep.	Y	Y*	25
Ecuador		Y*	83
Egypt	Y	Y	134
Ethiopia	Y	Y	123
Gabon	Y		124
Honduras		Y†	84
Hungary		Y	54
Indonesia	Y		49
Italy	Y	Y	21
Jordan	Y		120
Kazakhstan	Y	Y	140
Kenya	Y		93
Lebanon	Y	Y†	102
Luxembourg		Y*	11
Macedonia	Y		78
Malaysia	Y	Y	68
Mexico	Y	Y	66
Mongolia	Y	Y*	62
Morocco	Y	Y	107
Nigeria	Y	Y	108
Oman	Y	Y	142
Panama		Y	45
Paraguay	Y		71
Poland		Y	48
Romania	Y		59
Russia		Y*	132
Saudi Arabia	Y	Y	160
Serbia	Y		58
Singapore		Y*	74
Slovenia	Y		36
South Africa	Y		37
South Korea		Y	22
Spain	Y	Y*	17
Sudan		Y	151
Switzerland		Y†	6
Taiwan	Y		31
Thailand		Y	98
Turkey	Y	Y	97
Turkmenistan	Y		162
UAE		Y	148
Uganda		Y†	96
USA		Y*	20
Uzbekistan		Y	158
Vietnam		Y†	128

Table 3.1: Government users of FinSpy and RCS, according to scanning and leaked Hacking Team data. Y: Countries where we detected a FinSpy or RCS endpoint; Y*: Countries that were RCS users during our scanning period that we failed to detect, according to 2015 leaked data; Y†: Countries that became RCS users after the end of our scanning, according to 2015 leaked data. Note that all countries where we detected an RCS endpoint had at least one government client during our scan period. Leaked data from FinFisher did not include identities of clients.

3.4 Stealth Falcon

While analyzing Stealth Falcon (Section 2.4.3), we noted that they used a malicious URL shortener, `aax.me`, and contacted some victims publicly on Twitter. We first searched Twit-

ter for instances of `aax.me` links. The links we found indicated that all `aax.me` links conformed to a simple pattern, affording us the opportunity to exhaustively probe `aax.me` for all possible short URLs. Our findings point to a UAE-focused operator, whose bait content and targets are predominately linked to the Emirates.

Our search of Twitter for `aax.me` links yielded public tweets targeting 24 accounts. Four of these accounts were suspended or had apparently been deleted. The remainder of the accounts all had compelling UAE links, such as tweeting in solidarity with UAE political prisoners, or listing their location as the UAE. We were able to link seven of the accounts to specific individuals or cases (Table 3.2). Of the accounts we have been able to identify, several individuals were subsequently arrested or convicted in absentia by the UAE Government in relation to their online activities.

Enumerating aax.me for Bait Content

All of the public `aax.me` links we found, as well as the links sent to Donaghy (Section 2.4.3), matched the regular expression `/aax\.me\/[0-9a-f]{5}/`. Assuming all links shortened via aax.me match this regular expression, there are only 16^5 (1,048,576) possible short URLs. We sent a request to `aax.me` for each possible URL, and observed the returned page or redirect. We found 57 URLs that exhibited the redirect.php profiling behavior, and 524 URLs that returned an `HTTP 302` redirect to an expanded URL. The other 1,047,995 `aax.me` links returned a `HTTP 302` redirect to the `aax.me` homepage; we assume these short URLs were unassigned to an expanded URL, as of the time of our scan.

We assigned labels to the long URLs where the URLs were still active, or where we could find an archived copy of, or some information about, the URL. We were able to assign at least one label to 535 URLs, and failed to label 46 URLs as the corresponding websites were down, and we could not find reliable information about what content the URLs contained. We labeled 133 URLs as "advertisement" (25% of all labeled URLs), as they appeared to represent an advertisement for a product. The vast majority of these advertisements were for products typically marketed via spam (e.g., "dietary supplement" or "green coffee"). We suspect that these links may have been shortened by spammers, as the `aax.me` URL shortening page was publicly accessible and indexed by Google, and YOURLS advises that publicly accessible URL shorteners will receive spam [173]. All "advertisement" links were 302 redirects, and none were `redirect.php` links. This is consistent with our observation that the `aax.me` public interface only permits visitors to shorten links using the 302 redirect method.

We filtered out the short URLs labeled as "advertisement," suspecting these to be the work of spammers rather than operators. There were 402 non-advertisement short URLs that we labeled. We display a summary of the top ten labels in Table 3.3.

We noted that a number of long URLs had multiple corresponding short URLs. We display the top ten long URLs in Table 3.4.

Handle	Targeting	Arrests or Convictions	Note
@omran83	14/1/2012 [19]	16/7/2012 (arrested) [88]	UAE94 prisoner; serving 7 years in prison [87].
@weldbudhabi	5/8/2012 [96]; 20/10/2012 [2]	14/12/2012 (arrested) [114]	Identified "plainclothes officers who work for the [UAE] interior ministry's security agency."
@intihakat	5/8/2012 [97]	25/12/2013 (convicted) [86]	Qatari convicted in absentia of damaging the reputation of UAE Supreme Court; sentenced to 5 years in prison.
@bukhaledobaid	24/4/2013 [47]	2/7/2013 [17]; 12/12/2013 (arrested) [44]	Brother of UAE94 prisoner; acquitted of charges; indefinitely detained in prisoners ward of hospital.
@northsniper	7/11/2013 [48]	18/5/2015 (convicted) [103]	Five Qataris convicted for insulting UAE rulers; sentences ranged from 10 years to life in prison.
@71uae	9/1/2012 [141]		Last tweeted 1 July 2013, a day before arrest of **@bukhaledobaid**.
@kh_oz	10/1/2012 [142]		Likely son of **@bukhaledobaid** [104].
@shamsiuae58	9/5/2013 [169]		Signed 2011 UAE pro-democracy petition that Ahmed Mansoor was arrested after signing.
@newbedon	9/1/2012 [158]		Donaghy describes the account as "ensur[ing that] details of mistreatment [by UAE security forces] are readily available" [40].
@bomsabih	9/1/2012 [98]		Inactive since 8/10/2014. Owner claimed affiliation with UAE State Security Apparatus.

Table 3.2: Twitter users linked to the UAE who were contacted by Stealth Falcon attackers.

Label	Number of Short URLs	% of non-ad URLs
UAE	292	73%
Torture	57	14%
Security Forces	49	12%
Denaturalization	46	11%
Isa bin Zayed	42	10%
Rule of Law	40	10%
Criticism	40	10%
ABC News	40	10%
Violations	33	8%
Islam	29	7%

Table 3.3: Top ten labels we assigned to `aax.me` bait content.

Infrastructure Analysis of Stealth Falcon Command & Control

This section describes how we traced the spyware Rori Donaghy received to a network of live C&C servers and domain names.

We sent a `HTTP GET /` request to, and examined the response from, `adhostingcaches.com`, the replacement domain name for `adhostingcache.com`, the C&C server for the stage1 spyware that Donaghy received. We noted that the server returned an unusual response similar to the one in Figure 3.21. Oddly, the response indicated it was returning a 0-byte `application/x-shockwave-flash` file. We searched Internet scanning results using the fingerprint in Figure 3.22, and found 14 additional IP addresses, which we linked to 11 domain names using PassiveTotal [162].

```
HTTP/1.1 200 OK
Date: Mon, 29 Feb 2016 18:43:04 GMT
Content-Length: 0
Content-Type: application/x-shockwave-flash
Server: Apache/2.4.9
```

Figure 3.21: An example of a Stealth Falcon stage1 server response.

```
/^HTTP\/1.1 200 OK OK\r\n.*\r\nContent-Length: 0\r\nContent-Type: application
\/x-shockwave-flash\r\n.*/
```

Figure 3.22: Our stage1 Stealth Falcon fingerprint.

Long URL	# Short URLs	Description
http://www.youtube.com/watch?v=F6NU4pc378k	40	ABC News report featuring video of Abu Dhabi Crown Prince's brother, Sheikh Isa bin Zayed al-Nahyan, torturing an Afghani grain salesman.
http://mohaamoon.com/uae/17.htm	40	Personal website criticizing rule of law and human rights issues in the UAE, including torture, slavery, and imprisonment for debts.
https://r7aluae2.wordpress.com/2012/01/09/اتحاد-المنظمات-الإسلامية-في-أوروبا-يس/	19	Copied statement from the Federation of Islamic Organizations in Europe (FIOE), criticizing the UAE's denaturalization of citizens.
https://www.a7rarelemarat.com/vb	10	Purported to be an opposition web forum for discussing Emirati issues, and providing proxy tools. The site is now down, so we cannot inspect the specific forum posting.
http://google.com	9	Google search engine.
https://www.a7rarelemarat.com/vb/showthread.php?p=3423#post3423	6	(see a7rarelemarat above)
http://www.youtube.com/watch?v=Xcc9Tdc_Hxg&feature=player_embedded#!	5	Video montage talking about torture by UAE security forces.
http://www.youtube.com/watch?v=izeSn9Am6us&list=UU2wwG6r1J_GRgXuMGi9m8FQ&index=1&feature=plcp	5	Video unavailable.
https://www.youtube.com/watch?feature=player_embedded&v=Q3aQpfyXSrg	5	Video published by Al Islah, which appears to be a montage of UAE political detainees.
https://www.a7rarelemarat.com/vb/forumdisplay.php?f=3	5	(see a7rarelemarat above)

Table 3.4: Most popular bait content on aax.me.

Nine domain names were chosen to look like Internet backend servers (e.g., `simpleadbanners.com`, `clickstatistic.com`), whereas two appear to be thematically related to travel (`bestairlinepricetags.com`, `fasttravelclearance.com`), perhaps indicating that some bait content or targets were related to travel.

We next examined the behavior of `incapsulawebcache.com`, the C&C server for the stage2 spyware that Donaghy received, in response to an `HTTPS GET /` request. We found that it returned a response similar to the one in Figure 3.23. At first glance, the response appears similar to that of a server running Python's *TwistedWeb* [111] web server. However, TwistedWeb's default session management behavior is to set a cookie called `TWISTED_SESSION` [112]. In our case, we observed a cookie simply called `SESSION`. We searched Internet scanning results for the fingerprint in Figure 3.24.

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
Date: Tue, 18 Oct 2016 14:43:45 GMT
Content-Type: text/html
Server: TwistedWeb/15.0.0
Set-Cookie: SESSION=dc5b49bc50a32f2eadc531f869271a46; Path=/
```

Figure 3.23: An example of a Stealth Falcon stage2 server response.

```
/^HTTP\/1.1 200 OK\r\nTransfer-Encoding: chunked\r\n.*\r\nServer: TwistedWeb
\/.*\r\nSet-Cookie: SESSION=.*/
```

Figure 3.24: Our stage2 Stealth Falcon fingerprint.

In total, we associated 67 active (and 30 historical) IP addresses with the stage2 spyware. We used PassiveTotal to link 69 domain names with these IP addresses, the earliest registered on 28 January 2013, and the most recent registered on 19 April 2016. The vast majority of the domains are named like generic Internet backend servers. One domain name appears to be travel-themed (`airlineadverts.com`), and two appear to be news and/or government themed (`ministrynewschannel.com`, `ministrynewsinfo.com`).

The domain names we found were typically registered with WHOIS privacy providers. In some cases we were able to obtain the true registration email through historical WHOIS, or the domain's DNS SOA record. We traced several additional domains to Stealth Falcon using WHOIS information. Of these, one was designed to impersonate a China-based provider of VoIP solutions (`yeastarr.com`), and two appeared to perhaps contain the Arabic word for security, "amn," (`amnkeysvc.com`, `amnkeysvcs.com`).

Even when they failed to use WHOIS privacy protection, the operators typically practiced disciplined operational security: we rarely found an email address that was used to register two domains, and we rarely found two domains linked to the same IP address. However,

there were a few cases where operators made mistakes, which allowed us to greatly expand our insight into their activity. We describe one such mistake in Section 3.5.

3.5 NSO Group Pegasus

In this section, we describe how we attributed the attack on Mansoor (Section 2.4.5) to NSO Group, using both elements of the attack, as well as through our empirical characterization of Stealth Falcon. Figure 3.25 summarizes our attribution.

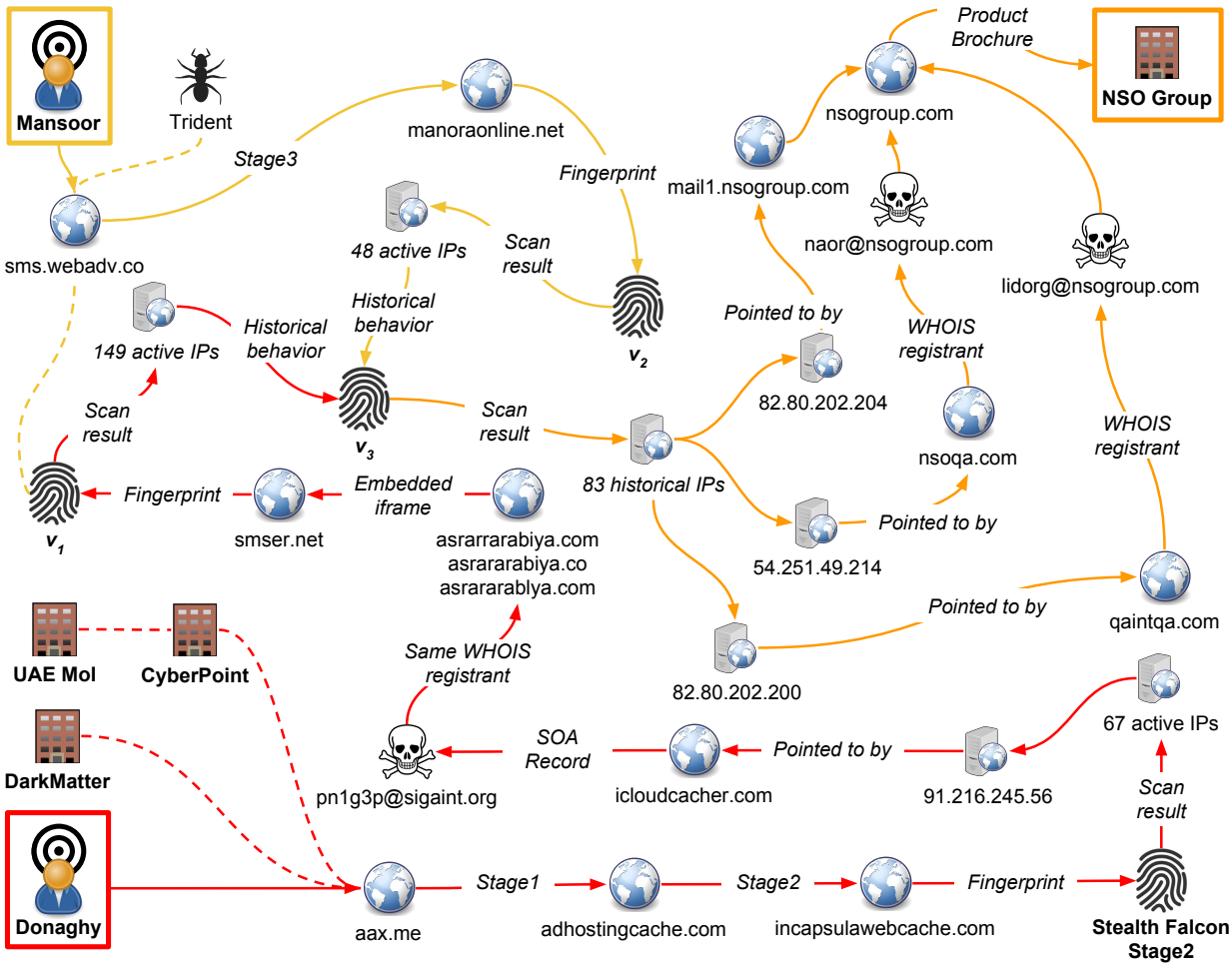


Figure 3.25: How we attributed to NSO Group the spyware that Mansoor received. We started from the Stealth Falcon targeting of Donaghy (bottom left; red), and the SMS targeting of Mansoor (top left; yellow), and worked our way to the NSO attribution (top right; orange).

Recall the attack on **Donaghy** (Section 2.4.3), which we show in the bottom-left of Figure 3.25 (red elements). Donaghy received a link to `aax.me`, from which a stage1 was downloaded, communicating with `adhostingcache.com`, which in turn downloaded a stage2 communicating with `incapsulawebcache.com`. We fingerprinted the stage2 C&C server (Section 3.4), and discovered 67 IP addresses.

One IP address we found, `91.216.245.56`, was pointed to by a domain name `icloudcacher.com`,[6] registered to `pn1g3p@sigaint.org`, according to data in its DNS SOA record. The same email address appeared in WHOIS records for the following three domains:

```
asrarrarabiya.com
asrararabiya.co
asrararablya.com
```

When we became interested in these domains, the IP address they pointed to no longer accepted requests on ports 80 (HTTP) or 443 (HTTPS). However, we found an entry in Google's Cache for each domain. Google obtained the response in Figure 3.26 from each domain. The response contains an `iframe` pointing to the website `asrararabiya.com` ("Asrar Arabiya," or "Arabian Secrets" in English), which appears to be a benign website that takes a critical view of the Arab World's "dictatorships." The response also contains a nearly invisible iframe pointing to a page on another site, `smser.net`.

```
<iframe src="https://smser.net/9918216t/" width="1" height="1" border="0"></
iframe>
<iframe src="http://asrararabiya.com/" style="width:100%; height:1200px;
position:absolute; top:-5px; left:-5px;" border="0"></iframe>
```

Figure 3.26: Source code of Google Cache result for fake *Asrar Arabiya* domains linked to Stealth Falcon.

We suspected that the three domains we identified were attempting to mislead users into believing they were visiting the legitimate `asrararabiya.com` website. Since we had linked the domains to Stealth Falcon, we suspected that the additional domain, `smser.net`, might be an attack domain. We visited the URL in the `iframe`, `https://smser.net/9918216t/`, and were redirected to `https://smser.net/redirect.aspx`. That URL returned the response in Figure 3.27 (headers omitted). We called this response fingerprint ν_1, and scanned the Internet by sending an HTTPS `GET /redirect.aspx` request, and checking if the response matched. We found 149 IPs matching ν_1, including an IP addres pointed to by `sms.webadv.co`, the domain name of the spyware link that Mansoor received (Section 2.4.5).

[6]The phrase "cache" was popular in Stealth Falcon domains, appearing in 2/13 stage1 domain names, and 22/69 stage2 domain names.

```
<html><head><meta http-equiv='refresh' content='0;url=http://www.google.com'
/><meta http-equiv='refresh' content='1;url=http://www.google.com' /><title></
title></head><body></body></html>
```

Figure 3.27: NSO Pegasus fingerprint ν_1.

The spyware payload installed by the link that Mansoor received (yellow elements in Figure 3.25) communicated with `https://manoraonline.net/Support.aspx`. When we sent an HTTPS `GET /Support.aspx` to `manoraonline.net`, we received the response in Figure 3.28. From this, we devised fingerprint ν_2 (Figure 3.29). We scanned the Internet, sending HTTPS `GET /Support.aspx` requests, and found 48 IPs matching ν_2.

```
HTTP/1.1 302 Found
Location: http://www.google.com/
Date: Sun, 14 Aug 2016 02:38:39 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 139
Connection: keep-alive
Cache-Control: Private
```

Figure 3.28: An example of an NSO Pegasus server response matching ν_2.

```
/^HTTP\/1.1 302 Found\r\nLocation: http:\/\/www\.google\.com\//
```

Figure 3.29: NSO Pegasus fingerprint ν_2.

3.5.1 Attribution to NSO Group

We examined the IP addresses we found matching ν_1 and ν_2 in historical HTTP scanning results (*sonar-http* [159]), and found 19 of these IPs previously returned a different distinctive response, which we called fingerprint ν_3 (Figure 3.30). The response ν_3 seemed distinctive because of its capitalization, the positioning of newlines, and the presence of the Unicode "byte order mark" (BOM), `\xef\xbb\xbf`. We searched the same historical HTTP scanning data for ν_3, and found 83 IPs that returned it (orange elements in Figure 3.25).

Three of these 83 IPs were connected to NSO Group. The IP `82.80.202.200` matched ν_3 between October 2013 and September 2014, at the same time that the domain name `qaintqa.com` pointed to it. According to WHOIS, The domain's registrant is `lidorg@nsogroup.com`. The `nsogroup.com` website is mentioned in an NSO Group brochure [71] that was posted on SIBAT (The International Defense Cooperation Directorate

```
\xef\xbb\xbf<HTML><HEAD><META HTTP-EQUIV="refresh" CONTENT="0;URL=http://www.
google.com/">
<TITLE></TITLE></HEAD><BODY>
</BODY></HTML>
```

Figure 3.30: NSO Pegasus fingerprint ν_3.

of the Israel Ministry of Defense). The IPs `82.80.202.204` and `54.251.49.214` matched ν_3 in March 2014. The former was pointed to by `mail1.nsogroup.com` from September 2014 until May 2015,[7] the latter was pointed to by `nsoqa.com` from September 2015 onwards.[8] Both domains are registered to **NSO Group** (orange box in Figure 3.25).

3.5.2 Pegasus domain names

We suspected that the IPs we found matching ν_1 were Pegasus *installation severs*, and the IPs we found matching ν_2 were Pegasus *data servers*.

We identified domain names associated with IPs we found matching ν_1, ν_2, and ν_3 both from SSL certificates, and passive DNS data. In total, we identified 312 domain names. We categorized the domain names we found, and identified several common themes, perhaps indicating the type of bait content that targets would receive. Interestingly, the most common theme among the domains we identified was "News Media," perhaps indicating the use of fake news articles to trick targets into clicking on spyware links. We display the top 15 categories in Figure 3.31, and some examples for each category in Table 3.5.

3.6 Conclusion

Despite vendor claims that tracing attacks conducted with their products back to specific nation-state attackers is "impossible" [93, 183], we showed that spyware products from two companies, FinFisher and Hacking Team, had deficiencies that easily supported this kind of attribution. Even in cases where we were unable to conclusively establish the identity of attackers, we were able to characterize or enumerate the bait content used to attack targets.

The commercial spyware tools for targeted surveillance that we examined appear to be broadly used around the world, by democracies and dictatorships alike, calling into question the quality of human rights "oversight" that surveillance companies claim to practice over their sales [65]. Our findings of several government users ranked as "authoritarian regimes" by the Economist Intelligence Unit suggests that future engagement with potential targets in these areas may yield evidence of abusive surveillance.

[7] According to PassiveTotal.
[8] According to DomainTools.

Type	Example	Impersonating
News Media	alljazeera.co	Al Jazeera
	bbc-africa.com	BBC
	cnn-africa.co	CNN
	unonoticias.net	Las Ultimas Noticias
	univision.click	Univision
SMS Gateway	bahrainsms.co	
	kenyasms.org	
ISP - Telco	mz-vodacom.info	Vodacom (Mozambique)
	iusacell-movil.com.mx	Iusacell (Mexico)
	sabafon.info	Sabafon (Yemen)
	newtarrifs.net	Generic
Streaming - Sharing Media	uzbektube.net	
	hotmusic.ae	
Ecommerce	thedubaimal.net	Dubai Mall
	bestbargain.co.il	
Unsubscribe	unsubscribeinhere.com removefrommailinglist.com	
Account Information	accounts.mx adjust-local-settings.com	
Shipment Tracking	track-your-fedex-package.org	FedEx
Social Networking	fb-accounts.com	Facebook
	whatsapp-app.com	WhatsApp
Corporate Account	addingclients.com	
Religious	quranblogmaster.com	
Government	topcontactco.com	Teleperformance Visa Application Processing Portal
	juridica-data.com	Central de Informacion Judicial
Document - Sharing	download-drive.online	
	document-pdf.biz	
Other	redcrossworld.com	Red Cross
	pickuchu.com	Pokemon Company
	turkishairines.info	Turkish Airlines

Table 3.5: Examples of some domain names pointing to NSO Pegasus Installation Servers. These domains would likely be used as part of bait messages sent to targets.

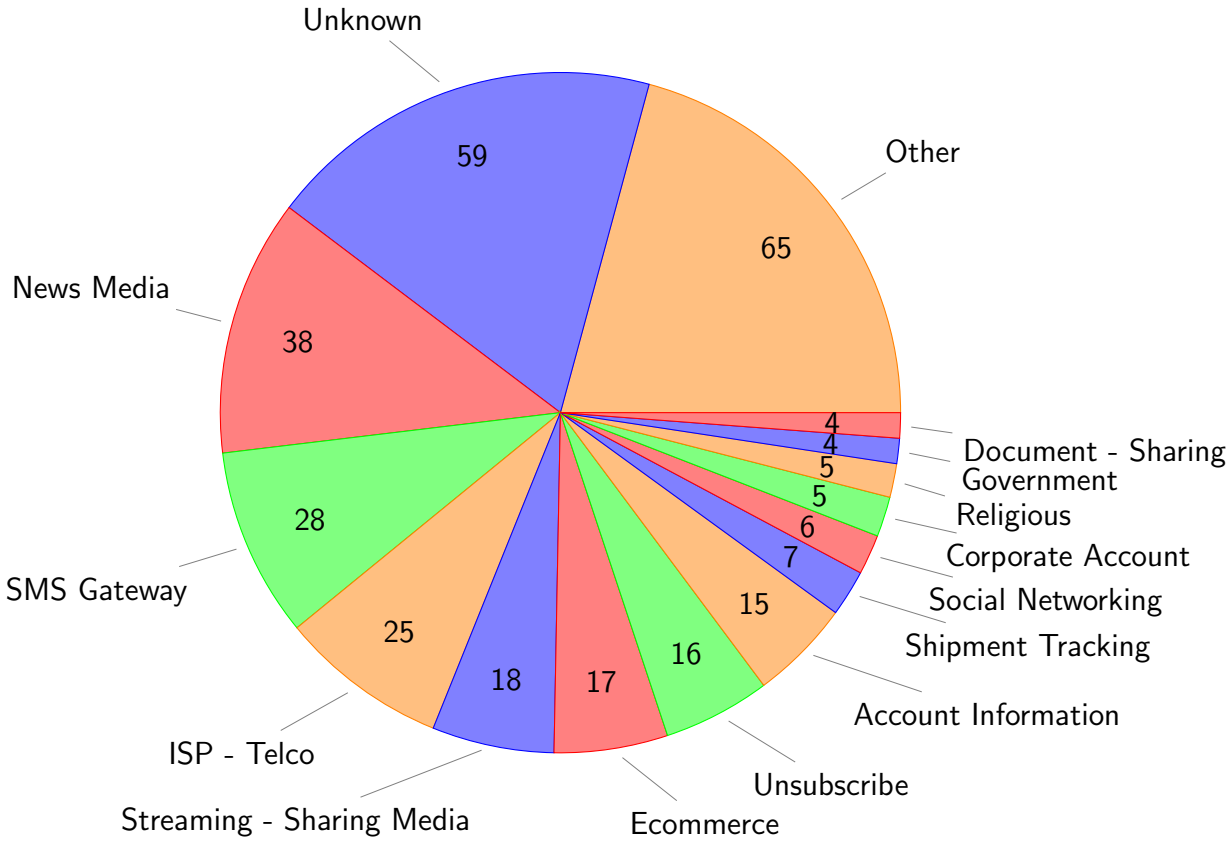


Figure 3.31: Most commonly recurring themes in Pegasus domain names.

Chapter 4

Fieldwork

4.1 Introduction

It is understood that activists, NGOs, and civil society targets are at risk for surveillance; we extensively documented methods of compromise in Chapter 2, and showed similar surveillance tools are in use in countries all around the world (Chapter 3). However, far less is known about how such groups perceive surveillance risks, any relevant training they have received, and how their perceptions and knowledge, along with their dissident activities, shape their security behaviors and risks. To devise effective surveillance defenses for targeted groups, we need to first interact with such groups in detail.

We engaged with 30 targets, on the ground in two Middle Eastern countries, as well as Middle East and Horn of Africa diaspora members overseas, through in-depth (IRB-approved) structured cybersecurity interviews. While the subjects we interviewed reflect the behavior of ordinary users in a number of ways, we also find important differences in terms of the subjects' perceptions of risk (surveillance resulting in government punishment was feared by more than half of on-the-ground activists) and specific security behaviors (e.g., using out-of-country human "password managers" to maintain security of online accounts).

Despite their heightened awareness of risk and steps taken in response to it, on the whole however our results indicate that activists, NGOs, and civil society remain vulnerable to attacks involving *social engineering*, where an attacker tries to convince a target to open a link or attachment included in a message. For example, while subjects reported performing basic vetting before interacting with links and attachments—such as checking the sender ($> 2/3$ of subjects), or evaluating the context of the message ($> 1/3$)—this sort of checking can still prove vulnerable to subtle social engineering, including sender spoofing and "doppelgänger" accounts. This threat particularly looms for attackers with access to a friend or contact's compromised (or seized) account, and indeed 40% of subjects had no strategy to recover their compromised accounts, and 57% reported no strategy if they lost their phone.

Even subjects who report positive security behavior can come up short in implementing it correctly. One subject we interviewed who reported checking a message's sender and con-

text for vetting contacted us several weeks after the interview, stating that they had opened an attachment from an email that they later realized was sent from an account designed to impersonate one of their friends. The attachment was benign, but a link included in the email contained spyware. Similarly, the archival analysis by our *Himaya* tool (Chapter 5) of subjects' past messages reveals instances where subjects who report consciously vetting messages still clicked on social-engineering links. Overall, comparatively fewer subjects reported using tools such as online scanning tools on links or attachments they received, or following up with a message's purported sender through another means of communication before interacting with the message.

Our survey blended interview questions regarding subjects' perception of risk and security behaviors with examination of their computers and phones. After concluding our intervention with a subject, we offered to answer any additional subject questions, and provided them with customized security advice (subjects were not paid for their participation). Subjects asked about the safety of specific chat apps, the capabilities that governments were likely to employ against them, as well as more general technical questions including how to recover deleted files. We review related work in Section 4.2, describe our survey methodology in Section 4.3, present results in Sections 4.4, and identify key take-aways in Section 4.5.

4.2 Related Work

Social engineering of civil society

Previous academic work illustrates targeted nation-state social engineering of activists and civil society [16, 129, 78]. Attackers include government agencies themselves, cyber mercenaries (hackers for hire), and cyber militia groups. Tools used by attackers include malware, exploits, and links sent to pseudonymous accounts that record a clicker's IP address to aid deanonymization by a government. In some cases, attackers use malware purchased from commercial "lawful intercept" vendors such as FinFisher [57], Hacking Team [75], and NSO Group [71]. In other cases, attackers write malware themselves, or employ common Remote Access Trojans (RATs) developed by the cybercrime underground.

In addition, substantial leaked data from FinFisher [27] and Hacking Team [70] reveals product functionality and the operation of surveillance markets. While convincing a target to open a malicious link or file (either with or without an exploit) is a a main vector advertised by these companies, they also offer stealthier infection options including *network injection* hardware, which can be installed on an ISP's backbone to inject malware in targets' unencrypted Internet connections [56, 76].

Previous work also indicates that attackers shift tactics in response to targets' security behaviors. For instance, targets who employed two-factor authentication received specially designed phishing crafted to capture both passwords and authorization codes [167], and a campaign urging Tibetan activists to "detach from attachments" [91] led attackers to instead distribute malicious files via Google Drive links [105].

Studies of user security behavior

Previous work has studied user security behaviors generally, as well as among specific groups. For instance, McGregor et al. [135] studied digital security practices of American and French journalists, a group that reported facing somewhat similar risks to our subjects (e.g., prison, physical danger, and discovery of identity). Authors conducted 15 in-depth interviews, focusing in particular on how journalists' workflows influence their behaviors and use of security tools. Authors noted that several interviewees employed *ad hoc defensive strategies* that sometimes introduced additional vulnerabilities, a finding we share. In contrast, our study focuses primarily on the documented threat of targeted attacks through malicious messages, rather than surveillance more broadly.

Some examples of work studying more general subject populations are Forget, et al. [61], and the *AOL/NCSA Online Safety Study* [7]. In [61], the authors enlisted subjects to install a monitoring agent on their computers to transmit their behavior to researchers for analysis. While the study identified some similar security deficiencies as ours (e.g., lack of security software, out-of-date operating systems and plugins such as Adobe Flash), their subject population (recruited via a university service) likely faces different risks than ours.

In [7], the authors interviewed a sample of 329 adult Internet subscribers in 2004, selected by an "independent market analysis organization" in 22 different American cities and towns. They asked subjects a variety of questions, including how safe subjects felt their computer was from "viruses," "hackers," and "online threats," whether subjects employed antivirus software and firewalls, and how often they updated these. The authors then ran a "scan" of each subject's computer, to verify whether antivirus software and firewalls were present, correctly configured, and up-to-date. A majority of their subjects felt "very safe" or "somewhat safe" from "online threats" (77%), "viruses" (73%), and "hackers" (60%), whereas we classified only 47% of our subjects as believing that their "online activities" placed them at "low risk" (Section 4.4.2). The authors also found that 85% of subjects had antivirus software installed (83% thought they did), though only 33% had virus definitions that were up-to-date (within a week), and 12% had definitions older than six months. Among our subject population, 72% of computers had antivirus software installed, and 14% of installed antivirus software was not up-to-date (in all cases because the update subscription had expired); the discrepancy in update rates may be due to the increasing prevalence (since 2004) of default automatic updates in antivirus programs, and OS warnings if antivirus software is not configured.

It is worth noting that our population is significantly more specialized than general survey populations, such as the sample interviewed by Northwestern University's 2015 *Media Use in the Middle East* survey [137], which polled more than 6,000 Internet users in six Middle Eastern countries about censorship, surveillance, and other issues. Overall, 38% of individuals were "worried about governments checking what I do online." In comparison, we found that 28/30 subjects were concerned about at least one government targeting them.

4.3 Methodology

We interviewed thirty subjects (randomly assigned identifiers S_1–S_{30}) over a two year period between March 2014 and March 2016. We conducted Interviews in two GCC[1] countries, as well in the United States and United Kingdom, where we interviewed human rights workers, and activists originally from the GCC and Horn of Africa (HoA) but now residing abroad.

We obtained verbal consent (in GCC countries) and signed consent (in the US and UK) before proceeding. Consent materials were available in both Arabic and English. All interviews were conducted in English, with the exception of two interviews in GCC countries, which were conducted with translation aid provided by an Arabic speaker proficient in the local dialect.

While in the GCC, we rigidly practiced careful IRB-approved operational security. To protect participants' identities and responses, responses, we implemented IRB-approved measures to reduce our susceptibility to remote tracking, and minimize the amount of information that authorities could recover if we were arrested and forced to reveal passwords while in-country. To avoid undetected physical compromise, we implemented procedures to determine whether our electronic devices had been subject to surreptitious tampering or inspection.

We recruited GCC-based subjects through trusted activist connections on the ground, who invited potential subjects they believed to be at risk of government surveillance. All interviews in the GCC were conducted at a place of the subject's choosing, and we abided by all conditions set by contacts (e.g., one subject requested that we carry on a fictitious conversation suggesting we were old friends while traveling in his vehicle, until we had reached a location where he was comfortable talking freely).

We recruited subjects outside of the GCC by reaching out to our contacts in civil society and human rights organizations.

4.4 Study results

4.4.1 Demographics

Four subjects were from a global human rights organization, Ⓗ, eight were local GCC activists, Ⓖ, eight were GCC activists living abroad, Ⓓ, and ten worked at an out-of-country Horn of Africa (HoA) media outlet, Ⓜ.

Throughout the interview, we asked subjects basic demographic questions (Table 4.1), including whether they had received formal digital security training (**Received training**). The vast majority (>2/3) of subjects had not received formal digital security training. Two subjects mentioned that they provided digital security trainings for others, including one who had not received training themselves.

[1]Gulf Cooperation Council: Bahrain, Kuwait, Oman, Qatar, Saudi Arabia, UAE.

[2]Three were trained by Ⓗ, one by Tactical Tech.

[3]Both trained by same local activist.

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
Female:male	2:2	0:8	3:5	1:9	6:24
Median age	46	31.5	39	38	38
Received training	4/4[2]	2/8[3]	2/8[4]	1/10[5]	9/30
Provided training	0/4	2/8	0/8	0/10	2/30
Highest level of education					
High school	0/4	0/8	1/8	1/10	2/30
Associates	0/4	0/8	0/8	1/10	1/30
High Diploma[6]	0/4	2/8	1/8	0/10	3/30
Bachelors	0/4	2/8	2/8	5/10	9/30
Masters	1/4	3/8	2/8	2/10	8/30
Doctorate	3/4	1/8	1/8	1/10	6/30

Table 4.1: Demographics of study groups.

While we did not ask a specific question about whether respondents had suffered consequences that they believed were linked to their online activity, seven individuals (23%) volunteered this information during the course of the interviews. Three individuals from Ⓖ reported that they had suffered such consequences. One activist (S_2) had been subjected to physical assault. One subject (S_{29}) served a year in prison, and upon his release, found that his Twitter password had been changed. One subject (S_9) was sacked from his job. Two individuals from Ⓓ cited hacking of their email and social media accounts (S_{16}, S_{21}). One subject from Ⓜ (S_{11}) reported receiving threats from authorities: *"I was in* [HoA Country] *last year, authorities told me to leave or die, so I left."* One subject from Ⓜ (S_{22}) reported that his Facebook account was hacked, and hackers posted messages stating that he was working with his country's government. The same subject also reported that a copy of a book he was writing was leaked online (he speculated hacking of the computer or email account of him or his friend he had shared a copy with). The same subject also reported in-country harassment of his brother.

4.4.2 Surveillance risks

To understand perceptions of government targeting, we asked subjects about risks they associated with their online activity, as well as what sorts of attackers they felt might target them online, and to what extent they believed a government was tracking their activities. The overwhelming majority (93%) of subjects mentioned potential government attackers,

[4]One trained by NED [147], one trained by Frontline Defenders [63].
[5]InterNews USA [95] and Google.
[6]One year less than a Bachelors degree.

and 90% indicated the likelihood of the government tracking their online activities to be 50% or greater.

We first asked: *"To what degree do you believe your online activities are safe or place you at risk?"* (Table 4.2). Two answered that they did not know. We classified the rest of the answers into two categories, **High risk** (e.g., *"at risk," "high risk," "70% unsafe"*), and **Low risk** (e.g., *"variable," "50%," "light risk."*)

We next asked subjects, *"What are the risks?"* (Table 4.2). One subject responded that they did not know. Based on subject responses, we devised eight categories: **Surveillance**, which includes answers that cited surveillance of, or theft of private information from computers, phones, or online accounts; **Punishment**, which includes denial of due process, arrest, prison, or physical assault; **Publicity**, which includes smear campaigns, blackmail, or any public disclosure of private information; **Friends**, which includes targeting friends, family members, or contacts; **Financial**, which includes theft of financial information; **Access**, which includes travel bans or deportation; **Damage**, which includes damage to devices or loss of data, and **Cloud**, which includes concern about sharing data with cloud providers.

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
High risk	2/4	3/8	6/8	5/10	16/30
Light risk	2/4	5/8	2/8	3/10	12/30
Risks					
Surveillance	4/4	3/8	7/8	6/10	21/30
Punishment	0/4	5/8	1/8	1/10	6/30
Publicity	1/4	2/8	2/8	0/10	5/30
Friends	0/4	1/8	1/8	2/10	4/30
Financial	0/4	0/8	0/8	2/10	2/30
Access	0/4	2/8	0/8	0/10	2/30
Damage	0/4	0/8	0/8	2/10	2/30
Cloud	0/4	0/8	0/8	1/10	1/30
Total	4/4	8/8	8/8	9/10	29/30

Table 4.2: Subject perception of risks.

We then asked *"Who do you believe might be targeting you due to your online activities?"* If they did not mention a government actor, we asked them *"Do you think you might be targeted by the government?"* 28 respondents were concerned about at least one government targeting them (Table 4.3); one was not concerned about government targeting, but was concerned about targeting by private actors aligned with the government (**Pro-gov**). One respondent (from Ⓜ) was unsure who might be targeting them.

[7]Subjects mentioned governments of countries where they lived, and other countries where they worked.

[8]One subject cited nongovernmental actors that were the targets of their investigations.

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
GCC gov	0/4	7/8	8/8	0/10	15/30
HoA gov	0/4	0/8	0/8	9/10	9/30
Other gov[7]	4/4	1/8	5/8	0/10	10/30
Pro-gov	0/4	4/8	1/8	2/10	7/30
Others	1/4[8]	1/8[9]	2/8[10]	0/10	4/30

Table 4.3: Subject perception of attackers.

We asked subjects to rate the likelihood of the governments they mentioned tracking their online activities on a scale from 1–5 (Figure 4.1).

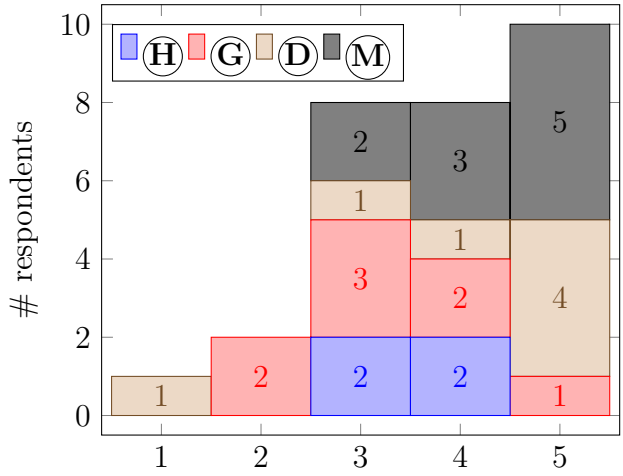


Figure 4.1: "How likely is it that the government is tracking your online activities?" (1: definitely not; 5: definitely)

4.4.3 PC security

We examined subject PCs (and mobile devices, per Section 4.4.4) to see if they exhibited well-known security deficiencies, such as issues with security software, or old versions of plugins. We found that 27% of PCs had no security software, 14% of PCs had expired security software, and 27% of PCs had an old version of Adobe Flash, a common exploit vector for social engineering attacks involving malicious documents.

Of the 29 computer users (S_9 from Ⓖ reported they used an iPad instead of a computer), we were able to examine a total of 22 operating systems on 21 computers (one computer each of 21 subjects who had their computer with them at the interview). We examined seventeen

[9]One subject mentioned "other groups."

[10]One subject mentioned "everybody," another subject mentioned "others."

Windows computers (and an Ubuntu partition on one of these), 3 OSX computers, and one Linux computer.

Table 4.4: PC security deficiencies.

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
No encryption	0/4	5/6	4/5	7/7	16/22
No AV	0/4	3/6	1/5	2/7	6/22
Expired AV	0/4	2/6	1/5	0/7	3/22
Old Flash	1/4	1/6	1/5	3/7	6/22

19 operating systems had Adobe Flash installed. 13 were selected to allow Adobe to automatically install Flash updates (two of these were 991 and 1,064 days out-of-date; we are unsure why). Four Flash installations were set to prompt the user before installing an update (28, 63, 116, and 273 days out-of-date), and two Linux systems had Flash installed through their respective package managers. We denote out-of-date Flash versions as **Old Flash** in Table 4.4.

Of the six operating systems (OSes) we investigated that did not appear to have any antivirus (AV) software (**No AV**), two ran Linux, one ran OSX, and three ran Windows. Three additional operating systems had only an expired AV program (**Expired AV**), which was McAfee in each case. While advanced targeted threats may be engineered to evade antivirus software, these products can help protect against some more common threats [113].

The vast majority of operating systems (73%) did not have disk encryption enabled (**No encryption**). Of the OSes with encryption enabled, four were Windows using BitLocker (a company policy at Ⓗ), one was OSX using FileVault, and one was Linux using DM-Crypt.

4.4.4 Mobile device security

We examined subject mobile phones to check for issues including lax security settings that can ease social engineering (sideloading, rooting), outdated OS versions that can increase odds of successful compromise, and physical security concerns (passwords, encryption, contingency plans for lost devices) that can lead to theft of data or compromise of online accounts by obtaining physical access to a device. We also asked subjects about security behaviors including use of security apps, and responses to application permissions dialogs. We found device physical security to be the area of greatest concern: 68% of subjects did not have an encrypted device, including 5/8 subjects in GCC countries, and 32% did not have a device password. Further, 57% of subjects did not have a contingency plan if they lost their device.

Use of phones

We examined 28 phones, one per subject, except for two subjects (both from Ⓓ) who had time constraints (Table 4.5).

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
Android	3/4	3/8	4/6	8/10	18/28
iOS	0/4	5/8	2/6	2/10	9/28
BlackBerry	1/4	0/8	0/6	0/10	1/28

Table 4.5: Phones we examined.

Multiple Phones: The four members of Ⓗ used BlackBerry phones for their organizational/work email account. Three of them used other phones as well, including for work-related activities. For the subject that exclusively used a BlackBerry, we examined their BlackBerry. In the other cases, we examined their other phones. Three other subjects reported using multiple phones, including one who used two iPhones with different mobile providers, because one had better coverage at his house, and one had better coverage at his office; one who used both iPhone and Android; and one who used three different Android phones. We asked to examine these subjects' primary phone.

One subject (S_{12}) also used several triple-SIM phones that were not smartphones, in order to prevent calls he made to different individuals (using different SIMs) from being linked together by governments in countries where he works. He stated that he has a multitude of SIM cards, and uses certain SIM cards only to talk to certain people, in some cases only calling a single contact from a SIM card. He also stated that he swaps SIM cards frequently as he travels. (We explained that each SIM slot is also uniquely identified with an IMEI, and thus if he swapped the SIM cards around to different slots, this would be one way his SIM cards could be linked together.)

Updates

We measured how many days out-of-date a subject's phone's operating system was by taking the difference between the release date of its OS version, and the release date of the latest version available as of the interview date. Out-of-date phone OS versions can be used to target victims. For instance, Hacking Team had a zero-day exploit for the default web browser in older Android versions 4.0-4.3 [130]. Hacking Team also used several known exploits to *root* older versions of the Android OS, including CVE-2012-6422, CVE-2013-6282, and CVE-2014-3153 [134, 130].

One of the nine iPhones we examined was out-of-date (43 days); the rest of the iPhones were up to date. We plot the results for the 18 Android phones we examined in Figure 4.2.

Security Settings

We checked a variety of security settings on subjects' phones (Table 4.6).

Rooting is the process of gaining administrator-level privileges on a device; on an iPhone, *jailbreaking* includes rooting, as well as circumventing other iPhone security measures. We marked a phone **Rooted** if it was a rooted Android phone or a jailbroken iPhone.

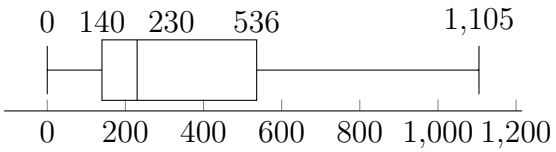


Figure 4.2: Days out-of-date (Android OS version). Horizontal box plot showing quartiles

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
Unencrypted	3/4	5/8	3/6	8/10	19/28
WAP Push	3/4	3/8	3/6	6/10	15/28
Sideloading	1/4	2/8	1/6	2/10	6/28
Rooted	0/4	2/8	0/6	0/10	2/28

Table 4.6: Phone security settings.

One subject (S_8) from Ⓖ jailbroke his iPhone in order to install a second copy of WhatsApp to use with an overseas phone number not linked to his name (to remain anonymous when communicating with certain people). A second individual from Ⓖ (S_{14}) rooted his Android in order to install *X Privacy*, an app that enables fine-grained permissions control over Android apps. Rooting and jailbreaking can be a security risk; both FinFisher and Hacking Team require rooting or jailbreak for all mobile spyware features to be available [55, 197].

Sideloading is the process of installing apps from outside of the phone's app store. Any Android user can enable sideloading in the phone's security settings.[11] The iPhone does not have such an option, but jailbreaking allows sideloading. We counted the number of phones that had sideloading enabled as of interview time. S_{27} from Ⓗ enabled sideloading to install Grooveshark, a (now former) music streaming service of undetermined legality. S_{10} from Ⓓ enabled sideloading to install Aptoide (an alternative app store), and S_{20} from Ⓖ enabled sideloading to install Popcorn Time (a P2P Netflix clone). Two individuals from Ⓜ (S_3, S_{23}) were unsure as to why sideloading was enabled on their phones.

One subject (S_{11}), who did not have sideloading enabled on their current phone, remarked about their past usage of the feature: "*[I installed] an app transfer software. I wanted to transfer one of these apps to my friend. A friend in* [HoA Country] *gave me an APK for the app transfer software, to transfer apps through Bluetooth. The Internet in* [HoA Country] *is so slow, really hard to get a connection.*"

Enabling sideloading can be a security risk, as this allows the installation of apps that have not been vetted by the app store. One attack apparently asked dissidents to install an APK file from a link [134].

[11] When attempting to install an APK file on an Android where sideloading is disabled, the user receives an "Install Blocked" message, with a link to the Android settings page where the user may enable sideloading.

Sideloading can pose an additional concern in concert with **WAP Push** *service* messages. WAP Push messages are SMS messages that may be presented to the user by the phone in a way that makes them appear to originate from the user's mobile phone carrier. WAP *Service Indication (SI)* messages may contain text and a link. Hacking Team and FinFisher documentation suggest that this technique may be used to send targets links to fake updates [118, 54]. WAP *Service Loading (SL)* messages may try to execute an action on a recipient's phone, such as installing an app from a file on a website. As far as we are aware, WAP Push service messages are not supported on iPhones. Fifteen Android phones had WAP Push messages enabled, while requiring a prompt before any action associated with an SL message is taken (the default). Five Android phones did not have any WAP Push message options, and we assumed these did not support WAP Push messages.

We consider iOS versions from 8 onward to be encrypted [9], as well as any Android that has the encryption option enabled.[12] Other phones are marked **Unencrypted**.

One subject whose phone appeared to be encrypted (according to Android settings) complained that they had enabled encryption, but at some point their device stopped prompting them for a password when it booted. Therefore they were unsure as to whether their phone was still encrypted.[13]

One subject whose phone was not encrypted (S_{29} from Ⓖ) mentioned that they wanted to use a pattern password with encryption, but were unable to do this, and that having a password they had to type out was *"too inconvenient."* The same subject was also concerned about the auto-wipe feature. He mentioned a previous occasion on which he set up encryption and enabled auto-wipe, which was set to wipe his phone after five incorrect passwords. The subject was subsequently arrested; police confiscated his phone and input five incorrect passwords, and he lost all of the data on his phone. Another subject (S_9 from Ⓖ) whose phone was not encrypted mentioned that the instructions were too complex, and believed that if he enabled encryption, he could only exchange files with people who had the same version of Android.

We checked what type of password was enabled on each phone (Table 4.4.4).

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
No password	1/4	2/8	1/6	5/10	9/28
Pattern	2/4	2/8	2/6	2/10	8/28
4 digit	1/4	3/8	2/6	1/10	7/28
6 digit	0/4	0/8	0/6	2/10	2/28
Alphanumeric	0/4	1/8	1/6	0/10	2/28

Table 4.7: Phone password settings.

[12] We interviewed one individual in May 2014, who was using iOS 7.1.1 (the latest at the time). We counted their phone as unencrypted.

[13] We still counted this phone as encrypted, because the option for encryption was enabled.

Lost Phone

We asked subjects what they would do if they *"lost or misplaced"* their phone (Table 4.8).

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
No strategy	2/4	4/8	5/8	8/10	17/30
Trace phone	0/4	2/8	2/8	1/10	5/30
Wipe phone	0/4	1/8	0/8	0/10	1/30
Use backup	0/4	1/8	1/8	1/8	3/30
Other	2/4[14]	1/8[15]	1/8[16]	1/10[17]	5/30

Table 4.8: Subjects' action if phone "lost or misplaced."

We classified subjects into the **No strategy** category if they had a response similar to *"nothing," "I don't know,"* or *"pray"* (S_{29}). Of the subjects who had no strategy, one subject from Ⓖ (S_7) said they were aware of the option to remotely wipe their phone, but thought it was too complicated to set up. One subject from Ⓓ (S_{10}) said they formerly used a remote wipe program, but did not set it up on their current phone. One subject from Ⓜ (S_{23}) had a *"vague recollection"* of setting up software to track their phone if it was lost, but had no idea how to use it. Another subject from Ⓜ (S_{22}) said *"I never think about these questions."* One subject from Ⓜ (S_{26}) remarked only: *"I am supposed to back up everything on the cloud, which I have not"* in response to the question.

Security apps

We checked subjects' phones to see if they used several popular messaging programs (Table 4.9).

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
WhatsApp	1/4	8/8	6/6	4/10	21/28
BBM	0/4	1/8	1/6	0/10	3/28
Viber	0/4	3/8	5/6	8/10	17/28
Telegram	0/4	6/8	2/6	1/10	10/28
Signal[18]	0/4	3/8	2/6	0/10	5/28
Line	2/4	2/8	3/6	0/10	8/28

Table 4.9: Installed mobile messaging apps.

[14]Two subjects said they would contact their IT department.
[15]One subject said they would change their passwords.
[16]One subject said they would call the police.
[17]One subject said they would call their phone manufacturer or telecom company.

We asked subjects if they used any security or privacy apps on their phones (Table 4.10).

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
Antivirus	0/4	2/8	2/8	2/10	6/30
Secure chat	0/4	2/8	2/8	0/10	4/30
App lock	0/4	1/8	1/8	0/10	2/30
Other	0/4	4/8[19]	2/8[20]	0/10	4/30

Table 4.10: "Security or privacy apps" mentioned by subjects.

The **Antivirus** apps used by subjects included *Avast*, *Avira*, *Clean Master*, *Lookout*, *Malwarebytes*, and *McAfee*. One subject, S_{21}, was using three different antivirus apps at the same time, and the five other current users of **Antivirus** apps used a single app. Several subjects noted issues with antivirus apps. Subject S_{21} remarked that they formerly used messaging app Telegram, but uninstalled it because Malwarebytes indicated it was malicious. Subject S_5, a McAfee user, expressed their annoyance at how the app would bother them with *"frequent messages and promotions."* Former antivirus app users also noted issues that caused them to stop using such apps. Subject S_{11} said they formerly used *AVG* antivirus on their phone, but uninstalled it because they said they *"don't usually use"* it, and wanted to free up space. Subject S_{23} said they formerly used Avast, but uninstalled it because *"there were too many notifications, it killed most of the activities of my phone, it appeared to be a virus."*

Four subjects cited their use of **Secure chat** apps. S_2 cited *Chatsecure*, S_{15} and S_{21} cited *SureSpot*, and S_{25} cited their use of *Signal*.

Two subjects, S_2 and S_{21}, reported using an app that allowed them to "password protect" other apps (**App lock**). Both subjects used Clean Master for this purpose.

We asked Android subjects: *"Have you ever declined to install an app based on the permissions it requested?"* and iPhone subjects *"Have you ever declined a permission request for an app?"* (Table 4.11). One subject was unsure, four subjects reported that they did not install apps (**Don't install**).

Ten subjects provided additional detail about under what circumstances they declined permissions requests. Subjects cited **Updates** that requested additional permissions, access to their **Location** or **Contacts**, or whether they felt the request was **Unreasonable**. There was no overlap between responses.

Of the subjects who did not decline permissions, S_3 remarked that they *"don't read any of the permissions, just click the agreement and go for it,"* but added that they *"don't use a*

[18]For interviews conducted before the advent of Signal for Android, we also count TextSecure.

[19]Subject S_{14} mentioned using APG with K-9 Mail, X Privacy, Avast Anti-Theft, SyncThing, and Ccleaner. Subject S_{15} mentioned *unfurlr*, and *Video Downloader Pro*, which they employed to "lock files with a PIN code." Subject S_8 mentioned their use of VPN.

[20]Subject S_{10} mentioned "Samsung Security Policy Update." Subject S_{21} mentioned their use of VPN.

[21]S_{27} remarked that they used to decline permissions, but do not do so anymore.

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
Yes	2/4[21]	8/8	6/8	3/10	19/30
No	0/4	0/8	1/8	5/10	7/30
Don't install	2/4	0/8	0/8	2/10	4/30
What is declined					
Location	0/4	2/8	1/8	1/10	4/30
Unreasonable	0/4	2/8	0/8	1/10	3/30
Updates	1/4	0/8	1/8	0/10	2/30
Contacts	0/4	0/8	1/8	0/10	1/30

Table 4.11: Declined permissions requests or app installs.

lot of apps." Subject S_{11} remarked that they *"thought Play store was trusted, so didn't really think about declining because of permissions."* Subject S_6 (an Android user) said they did not decline installation due to permissions, but remarked that they sometimes did not use an app after install if the app prompted them to enter their password, or credit card details.

4.4.5 Internet browsing

We asked respondents if they used Tor or a VPN (Figure 4.12). These tools can be useful for preventing government deanonymization of online accounts, and can additionally help avoid local government tracking of visited websites and interception of sensitive plaintext data via passive surveillance, as well as local government *network injection*. Subject responses also revealed some pitfalls of security tools.

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
VPN phone	0/4	5/8	1/8	0/10	6/30
VPN PC	0/4	2/8	1/8	0/10	3/30
Tor PC	0/4	3/8	0/8	1/10	4/30
Total	0/4	6/8	1/8	1/10	8/30

Table 4.12: Subject use of VPNs and Tor.

One subject from Ⓓ (S_{21}) and one subject from Ⓖ (S_{14}) reported using a VPN all the time on their computer and phone. One subject from Ⓖ (S_8) reported using two different VPNs; they had heard that one was *"more secure,"* and one functioned better with *"slow Internet speeds."* Subject S_8 used the VPNs to post to a pseudonymous online account. One subject from Ⓖ (S_{15}) said they used a (different) VPN on their computer and phone if they felt what they were doing was *"sensitive."* One subject from Ⓖ (S_9) said they used a VPN on their phone when they *"suspect something is being checked or monitored, or to*

get around censorship." One subject from Ⓖ (S_{20}) said they used a VPN on their phone to watch Western TV shows.

Two subjects from Ⓖ (S_2, S_7) said they used to use a VPN; S_2 said they switched to Tor browser, and S_7 said they found VPN usage to be too complicated.

Subject S_8, who was concerned about interception of their passwords, requested that we test to see whether their passwords were being transmitted in plaintext from their phone. We connected the subject's phone to the Internet via our laptop, and observed their Internet traffic for a brief period. We were able to capture a password for one of their pseudonymous accounts on a blogging site. They were using a blogging app on their phone which transmitted the password in plaintext whenever the app was opened. They had been told to make sure to use the VPN whenever *posting* to a blog, so they first opened the app, and then connected to the VPN before submitting a post. We advised S_8 to connect to the VPN before opening the blogging app.

We examined two computers (both from Ⓖ) that had Tor Browser installed. Both copies of Tor Browser were out-of-date. Subject S_2 exclusively accessed their email account through Tor. This subject (using Ubuntu) said their Tor browser was out-of-date because on one previous occasion, installing updates had rendered their Tor browser unusable. Therefore, S_2 was not updating their Tor browser, or their Ubuntu system (they had 600+ updates pending in Ubuntu).

We helped the subject get their system up to date and ensure that Tor browser continued to work. The other Tor subject, S_{15}, said they used Tor only occasionally for *"sensitive"* things. One subject from Ⓖ (S_{14}) and one from Ⓜ (S_{26}) said they used Tor rarely or occasionally; we did not find Tor Browser on the computers of theirs that we examined.

Five subjects claimed to be former Tor users. Subject S_4 from Ⓜ stated that they had tried it once, but *"sometimes it won't let you go to certain pages."* Subject S_{19} from Ⓜ stated that they *"learned about it at the training...but don't use it anymore."* Subject S_{11} from Ⓜ stated that *"somehow, it crashed 2 or 3 times so I uninstalled it."* Subject S_{21} from Ⓓ stated that they *"used to, but I mean generally, it slows down everything so I stopped using it a few years ago."* Subject S_7 from Ⓖ said they tried to use Tor, but found it too complex. They said that solutions like Tor *"aren't catered to part-time activists, only for hardcore activists that do it all the time."*

4.4.6 Security of Online Accounts

We asked subjects how they would recover their email and social media accounts if they lost access (Table 4.13). Losing account access may be a symptom of phishing or device compromise; if an attacker takes over a victim's account, they may use it to target the victim's friends and contacts. We also asked for subject perspectives on whether governments would seek to use accounts of imprisoned dissidents in this fashion: 73% of subjects viewed this as a certainty.

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
Recovery account	0/4	6/8	3/8	7/10	16/30
No strategy	4/4	0/8	5/8	3/10	12/30
Other	0/4	2/8	0/8	0/10	2/30

Table 4.13: Subjects' action if they lose access to online account.

We classified 12 subjects into the **No strategy** category, because they remarked that they did not know (9 subjects), their recovery account information was out-of-date (2 subjects), or they indicated they would do nothing (S_{23} from Ⓜ indicated they would leave their old accounts *"for dead"* and create new accounts). Sixteen subjects believed they had up to date **Recovery account** information (either a recovery email account or recovery phone or both).

Two subjects mentioned **Other** methods of recovery. One subject from Ⓖ, who was part of an organized activist group that was the subject of recent arrests, remarked that their organization used a person outside of the country as a *"password manager."* That person had access to all of the passwords used by the various groups, and could recover accounts or change passwords if necessary. One subject (S_7) remarked that if they lost access to their accounts, they would reach out to friends at Facebook and Twitter to recover access.

We asked respondents how likely (on a scale from 1–5) they thought it was that governments would try to take the passwords of arrested activists (Figure 4.3), and use accounts of arrested activists to target friends (Figure 4.4). This question was motivated by the case of Red Sky in Section 2.3.2.

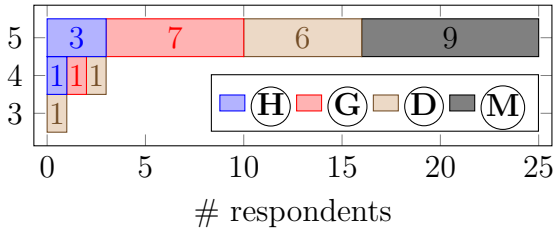


Figure 4.3: "How likely is it that the government will steal ... passwords from an activist's phone or computer when they are arrested?" (1: definitely not; 5: definitely)

4.4.7 Checking links and attachments

Malicious links and attachments are a common vector for social engineering attacks on dissidents. While most subjects reported that they vet such messages, and perceive reduced risks from interacting with the message afterwards, the methods they use appear to be vulnerable to social engineering.

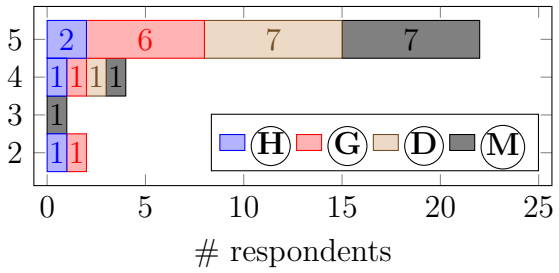


Figure 4.4: "How likely is it that the government will use arrested activists' ... accounts to target their acquaintances?" (1: definitely not; 5: definitely

Message vetting techniques

We asked subjects about how they checked messages containing links and attachments that they receive: *"Do you check links before opening them to see if they are safe? How?"* (and repeated for attachments). The results for links and attachments were broadly similar, except two subjects (both from Ⓓ) reported checking links but not attachments. We record how subjects checked links and/or attachments in Table 4.14. Five subjects reported that they did not check either links or attachments.

	Ⓗ	Ⓖ	Ⓓ	Ⓜ	**Total**
Sender	3/4	5/8	3/8	10/10	21/30
Context	3/4	4/8	0/8	4/10	11/30
2nd Factor	2/4	0/8	0/8	4/10	6/30
Tools	0/4	2/8	1/8	0/10	3/30
Gmail	0/4	2/8	2/8	2/10	6/30
Other	2/4[22]	1/8[23]	0/8	1/10[24]	4/30
Total	3/4	7/8	5/8	10/10	25/30

Table 4.14: How do subjects check links or attachments?

Some subjects reported checking the **Sender** of the message; some checked the **Context** surrounding the message (e.g., if the message was *"out of the ordinary"* for the sender, whether it was *"generally addressed,"* whether the sender properly greeted the recipient); some made use of a **2nd Factor**, e.g., placing a phone call to the purported sender, to ask if they had sent the message; some subjects used **Tools** to check links and attachments themselves, such as VirusTotal, their computer's antivirus (before opening an attachment), or various websites to unshorten URLs; and some relied on **Gmail** warnings, or used the preview feature in Gmail to check the attachment before they downloaded it.

[22]Hovers over links.
[23]One mentioned checking extensions on links, and not clicking on shortened links.
[24]One mentioned checking the URL of links, and extensions on attachments.

When prompted about their checking of links and attachments, three subjects from Ⓜ mentioned that sometimes their computers run slow after opening a link or attachment, and this leads them to be suspicious about surveillance (S_6, S_{11}, and S_{23}).

Two subjects from Ⓜ articulated a feeling that they were safe using Gmail: S_{19} stated: *"Experts say Gmail is safe,"* and subject S_{26} said: *"Gmail always gives you a sign if something's wrong. They have some kind of notification system that this may contain a virus."*

Subject S_{26} also remarked that he could tell if a message was malicious: *"when you experience a lot of being victimized and targeted, you develop intuition and are cautious about these things."* A few weeks after the interview, the subject forwarded a message to the authors that he thought was suspicious. He remarked that he had downloaded and opened the attachment on his computer, and only later realized that the sender account was crafted to look like one of his friends, but did not actually belong to the friend. The attachment did not contain spyware, but a link in the email led to a website that contained spyware. The subject had not interacted with the website sufficiently to infect his computer.

Subject perception of vetting efficacy

We asked subjects to rate the safety of opening links and attachments before and after they had checked them (on a scale from 1–5). We show the results for attachments in Figure 4.5. We excluded five subjects who did not check attachments, and four subjects who answered *"don't know."*

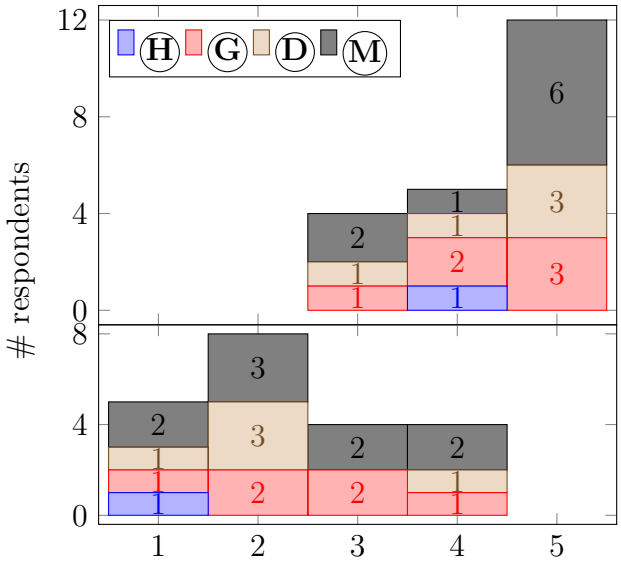


Figure 4.5: Subjects' perceived risk of opening attachments before (top) and after (bottom) "checking." (1: completely safe; 5: not at all safe)

4.5 Take-aways

Most respondents seemed concerned with ensuring the privacy of information on their computers, phones, and online accounts, or consequences stemming from the compromise of private information.

Optional vs mandatory security

A significant number of subjects did not enable optional security features on their devices or online accounts; only 68% of phones had a password of some type; 53% of subjects had a strategy for recovering access to an online account if they lost access; 44% of subjects had a strategy for recovering from loss of a mobile device; and 32% of phones were encrypted. Focusing on making such features mandatory while causing minimal inconvenience for users could help ease these security concerns.

Folk models and ad hoc defenses

Several subjects appear to use *ad hoc defensive strategies* for responding to perceived security risks. For instance, two subjects (S_2 and S_{21}) reported using app lock software, and one (S_{15}) reported using file lock software. It is an open question how effective such apps are; Trend Micro found vulnerabilities in some app locking products it analyzed [84].

In some cases, these strategies may introduce new security vulnerabilities. For instance, the subject's (S_{12}) use of triple-SIM phones and frequent SIM swapping to prevent linkage of communications with different contacts (Section 4.4.4) could introduce issues if the same SIM card is ever used in more than one slot. The subject's (S_8) jailbreaking of their iPhone to install a second copy of WhatsApp with a number not registered in their name may help preserve anonymity of WhatsApp communications, but could increase risks from spyware. Similarly, the subject's (S_{14}) rooting of their Android to install X Privacy may help the subject better control information shared by their apps, but could increase spyware risks.

We also identified the possible presence of security *folk models*: mental models that are not necessarily correct, and may lead to suboptimal security decision making [199]. For instance, S_{11}'s statement that the "[Google] Play store [is] trusted" leading them to not check permissions of apps they install, S_{10}'s belief that "Gmail is safe," and S_{26}'s belief that "Gmail always gives you a sign if something's wrong." Three subjects believed that computers running slow was a possible sign of surveillance (S_6, S_{11}, and S_{23}).

Vulnerability to social engineering

Broadly, subjects appear to be vulnerable to social engineering along several different dimensions.

Subjects' methods for checking links and attachments (primarily checking sender addresses and vetting a message's context) would seem to be vulnerable to more advanced social engineering. Indeed, one subject, S_{26}, who indicated that they checked a message's

sender address and context, opened an attachment from an account designed to impersonate a friend (Section 4.1).

We also identified two subjects (who were also subjects of *Himaya*) who claimed to perform the same checking, but interacted with malicious messages: the case of S_{25} mentioned in Section 5.4, and two cases involving S_{13}.

Deficiencies in subject security settings, physical security, as well as in contingency plans for recovering from account and phone loss, could lead compromised accounts to be used in social engineering, as has been documented in previous work [129].

Further, some subjects expressed a desire to focus on their work, or a frustration with digital security. As S_{19} from Ⓜ put it *"Life is intense, you focus on your work. People ... are suffering and in prison, when you focus on that you forget yourself."* Subject S_{27} from Ⓗ wondered *"Should I spend half a day figuring out digital security, or do work?"* Subject S_7 from Ⓖ stated *"you have to open attachments. you can't allow yourself to be a permanent victim of malicious intent. life has to move on."*

4.6 Conclusion

Our survey of potential targets of abusive government attacks finds they have numerous vulnerabilities to new methods of dissident surveillance that involve social engineering. Despite the availability of free online tools to check links and attachments, our subject population does not appear to widely use such resources.

Our results suggest that a tool supporting *automatic checking* of email messages may provide some benefit to our study population. Per Chapter 2, many attack campaigns targeting this population employ similar techniques (e.g., Microsoft Office documents that try to run executable files, IP logging links). A defensive tool could look for certain behavioral signatures that are more likely to be malicious in the context of potentially targeted users than in an ordinary population (e.g., IP logging links), as well as conducting scans against indicators (e.g., [108]) known to be part of targeted attacks. Depending on subjects' preferences, scanning could go beyond the traditional scanning provided by email services, by attempting to unshorten and download certain links included in emails to check their contents.

In Chapter 5, we present *Himaya*, an architecture and prototype implementation of a scanning tool for subjects' email accounts, guided by subject sentiment regarding security apps, e.g., a desire for stability and positive first interactions (Section 4.4.5), and balancing intrusiveness with engagement (Section 4.4.4).

Chapter 5

Himaya

5.1 Introduction

Informed by the results of our ecosystem studies (Chapter 2), and our fieldwork (Chapter 4), we developed *Himaya*, a defensive approach that readily integrates with targets' workflow to provide near real-time scanning of email messages for threats. *Himaya* currently protects 36 subjects, and has found a number of attacks both from retrospective scanning of messages, as well as in live activity.

Subjects authorize *Himaya* to access their email accounts (via OAuth), enabling the system to check message elements against a variety of both static and behavioral signatures in order to alert users to instances of malicious messages. In the long term, we hope that *Himaya* will provide deeper insight into how activists are targeted, and fundamentally advance the surveillance arms race by rendering an entire class of attacks likely to be detected by surveillance targets. Our prototype implementation of *Himaya* supports Gmail accounts.

Given that *Himaya* holds keys to user accounts, and processes user messages, we have carefully designed it to resist compromise by sophisticated adversaries. We separate *Himaya*'s functions into six distinct physical machines that are internally firewalled from each other. Our goal is to prevent a compromise of one machine from resulting in disclosure of subjects' messages, or the OAuth tokens that *Himaya* uses to access subjects' accounts.

While *protecting* users from malicious messages is *Himaya*'s primary goal, establishing evidence of past targeting through retrospective analysis is an important secondary goal. Such evidence can indicate the need for incident response, and aid civil society advocacy campaigns.

5.2 Architecture

We provide an overview of the logical topology of *Himaya* in Figure 5.1.

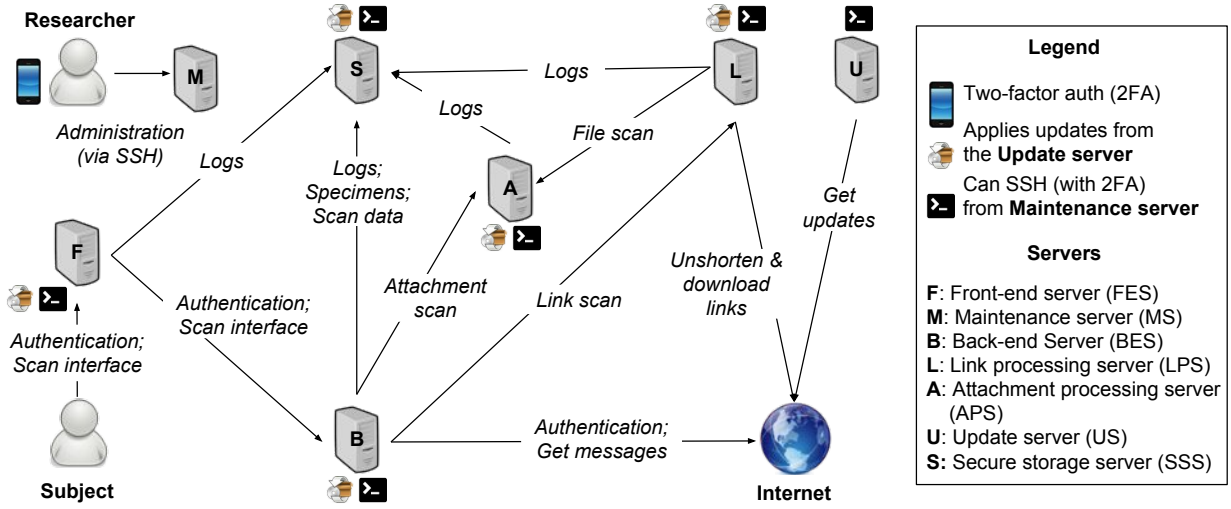


Figure 5.1: Logical topology of *Himaya*.

5.2.1 Enrollment

A subject's interaction with *Himaya* starts when they visit the experiment page in their web browser, hosted on a **Front-end server** (FES). A subject wishing to scan one of their accounts with *Himaya* selects the "Scan a new account" option, and is presented with a consent form that explains the procedures, benefits, and risks of *Himaya*. When the FES sends the consent form to the user's browser, the FES creates a new unique session identifier, `session_uuid`, and returns a digitally signed copy of this value to the user in a `state` cookie, as well as in a hidden `form` field in the consent form's HTML code. We use this to implement the *double submit cookies* technique for preventing cross-site request forgery (CSRF) [152].

On the consent form, the subject selects whether they want *Himaya* to scan their old messages (messages received before enrollment), new messages (messages received after enrollment), or both. The subject also selects a variety of other options, including whether they want us to scan "shortened" links, preserve their message metadata to check against attacks identified in the future, or be contacted in the future by us, if we wish to request permission to disclose personal information regarding the subject in a publication. The subject further provides a contact email for alerting purposes.

Once a subject submits the consent form to the FES, the FES checks whether the `state` cookie matches the value of the hidden CSRF `form` field, and whether this value bears a valid digital signature from the FES. If both checks pass, the FES stores the preferences in a `prefs` cookie, set as the FES redirects the subject to Gmail to sign into their account. The FES passes the value of the `state` cookie in the `state` field to Gmail, which Gmail will later return to the FES.

Once the subject signs in, Gmail redirects the subject back to *Himaya*'s FES, passing the OAuth code and `state` parameter via an HTTPS URL's query string. After verifying

the `state` parameter matches the `state` cookie, and bears a valid digital signature, *Himaya* forwards the only copy of this OAuth code to the **Back-end server** (BES), as well as the preferences selected by the user from the `prefs` cookie, and the user's `session_uuid`, in a `token_response` message (Table 5.2 on page 103). The BES contacts Gmail to exchange the OAuth code for tokens to access the subject's account. The BES sends the OAuth refresh token to the **Secure storage server** (SSS) in an `insert_scan_data` message, along with a unique identifier generated for the scan (`scan_uuid`), the address of the email account being scanned by *Himaya*, and the preferences selected by the subject.

Messaging: Internal messages between *Himaya* components are sent by invoking an API providing the methods `send_any`, `send_one`, and `recv_all`, which take a server type (e.g., "BES," "FES," etc.) as an argument, and allow for the sending of messages to any node of a given type, or a particular node (e.g., "send to any BES" or "send to this specific BES"), and the receipt of messages from all nodes of a given type. The `recv_all` method passes the identity of the node from which the message was received to its event handler.

The API also contains `inc_q` and `dec_q` methods by which a component can indicate the size of its queue (`q`); messages announcing changes in the current queue size of each node are periodically sent in the opposite direction of the arrows in Figure 5.1. The `send_any` method sends to the node with the smallest queue size, and returns the identity of the node chosen, so that the sender may store this information, and contact this specific node for follow-up queries via `send_one`. The API ultimately passes messages through *zeromq* [209] sockets, to facilitate atomic transmission of individual messages. As an example, the FES sends the `token_response` (Table 5.2 on page 103) to *any* BES, and then notes internally that follow-up messages regarding the same session should be sent to that specific BES. The messaging API facilitates the simultaneous operation of several different physical machines each having the same role (e.g., BES).

5.2.2 Scan

After a BES has obtained the OAuth tokens and stored these on the SSS, it uses the tokens to connect to the subject's Gmail account via IMAP (using TLS). Note that the subject does not need to have the option for IMAP enabled in their Gmail account (which is disabled by default) to allow this kind of access.

If the user has selected for only *new* messages to be scanned, then the BES immediately sends a "welcome" email (via Gmail) that advises the user how to access the FES to check future scan results, and begins listening on the IMAP connection for `IDLE` messages from Gmail indicating that new emails have arrived. When the welcome message has been successfully sent, the BES informs the SSS via a `set_welcome` message.

If the user has selected for *old* or both old and new messages to be scanned, the BES queries Gmail for a list of all old messages. The number of messages returned by Gmail is sent to the SSS via an `update_scan_data_set_total` message to update the SSS's view of the scan progress.

After the BES downloads a message and assigns it a unique `msg_uuid`, the BES checks the message's body and metadata against a number of signatures based on headers and body contents.

Signatures: Head and body signatures are specified in a CSV (comma-separated values) file. Each line of the signature file represents a single signature, and contains the following information:

1. a *start date* and *end date* in UNIX timestamp form used to limit the signature's applicability in time; these are matched against the the email's date of receipt as recorded by Gmail
2. a *type* that specifies what the signature should be matched against; BES signatures have two valid types, either "headers" or "body"
3. a *signature name*, which is a word (e.g., "ipspy") that explains the signature; in the future, we plan to present to the user more detailed descriptions associated with each signature name
4. a *severity*, currently either "fatal" or "warn," which indicates how a signature match should be presented to the user
5. a *regular expression* representing the signature

An example of two signatures used by the BES appears in Figure 5.2. The first one is a "warn," "header" signature designed to detect a particular header inserted by `readnotify.com`, which allows senders to track the IP address and browser headers of a target who opens an email sent through the service (Section 2.3.2). The second is a "fatal," "body" signature associated with the phishing attacks we profile in Section 2.4.2, representing a particular misspelling of Google's corporate address (the correct address is included with all legitimate Gmail account service emails). Note that both signatures contain no start and end date, indicating that they are valid regardless of the email's date.

```
,,header,ipspy,warn,/X-Mai1er: RNwebmail/
,,body,apt_noisyfalcon,fatal,/1700\.Ambhitheatre\.Parkway/
```

Figure 5.2: Two examples of signatures used by a *Himaya* BES.

Once the BES is done parsing and checking a message, it assigns a unique `att_uuid` to each attachment, and forwards (via a `file` message) each attachment to *any* **Attachment processing server** (APS), along with the `msg_uuid`, `att_uuid`, the message's date, the extension as indicated in the attachment's filename, and the message's *wordlist*, a list of every word in the message (for password guessing if the attachment is encrypted).

The BES also forwards a list of all URLs in the message, along with the `msg_uuid`, the message's date, and the message's wordlist, to *any* **Link processing server** (LPS). The

URLs are extracted via a regular expression, so the list includes not only `href`s, but also URLs in `img` and other HTML tags.

The LPS notifies the BES when it is finished scanning all links, and the APS notifies the BES when it is done scanning each attachment. The BES keeps track of which attachments it is waiting to receive scan results for, and whether it has received scan results for the links. After a message has finished processing, including the scanning of all links and attachments, the BES forwards the message and metadata to the SSS if malicious (via `store_msg` and `insert_email`), and if not malicious, only the metadata, if the subject has consented (`insert_email`), or simply a record that the message UID has been scanned (`update_scan_data_progress`), if the user has not consented to preserving metadata.

If the message is malicious, the BES also sends an email notification to the contact address that the user provided during enrollment, unless the message being scanned is an *old* message (i.e., received before the user enrolled in *Himaya*, and the user selected the option for *Himaya* to scan their old messages).

Also, if the user is logged in to the experiment website, the BES will send a notification (`msgprogress`) to the appropriate FES indicating how many message have been scanned, and how many are left to scan, as well as four pieces of metadata from any message detected as malicious: the message's sender, subject, date, and list of detected threats.

If all of the user's *old* message have been scanned according to the user's preferences, the BES sends a "welcome" email, that includes the number of malicious messages detected so far, and advises the user how to access the FES to check their scan results. The BES informs the SSS via a `set_welcome` message. The BES then begins listening for `IDLE` messages on the IMAP connection to indicate new emails.

If the BES receives an `IDLE` message indicating some number of new emails, it first sends an `update_scan_data_inc_total` message to the SSS to update the SSS's scan state, and then processes the new messages as above.

The FES sends information about scan progress to the user's browser (via *socket.io* [177]) to update an interface (Figure 5.3), which visualizes the progress of the scan, as well as any detected threats. The progress bar is updated in near-real-time, both during the scanning of past messages, and as new messages arrive. As a security measure, an FES feature called the *threat encyclopedia* contains a list of threats that *Himaya* can detect. If the FES receives a threat not in the encyclopedia, it does not forward it to the user's browser, in order to prevent compromised components that report threats (e.g., an APS) from exfiltrating data this way (Section 5.3.4).

Right now, "fatal" threats are rendered in red in Figure 5.3, whereas "*warn*" threats are rendered in yellow. "Fatal" is reserved for clearly malicious threats, whereas "warn" is used for items that may or may not be threats, depending on the context of the message.

Scanning links

When an LPS receives a batch of links from a BES, it will first increment its queue (`inc_q`) by the number of links received. If the subject wishes for *Himaya* to attempt to unshorten

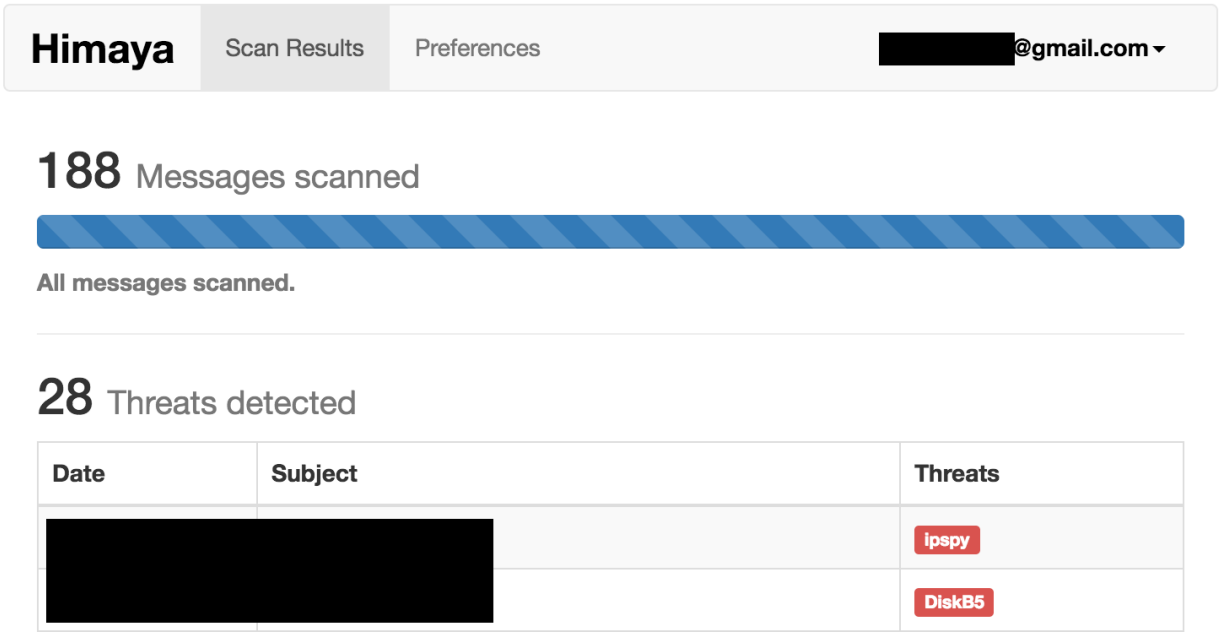


Figure 5.3: *Himaya* interface to display scan results.

URLs, the LPS will examine each link received against the following regular expression designed to detect short URLs:

```
/:\/\/[^\/]+\/[a-z0-9]{3,15}$/i
```

The LPS unshortens URLs by recursively issuing HTTP HEAD requests until it receives a non-`30X` response code, or reaches a pre-defined maximum depth (currently set at 5). The output of the unshortening process is a chain consisting of intermediate hops, and a final URL. Note that we do not attempt to unshorten or access all URLs, as emails may sometimes contain *side-effecting* links, such as links that unsubscribe a user from a mailing list[1] or RSVP to an event.

The LPS checks each element of the chain against a variety of signatures, which are specified using the same CSV schema as the BES signatures. Signatures containing known malware or phishing domains or patterns are marked with a severity of "fatal," whereas signatures designed to detect services that facilitate IP address discovery are marked as "warn," as these may be malicious depending on the context, but may also be benign in spam or marketing emails. See Figure 5.4 for an example of some LPS signatures used by *Himaya*. The first is a "fatal" signature based on a domain name we saw used by *Stealth Falcon* (Section 2.4.3) to distribute malware, the second is a "warn" signature based on a service,

[1]We have observed unsubscriptions from some mailing lists when we send a HEAD request to the unsubscribe URL.

`readnotify.com`, that allows senders to track the IP address and browser headers of a target who opens an email sent through the service (Section 2.3.2). The first signature has a start date and end date corresponding to the period during which the domain `adlinkmetric.com` was under attacker control. The second has a start date equal to the registration date of `readnotify.com`, and no end date, as the domain is still in use.

```
1427562000,1490720400,link,apt_stealthfalcon,fatal,/\/\/((.*)\.)?adlinkmetric.
com\//
970286400,,link,ipspy,warn,/\/\/readnotify\.com\/[^\/]+\//
```

Figure 5.4: Two examples of signatures used by a *Himaya* LPS.

The use of a *start date* and *end date* in specifying signatures is particularly important for the LPS, as it allows us to write signatures for domain names known to be under attacker control at particular times; attack domain names may be later transferred or re-registered for benign uses.

The LPS will also attempt to download (up to a maximum size, currently configured at 32MB) the last URL in the redirect chain (or the original URL if it is not a short URL) if it contains an obvious extension that matches an extension that we have typically seen used to distribute malware (e.g., executables, Microsoft Office documents), or if the URL points to a file sharing website that is supported by the *Plowshare* automatic downloading tool [30]. If a file is successfully downloaded, the LPS will forward the file to an APS in the same type of `file` message sent by a BES.

Once an LPS has finished scanning a link, and the APS has finished scanning its downloaded file (if applicable), the LPS will decrement its queue by one (`dec_q`). Once all links in a message have been completely scanned, the LPS notifies the BES of the scan results, as well as the full chain of URLs observed if the link was unshortened, and sends the contents of any downloaded files via a `done_batch` message. The BES sends any downloaded files for emails detected as malicious to the SSS via a `store_link_dl` message.

Scanning attachments

When an APS receives a file from a BES or LPS, the APS recursively extracts it, up to a maximum extraction depth (currently set at 5). At each stage, the APS attempts to use both the `unrar` [160] tool as well as `7-zip` [1]. If these tools indicate that the archive is encrypted, the APS will try every word in the message's wordlist as a possible password.

Once extraction is completed, extracted files with certain extensions (e.g., Office Documents and scripts) are run in an instance of Cuckoo Sandbox to check for malicious behavior. The APS does not have access to the Internet, so any network communications initiated by the file are captured by Cuckoo, but not routed to the Internet, or any *Himaya* nodes.

Currently, some malicious behaviors recognized by *Himaya* as "fatal" include Office Documents that drop executable files on disk or try to invoke PowerShell. Other behaviors

including Office Documents that connect to websites to download additional content are marked as "warn," as these may be malicious depending on the context.[2]

The APS also scans extracted files against Yara [208] signatures maintained by us, and Citizen Lab [108], for certain types of targeted malware. For completeness, extracted files are also scanned by free versions of a number of commercial antivirus products. Yara and antivirus hits are marked as "fatal." Once a file submitted to an APS has been completely scanned, the BES (or LPS) is notified via a `done` message, which contains a tree structure with metadata (filenames, hashes, sizes) of extracted files, as well as any threats detected.

5.2.3 Login

After a subject has enrolled in *Himaya*, they may later access scan results (Figure 5.3) by navigating to the experiment website, and selecting "Check results of existing scan." The subject is redirected to Gmail for authentication in a similar manner as to when they enroll, except the subject does not have to go through consent procedures. The FES sends the BES a `token_response` message without a `prefs` argument, signifying a login. If there is some error with the login (e.g., the subject is trying to log in to a nonexistent account), the BES may send an `errmsg` message to the FES to display an error popup in the user's browser. Certain errors may also cause the BES to send a `force_logout` message to the FES, in order to force the user to log in again.

After the subject authenticates with Gmail, the BES will detect that the subject has a scan in progress, and send a `get_initial_data` message to the SSS. The SSS responds via the `get_initial_data_result` message with information about the scan progress (i.e., how many messages have been scanned, and how many are left to scan), as well as any threats detected so far. The BES forwards this information to the FES in an `initial_data` message. This information is then sent to the subject's browser to populate the scan interface (Figure 5.3).

If a subject decides to log out, they may click on their email address in the upper-right corner of the interface (Figure 5.3) which will open a drop-down menu with a "Logout" option. If the user clicks on this option, they are redirected back to the FES's homepage, and their `state` cookie is cleared. The FES also sends a `scan_logout` message to the BES, indicating that the BES should no longer inform the FES about scan progress updates for that session.

Once a subject is logged in, they may modify the scan preferences they supplied during enrollment using the "Preferences" tab (Figure 5.3). Through this tab, the subject may also choose to withdraw at any point, and all data stored by *Himaya* about their account will be deleted. A subject can withdraw either by clicking on the "Stop using *Himaya* and delete all of my information" button, or by contacting researchers directly. Similar to the process for enrolling or logging in, the preferences page and FES implement the "double submit cookies"

[2]For instance, copying from a web browser and pasting into Microsoft Word sometimes generates embedded remote images in the Word document. On the other hand, malicious Word documents sometimes attempt to download remote executable files disguised with an extension like `.jpg`.

technique for CSRF protection so that a third-party website cannot change a user's *Himaya* preferences.

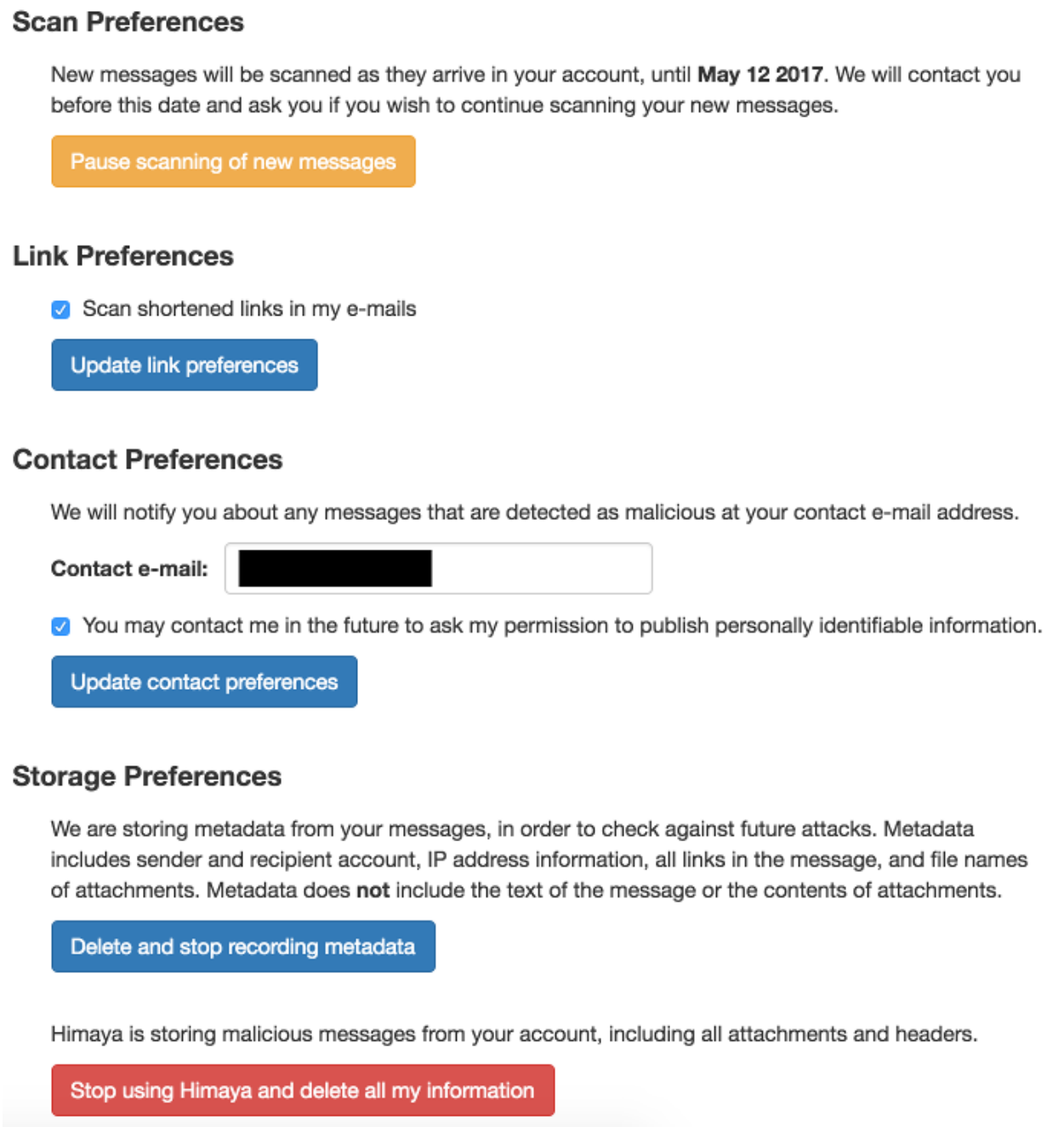


Figure 5.5: *Himaya* interface to change a subject's scan preferences.

Clicking on the "Preferences" tab causes the FES to generate a `get_prefs` message to the BES, which maintains the subject's preferences. The BES replies with current subject

preferences via a `get_prefs_response` message, which are then used to populate the "Preferences" page in the subject's browser. If a subject chooses to modify their preferences, this causes the FES to generate a `set_prefs` message to the BES, which in turn sends a `set_prefs` message to the SSS.

If the subject clicks on the "Stop using *Himaya* and delete all of my information" button, they are presented with a confirmation popup; if the user accepts, then the FES sends a `kill` message to the BES, which immediately stops the scan, and contacts Gmail with an OAuth deauthorization request for *Himaya*, so that *Himaya* cannot access the subject's account in the future. The BES also sends a `kill` message to the SSS, which searches for, and deletes, all metadata and data gathered about that user.

If a subject earlier consented to metadata collection, they may also opt out of this at any time through the "Preferences" tab, by clicking on the "Delete and stop recording metadata" button. After user confirmation, the FES will send a `delete_metadata` message to the BES, which in turn sends a `delete_metadata` message to the SSS, causing the SSS to delete all metadata regarding messages not detected as malicious.

Administration

We administrate *Himaya* by logging in through the **Maintenance server** (MS) (Figure 5.1) with SSH using the Google Authenticator app [68] with *TOTP* two-factor authentication [145]. From the MS, we can SSH into any of the other machines, also using two-factor authentication.

5.2.4 Internals

Most *Himaya* code is written in JavaScript, using the *Node.js* framework [149]. Each machine that is part of *Himaya* runs CentOS, and employs full disk encryption using DM-Crypt with LUKS [202]. The FES, BES, LPS, APS, and SSS are updated daily via the **Update server** (US), which is itself updated daily. The US serves as a local CentOS mirror. The US also updates all external *Node.js* packages and antivirus products, both via HTTPS. The FES, BES, LPS, and APS in turn download *Node.js* package updates on a daily basis from the US. The FES, BES, LPS, and APS send activity logs to the SSS for storage by sending a `store_log` message. Activity logs do not include any data or metadata from messages, nor any browser information or IP addresses used by subjects to access *Himaya*.

When a component of *Himaya* is updated, it may require a restart. When a BES is shut down, all existing scans are temporarily halted. When the BES restarts, it needs to obtain data from the SSS on which scans it should resume, as no component except the SSS stores data locally. When a BES starts up, it contacts the SSS with a `get_scan_data` message. The SSS responds with a `get_scan_data_result` message, containing OAuth refresh tokens, preferences, and progress associated with each ongoing scan it is responsible for. If the BES fails to authenticate via OAuth for a particular scan, it may be because the user has either de-authorized the app directly through Gmail without visiting the experiment website, or

changed their password. In this case, the BES sends an email to the user explaining how to re-authorize *Himaya* to access their account.

Service	Rules
Admin (SSH + 2FA)	🌍 $\rightarrow MS$ (Port 22) $FES \rightarrow *$
Updates (HTTP, NTP)	$* \rightarrow US$ (Port 80) $US \rightarrow$ 🌍
Himaya	$FES \rightarrow BES, SSS$ (Port 8081) $BES \rightarrow LPS, APS, SSS$ (Port 8082) $LPS \rightarrow APS, SSS$ (Port 8084) $APS \rightarrow SSS$ (Port 8083)
Web Interface (SSL)	🌍 $\rightarrow FES$ (Port 443)

Table 5.1: Communication allowed by Firewall.

All six components of *Himaya* are physically connected to a managed switch, which serves as a firewall, and the only physical connection to the Internet. The switch denies all communication by default, with a whitelist to allow certain communication between certain switch ports on certain TCP ports. We show the firewall rules in Table 5.1, where * represents "all except Internet," and 🌍 represents "Internet."

5.3 Risks

Clearly, we wish to avoid harms to users of *Himaya*. In this section we outline potential risks, and explain how these are mitigated by our architecture.

We identify six primary risk vectors involving *Himaya*, and address each in turn:

1. Discovery of *Himaya* users through surveillance,
2. Impersonation of subjects,
3. Impersonation of *Himaya*,
4. Compromise of *Himaya*'s systems or hardware,
5. Use of *Himaya* to develop attacks resistant to *Himaya*, and
6. Compromise of *Himaya* administrators.

5.3.1 Discovery of *Himaya* users through surveillance

A government may be able to identify a user of *Himaya* by compromising the Gmail account enrolled in *Himaya*, the email account where *Himaya* sends notifications, or a device on which

Source → Destination	Message
FES → BES	`scan_logout (session_uuid)`
	`kill (session_uuid)`
	`set_prefs (session_uuid, prefs)`
	`get_prefs (session_uuid)`
	`delete_metdata (session_uuid)`
	`token_response (session_uuid, oauth_code, prefs)`
BES → FES	`force_logout (session_uuid)`
	`get_prefs_response (session_uuid, prefs)`
	`msgprogress (num_scanned_emails, num_total_emails)`
	`initial_data (session_uuid, account_name, num_scanned_emails, num_total_emails, threats)`
	`errmsg (session_uuid, msg)`
BES → SSS	`set_welcome (scan_uuid)`
	`update_scan_data_set_total (scan_uuid, new_total)`
	`update_scan_data_inc_total (scan_uuid, new_num)`
	`get_scan_data ()`
	`get_initial_data (session_uuid, account_name)`
	`kill (scan_uuid)`
	`set_prefs (scan_uuid, prefs)`
	`delete_metadata (scan_uuid)`
	`insert_scan_data (scan_uuid, account_name, prefs, refresh_token)`
	`insert_email (scan_uuid, msg_uuid, attributes, headers, links_meta, attachments_meta, threats)`
	`update_scan_data_progress (scan_uuid, imap_uid, num_attachments, num_links, is_bad?)`
	`store_msg_file (msg_uuid, file)`
	`store_link_dl (msg_uuid, lnk_uuid, file)`
BES → LPS	`links (msg_uuid, wordlist, date, unshorten?, links)`
BES, LPS → APS	`file (msg_uuid, att_uuid, wordlist, date, file)`
APS → BES, LPS	`done (att_uuid, extraction_tree, threats)`
LPS → BES	`done_batch (links, threats)`
SSS → BES	`get_initial_data_result (scan_uuid, session_uuid, account_name, num_scanned_emails, num_total_emails, threats)`
	`get_scan_data_result (scan_data)`

Table 5.2: Internal messages exchanged by *Himaya* components.

the user accesses these accounts. These sorts of compromise, however, would introduce far greater risks to the subject than their use of *Himaya*.

However, a government may also be able to identify users of *Himaya* through passive monitoring, after discovering a domain name associated with *Himaya*, and querying the government's passive monitoring infrastructure for any DNS requests or TLS certificates involving the domain name of the FES, or for notification emails delivered by *Himaya*'s mail server to a recipient mail server without using TLS. At the current scale of *Himaya*, we think that this sort of analysis is unlikely, though we have designed *Himaya* to support multiple FES servers, each associated with a different domain name and SSL certificate. In the future, we could run a separate FES for every few users, reducing the harm associated with a government discovering any one FES domain.

Right now, *Himaya* users are invited by us, or referred by an existing subject. Government discovery of *Himaya* users may suggest that the subject is "interesting," or the government may wish to take some adverse action against the subject for collaborating with us. As we scale up *Himaya* by making it available to both dissident and non-dissident populations, we believe that the mere fact that a subject is participating in *Himaya* will be less "interesting" to governments.

5.3.2 Impersonation of subjects

Since we make ourselves available to answer subjects' questions about *Himaya*, someone who impersonates a subject could potentially socially engineer us for information about the subject. To combat this, we only answer subject questions pertaining to a specific message received by the subject that is classified malicious by *Himaya*. If subjects have questions about messages not detected as malicious, we ask them to forward us the message in question. We believe this limits the information that an adversary could gain by impersonating a subject to us.

5.3.3 Impersonation of *Himaya*

On the other hand, attackers could phish targets by providing them a link to a malicious app that impersonates *Himaya*, and inducing them to authorize the malicious app to access their accounts. Indeed, we have seen some past phishing against dissidents employing fake antivirus or encryption tools [64, 194]. However, we view this as a risk even in the absence of *Himaya*, as subjects already use online tools and services that may be subject to phishing. Indeed, we profile several phishing attacks aimed at obtaining Gmail credentials of subjects in Section 2.4.2.

5.3.4 Compromise of *Himaya*'s systems or hardware

An adversary may try to compromise *Himaya*'s systems or hardware in order to view subjects' messages, or obtain the OAuth keys we store to scan subjects' accounts.

To avoid physical compromise, we house *Himaya*'s hardware in a secure facility that includes 24-hour guards, and thus assume that attackers cannot readily gain access to physically compromise the system.

An adversary could attempt to compromise any of several components of *Himaya* remotely. While the FES and MS are the only components directly accessible from the Internet, if an attacker sends a message to a *Himaya* subject, it will be processed by the BES, LPS, and APS. We first consider the BES, LPS, and APS; the components with which an adversary may have indirect interaction.

BES

The BES accepts no incoming connections from the Internet. The BES places an outgoing connection to Gmail to download messages. Downloaded messages are parsed, and further processing occurs on the LPS and APS. An attacker could exploit a vulnerability in one of the libraries used by the BES to parse messages, if such a vulnerability exists. However, as the message parsing library is implemented in `Node.js`, as opposed to a non-memory-safe language like C, we view its compromise as unlikely. If the attacker compromises the BES, they would be able to exfiltrate any messages it scans, and any OAuth tokens for users who enroll or sign in.

APS

The APS executes malicious artifacts. An attachment may contain an exploit designed to break out of the virtual machine that the APS runs it in. An attachment also might contain an exploit for an unarchiving or anti-virus program (either to target *Himaya*, or a regular user downloading and extracting the file). Such a malicious file could could compromise the APS. However, if the APS is compromised, the attacker cannot exfiltrate data, because the APS is isolated from the Internet (Table 5.1) and cannot task other components beyond writing log files to the SSS, and submitting scan results to the BES and LPS (Table 5.2). An attacker could write bogus log entries to the SSS, and submit fraudulent file analysis results to the BES and LPS.

A clever attacker might try to cause the APS to exfiltrate data through the FES, to an authenticated malicious user, for instance by sending `done` messages to a BES or LPS containing information to exfiltrate in the `threats` list—recall that this information is ultimately rendered to the user's browser via the FES. The FES combats this by sending only threats in the FES's threat encyclopedia to the browser.

LPS

If an **LPS** is compromised, an attacker could view URLs and associated message wordlists, and would have the capability to exfiltrate this data. However, we think it is unlikely that an LPS could be compromised. The LPS interacts with the Internet by performing HTTP

HEAD requests to unshorten URLs, and HTTP(S) requests to download files. Downloaded files are not executed or parsed by the LPS.

SSS

We think it is unlikely that the **SSS** would be compromised, as it simply receives information and writes it to disk. The SSS also returns information to the BES about scan progress, and metadata regarding detected threats.

US

If an attacker compromises the **US**, they could cause the installation of malicious system or `Node.js` packages, or antivirus updates, on the other components. However, we believe it unlikely that the US could be compromised, as it simply downloads updates and stores them to disk using `HTTPS` and `RSYNC`, while processing `HTTP` requests from the other components. Compromise of update sources themselves is outside of our threat model.

FES

An **FES** could be compromised via an exploit in the `Node.js` web server component, or system libraries involved in serving an `HTTPS` webpage. If an attacker compromises the FES, they could set up a fake experiment website to capture OAuth tokens returned by Google for those who sign up and log in after compromise. The attacker could then break in to subject accounts. In order to minimize the FES's attack surface, we make use of Node.js's built-in HTTPServer class, in lieu of a more full-featured web server such as Apache.

MS

An **MS** could be compromised via an exploit in the `SSH` server, though such an exploit would imply a global Internet disaster, which is outside of our threat model.

5.3.5 Use of *Himaya* to develop attacks resistant to *Himaya*

When we scale up enrollment in *Himaya* by publicizing it more widely, there is a chance that an attacker or surveillance vendor could enroll a "test" email account in *Himaya*, repeatedly submit attacks, and obtain detection results, in order to devise an attack that would evade *Himaya*. Product materials from two commercial vendors, FinFisher and Hacking Team, indicate that these companies regularly test their products against antivirus and anti-malware tools [182, 52] in this manner.

However, testing against traditional antivirus products can be done offline, without risk of disclosure of attacks to researchers. Since the architecture of *Himaya* necessitates that attack samples be submitted to *Himaya*'s servers in order to view detection results. We believe that attackers are unlikely to make this tradeoff, as this can facilitate tracing their attack

infrastructures. For instance, Hacking Team advises its clients not to submit samples of its product to cloud-based scanning services like VirusTotal [196], remarking that researchers gaining visibility into samples submitted to VirusTotal is a "big issue" [26].

5.3.6 Compromise of *Himaya* administrators

One risk we see going forward is that the administrators of *Himaya* may be targeted as a way to compromise *Himaya*'s systems. For instance, an attacker may infect a device used by a *Himaya* administrator to log into the MS, in such a way that the administrator's login to the MS results in implantation of malware on the MS, which could then be propagated to other components as the administrator accesses them via SSH—for instance if the malware overwrites the `ssh` command. In the future, we plan to mitigate this risk by dedicating a separate machine dedicated to *Himaya* administration. Importantly, the dedicated machine will not be used for the administrator's other day-to-day web browsing or work activities, which may be a vector for compromise.

5.4 Initial results

A prototype implementation of *Himaya* has been in operation for a period of twelve months. Our implementation has 8 APS, and one of every other type of node. *Himaya* has scanned approximately 1.3 million emails, and has found 766 malicious messages (each reply that includes a malicious original email is counted as a separate malicious message). We provide a summary of the 36 subjects enrolled in *Himaya* in Table 5.3. All but three subjects elected to have *Himaya* **Scan** both their old and new emails; three chose to scan only their old messages; all but four subjects consented to have certain URLs **Unshorten**ed for scanning; 64% allowed us to preserve **Metadata** even for non-malicious messages, and 72% were willing to have us **Contact** them to ask permission if we wanted to publish their personal information in relation to *Himaya*.

We provide a brief overview of some attacks detected by *Himaya*.

Commercial surveillance tools

Himaya's retrospective analysis found that Ahmed Mansoor was targeted with FinFisher in 2011, providing the earliest known case of FinFisher's use in that country. In that case, the attack was an executable file disguised as a PDF inside a RAR archive. Mansoor did not fall for the attack; he responded to the attacker remarking that he would not open the attached executable file. The attacker replied with a nonmalicious PDF file.

Retrospective analysis also discovered that five *Himaya* subjects were targeted with FinSpy in 2012; the spyware communicated with the same C&C server as FinSpy samples we linked to Bahrain's government (Section 2.3.1).

# Emails	# Bad	Scan	Unshorten	Metadata	Contact
107	0	Both	Y	Y	Y
41	0	Both	Y	Y	Y
1,677	8	Both	Y		Y
222	3	Both	Y		Y
753	0	Both	Y	Y	Y
225	0	Both	Y		
23,818	1	Both	Y	Y	Y
10,119	109	Both	Y	Y	Y
18,501	70	Both	Y		
40,404	0	Both	Y		
3,310	0	Both	Y		
5,634	0	Both	Y		
44,916	0	Both		Y	Y
137,579	0	Old			Y
45,011	1	Both	Y	Y	Y
47,807	0	Both	Y	Y	Y
22,214	296	Both	Y	Y	
36,689	55	Old	Y		Y
178,581	97	Both	Y	Y	
42,900	11	Old	Y		Y
47,243	0	Both	Y	Y	Y
22,569	0	Both	Y	Y	
58,061	12	Both	Y		Y
14,566	2	Both	Y	Y	Y
108,823	67	Both	Y	Y	Y
88,841	0	Both	Y		
38,996	1	Both		Y	Y
156,957	1	Both	Y	Y	Y
12,078	1	Both	Y		
13,241	0	Both	Y	Y	Y
14,402	3	Both	Y	Y	Y
8,617	0	Both	Y	Y	Y
26,907	0	Both	Y	Y	Y
1,833	0	Both	Y	Y	Y
272	0	Both		Y	Y
188	28	Both	Y	Y	Y

Table 5.3: Summary of the 36 *Himaya* subject accounts and their preferences.

Interactions with malicious messages

One subject from our interviews (S_{25} in Chapter 4), reported vetting message senders, but inadvertently forwarded a malicious message to a large International mailing list. Subject S_{21}, also a subject of *Himaya*, received the message from the mailing list.

Subject S_{13} (who reports that they vet a message's sender and context) appeared to have twice interacted with malicious content identified by *Himaya*: one case discovered by retrospective analysis, and one case reported by the subject after enrollment. Retrospective analysis identified an email containing a link via the `goo.gl` link shortener. Analytics from `goo.gl` indicated the link had been clicked once. We asked S_{13} about the email, who indicated that they viewed it as legitimate, and also responded to the message. S_{13} later received an email containing a malicious attachment after they had enrolled. *Himaya* alerted them to the email (via an email alert in their inbox) approximately 90 seconds after they received it. The subject did not see the alert until they had already attempted to open a malicious attachment (they were stepping through their messages oldest-to-newest).

Other attacks

We found one instance where a *Himaya* subject received a link to a malicious password-protected RAR file hosted on a file hosting service. *Himaya* automatically downloaded the archive, cracked the password based on the contents of the email, ran the enclosed file in a sandbox, and flagged it as malicious when it tried to run an executable it dropped on disk.

We found three *Himaya* subjects who were targeted with Gmail password phishing links in 2014. It is unknown if they clicked on them. *Himaya* does not yet implement comprehensive phishing detection; we flagged the emails because the phishing links redirected through *ReadNotify* (Section 2.3.2), a service for tracking the IP address of targets who open messages (via embedded remote images).

Issues

We noted a number of issues with our initial implementation of *Himaya*. We initially experimented with alerting users when a single antivirus program detected a file as malicious. However, we noted that this sometimes resulted in false positives. We temporarily raised the threshold for malicious AV detections to 2, while we consider long-term fixes for the issue.

5.5 Discussion and future work

Himaya shows early promise not only in protecting subjects from attacks they may have otherwise fallen for, but also in mapping out past attacker activity. In the future, we plan to gather feedback from subjects to help improve *Himaya* alerts. We are particularly interested in how quickly subjects click on alerts, whether they find alerts useful, and to what extent their perception of malicious messages agrees with *Himaya*'s determinations.

Himaya does not have generalized support for detecting phishing messages. Rather, detection of phishing relies on BES and LPS signatures for certain campaigns. In the future, we plan to explore performing analysis of email structure [23] and visual presentation [3] in order to detect phishing.

Chapter 6

Concluding Remarks

Targeted nation-state surveillance of dissidents poses a challenging security problem, with significant real-world consequences for targets.

In our case studies of Bahrain and the UAE, we found attackers using tools and services from the cybercrime underground, as well as commercial government-exclusive hacking tools from three different vendors (FinFisher, Hacking Team, and NSO Group) to target dissidents. While these companies market their products under the theme of "lawful interception," we have been able to demonstrate ongoing abuse of these tools, and sales to authoritarian regimes where surveillance oversight may be lacking. Our findings indicate a lack of effective regulations on the market for hacking tools.

As nation-states have increasingly augmented their surveillance apparatuses with these capabilities, they open themselves to scrutiny. Targeted attacks often require the attacker to intervene in the target's environment, such as sending a carefully crafted message with malicious content. Unlike purely passive surveillance, such as wiretaps, these interventions are measurable and often *logged by default* in targets' inboxes. Once an attack is detected, it is often possible to leverage errors in the attacker's tools or operational security to map out other activity by the same or related adversaries. Indeed, we discovered 46 other countries using the same FinFisher and Hacking Team attack tools as Bahrain and the UAE.

While we have perhaps gained unprecedented visibility into government surveillance, it is clear that hacking tools afford governments unprecedented visibility into their targets lives. Our fieldwork highlights the fact that dissidents are vulnerable to careful social engineering, and that many have a desire to focus on their work rather than their digital security. To help subjects readily decide whether a message may be malicious, we developed *Himaya*, which shows promising early results in behaviorally detecting previously unknown attacks. As we continue to scale up *Himaya* to additional users, our hope is to remove a key barrier to surveillance research, by releasing datasets of *Himaya* attacks.

Ultimately, we aim with this dissertation to inspire more research efforts into abusive nation-state surveillance of dissidents. There are highly challenging issues left to tackle, including robust detection of novel attack vectors like Internet connection tampering to insert malware; but the potential stakes are likewise very high.

Bibliography

[1] *7-Zip*. URL: http://www.7-zip.org/.

[2] @a7rarelemarat. Twitter post. Oct. 2012. URL: https://twitter.com/a7rarelemarat/status/259883131807621120.

[3] Sadia Afroz and Rachel Greenstadt. "Phishzoo: Detecting phishing websites by looking at them." In: *Fifth IEEE International Conference on Semantic Computing (ICSC)*. 2011.

[4] *Ahmed Mansoor and Four Other Pro-Democracy Activists Pardoned and Freed*. 2013. URL: http://bit.ly/18pHpis.

[5] Sabreena Ahmed. *Cyber Crime Unit: Cyber and Cyber Crime*. URL: https://www.facebook.com/sabreeena30/posts/175194379211018.

[6] Salman H. AlJalahma. *Response to The Guardian—UK companys software used against Bahrain activist*. May 2013. URL: http://bit.ly/19iVUUP.

[7] *AOL/NCSA Online Safety Study*. Oct. 2004. URL: https://web.archive.org/web/20051102045804/http://www.staysafeonline.info/pdf/safety_study_v04.pdf.

[8] *Appin Technology Lab*. URL: http://www.appinonline.com/.

[9] Apple. *Government Information Requests*. URL: https://www.apple.com/privacy/government-information-requests/.

[10] arma. *Greg's browserfeedwriter javascript finds TBB's full path on Windows*. Tor Project Trac ticket. URL: https://trac.torproject.org/projects/tor/ticket/5922.

[11] *Asprotect SKE*. URL: http://www.aspack.com/asprotect32.html.

[12] Martin Ennals Award. *Ahmed Mansoor Selected as the 2015 Laureate Martin Ennals Award for Human Rights Defenders*. Oct. 2015. URL: http://www.martinennalsaward.org/?p=522.

[13] *Bahrain Independent Commission of Inquiry*. URL: http://www.bici.org.bh/.

[14] *BlackBerry rogue software leaves sour taste in UAE*. 2009. URL: http://on.ft.com/HVXvJP.

[15] blazegmc. *unknown ip connecting to port 6646*. McAfee Community. Feb. 2010. URL: https://community.mcafee.com/thread/21790?tstart=0.

[16] Stevens Le Blond et al. "A Look at Targeted Attacks Through the Lense of an NGO." In: *USENIX Security 2014*. Aug. 2014. URL: https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/le-blond.

[17] Reporters Without Borders. *Journalist held incommunicado, netizens arrested, censorship*. Aug. 2013. URL: http://bit.ly/2hxx0jU.

[18] Tom Brewster. *From Bahrain To Belarus: Attack Of The Fake Activists*. July 2013. URL: http://bit.ly/1gIgwhW.

[19] @bu_saeed2. Twitter post. Jan. 2011. URL: https://twitter.com/Bu_saeed2/status/158267593269063680.

[20] c0rnholio. *[Security Advisory] Samsung leaves it's Android Smartphones with WAP-Push Feature Open to Attacks (one sms to rule them all)*. July 2012. URL: https://www.silentservices.de/security-advisory-samsung-leaves-its-android-smartphones-with-wap-push-feature-open-to-attacks-one-sms-to-rule-them-all/.

[21] Index on Censorship. *United Arab Emirates: Stop the charade and release activists convicted at the mass UAE 94 trial*. Mar. 2015. URL: https://www.indexoncensorship.org/2015/03/united-arab-emirates-stop-the-charade-and-release-activists-convicted-at-the-mass-uae-94-trial/.

[22] *Censys*. URL: https://censys.io/.

[23] Hyunsang Choi, Bin B. Zhu, and Heejo Lee. "Detecting Malicious Web Links and Identifying Their Attack Types." In: USENIX Conference on Web Application Development, 2011. URL: http://research.microsoft.com/apps/pubs/default.aspx?id=193308.

[24] Richard Clayton, Steven J Murdoch, and Robert NM Watson. "Ignoring the Great Firewall of China." In: *PETS*. 2006.

[25] Lucian Constantin. *Facebook to roll out HTTPS by default to all users*. 2012. URL: http://bit.ly/1bsLBCm.

[26] Fabrizio Cornelli. *Re: Leveraging RiTE*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/120745.

[27] Joseph Cox. *A Hacker Claims to Have Leaked 40GB of Docs on Government Spy Tool FinFisher*. Aug. 2014. URL: http://motherboard.vice.com/read/a-hacker-claims-to-have-leaked-40gb-of-docs-on-government-spy-tool-finfisher.

[28] Jedidiah R Crandall et al. "ConceptDoppler: A Weather Tracker for Internet Censorship." In: *ACM CCS*. 2007.

[29] Jedidiah R. Crandall, Masashi Crete-Nishihata, and Jeffrey Knockel. "Chat program censorship and surveillance in China: Tracking TOM-Skype and Sina UC." In: (July 2013). URL: http://bit.ly/1fzNcHl.

[30] Matthieu Crapet. *plowshare-modules-legacy*. GitHub repository. URL: https://github.com/mcrapet/plowshare-modules-legacy.

[31] *Cross-platform Trojan controls Windows and Mac machines*. 2012. URL: http://bit.ly/1eJnJgZ.

[32] *CVE-2010-3333*. URL: http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-3333.

[33] *CVE-2013-0422*. URL: http://bit.ly/NA1O0A.

[34] *Cydia Substrate*. URL: http://www.cydiasubstrate.com/.

[35] *Dark Secrets—Hacking Team commercial*. URL: http://bit.ly/1bCh57v.

[36] *Default https access for Gmail*. 2010. URL: http://bit.ly/1bBktPM.

[37] Ronald Deibert and Rafal Rohozinski. "Tracking GhostNet: Investigating a Cyber Espionage Network." In: *Information Warfare Monitor* (2009).

[38] *Democracy Index 2015: Democracy in an age of anxiety*. 2015. URL: http://www.eiu.com/democracy2015.

[39] Rori Donaghy. *Equating human rights principles with a business strategy*. Apr. 2012. URL: http://www.youthdiplomaticservice.com/zzold-business-blog/category/business.

[40] Rori Donaghy. *UAE 94 Verdict: Unfair Trial and Torture Ensure the Story is Just Beginning*. July 2013. URL: http://www.huffingtonpost.co.uk/rori-donaghy/uae-94-verdict_b_3549671.html.

[41] Rori Donaghy. *UAE paid PR firm millions to brief UK journalists on Qatar, Brotherhood attacks*. Oct. 2015. URL: http://www.middleeasteye.net/news/uae-paid-pr-firm-millions-brief-uk-journalists-qatar-muslim-brotherhood-attacks-1058875159.

[42] Rori Donaghy. *UN envoy 'asked permission' from UAE to name Libya PM candidate*. Feb. 2016. URL: http://www.middleeasteye.net/news/un-envoy-asked-permission-uae-name-libya-prime-minister-candidate-210511543.

[43] Rori Donaghy. *UN Libya envoy secretly worked with UAE to back a side in civil war*. Nov. 2015. URL: http://www.middleeasteye.net/news/un-libya-envoy-secretly-worked-uae-back-side-civil-war-1441350981.

[44] Matt J. Duffy. *UAE Twitter activist not guilty but still in custody*. July 2014. URL: http://www.al-monitor.com/pulse/originals/2014/07/uae-twitter-imprisoned-not-guilty-activist-cyber-crime.html.

[45] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. "ZMap: Fast Internet-Wide Scanning and its Security Applications." In: *Proceedings of the 22nd USENIX Security Symposium*. Aug. 2013.

[46] Zakir Durumeric et al. "Analysis of the HTTPS Certificate Ecosystem." In: *Proceedings of the 13th Internet Measurement Conference*. Oct. 2013.

[47] @Dwight389. Twitter post. Apr. 2013. URL: https://twitter.com/Dwight389/status/327033672979079168.

[48] @Dwight389. Twitter post. Nov. 2013. URL: https://twitter.com/Dwight389/status/398413653315031041.

[49] Snorre Fagerland. *Systematic cyber attacks against Israeli and Palestinian targets going on for a year*. 2012. URL: http://bit.ly/1aSdw07.

[50] Snorre Fagerland et al. *Operation Hangover: Unveiling an Indian Cyberattack Infrastructure*. 2013. URL: http://enterprise-manage.norman.c.bitbit.net/resources/files/Unveiling_an_Indian_Cyberattack_Infrastructure.pdf.

[51] R. Fielding et al. *Hypertext Transfer Protocol – HTTP/1.1*. RFC 2616. June 1999. URL: https://tools.ietf.org/html/rfc2616.

[52] FinFisher. *Anti Virus Results FinSpy PC 4.51*. FinFisher leak. URL: https://wikileaks.org/spyfiles4/documents/Anti-Virus-Results-FinSpy-PC-4.51.xlsm.

[53] FinFisher. *FinFisher: Governmental IT Intrusion and Remote Monitoring Solutions*. Privacy International Surveillance Industry Index (SII). URL: http://bit.ly/2hzJEN0.

[54] FinFisher. *FinSpy Mobile 4.00 User Manual*. FinFisher leak. URL: https://wikileaks.org/spyfiles4/documents/FinSpyMobile-4.00-User-Manual.docx.

[55] FinFisher. *FinSpy Mobile 4.51–Release Notes*. FinFisher leak. 2014. URL: https://netzpolitik.org/wp-upload/Release-Notes-FinSpy-Mobile-4.51.pdf.

[56] FinFisher. *Remote Monitoring & Infection Solutions: FinFly ISP*. FinFisher leak. URL: https://wikileaks.org/spyfiles/files/0/297_GAMMA-201110-FinFly_ISP.pdf.

[57] *FinFisher - Excellence in IT Investigation*. URL: http://www.finfisher.com/.

[58] Gregory Fleischer. "Attacking Tor at the Application Layer." In: *Defcon 2009*. July 2009. URL: https://www.defcon.org/images/defcon-17/dc-17-presentations/defcon-17-gregory_fleischer-attacking_tor.pdf.

[59] Directorate General of Forces Intelligence. *About DGFI*. URL: http://www.dgfi.gov.bd/index.php/about.

[60] ISF - Internal Security Forces. *About Us*. URL: http://www.isf.gov.lb/en/about.

[61] Alain Forget et al. “”My Daughter Fixes All My Mistakes”: A Qualitative Study on User Engagement and Computer Security Outcomes.” In: *SOUPS*. 2016.

[62] The Apache Software Foundation. *httpd: index.html*. Source Code. URL: https://github.com/apache/httpd/blob/trunk/docs/docroot/index.html.

[63] *Front Line Defenders*. URL: https://www.frontlinedefenders.org/.

[64] Eva Galperin and Morgan Marquis-Boire. *Pro-Syrian Government Hackers Target Activists With Fake Anti-Hacking Tool*. Aug. 2012. URL: http://bit.ly/1eJj12T.

[65] David Gilbert. *Hacking Team and the Murky World of State-Sponsored Spying*. Mar. 2013. URL: http://bit.ly/17tBBtm.

[66] Fulvio de Giovanni. *Re: R: UAEAF - Stato dell'Arte*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/608068.

[67] Sergey Golovanov. *Adobe Flash Player 0-day and HackingTeam's Remote Control System*. 2013. URL: http://bit.ly/17n12ro.

[68] Google. *google-authenticator*. GitHub repository. URL: https://github.com/google/google-authenticator.

[69] Google. *Sign in using application-specific passwords*. URL: https://support.google.com/accounts/answer/185833?hl=en.

[70] Andy Greenberg. *Hacking Team Breach Shows a Global Spying Firm Run Amok*. July 2015. URL: https://www.wired.com/2015/07/hacking-team-breach-shows-global-spying-firm-run-amok/.

[71] NSO Group. *NSO Group*. URL: https://web.archive.org/web/20120813064018/http://www.sibat.mod.gov.il/NR/rdonlyres/DADE8D1E-DFAA-4143-BB48-A73C77C88CBA/0/NSOGROUPE.pdf.

[72] Claudio Guarnieri. *Analysis of the FinFisher Lawful Interception Malware*. 2012. URL: http://bit.ly/1eJjVMV.

[73] Luca Guerra. *Exploit deployment guidelines*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/121783.

[74] *Guy Molho — LinkedIn*. URL: https://www.linkedin.com/in/guymolho.

[75] *Hacking Team*. URL: http://www.hackingteam.it/.

[76] Hacking Team. *Hacking Team: Network Injector Appliance*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/fileid/447727/212805.

[77] Thoufique Haq and Ned Moran. *Now You See Me - H-worm by Houdini*. Sept. 2013. URL: https://www.fireeye.com/blog/threat-research/2013/09/now-you-see-me-h-worm-by-houdini.html.

[78] Seth Hardy et al. “Targeted Threat Index: Characterizing and Quantifying Politically-Motivated Targeted Malware.” In: *USENIX Security 2014*. Aug. 2014.

[79] Shane Harris. *U.S. Hired Dictators' Favorite Hackers*. July 2015. URL: http://www.thedailybeast.com/articles/2015/07/06/u-s-hired-dictators-favorite-hackers.html.

[80] Mehdi Hasan. *The British PM, the Middle East, and human rights*. Nov. 2015. URL: http://www.aljazeera.com/indepth/opinion/2015/11/british-pm-middle-east-human-rights-151103070038231.html.

[81] Cecily Hilleary. *UPDATE: Shez Cassim Back Home After Months in UAE Jail*. Jan. 2014. URL: http://blogs.voanews.com/repressed/2014/01/14/update-shez-cassim-back-home-after-months-in-uae-jail/.

[82] *Home of Crossbear and OONIBear*. Accessed: 27-February-2014. URL: https://pki.net.in.tum.de/.

[83] *Hping - Active Network Security Tool*. URL: http://www.hping.org/.

[84] Simon Huang. *The Severe Flaw Found in Certain File Locker Apps*. 2014. URL: http://blog.trendmicro.com/trendlabs-security-intelligence/the-severe-flaw-found-in-certain-file-locker-apps/.

[85] Ben Hubbard. *Emirates Balk at Activism in Region Hit by Uprisings*. June 2013. URL: http://nyti.ms/I4n2Aw.

[86] Emirates Centre for Human Rights. *19-year-old Emirati activist jailed for tweets*. Dec. 2013. URL: https://web.archive.org/web/20160326013707/http://www.echr.org.uk/?p=1104.

[87] Emirates Centre for Human Rights. *Current Political Prisoners*. Jan. 2015. URL: https://web.archive.org/web/20160404212357/http://www.echr.org.uk/?page_id=207.

[88] Gulf Center for Human Rights. *UAE: Human rights defenders arrested in crackdown by security forces*. July 2012. URL: http://www.gc4hr.org/news/view/198.

[89] Apple Inc. *About the security content of iOS 9.3.5*. Aug. 2016. URL: https://support.apple.com/en-us/HT207107.

[90] Lookout Inc. *Technical Analysis of the Pegasus Exploits on iOS*. Nov. 2016. URL: https://info.lookout.com/rs/051-ESQ-475/images/pegasus-exploits-technical-details.pdf.

[91] Tibet Action Institute. *Detach From Attachments!* Dec. 2011. URL: https://vimeo.com/32992617.

[92] The Department of Internal Affairs (New Zealand). *Email Scams - February 2013*. Feb. 2013. URL: http://www.dia.govt.nz/diawebsite.nsf/wpg_URL/Services-Anti-Spam-Email-Scams-February-2013.

[93] Gamma Group International. *FinSpy 3.10 Specifications*. FinFisher leak. URL: https://wikileaks.org/spyfiles4/documents/FinSpy-3.10-Specifications.doc.

[94] *Internet Census 2012*. 2013. URL: http://bit.ly/1i7rRHs.

[95] *Internews*. URL: https://www.internews.org/.

[96] @islam_way_2030. Twitter post. Aug. 2012. URL: https://twitter.com/islam_way_2030/status/232392466760863744.

[97] @islam_way_2030. Twitter post. Aug. 2012. URL: https://twitter.com/islam_way_2030/status/232393358243401728.

[98] @islam_way_2030. Twitter post. Aug. 2012. URL: https://twitter.com/islam_way_2030/status/232394930285318144.

[99] Adrianne Jeffries. *Meet Hacking Team, the company that helps the police hack you*. Sept. 2013. URL: http://bit.ly/1bCajyl.

[100] *jRAT - Java Remote Administration Tool*. URL: https://jrat.io/.

[101] Takashi Katsuki. *Crisis for Windows Sneaks onto Virtual Machines*. Aug. 2012. URL: http://bit.ly/MzheRJ.

[102] Ayesha Al Khoori. *Defendant denies insulting leaders of UAE on social media*. Mar. 2015. URL: http://www.thenational.ae/uae/courts/defendant-denies-insulting-leaders-of-uae-on-social-media.

[103] Ayesha Al Khoori. *Five Qataris found guilty of insulting UAE royals*. May 2015. URL: http://www.thenational.ae/uae/courts/20150518/five-qataris-found-guilty-of-insulting-uae-royals.

[104] @Kh_OZ. Twitter post. July 2013. URL: https://twitter.com/kh_oz/status/351828658371039233.

[105] Katie Kleemola, Masashi Crete-Nishihata, and John Scott-Railton. *Targeted Attacks against Tibetan and Hong Kong Groups Exploiting CVE-2014-4114*. June 2015. URL: https://citizenlab.org/2015/06/targeted-attacks-against-tibetan-and-hong-kong-groups-exploiting-cve-2014-4114/.

[106] Peter Kovessy and Heba Fahmy. *Report: UAE court convicts Qataris for social media insults*. May 2015. URL: http://dohanews.co/uae-court-convicts-qataris-for-insulting-royals-on-social-media/.

[107] Lavakumar Kuppan. "Attacking with HTML5." In: *Blackhat Abu Dhabi 2010*. Nov. 2010. URL: https://media.blackhat.com/bh-ad-10/Kuppan/Blackhat-AD-2010-Kuppan-Attacking-with-HTML5-slides.pdf.

[108] Citizen Lab. *Malware Signatures*. GitHub repository. URL: https://github.com/citizenlab/malware-signatures.

[109] Kaspersky Lab. *Features of using Kaspersky Anti-Virus 2015 with third-party firewalls*. June 2014. URL: http://support.kaspersky.com/us/11255.

[110] Attack Defense Labs. *JS-Recon: HTML5 based JavaScript Network Reconnaissance Tool*. URL: http://www.andlabs.org/tools/jsrecon.html.

[111] Twisted Matrix Labs. *Twisted Web*. URL: http://twistedmatrix.com/trac/wiki/TwistedWeb.

[112] Twisted Matrix Labs. *Twisted Web In 60 Seconds: Session Basics*. URL: http://twistedmatrix.com/documents/current/web/howto/web-in-60/session-basics.html.

[113] Fanny Lalonde Levesque et al. "A Clinical Study of Risk Factors Related to Malware Infections." In: *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*. New York, NY, USA: ACM, 2013.

[114] Bill Law. *Eight online activists 'arrested in UAE'*. Dec. 2012. URL: http://www.bbc.com/news/world-middle-east-20768205.

[115] Frankie Li, Anthony Lai, and Ddl Ddl. "Evidence of Advanced Persistent Threat: A case study of malware for political espionage." In: *MALWARE*. 2011.

[116] *Lookout, Inc.* URL: https://www.lookout.com.

[117] luckybuilding. *Port 80 is redirected to 30606 and No webpage is opened!* Wilders Security Forums. June 2008. URL: http://www.wilderssecurity.com/threads/port-80-is-redirected-to-30606-and-no-webpage-is-opened.212599/.

[118] Massimiliano Luppi. *I: I: BULL. RMI addtional questions*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/437543.

[119] Massimiliano Luppi. *R: RE: 2 OPPORTUNITIES*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/21775.

[120] Mostapha Maana. *Egypt: June 18-19 2013*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/602607.

[121] Daniel Maglietta. *RE: Expression of Interest*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/575806.

[122] *Making Twitter more secure: HTTPS*. 2011. URL: http://bit.ly/1i7l9kM.

[123] Mandiant. *APT1: Exposing One of China's Cyber Espionage Units*. 2013.

[124] Mandiant. *The Advanced Persistent Threat*. 2010.

[125] Bill Marczak. *spyware-scan: 1.1.py*. URL: https://github.com/citizenlab/spyware-scan/blob/master/ff/fingerprint-1.1/1.1.py.

[126] Bill Marczak and John Scott-Railton. *The Million Dollar Dissident: NSO Group's iPhone Zero-Days used against a UAE Human Rights Defender*. Aug. 2016. URL: https://citizenlab.org/2016/08/million-dollar-dissident-iphone-zero-day-nso-group-uae/.

[127] Bill Marczak, John Scott-Railton, and Sarah McKune. *Hacking Team Reloaded? US-Based Ethiopian Journalists Again Targeted with Spyware*. Mar. 2015. URL: https://citizenlab.org/2015/03/hacking-team-reloaded-us-based-ethiopian-journalists-targeted-spyware/.

[128] Bill Marczak et al. *Hacking Team and the Targeting of Ethiopian Journalists*. Feb. 2014. URL: https://citizenlab.org/2014/02/hacking-team-targeting-ethiopian-journalists/.

[129] William R. Marczak et al. "When Governments Hack Opponents: A Look at Actors and Technology." In: *USENIX Security 2014*. Aug. 2014.

[130] Attila Marosi. "Hacking Team: how they infected your Android device by 0days." In: *Hack.lu 2015*. 2015. URL: http://archive.hack.lu/2015/HT_Android_hack_lu2015_v1.0.pdf.

[131] Attila Marosi. *Inside Spying – FinSpy for Android*. 2014. URL: http://archive.hack.lu/2014/inside_spying_v1.4.pdf.

[132] Morgan Marquis-Boire. *Backdoors are Forever: Hacking Team and the Targeting of Dissent?* Oct. 2012. URL: https://citizenlab.org/2012/10/backdoors-are-forever-hacking-team-and-the-targeting-of-dissent/.

[133] Morgan Marquis-Boire and Bill Marczak. *From Bahrain With Love: FinFisher's Spy Kit Exposed?* July 2012. URL: https://citizenlab.org/2012/07/from-bahrain-with-love-finfishers-spy-kit-exposed/.

[134] Morgan Marquis-Boire et al. *Police Story: Hacking Team's Government Surveillance Malware*. June 2012. URL: https://citizenlab.org/2014/06/backdoor-hacking-teams-tradecraft-android-implant/.

[135] Susan E. McGregor et al. "Investigating the Computer Security Practices and Needs of Journalists." In: *USENIX Security 2015*. Aug. 2015.

[136] Jenna McLaughlin. *Spies for Hire*. Oct. 2016. URL: https://theintercept.com/2016/10/24/darkmatter-united-arab-emirates-spies-for-hire/.

[137] *Media Use in the Middle East*. 2015. URL: http://www.mideastmedia.org/survey/2015/.

[138] Trend Micro. *Cricut Gypsy does not synchronize when Tmproxy is enabled in Worry-Free Business Security (WFBS)*. Dec. 2015. URL: https://success.trendmicro.com/solution/1112123-cricut-gypsy-does-not-synchronize-when-tmproxy-is-enabled-in-worry-free-business-security-wfbs.

[139] mikeperry. *Tor Browser 3.6 is released*. Apr. 2014. URL: https://blog.torproject.org/blog/tor-browser-36-released.

[140] Daniele Milan. *Re: Fwd: The Globalization Of Cyberespionage*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/824583.

[141] @MiriamKhaled. Twitter post. Jan. 2012. URL: https://twitter.com/MiriamKhaled/status/156625204280434688.

[142] @MiriamKhaled. Twitter post. Jan. 2012. URL: https://twitter.com/Bu_saeed2/status/156781983983349760.

[143] *mitmproxy*. URL: https://mitmproxy.org/.

[144] HD Moore. *Critical Research: Internet Security Survey*. 2012. URL: https://scans.io/study/sonar.cio.

[145] D. M'Raihi et al. *TOTP: Time-Based One-Time Password Algorithm*. RFC 6238. May 2011. URL: https://tools.ietf.org/html/rfc6238.

[146] Shishir Nagaraja and Ross Anderson. *The snooping dragon: social-malware surveillance of the Tibetan movement*. Tech. rep. 2009.

[147] *National Endowment for Democracy*. URL: http://www.ned.org/.

[148] njq8. *New java drive-by 2013-1-11*. Jan. 2013. URL: http://www.dev-point.com/vb/t357796.html.

[149] *Node.js*. URL: https://nodejs.org/en/.

[150] Cyrus Ombati. *300 new spies will soon be in your midst*. Sept. 2010. URL: http://www.standardmedia.co.ke/business/article/2000017897/300-new-spies-will-soon-be-in-your-midst.

[151] Alberto Ornaghi. *em-http-server*. GitHub repository. URL: https://github.com/alor/em-http-server.

[152] OWASP. *Cross-Site Request Forgery (CSRF) Prevention Cheat Sheet*. URL: https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)_Prevention_Cheat_Sheet#Double_Submit_Cookie.

[153] ownCloud. *History*. URL: https://owncloud.org/history/.

[154] Peace1. *Avira Antivir Premium 44081 connection?* Wilders Security Forums. Dec. 2011. URL: http://www.wilderssecurity.com/threads/avira-antivir-premium-44081-connection.314989/.

[155] Armando Perez. *Re: QUOTE MEXICO URGENT*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/5391.

[156] proofpoint. *Threat Protection and Compliance for Office 365*. URL: https://www.proofpoint.com/us/solutions/threat-protection-compliance-office-365.

[157] Quasar. *QuasarRAT: InternetExplorer.cs*. Source code. URL: http://bit.ly/2hovQXx.

[158] @r7aluae2. Twitter post. Jan. 2012. URL: https://twitter.com/r7aluae2/status/156418043424157696.

[159] Rapid7. *Project Sonar*. URL: https://sonar.labs.rapid7.com/.

[160] RARLAB. *WinRAR and RAR archiver addons*. URL: http://www.rarlab.com/rar_add.htm.

[161] *'Reinstate sacked official' call*. 2013. URL: http://bit.ly/1aRUZ4b.

[162] RiskIQ. *PassiveTotal*. URL: https://passivetotal.org/.

[163] Davide Romualdi. *RE: Foto TNI*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/1081335.

[164] *Royal Group*. URL: http://www.royalgroupuae.com/.

[165] Alessandro Scarafile. *I: Delivery Egypt (TREVOR)*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/1030236.

[166] Alessandro Scarafile. *I: RE: TRD-GNSE- DELIVERY*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/14684.

[167] John Scott-Railton and Katie Kleemola. *London Calling: Two-Factor Authentication Phishing From Iran*. Aug. 2015. URL: https://citizenlab.org/2015/08/iran_two_factor_phishing/.

[168] Sebastian. *New Tor Browser Bundles for Windows*. May 2012. URL: https://blog.torproject.org/blog/new-tor-browser-bundles-windows.

[169] @shamsiuae58. Twitter post. May 2013. URL: https://twitter.com/Dwight389/status/332452681325088768.

[170] Micah Sherr et al. "Can They Hear Me Now? A Security Analysis of Law Enforcement Wiretaps." In: *ACM CCS*. 2009.

[171] *Shodan*. URL: https://shodan.io.

[172] YOURLS: Your Own URL Shortener. *About YOURLS*. URL: https://yourls.org/#About.

[173] YOURLS: Your Own URL Shortener. *Spam*. URL: https://github.com/YOURLS/YOURLS/wiki/Spam.

[174] YOURLS: Your Own URL Shortener. *YOURLS: Your Own URL Shortener*. GitHub repository. URL: https://github.com/YOURLS/YOURLS.

[175] Vernon Silver. *Gamma Says No Spyware Sold to Bahrain; May Be Stolen Copy*. July 2012. URL: http://bloom.bg/17SOXQs.

[176] Vernon Silver. *Spyware Leaves Trail to Beaten Activist Through Microsoft Flaw*. Oct. 2012. URL: http://bloom.bg/1ja2geI.

[177] *Socket.IO*. URL: http://socket.io/.

[178] Fidelis Cybersecurity Solutions. ""njRAT" Uncovered." In: (2013). URL: http://bit.ly/1eJheel.

[179] *SPY NET*. URL: http://newspynetrat.blogspot.com/.

[180] Hacking Team. *Client Overview List*. Hacking Team leak. URL: http://bit.ly/2hRFuPx.

[181] Hacking Team. *Fatturato Offensiva 2008-2013*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/fileid/2886/1050.

[182] Hacking Team. *Hacking Team RCS Invisibility Report*. Hacking Team leak. URL: https://theintercept.com/document/2014/10/30/hacking-team-rcs-invisibility-report/.

[183] Hacking Team. *RCS9 System Administrator's Guide*. Sept. 2013. URL: https://s3.amazonaws.com/s3.documentcloud.org/documents/1348001/rcs-9-sysadmin-final.pdf.

[184] Hacking Team. *rcs-collector: events.rb*. Source Code. URL: https://github.com/hackedteam/rcs-collector/blob/master/lib/rcs-collector/events.rb.

[185] Microsoft TechNet. *TcpInitialRTT*. URL: https://technet.microsoft.com/en-us/library/cc938207.aspx.

[186] Microsoft TechNet. *Windows Time Service Tools and Settings*. May 2012. URL: https://technet.microsoft.com/en-us/library/cc773263(v=ws.10).aspx.

[187] *TEMU: The BitBlaze Dynamic Analysis Component*. URL: http://bit.ly/1clcxSZ.

[188] New Orleans The National WWII Museum. *Fighting for the Right to Fight: African American Experiences in WWII*. URL: http://righttofightexhibit.org/home/.

[189] Charissa Tobing. *Re: Expression of Interest*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/601732.

[190] *Ultimate Packer for eXecutables*. URL: http://upx.sourceforge.net/.

[191] *Unionist Questioned*. 2013. URL: http://bit.ly/1gHnBiS.

[192] *Unionist Sacked for Concealing Information*. 2013. URL: http://bit.ly/1bsKRNy.

[193] *Unpacking VBInject/VBCrypt/RunPE*. 2010. URL: http://bit.ly/1e28nS2.

[194] Nart Villeneuve. *Fake Skype Encryption Service Cloaks DarkComet Trojan*. Apr. 2012. URL: http://bit.ly/17SpA1c.

[195] David Vincenzetti. *Re: ISS Kuala Lumpur 2012*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/594340.

[196] David Vincenzetti. *Re: Wall Street Journal article*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/171252.

[197] Philippe Vinci. *Fwd: PUMA updated Proposal - version 3*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/1094866.

[198] *VirusTotal Intelligence - Your malware research telescope*. URL: https://virustotal.com/intelligence/.

[199] Rick Wash. "Folk Models of Home Computer Security." In: *SOUPS*. July 2010.

[200] Adam Weinberg. *New opprtunity - Bahrain*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/446624.

[201] Wikipedia. *Cross-origin resource sharing*. URL: https://en.wikipedia.org/wiki/Cross-origin_resource_sharing.

[202] Wikipedia. *Dm-crypt*. URL: https://en.wikipedia.org/wiki/Dm-crypt.

[203] *Wireshark*. URL: https://www.wireshark.org/.

[204] Scott Wolchok, Randy Yao, and J Alex Halderman. *Analysis of the Green Dam Censorware System*. Tech. rep. 2009.

[205] Serge Woon. *Japan Holds First, Government-Sanctioned Hacking Contest*. Hacking Team leak. URL: https://wikileaks.org/hackingteam/emails/emailid/565854.

[206] Xueyang Xu, Z Morley Mao, and J Alex Halderman. "Internet Censorship in China: Where Does the Filtering Occur?" In: *PAM*. 2011.

[207] Danny Yadron. *Can This Israeli Startup Hack Your Phone?* Aug. 2014. URL: http://blogs.wsj.com/digits/2014/08/01/can-this-israeli-startup-hack-your-phone/.

[208] *YARA - The pattern matching swiss knife for malware researchers*. URL: http://plusvic.github.io/yara/.

[209] ZeroMQ. *Distributed Messaging - ZeroMQ*. URL: http://zeromq.org/.