UNIVERSITY OF TORONTO

MUNK SCHOOL OF GLOBAL AFFAIRS

Join the Global Conversation

The Citizen Lab

Research Brief
June 2014

Police Story: Hacking Team's Government Surveillance Malware

Authors: Morgan Marquis-Boire, John Scott-Railton, Claudio Guarnieri, and Katie Kleemola

For **media coverage** of this report, see The Economist, Associated Press, Wired, VICE, International Business Times, Forbes, Ars Technica, Kaspersky's Threatpost, Slash Gear, Engadget, Help Net Security, The Verge, CIO Today, Voice of America (VOA), Computer World, Infosecurity Magazine, and Slate.

Read the Arabic Version / ال عرب ية ال نسخة translated by Cyber Arabs

The left newspapers might whine a bit

But the guys at the station they don't give a shit

Dispatch calls "Are you doin' something wicked?"

"No siree, Jack, we're just givin' tickets"

Police Truck, Dead Kennedys (1980)

SUMMARY

- In **Part 1**, we analyze a newly discovered Android implant that we attribute to Hacking Team and highlight the political subtext of the bait content and attack context.
- In **Part 2**, we expose the functionality and architecture of Hacking Team's Remote Control System (RCS) and operator tradecraft in never-before published detail.

INTRODUCTION

This report analyzes Hacking Team's Android implant, and uses new documents to illustrate how their Remote Control System (RCS) interception product works. This work builds on our previous research into the technologies and companies behind "lawful interception" malware. This technology is marketed as filling a gap between passive interception (such as network monitoring) and physical searches. In essence, it is malware sold to governments. Unlike phone monitoring and physical searches, however, most countries have few legal guidelines and oversight for the use of this new power. In light of the absence of guidelines and oversight, together with its clandestine nature, this technology is uniquely vulnerable to misuse. By analysing the tools, and their proliferation at the hands of companies like Hacking Team and Gamma Group, we hope to support efforts to ensure that these tools are used in an accountable way, and not to violate basic principles of human rights and rule of law.

In a report published earlier this year, we presented the results of a global scanning effort, and identified 21 countries with deployments of Hacking Team's Remote Control System monitoring solution. In addition, alongside other researchers, we have uncovered a range of cases where "lawful interception" software has been used against political targets by repressive regimes. Political and civil society targets have included Mamfakinch in Morocco, human rights activist Ahmed Mansoor in the UAE, and ESAT, a US-based news service focusing on Ethiopia. In all of these cases, a tool marketed for "law enforcement" was used against *political*, rather than *security* threats. In still other cases, like Malaysia [PDF], we have found bait documents and seeding suggestive of political targeting.

PART 1: AN ANDROID HACKING TEAM BACKDOOR IN SAUDI ARABIA

Protests in Saudi Arabia and Qatif

While Saudi Arabia has not seen protests comparable to those elsewhere during the Arab Spring, it has experienced protests since 2011, primarily in the Ash-Sharqīyah province. There are a number of reasons for political tensions, ranging from demographic pressures, cost of housing, and unemployment, to issues of women's and minority rights. The province is predominantly Shia, who have long-standing grievances over perceived political and cultural marginalization by the Sunni ruling regime. These grievances were magnified when, in early 2011, the Bahraini government violently suppressed Shia protests with the assistance of Saudi Arabian troops.

Protests then spread in a number of areas, including in the predominantly Shia Qatif Governorate. In 2011, Shia most protesters appear to have initially demanded reform, rather than the regime-change advocated in other Arab countries. Interestingly, Qatif has a history of Shia protest, most famously in widespread protests in 1979. In response to the protests, which demanded greater political and economic participation, the regime provided extensive economic concessions. In 2011, however, authorities responded with violence and arrests of prominent Shia figures. Protesters were wounded and others allegedly killed by security forces, according to Human Rights Watch. This crackdown may have contributed to shifting protesters' demands; today, some explicitly demand regime change using secular language, according to researchers and journalists directly familiar with recent developments who spoke with us. In what might be described as an inflammatory response, Saudi authorities also arrested an outspoken and highly visible Shia Sheikh.

"… the prosecutor demanded he face not only the death sentence, but an additional punishment mandated by sharia law for the most heinous offences in which the dead body is defiled by being hanged from a pole." -Reuters

The escalation, which has been accompanied by violence against security services among some Shia, is used by the regime to justify harsh measures, including "riot control," arrests, and sentences including death for protesters on charges of "Sedition." Others have been charged with espionage on behalf of Iran, in a case that has been claimed by many Shia to have been politically motivated.

Human rights organizations that have catalogued alleged abuses, like the Adala Center for Human Rights, have been denied the ability to register as formal organizations, subject to URL blocking, and had staff harassed and imprisoned. Journalists attempting to report on Qatif are blocked from entering, and regularly subjected to threats and government pressure.

Social media and mobile phones are a key part of how protests are organized, with protesters taking measures, like using pseudonymous accounts, to share their message. Nevertheless, according to people familiar with the events, digital operational security practices are often piecemeal, and do not match the capabilities of the security services.

Surveillance, Monitoring and Information Control in Saudi Arabia

Saudi Arabia is a unique and complex security environment, and its security services play a range of roles. On the one hand, Saudi Arabia faces *undeniable* foreign and domestic security threats from hostile groups, extremists and other governments. On the other hand, the regime has been exceptionally aggressive in its attempts to control and stifle dissent and political pluralism.

The security services in Saudi Arabia make use of a range of instruments of formal and informal state power to control the electronic information environment in the country. Beginning at government-maintained Internet chokepoints and extending to ISPs, the state blocks a wide range of political, religious and cultural content. This includes social media, whether specific users or whole platforms. Extending further, the state requires that news websites (defined broadly) register with the authorities. Registered websites are subject to extensive regulation, while unregistered operators that have not registered risk severe penalties. Site operators are encouraged to self-monitor and moderate content, under threat of financial penalties, jail time, and corporal punishment like lashes. In addition, anti-cybercrime legislation has also been used to prosecute online dialogue that most societies would consider acceptable political speech.

The public use of mobile monitoring extends into forms of social control that many societies would find highly objectionable. For example, the government earned international condemnation when it announced that it would implement a system to enable their male guardian to monitor the travel behavior of women under their care. Replacing an older permission-slip based system ("yellow cards"), male guardians receive text messages when women arrive on the premises of the international airport, asking whether the women are permitted to travel.

Internet and social media users are encouraged to self-censor and report on each other. The government engages in public advertising campaigns to encourage both behaviours, and makes it clear to Saudi citizens that they are watching, and listening. In particular, the state has implemented specific penalties for re-sharing, privately or publicly, content deemed objectionable. In addition to using the explicit tools of the law, it is widely believed that the state encourages an "electronic army" of pro-government individuals to swamp social media conversations with pro-regime voices and harass dissenters.

Speech involving religious themes is especially risky, as the government is willing to use serious religious charges, including the death penalty and corporal punishment, and tools of international jurisprudence like extradition, to detain and punish those who violate its strict norms for political, religious, and cultural speech.

In two notable cases, the operator of the Saudi Liberals discussion forum was eventually sentenced to 10 years in prison and 1000 lashes for maintaining a forum on the discussion of religion and reform. This was a reduction of sorts, as the prosecution demanded his execution. Many similar cases, using various charges, have been reported by human rights organizations and commentators.

These measures have an intentionally chilling effect on political speech, and are regularly the subject of criticism by the international human rights community. Nevertheless, social media remains the primary outlet for political speech. Many users practice some degree of self censorship, or indirect speech, while others use pseudonyms and other technical means to preserve their anonymity. To access banned content, the use virtual private networks (VPNs) is also common. In response, security services use police and investigative powers to unmask the posters, and punishes them severely, sometimes after arrests where the name of the detainee(s) is kept secret.

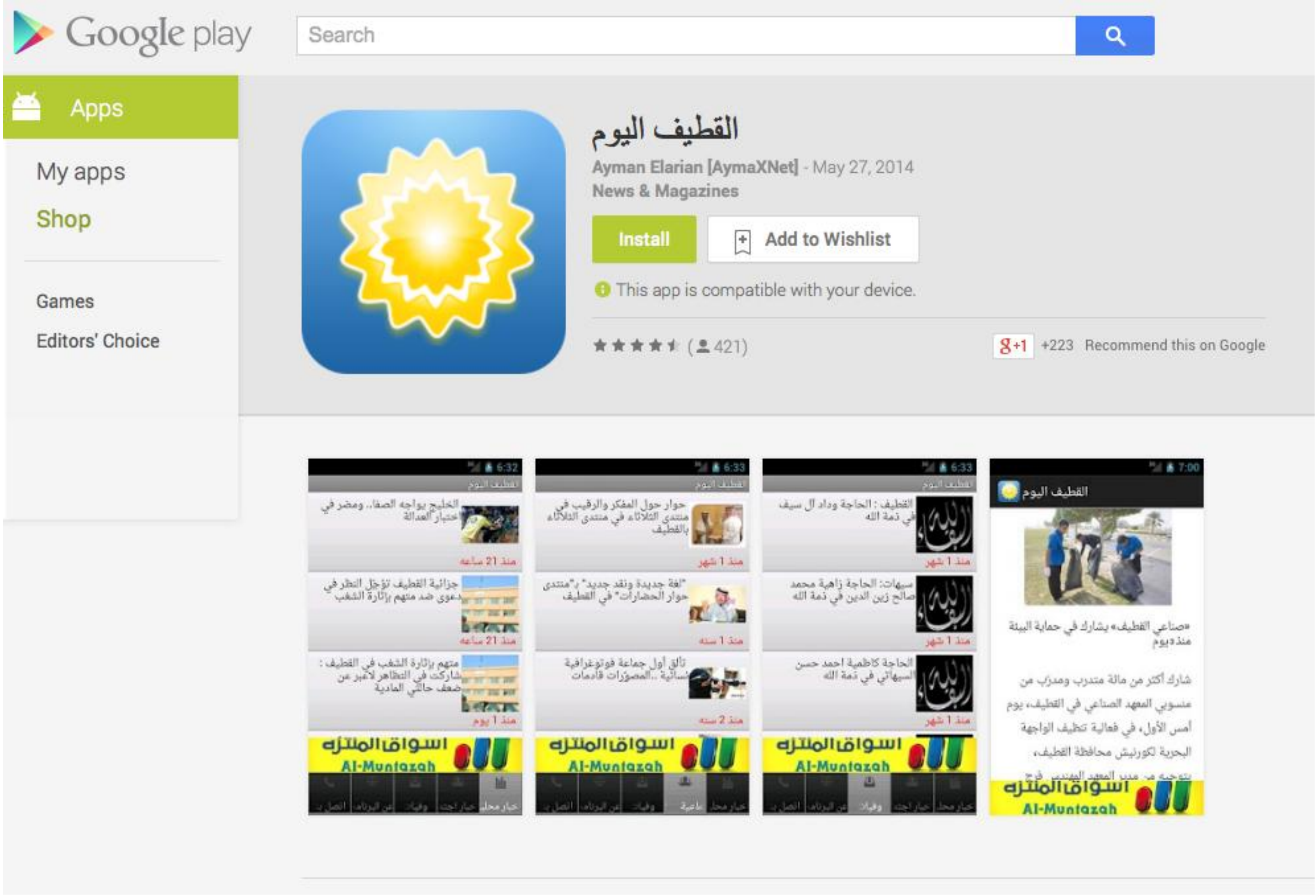
Phone use in Saudia Arabia has a penetration rate of 170%, with an estimated average of 30% of individual income spent on mobile phone and Internet costs in 2014. As a result, older mechanisms of Internet surveillance, like monitoring Internet cafes, are being replaced. Individual users are required to use real identities when registering mobile devices, and it is clear that the state is seeking greater visibility into encrypted traffic. In 2010, for example, Saudi Arabia successfully gained access to BlackBerry communications after making Saudi-located servers a quid-pro-quo of allowing the devices on Saudi Networks. More recently, the government's appetite for encrypted communications was revealed by Moxie Marlinspike, a security researcher and developer, who received an overture from Saudi telecom company Mobily seeking his assistance in accessing encrypted traffic. The firm was seeking an intercept solution (on request of the Saudi Government, they said) for access to a range of mobile chat clients (Viber, LINE, WhatsApp) as well as the mobile-version of Twitter.

The use of mobile malware can be understood as part of this desire, by no means limited to Saudi Arabia, to match the technologies in use by their population.

Seeding: A Lure with Political Subtext?

Using signatures developed as part of our ongoing research into "lawful intercept" malware developed by Hacking Team, we identified a suspicious Android installation package (APK). The file was a functional copy of the 'Qatif Today' (الـ يوم الـ قط يف) news application bundled with a Hacking Team payload. Documents we have reviewed suggest that Hacking Team refers to this kind of mobile implant as an "Installation Package," where a legitimate third party application file is bundled with the implant (See: Developing and Deploying Implants). This kind of tactic with Android package implants has been seen in other targeted malware attacks (that do not use commercial "lawful intercept" products) including the LuckyCat campaign, and in attacks against Tibetan activists, and groups in the Uyghur community.

The *genuine* 'Qatif Today' app is an Android (download here) and iPhone application that provides news and information in Arabic with a special relevance to the Qatif Governorate of Saudi Arabia.



القطيف اليوم

Ayman Elarian [AymaXNet] - May 27, 2014

News & Magazines

This app is compatible with your device.

★★★★★ (421)

+1 +223 Recommend this on Google

Qatif Today in the Google Play app store

The connection to Qatif is interesting, given the recent history of protest in Qatif as outlined above. We are not in a position to determine the identity of the group or individual targeted with this malware, however, we speculate that the attack may be linked to political protest in eastern Saudi Arabia.

Hacking Team Samples

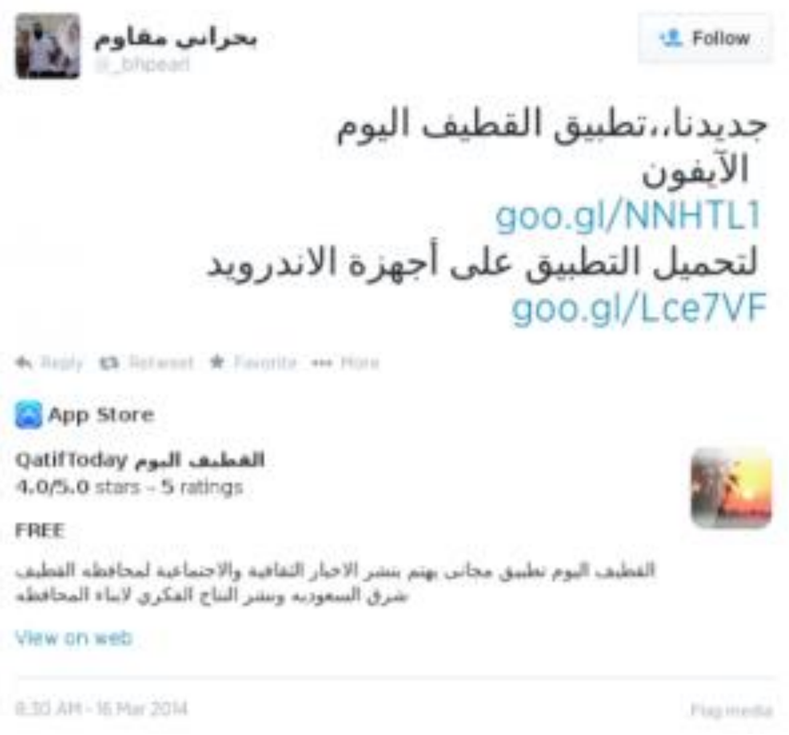
The malicious APK, QatifNews.apk, has the following hash:

8e64c38789c1bae752e7b4d0d58078399feb7cd3339712590cf727dfd90d254d

At the time of first submission to the VirusTotal database, the file was detected by zero out of 50 AntiVirus products in VirusTotal:

0 / 50 2014-03-11 09:28:49 2014-03-11 09:28:49

It appears that an APK of the same name was seeded on Twitter 5 days later by a Twitter account (@_bhpearl) linked to Bahrain, a country of great interest to Shia in Qatif.



بحراني مقاوم
@_bhpearl

جديدنا،،تطبيق القطيف اليوم
الآيفون
goo.gl/NNHTL1
لتحميل التطبيق على أجهزة الاندرويد
goo.gl/Lce7VF

App Store

QatifToday القطيف اليوم
4.0/5.0 stars - 5 ratings

FREE

القطيف اليوم تطبيق مجاني يهتم بنشر الاخبار الثقافيه والاجتماعية لمحافظه القطيف شرق السعوديه ونشر النتاج الفكري لابناء المحافظه

8:30 AM - 16 Mar 2014

Tweet with links by @_bhpearl (since deleted)

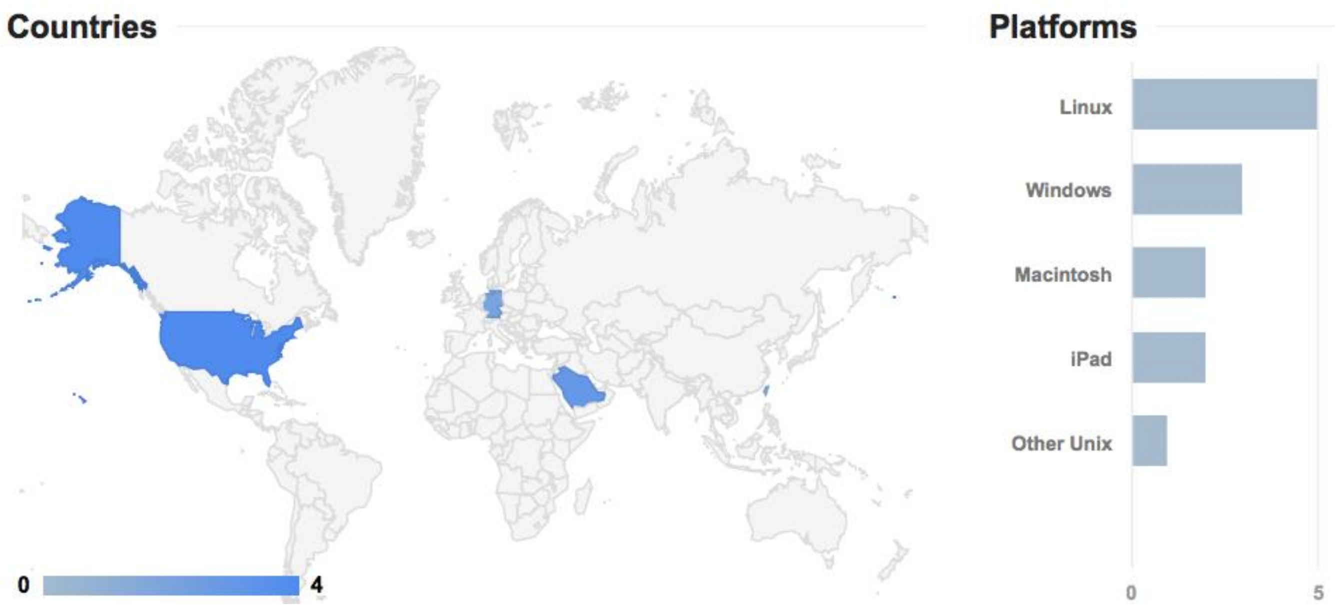
The tweeted links were shortened using the goo.gl service. These resolved to:

https://itunes.apple.com/app/qatiftoday-alqtyf-alywm/id584947120?mt=8

and

https://dl.dropboxusercontent.com/s/fw92fsu9r694iqc/QatifNews.apk

The first link to the iTunes store has 18841 clicks on the shortened link and appears legitimate. The second link, however, does not redirect to the genuine app at the Android App Store. Instead, it redirects to a Dropbox file that has since been removed. Examination of analytics on the shortened second link is interesting; there were only 13 clicks. We can discount 7 of these clicks (those in US and Germany) as researchers, while three are in Saudi Arabia. One click in Taiwan may be a VPN, or a security researcher.



Google Shortener Analytics

While we cannot confirm that the file on Dropbox was the same APK, we suspect the timing of the tweet, and the use of a non-standard method for sharing an Android application, is not a coincidence.

Malicious APK Code Signing

The malicious apk was signed by the following certificate:

Issuer:
DN: C=US, O=Sun, OU=JavaSoft, CN=Server
C: US
CN: Server
O: Sun
OU: JavaSoft
Subject:
DN: C=US, O=Sun, OU=JavaSoft, CN=Server
C: US
CN: Server
O: Sun
OU: JavaSoft
Serial: 1369041295
SHA256 fingerprint: 8ab03660fe537994b207e900f8e2f5711c08e61232e167ce97e52bb3fd77757f

A broader discussion of code-signing practices for Hacking Team implants is discussed later in the document in the section "Code Signing and Certificates".

Additional Samples

We were able to identify an additional six samples signed with the same certificate:

e85db2d92ae97f927905d6e931a0cb1f5b6def2475c23e023fe617249ddddb4
0b657e29a3e249b414fbde3a85e7be0829ddaad49b8ff2832350cfe5af190ba1
535070b5bd076f137052eb82257f16db4c3ba3e3516970b8934524e4a750a8f1
748e04aee9ec1a82abd2f0a9d3ddc33925a8a74b5058088dbf3d6a3c1e9698d8
8d2012d44208e79ea6e511847f86af0c45271e6f678ccc19639fe6c2eee75449
e6a77c8bf232636a505c31fae0215789caf8d73682102304a2541513de945526

The first sample of this list (…db4) was submitted to VirusTotal as:

Date	File Name	Submitter ID	Submitter Country
2012-08-28 10:06:01	rcs.apk	18e48a9b (api)	Germany

Interestingly, this was the first sample (chronologically) to be submitted to VirusTotal from this group of files. The name “rcs.apk” appears to be an abbreviation of the name of Hacking Team’s targeted surveillance solution, ‘Remote Control System.’

Permissions

The original app requires the following permissions:

android.permission.INTERNET
android.permission.GET_ACCOUNTS
android.permission.WAKE_LOCK
android.permission.READ_PHONE_STATE
android.permission.ACCESS_FINE_LOCATION
com.google.android.c2dm.permission.RECEIVE
com.aymax.qatiftoday.permission.C2D_MESSAGE
android.permission.USE_CREDENTIALS
android.permission.WRITE_EXTERNAL_STORAGE
android.permission.ACCESS_NETWORK_STATE
android.permission.VIBRATE

The implant requests the following permissions, which give it the ability to process calls, read and write SMS messages, monitor the user’s GPS location, and more.

android.permission.ACCESS_COARSE_LOCATION
android.permission.ACCESS_WIFI_STATE
android.permission.CALL_PHONE
android.permission.CAMERA
android.permission.CHANGE_NETWORK_STATE
android.permission.CHANGE_WIFI_STATE
android.permission.FLASHLIGHT
android.permission.PROCESS_OUTGOING_CALLS
android.permission.READ_CALENDAR
android.permission.READ_CONTACTS
android.permission.READ_LOGS

android.permission.READ_PHONE_STATE
android.permission.RECEIVE_BOOT_COMPLETED
android.permission.RECEIVE_SMS
android.permission.RECORD_AUDIO
android.permission.SEND_SMS
android.permission.SET_WALLPAPER
android.permission.USER_PRESENT
android.permission.WRITE_SMS

The following permission was requested by several of the other identified Hacking Team android implants:

android.permission.WRITE_APN_SETTINGS (write Access Point Name settings)

This value allows applications to change the APN (Access Point Name), a setting on a mobile phone that identifies an external network the phone can access for data. This value is marked as *not for use by third party applications* in the Android Developers reference.[1] For further discussion of this functionality, see: Other Capabilities.

Behavior & Network Communication

After analyzing the behavior of the application we were able to further confirm the malicious nature of the file.

When executed, the app performs a POST to:

http://iphone.al-motamiz.com/qatiftoday/gcms/register.php

This server is hosted at iWeb in Montreal:

IP Address	174.142.97.245
Host	server.5edma.com
Location	🇨🇦 CA, Canada
City	Montréal, QC H3E 1Z6
Organization	IWeb Technologies
ISP	IWeb Technologies
AS Number	AS32613 iWeb Technologies Inc.

We believe this to be *legitimate* communication by the decoy application.

We also observed command and control (C2) communication with two additional servers. We later found the same IP addresses in decrypted text from the implant's config file (see: Obfuscation / Implant Configuration). Incidentally, this was consistent with how Hacking Team implants store C2 addresses.

http://91.109.17.189/

http://106.186.17.60/

The first server is hosted on Leaseweb in Germany:

Host	91.109.17.189
Location	🇩🇪 DE, Germany
Organization	Leaseweb Germany GmbH (previously netdirekt e. K.)
AS Number	AS16265 LeaseWeb B.V.

The second is hosted on Linode in Japan:

IP Address	106.186.17.60
Host	li528-60.members.linode.com
Location	🇯🇵 JP, Japan
Organization	Kddi Corporation
AS Number	AS2516 KDDI KDDI CORPORATION

In previous work, we identified both the Leaseweb and the Linode IPs as part of a Hacking Team proxy chain uncovered in our prior reporting. At the time, we also identified a third IP (206.190.155.40) that belonged in the same group. This finding indirectly validates the methods used in the previous report, and highlights the third IP as likely part of the same collection infrastructure.

91.109.17.189
206.190.155.40
106.186.17.60

This additional IP is in a block owned by the hosting provider SoftLayer Technologies.

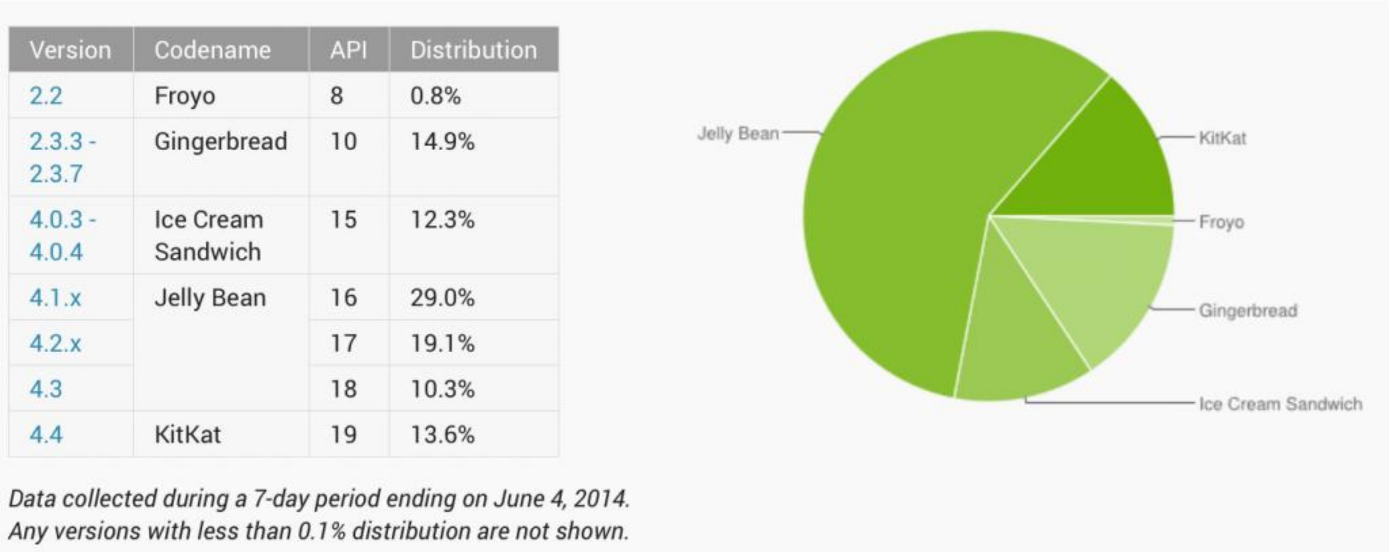
Rooting Exploit

The malicious implant attempts to stat() the following files:

/dev/exynos-mem
/dev/DspBridge
/dev/s5p-smem

This behavior is consistent with a known exploit (CVE-2012-6422) that permits a user without permissions to write to a compromised device's physical memory. An in depth analysis of this exploit is provided by Azimuth Security here.

While this exploit would not be effective against the latest version of the Android operating system, a high percentage of users still use legacy versions which may be vulnerable.[2]



Version	Codename	API	Distribution
2.2	Froyo	8	0.8%
2.3.3 - 2.3.7	Gingerbread	10	14.9%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	12.3%
4.1.x	Jelly Bean	16	29.0%
4.2.x		17	19.1%
4.3		18	10.3%
4.4	KitKat	19	13.6%

Data collected during a 7-day period ending on June 4, 2014.
Any versions with less than 0.1% distribution are not shown.

Image Credit: Google

Background on Rooting Android Devices

Rooting an Android phone typically involves two components: the *su* binary and a *supervisor application*.

In order to install the su binary, either the device manufacturer has allowed the boot loader to be unlocked or a vulnerability is exploited to temporarily elevate permissions. Once the su binary is installed, applications may use it in a controlled way to elevate their permissions for other purposes.

The su is much like the traditional Unix – it uses the suid flag so applications that call it elevate their permissions to that of the executable they are calling – in most cases, including this one, root. Each application on Android is run by its own user in order to prevent applications from accessing resources it does not have permissions for.

A supervisor application (usually Superuser.apk) is used by people who intentionally root their phone so the su binary can call the supervisor which will then notify the user and allow them to approve or deny the privilege escalation. Thus, even with a rooted phone the user is still notified when applications perform actions requiring root access.

How this applies to the RCS Android Implant

In this case a vulnerability is exploited to initially gain root access, and a binary is dropped. It copies itself to /system/bin/rilcap (through /proc/self/exe) and sets its owner to be root as well as setting the executable and suid flags. This allows for persistent root access.

/system/bin/rilcap 6250af9750606a4a06ca1ec0bfa127d0e37e1c7676e37773f461a91bfe0daf93

RILCAP Functionality

Examination of the binary revealed that basic techniques such as simple encryption of strings are used to hinder analysis.

Encryption algorithm:

```
if str is b[0]…b[n],

len = b[0] ^ b[1] ^ b[2]

for b[i] in b[3]…b[n]:
b[i] = b[i] ^ b[1]
b[i] = (b[i] – b[1]) & 0xff
b[i] = b[i] ^ b[0]
```

The binary contains the following functionality:

- Capture information from the framebuffer (/dev/graphics/fb0) — which can then be used to assemble screenshots
- Kill the volume daemon (which automatically detects and mounts storage devices when added)
- Mount /system as read only or read write
- Check /proc/mounts for an sdcard
- Execute commands from unprivileged applications using system()
- Execute commands from a file using execv /system/bin/sh
- Modify /data/system/device-policies.xml to add the application as device admin
- Copy files
- Modify file permissions
- Change file owners
- Unlink self
- Restart the device

This functionality allows RILCAP to act as the su binary would — only there is no supervisor application to notify the user of privilege escalation and offer the choice to allow or deny it.

A full list of of commands can be found in Appendix B.

Functionality

Looking at the samples as a group, we identified a range of behaviors indicative of the implant’s surveillance capabilities.

Attempts to access 3rd party chat / voice apps:

We found that the apps attempt to access the local files stored by popular social media, chat, and call apps including **Facebook, Viber, WhatsApp, Skype, LINE and QQ**.

/data/data/com.facebook.katana/databases
/data/data/com.facebook.orca/databases
/data/data/com.skype.raider/files/shared.xml
/data/data/com.whatsapp/shared_prefs
/data/data/com.whatsapp/shared_prefs/RegisterPhone.xml
/data/data/com.viber.voip/files
/data/data/com.tencent.mm/MicroMsg/
/data/data/com.viber.voip/files/preferences/reg_viber_phone_num
/data/data/jp.naver.line.android/databases/naver_line_myhome
/data/data/jp.naver.line.android
/data/data/jp.naver.line.android/databases

In addition, the app accesses the locally stored mail files belonging to the compromised user’s mail account.

/data/data/com.google.android.gm/databases/mailstore.{username}@gmail.com.db

Obfuscation / Implant Configuration

In order to prevent additional scrutiny into the internals of the backdoor, the source code appears to have been heavily obfuscated through the use of DexGuard (0r similar), which can provide encryption for strings, entire classes and assets, considerably slowing and complicating the reverse engineering process.

While analyzing the implant, we de-obfuscated the configuration file. Below, we highlight a number of lines that match functionality in RCS (for a more complete list see: Implant Modules and Functionality).

We find a range of audio recording, camera, video, key logging, “live mic,” chat, device info etc. configuration settings relevant to the surveillance functionality of the implant. We also note the presence of a “crisis” module, which provides anti-analysis functionality explained below (see: Anti-Analysis Functionality & Anti Forensics).

{“record”:true,”compression”:5,”buffer”:512000,”module”:”**call**“},{“module”:”**camera**“,”quality”:”med”},{“module”:”**chat**“},{“module”:”**clipboard**“},{“module”:”**conference**“,”number”:””},{“synchronize”:false,”position”:true,”module”:”**crisis**“,”mic”:true,”network”:{“processes”:[],”enabled”:false},”call”:true,”camera”:true,”hook”:{“processes”:[],”enabled”:true}},{“module”:”**device**“,”list”:false},{“module”:”**keylog**“},{“module”:”**livemic**“,”number”:””}

We also see what appear to be, location, screenshot-taking, and browsing activity modules.

{“cell”:true,”gps”:false,”wifi”:true,”module”:”**position**“},{“quality”:”med”,”module”:”**screenshot**“,”onlywindow”:false},{“module”:”**url**“}

We also found that the implant seems to have been deployed with a filter to specify a date range (“datefrom” and “dateto” for the Mail, SMS and MMS messages it is seeking.

{“**mail**“:{“filter”:{“datefrom”:”2014-03-10 00:00:00″,”maxsize”:100000,”dateto”:”2100-01-01 00:00:00″,”history”:true},”enabled”:true},”**mms**“:{“filter”:{“datefrom”:”2014-03-10 00:00:00″,”dateto”:”2100-01-01 00:00:00″,”history”:true},”enabled”:true},”module”:”messages”,”**sms**“:{“filter”:{“datefrom”:”2014-03-10 00:00:00″,”dateto”:”2100-01-01 00:00:00″,”history”:true},”enabled”:true}}

We also see information about how the implant exfiltrates data, along with its C2 servers. Interestingly, it appears that the implant is capable of monitoring the devices connectivity (e.g. wifi, cellular network), choosing connection type, and rate limiting the bandwidth. Note that these are the same servers we observed in the implant’s network communications.

{“desc”:”SYNC”,”subactions”:[{"host":"**91.109.17.189**","mindelay":0,"stop":true,"**cell**":true,"action":"synchronize","**wifi**":true,"maxdelay":0,"**bandwidth**":500000},{"host":"**106.186.17.60**","mindelay":0,"stop":false,"cell":true,"action":"synchronize","wifi":true,"maxdelay":0,"bandwidth":500000}]}]}

PART 2: HACKING TEAM RCS OPERATION

After we released our research on Hacking Team earlier this year, we were sent documentation by an anonymous individual pertaining to the set up and operation of Hacking Team’s RCS that Hacking Team provides to its customers. The documents appear to date from fall 2013. We have no knowledge as to the origin of the documents, and whoever sent them took steps to conceal their identity. While **the authenticity of these documents is unverified**, we have not identified inconsistencies with what is currently known about Hacking Team RCS (our latest findings included). We are concerned, however, that releasing the documents in their full form could bring risk to the source. The following is thus a general overview of some of the functionality and specifics of the Remote Control System, **according to these documents**. The screenshots that we excerpt are clearly from instances marked as “Demo*”* copies, rather than actual operations.

Architecture of a Hacking Team RCS Deployment

The following image suggests the logical architecture of a prototypical Hacking Team RCS deployment. In specific instances, according to these documents, a “distributed” architecture can be used (see: Appendix: Distributed Hacking Team RCS Architecture).

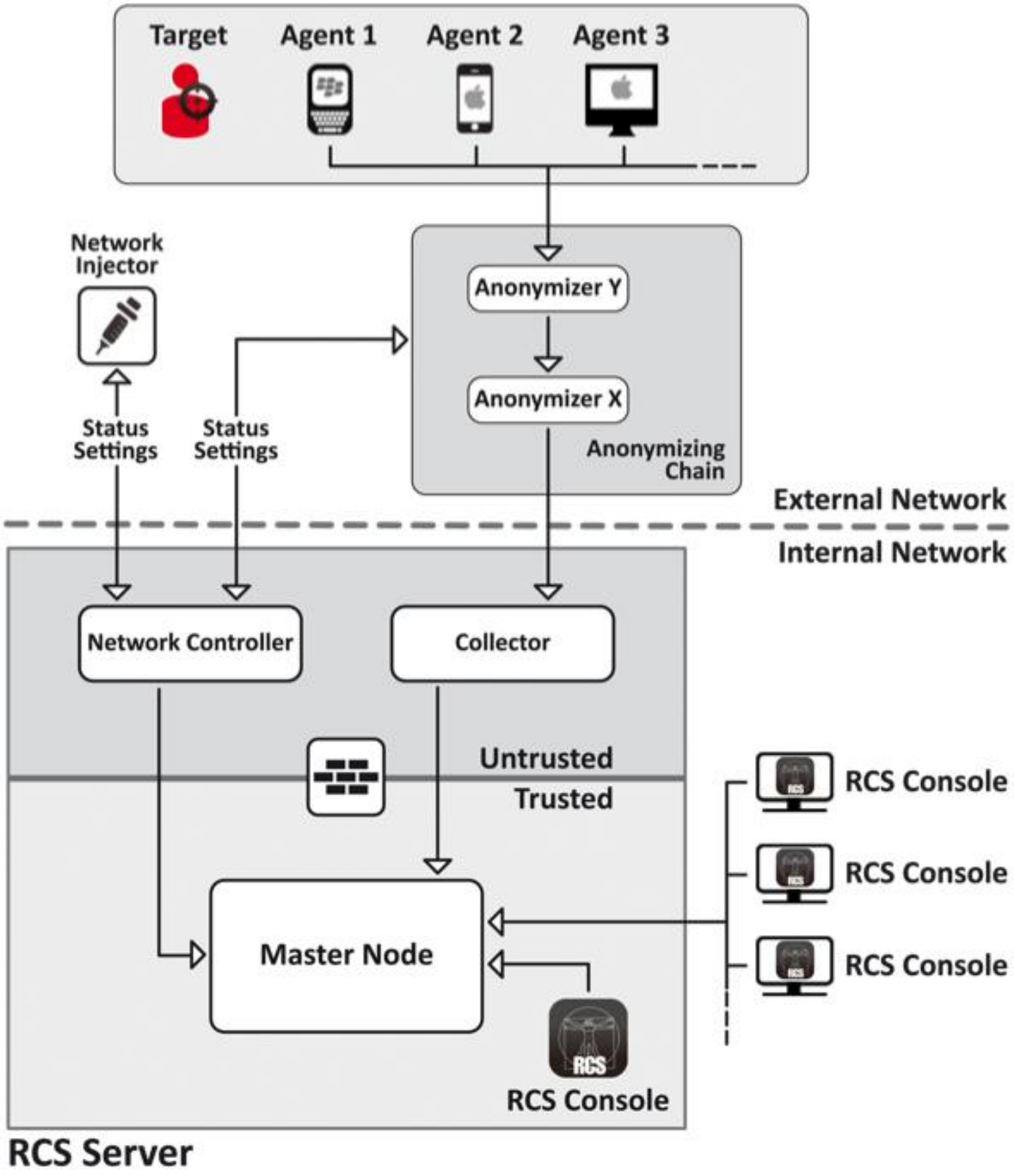


Image Source: "Hacking Team, RCS 9: The hacking suite for governmental interception, System Administrator's Guide," 2013.

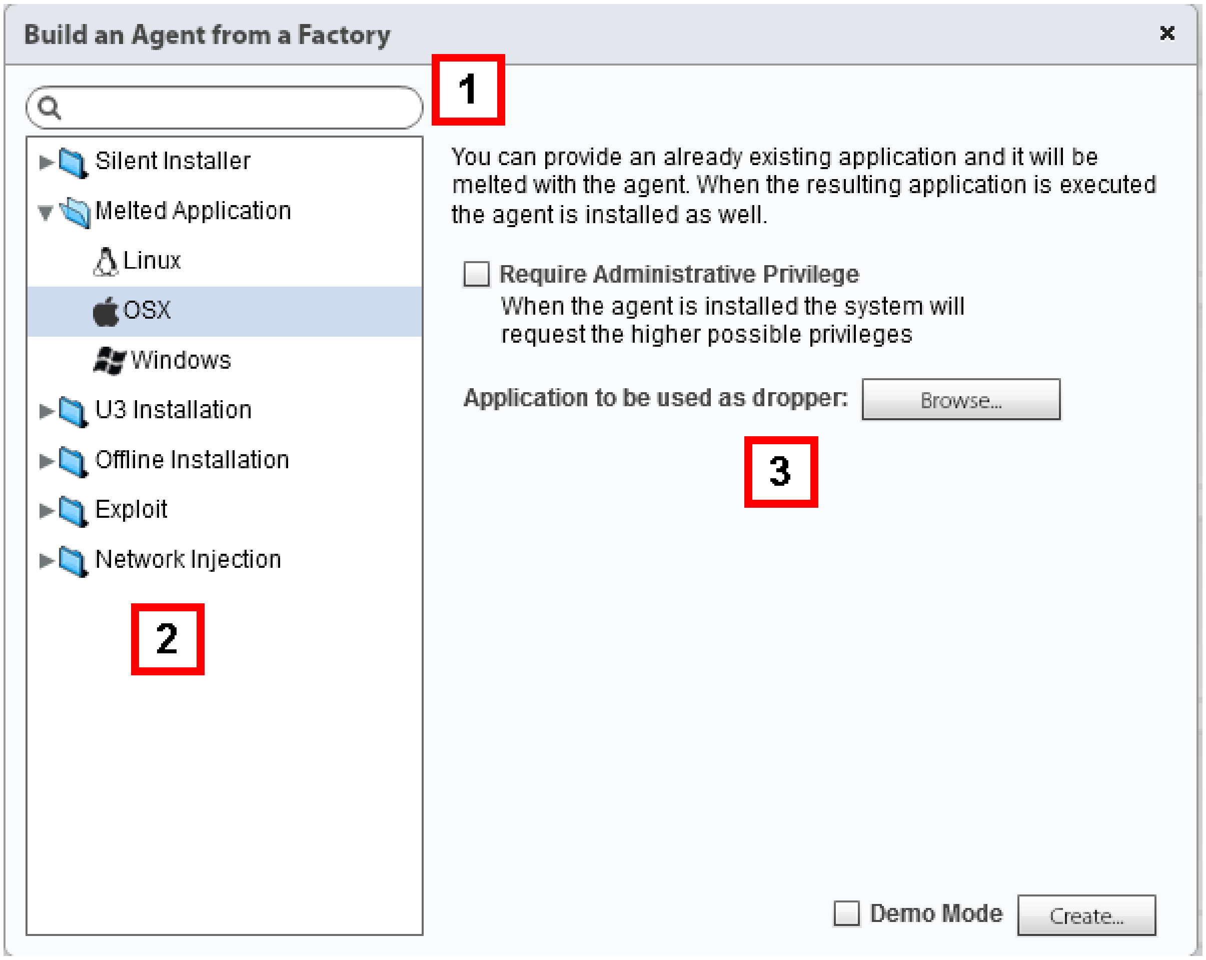
Players in System Operation

According to the documents, RCS has a series of defined roles, each with their own responsibilities and permissions on the system. In brief:

- A **System Administrator** appears to receive training from Hacking Team during the “Contract Phase”, as well as other system administration tasks, including installing and updating RCS server networks as well as network injectors.
- An **Administrator** creates accounts, and creates both operations and designates targets.
- A **Technician** is responsible for creating implants (named “agents” in the documentation) and managing network injectors.
- **Analysts** handle outputs from the system and perform intelligence analysis via the RCS console.

Developing and Deploying Implants

According to the documents, the software can create a full range of implants through a “factory” which is compiled specifically for an investigation. The **Technician** is responsible for creating these agents using the “factory.” Here is the prompt presented to the Technician:



Build an Agent from a Factory

You can provide an already existing application and it will be melted with the agent. When the resulting application is executed the agent is installed as well.

□ Require Administrative Privilege
When the agent is installed the system will request the higher possible privileges

Application to be used as dropper: Browse...

□ Demo Mode

In this image, the technician is preparing a Mac OSX implant. Image Source: “Hacking Team, RCS 9: The hacking suite for governmental interception,Technician’s Guide,” 2013.

The technician has a range of options from the Factory as well as the additional ability to use Network Injection (not pictured). These options include:

- **Network Injection:** via injected malicious traffic in cooperation with an ISP
- **Tactical Network Injection**: on LAN or WiFi

- **Melted Application**: bundling a Hacking Team dropper alongside a bait application
- **Installation Package**: a mobile installer
- **Exploit**: document-based exploit for mobile and desktop
- **Local Installation**: mobile installation via USB or SD card
- **Offline Installation:** create an ISO for a bootable SDHC, CD, or USB. This option includes the ability to infected hibernated and powered off devices
- **QR Code:** a mobile link that, when pictured, will infect the target
- **Applet Web:** likely a malicious website (depreciated after v. 8.4)
- **Silent Installer:** a desktop executable that will install the implant
- **Infected U3 USB**: an auto-infecting U3 USB
- **WAP Push Message**: the target will be infected if the user accepts the message (works on all mobile operating systems apart from iOS)

Infection: “Scout Agents” and “Agents”

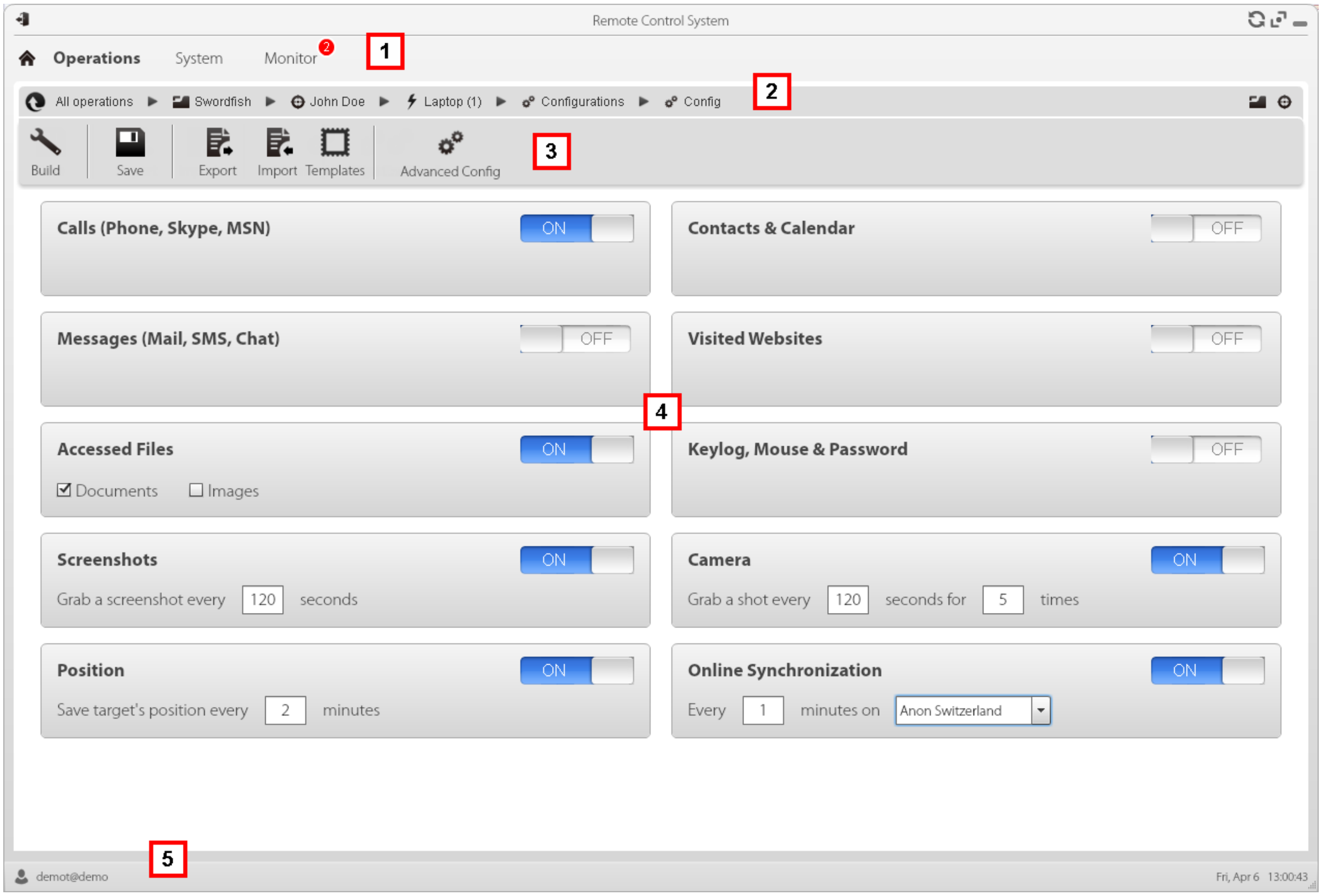
The documents indicate that RCS makes use of a “scout agent,” commonly known as a “validator” that helps to determine whether a full implant (“agent”) should be installed. This thin implant collects screenshots and device information to determine whether the target is of interest.

The documents suggest that Hacking Team is explicitly concerned with the possibility that their implant could become unmasked, and instructs technicians that a function of “scout agents” is to determine whether the system is ‘safe’ to infect, or whether it could risk unmasking the agent.

When the system is determined to be safe to infect, the “scout agent” is replaced by the full implant. According to the documents, Hacking Team advises users to gradually add functionality to implants. These implants exfiltrate data and receive instructions through a process called “synchronizing.”

Implant Modules and Functionality

The implant (“agent”) offers one-click functionality for requesting information from target devices. Technicians are encouraged to add functionality as needed.



Calls (Phone, Skype, MSN) ON

Contacts & Calendar OFF

Messages (Mail, SMS, Chat) OFF

Visited Websites OFF

Accessed Files ON

☑ Documents ☐ Images

Keylog, Mouse & Password OFF

Screenshots ON

Grab a screenshot every 120 seconds

Camera ON

Grab a shot every 120 seconds for 5 times

Position ON

Save target's position every 2 minutes

Online Synchronization ON

Every 1 minutes on Anon Switzerland

Image Source: “Hacking Team, RCS 9: The hacking suite for governmental interception,Technician’s Guide,” 2013.

In addition, a more advanced approach can be taken, allowing a sophisticated technician to determine a specific sequence of module activation upon infection, using a graphical flow model. This allows the user to define events that trigger particular actions, sub-actions, modules, and sequences.

The documents provide an example of such a sequence:

Event: Device is connected to power adapter
Sub Action: Send SMS, begin logging position, disable an event keyed to SIM card changes.

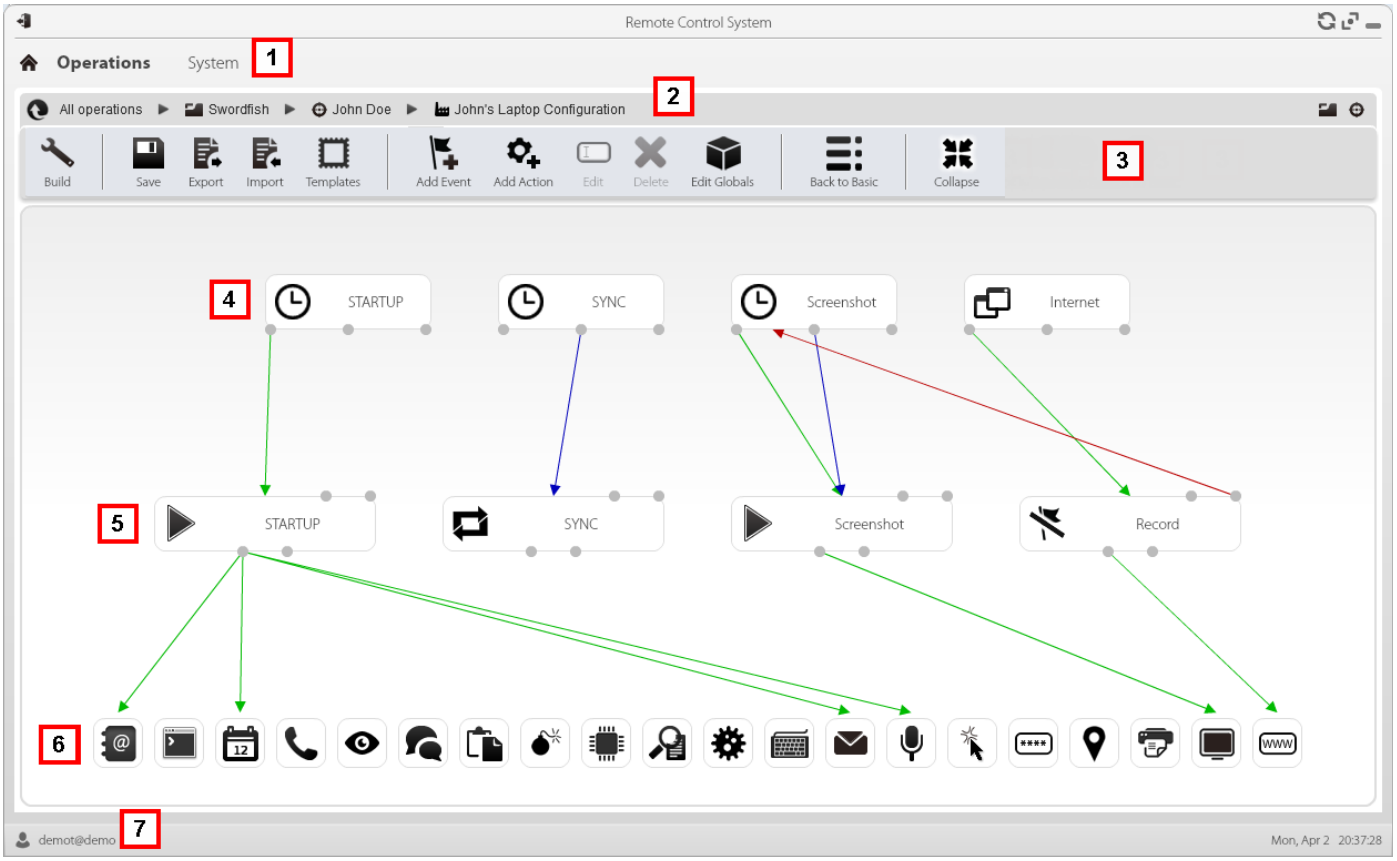


Image Source: “Hacking Team, RCS 9: The hacking suite for governmental interception,Technician’s Guide,” 2013.

Selection of available surveillance modules

- **Accessed files**
- **Address Book**
- **Applications used**
- **Calendar**
- **Contacts**
- **Device Type**
- **Files Accessed**
- **Keylogging**
- **Saved Passwords**
- **Mouse Activity** (intended to defeat virtual keyboards)
- **Record Calls and call data**
- **Screenshots**
- **Take Photographs with webcam**
- **Record Chats**
- **Copy Clipboard**
- **Record Audio from Microphone**
    - With additional Voice and silence detection to conserve space
- **Realtime audio surveillance** (“live mic:” module is only available for Windows Mobile)
- **Device Position**
- **URLs Visited**

- **Create conference calls** (with a silent 3rd party)
- **Infect other devices** (depreciated since v. 8.4)

Other Capabilities

Once an implant is operational its collection operations can be updated. In addition files can be sent to and received from the device.

In addition, implants have a default cap on “evidence” space of 1GB on the target device. Recording of new material stops when the space is reached. Operators also have the ability to delete not-yet-transmitted data on the device.

Synchronizing & Data Exfiltration

The synchronizing process has a number of steps, paraphrased here:

- Authorization between implant and server
- Time syncing between implant and RCS server
- Implant removal (if case specified as “closed” by technician)
- Implant configuration update
- Implant downloads material sent from the RCS server
- Upload to RCS server of “download” cue material
- Upload to RCS server of evidence, combined with secure delete
- Additional secure deletion of evidence

Mobile and desktop versions have slightly different parameters. While desktop implants can have bandwidth and delays in sending materials added, the mobile settings have some unique features.

Technicians can force the type of data channel (WiFi or Cell Network) that the implant uses to update. Interestingly, **technicians can specify login credentials for the APN used to exfiltrate data**. This allows the implant to avoid incurring data charges and displaying traffic to the victim. The feature is specified as only available on BlackBerry and Symbian OS in Version 9 of RCS.

Anti-Analysis Functionality & Anti Forensics

A functionality referred to as “crisis” allows for actions on detection of “hostile” activity (the documents give the example of a packet sniffer). This functionality can be triggered by a range of scenarios and have a number of options based on identifying processes. In desktop versions, these options can including pausing synchronizing, and not hooking programs. For mobile versions these options includes pausing audio, camera, location collecting, and synchronizing.

A **wipe** can also be triggered, and according to the documents, Hacking Team informs users that it will leave “no trace” of the implant. The **uninstall** action has several features that could be of interest in forensic analysis of potentially infected devices: (i) uninstall on BlackBerry triggers an automatic restart; (ii) uninstall on Android devices where root was not successfully gained results in a user prompt requesting permission to uninstall; (iii) on Windows Phone, uninstall deletes all files but does not remove the Application Icon from the list of programs.

Code Signing and Certificates

RCS is designed to incorporate code signing when creating implants. The documents helpfully suggest **Verisign, Thawte and GoDaddy** as sources of code signing certificates. In addition, the documents encourage users to contact Hacking Team technical support for assistance in obtaining a certificate.

Interesingly, for Symbian, users are instructed to purchase a “Developer Certificate” directly from TrustCenter and even provides the URL (https://www.trustcenter.de/en/products/tc_ publisher_id_for_symbian.htm).

For infecting Windows Phones, users are encouraged to register a Microsoft account (signup.live.com) and a Windows Phone Dev Center account (https://dev.windowsphone.com/en-us/join/).

The documents indicate that Hacking Team is concerned with ensuring users correctly manage the approval process (managed by Symantec) and instructs users to promptly reply to phone and e-mail communications from Symantec. Users also get instsructions in how to obtain an Enterprise Mobile Code Signing Certificate (https://products.websecurity.symantec.com/orders/enrollment/microsoftCert.do)

Android Specific Issues

In order to run some of its modules (Chat, Messages, Screenshot), the Android implant requires root privileges on the device. In some cases, where acquisition of root privileges is unsuccessful, the victim can be manually asked to give the application root access. In addition, in cases where root has not been successfully acquired on an Android device, the victim sees a prompt requesting permission if an uninstall has been triggered.

Hacking Team helpfully notes that the default APK file implant appears as a normal application named “DeviceInfo” that displays device information.

Anonymizer & Proxy Chain Architecture

This image highlights how a system administrator sees a distributed proxy chain (see: Appendix: Distributed Hacking Team RCS Architecture) and collector architecture.

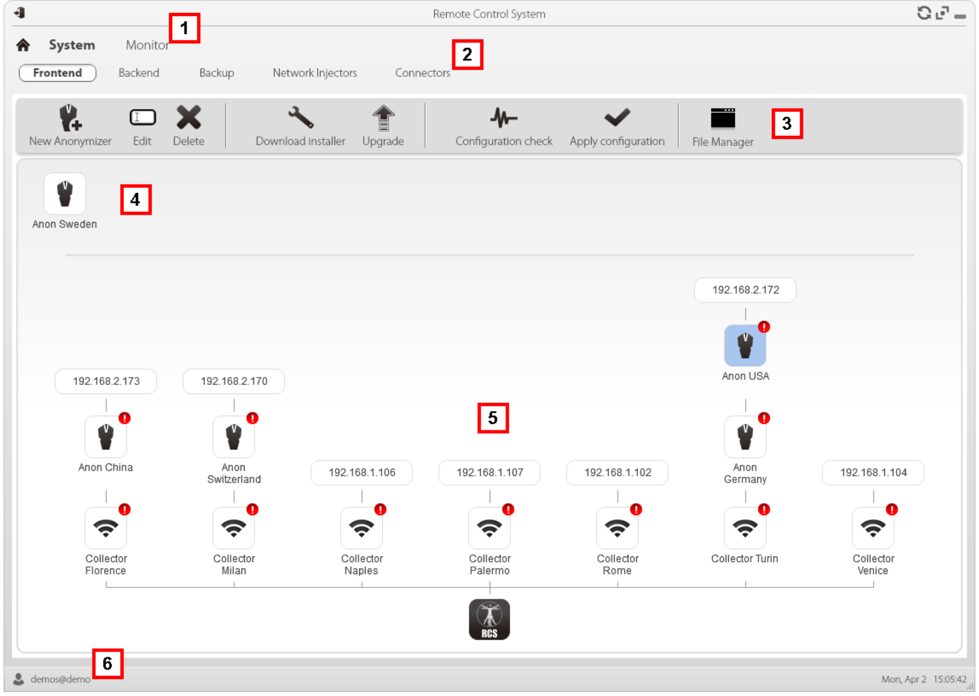


Image Source: “Hacking Team, RCS 9: The hacking suite for governmental interception, System Administrator’s Guide” 2013

Licensing, Updates

At the time of creation of a collection architecture, a license file from Hacking Team is required by the software, according to the documents. Other licenses are required for for specific functionality, as well as the more developed analysis toolkits.

The documents suggest that Hacking Team maintains an update cycle for its products. The updating process is performed by a System Administrator using updates provided by Hacking Team and includes updates both to the collection and injector infrastructure, and to implants.

Auditing

The program does appear to have a basic logging function and auditing capability, allowing authorized administrators to view logs of user actions throughout the system, including interactions with implants. This capability appears to have been developed to make the toolkit more compatible with evidence requirements for with digital material. We note however, the ease with which evidence, logs, implants, and “factories” can be irreversibly deleted by authorized users.

Analysts Eye View

This image graphically shows the ways in which multiple devices of a single target may be compromised and simultaneously monitored.

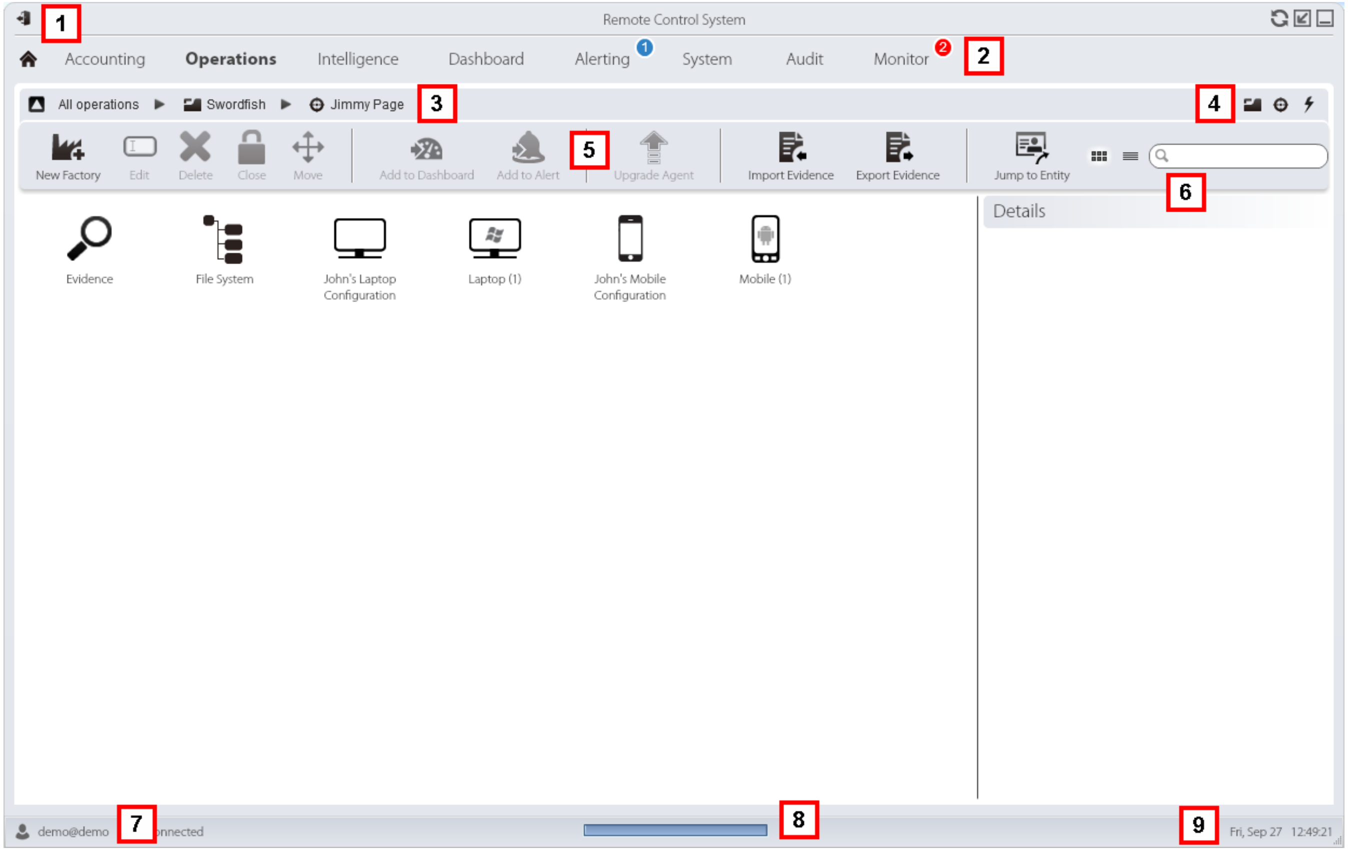
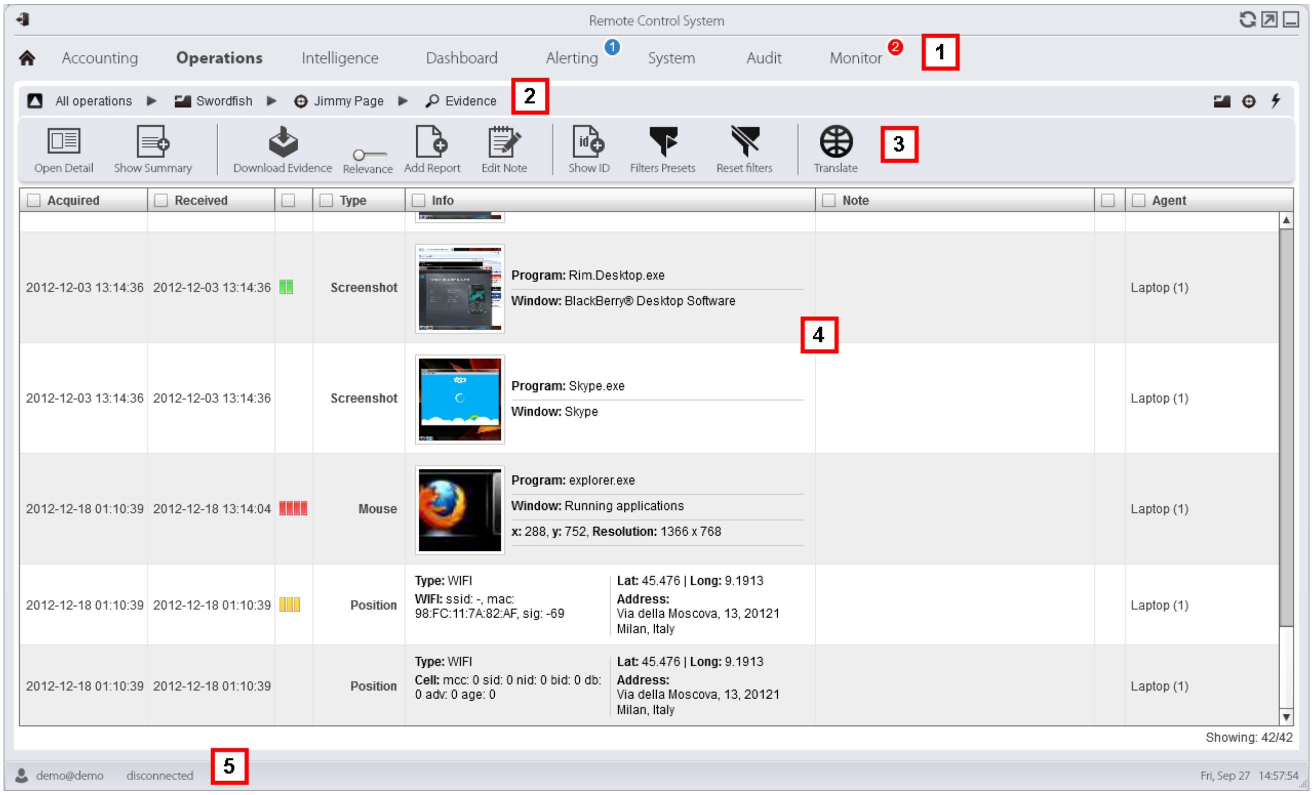


Image Source: "Hacking Team, RCS 9: The hacking suite for governmental interception, Analysts Guide," 2013

Analysts can perform searches on data using text strings as well as filtering and sorted data. In addition, Analysts can flag data for importance, as well as apply automated rules for flagging data.

Data storage makes use of the open source MongoDB, and documentation indicates that Hacking Team RCS uses MongoDB 2.4.8.

Next, also from a demo instance, we see the information exfiltrated from a target computer. Of note is the use of a basic color coded flagging system (red, green, yellow) to flag evidence, as well as the kind of information available to the analyst from each capture. Here we see screenshots of particular program operations (e.g. running BlackBerry Desktop software, and Skype) as well as location information. Addresses in the location information are identical to the publicly available address of Hacking Team.

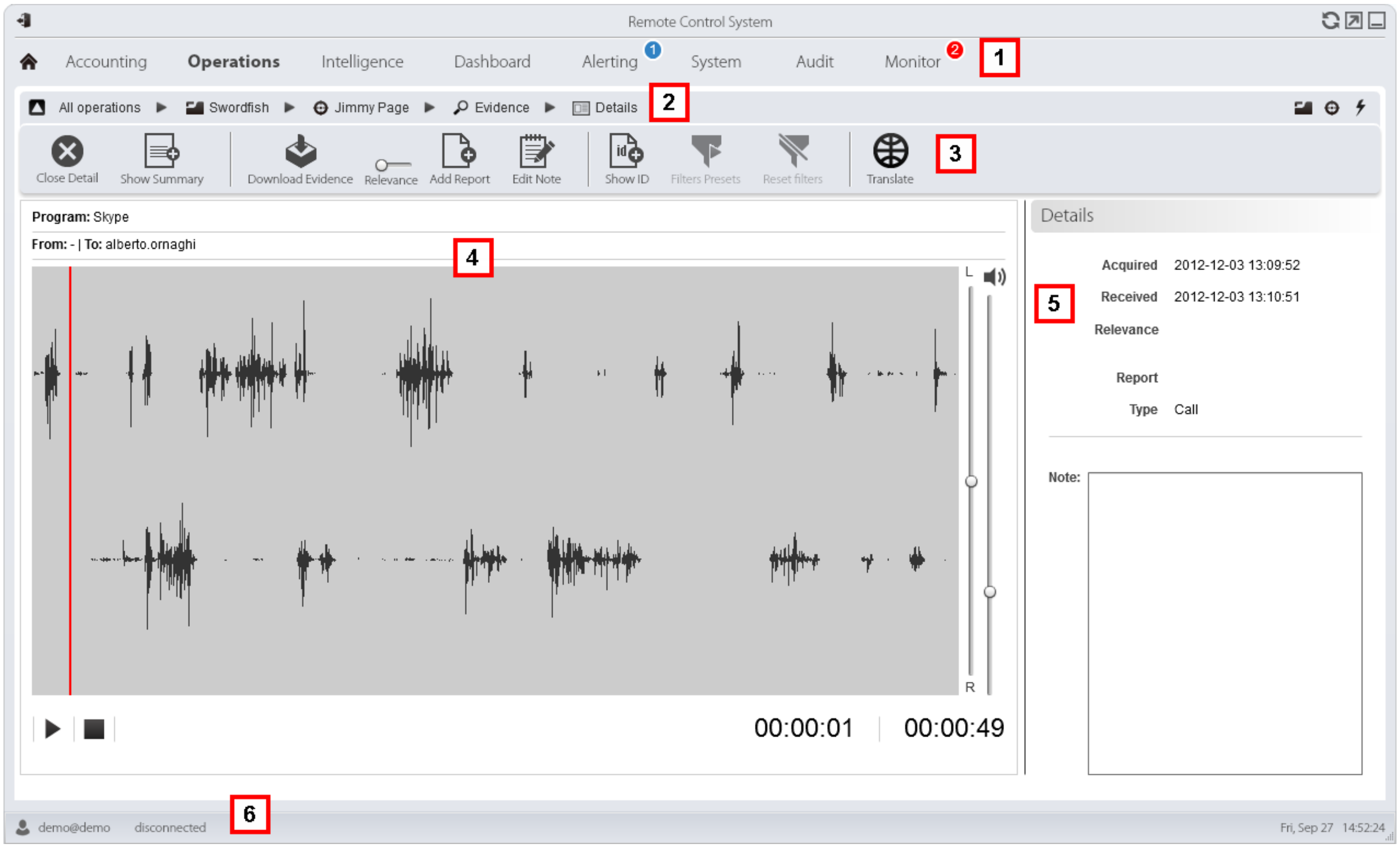


□ Acquired	□ Received	□	□ Type	□ Info	□ Note	□	□ Agent
2012-12-03 13:14:36	2012-12-03 13:14:36	🟩🟩	Screenshot	**Program:** Rim.Desktop.exe **Window:** BlackBerry® Desktop Software			Laptop (1)
2012-12-03 13:14:36	2012-12-03 13:14:36		Screenshot	**Program:** Skype.exe **Window:** Skype			Laptop (1)
2012-12-18 01:10:39	2012-12-18 13:14:04	🟥🟥🟥🟥	Mouse	**Program:** explorer.exe **Window:** Running applications **x:** 288, **y:** 752, **Resolution:** 1366 x 768			Laptop (1)
2012-12-18 01:10:39	2012-12-18 01:10:39	🟨🟨🟨	Position	**Type:** WIFI **WIFI:** ssid: -, mac: 98:FC:11:7A:82:AF, sig: -69 **Lat:** 45.476 | **Long:** 9.1913 **Address:** Via della Moscova, 13, 20121 Milan, Italy			Laptop (1)
2012-12-18 01:10:39	2012-12-18 01:10:39		Position	**Type:** WIFI **Cell:** mcc: 0 sid: 0 nid: 0 bid: 0 db: 0 adv: 0 age: 0 **Lat:** 45.476 | **Long:** 9.1913 **Address:** Via della Moscova, 13, 20121 Milan, Italy			Laptop (1)

Showing: 42/42

Image Source: “Hacking Team, RCS 9: The hacking suite for governmental interception, Analysts Guide,” 2013.

In the following image we see the analyst console playing back an intercepted Skype call.



Program: Skype

From: - | To: alberto.ornaghi

00:00:01 | 00:00:49

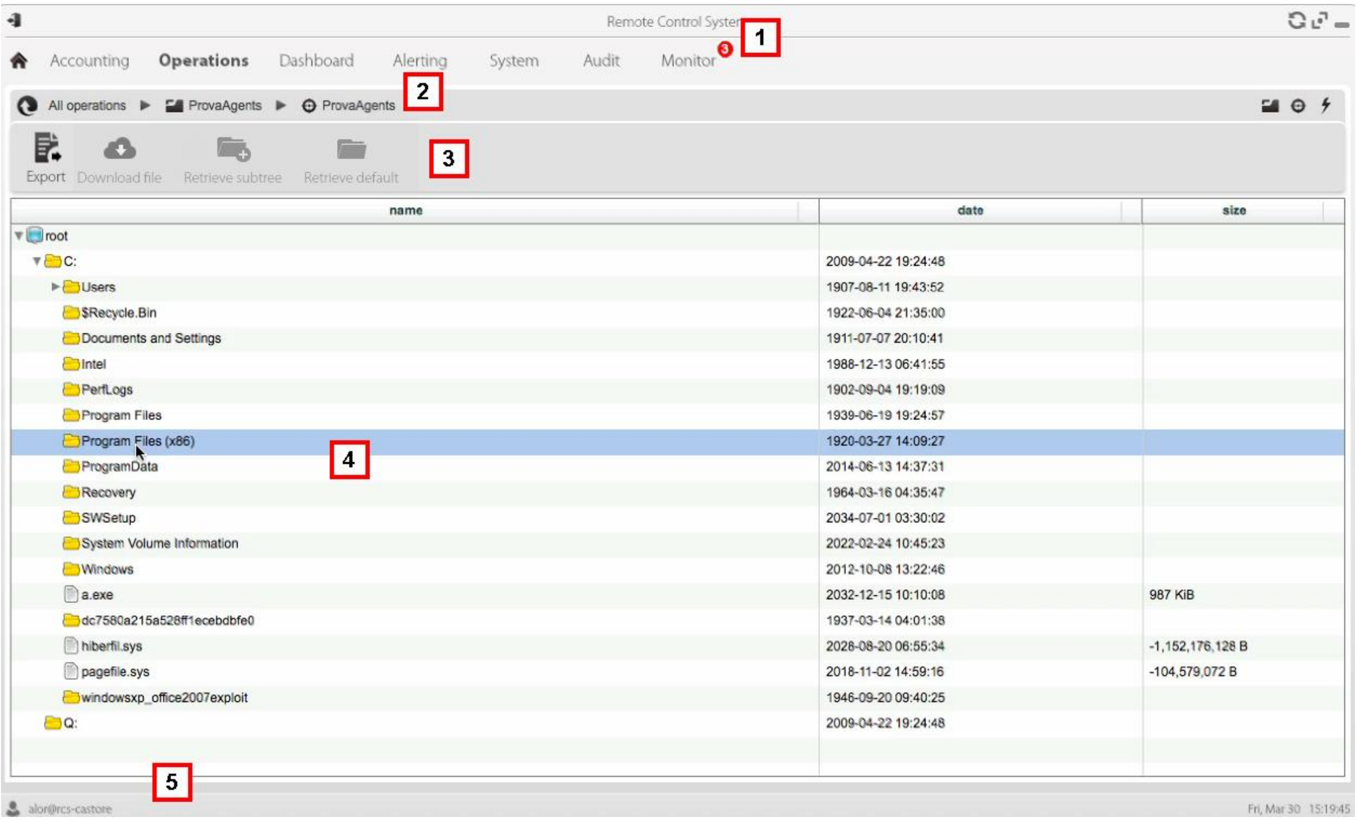
Details

Acquired	2012-12-03 13:09:52
Received	2012-12-03 13:10:51
Relevance	
Report	
Type	Call

Note:

Image Source: “Hacking Team, RCS 9: The hacking suite for governmental interception, Analysts Guide,” 2013.

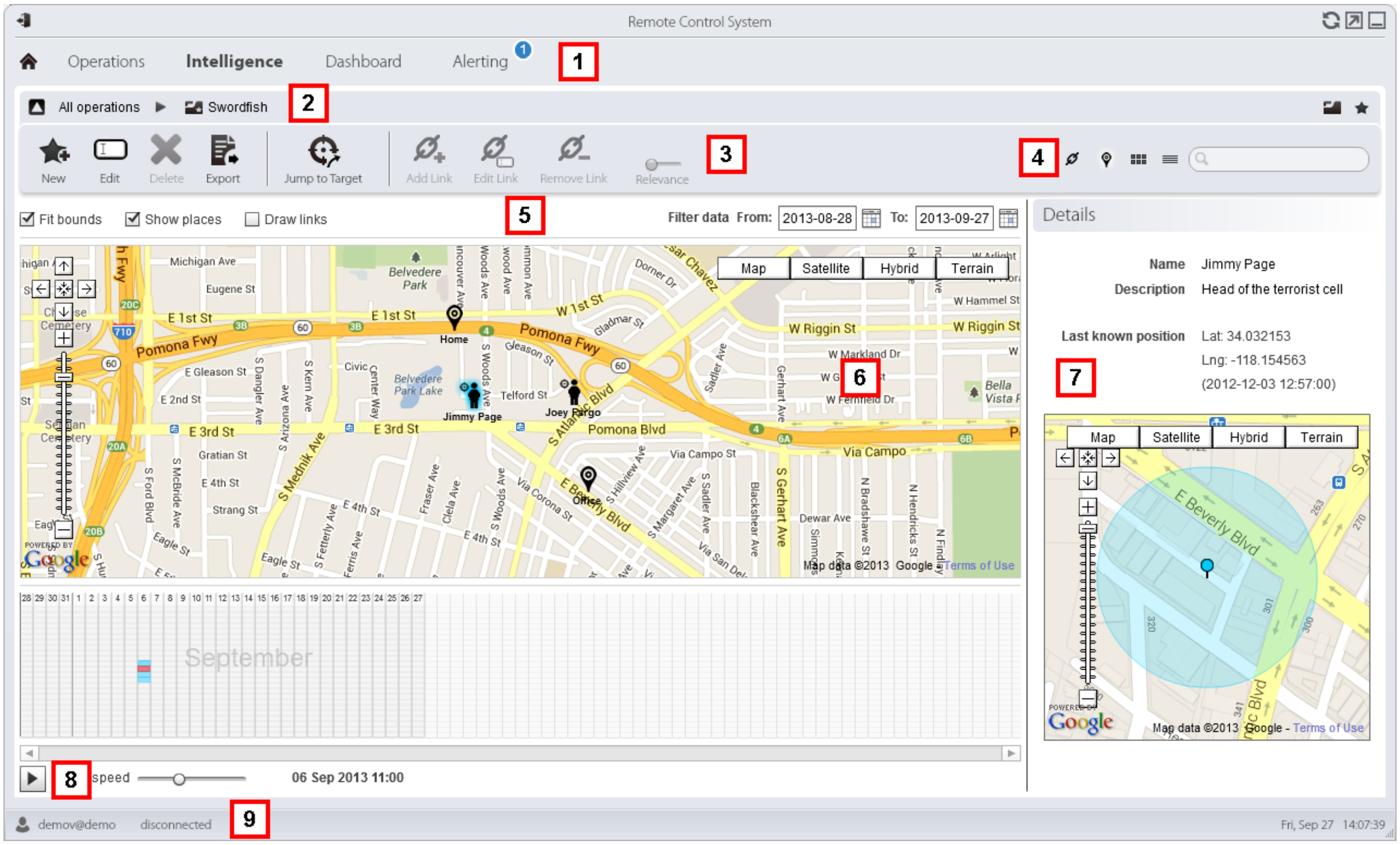
In the following image, we see live browsing of an infected system’s file tree. This is a functionality that can be enabled in certain cases.



name	date	size
root		
C:	2009-04-22 19:24:48	
Users	1907-08-11 19:43:52	
$Recycle.Bin	1922-06-04 21:35:00	
Documents and Settings	1911-07-07 20:10:41	
Intel	1988-12-13 06:41:55	
PerfLogs	1902-09-04 19:19:09	
Program Files	1939-06-19 19:24:57	
Program Files (x86)	1920-03-27 14:09:27	
ProgramData	2014-06-13 14:37:31	
Recovery	1964-03-16 04:35:47	
SWSetup	2034-07-01 03:30:02	
System Volume Information	2022-02-24 10:45:23	
Windows	2012-10-08 13:22:46	
a.exe	2032-12-15 10:10:08	987 KiB
dc7580a215a528ff1ecebdbfe0	1937-03-14 04:01:38	
hiberfil.sys	2028-08-20 06:55:34	-1,152,176,128 B
pagefile.sys	2018-11-02 14:59:16	-104,579,072 B
windowsxp_office2007exploit	1946-09-20 09:40:25	
Q:	2009-04-22 19:24:48	

Image Source: "Hacking Team, RCS 9: The hacking suite for governmental interception, Analysts Guide," 2013.

The image below shows a map and timeline view of the position of infected devices. Interestingly, in this image, used as an instructional illustration, it appears that Hacking Team **might have inadvertently revealed information about a demonstration to a US Law Enforcement to its other customers**. Specifically, the location of the "Jimmy Page" device on this map is in the access-controlled parking lot of the LA County Sheriff (and appears to be stamped with the date Sept 6 2013 *or* December 3 2012). It may be coincidental that Hacking Team was extensively represented at the ISS World held later in September 2013 in the US.



☑ Fit bounds ☑ Show places □ Draw links

Filter data From: 2013-08-28 To: 2013-09-27

Details

Name Jimmy Page
Description Head of the terrorist cell

Last known position Lat: 34.032153
Lng: -118.154563
(2012-12-03 12:57:00)

06 Sep 2013 11:00

demov@demo disconnected

Fri, Sep 27 14:07:39

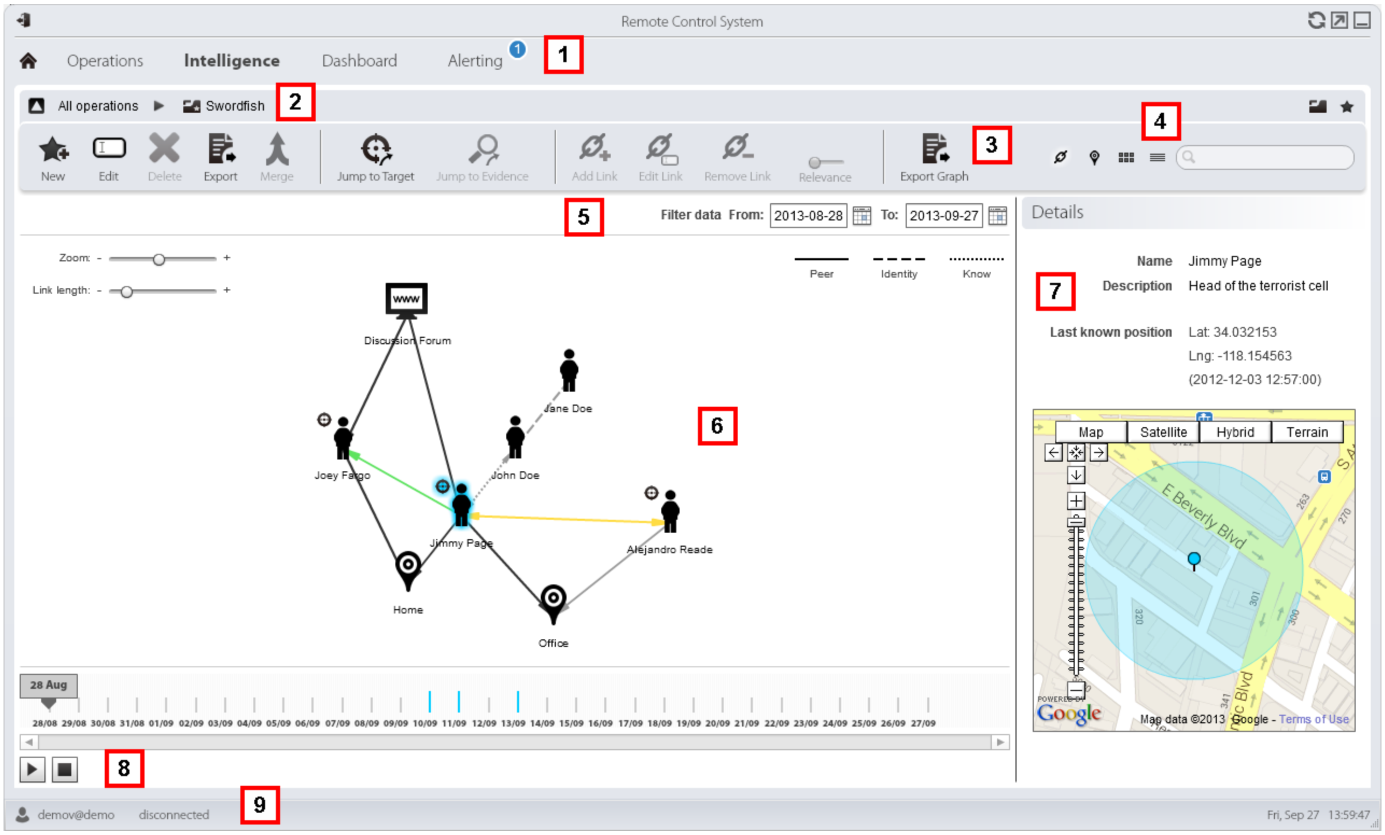
Image Source: “Hacking Team, RCS 9: The hacking suite for governmental interception, Analysts Guide,” 2013.

The use of Google maps in the tool provides a graphic view of the targets; however it may present a serious breach of operational security for clients, including betraying location data for ongoing investigations. Even if some of the point data is not transmitted, Google needs to know the map centroid (and hence target location) to deliver some of this map data.

It is possible that Hacking Team RCS is exposing highly sensitive investigation data of government clients to Google as they are making use of the Google Maps API to display this map.

In addition, Google Maps traffic, even using SSL, has been subject to successful traffic analysis by third parties in the past. While we have not demonstrated this vulnerability, we believe this possibility warrants further investigation.

Beyond collection of data, the documents indicate that Hacking Team is offering a basic investigation management toolkit including basic link analysis and other features with the purchase of an additional license. The software can build *very* basic link analysis using “Peer” (contact between entities like calls) and “Know” (one or both individuals found in others phone book) links. The system also automatically attempts to create identity links when details are shared between entities (e.g. a phone number). Time-series and locational data can also be displayed.



Filter data From: 2013-08-28 To: 2013-09-27

Name Jimmy Page
Description Head of the terrorist cell
Last known position Lat: 34.032153
Lng: -118.154563
(2012-12-03 12:57:00)

demov@demo disconnected
Fri, Sep 27 13:59:47

Image Source: “Hacking Team, RCS 9: The hacking suite for governmental interception, Analysts Guide,” 2013

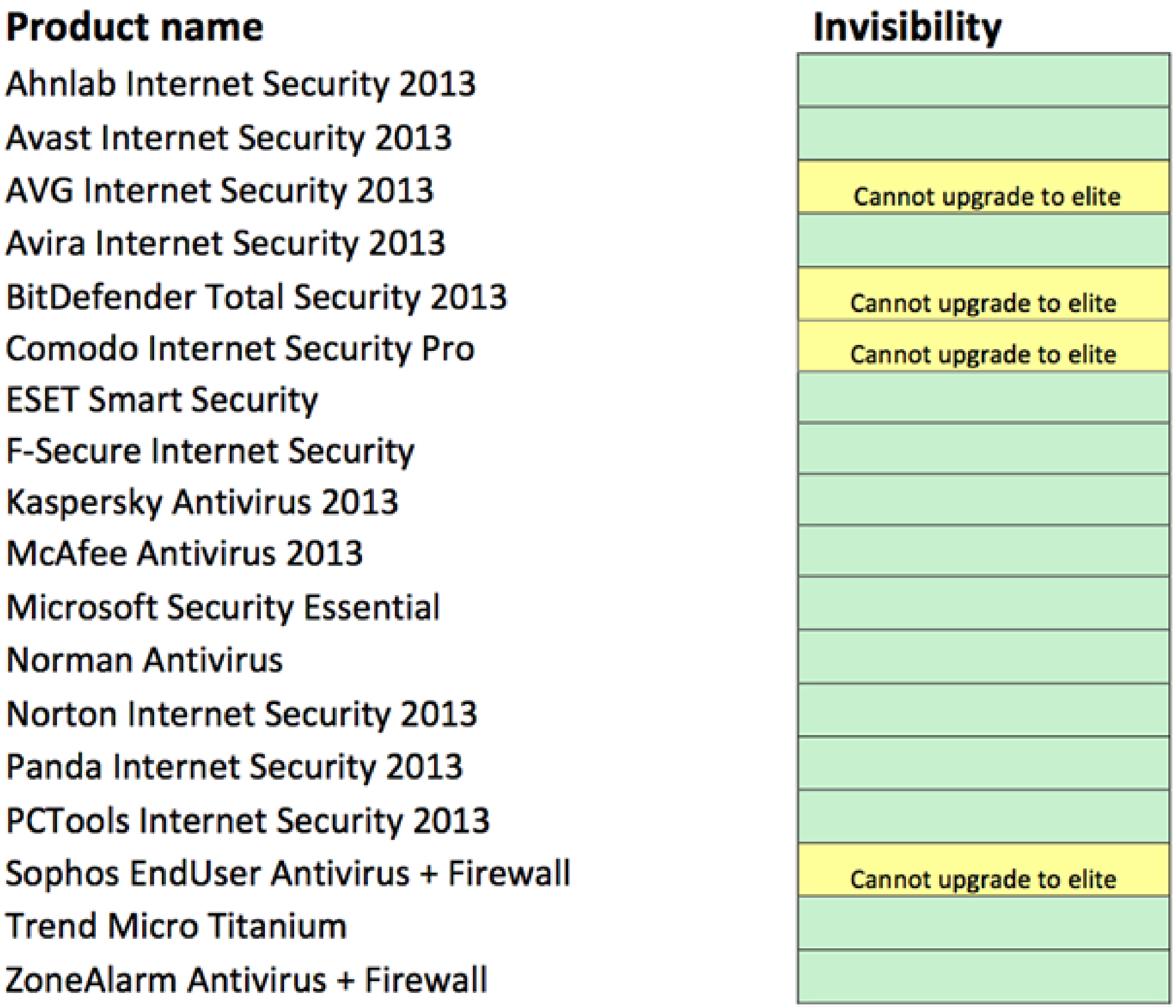
Evolving Terminology

The documents suggest that Hacking Team has made a number of changes in the language it uses to describe its tools. We found for example that in v7.6 and below they used the term “Backdoor” to refer to their implant, but changed the term in v 8.0 and above to “Agent.” Similarly, they changed the term “Backdoor Class” to “Factory” in the same time period.

“Invisibility”

According to the documents, Hacking Team appears to supply customers with an “Invisibility Report” for its RCS solultion. For example, we have reviewed the “Version 9.0 – Invisibility Report” for the Silent Installer, Melted application, Network Injector INJECT-EXE attack, and Offline CD.”

The document informs customers: Tests were performed on a default 64-bit Windows 7 installation.



Product name	Invisibility
Ahnlab Internet Security 2013	🟩
Avast Internet Security 2013	🟩
AVG Internet Security 2013	Cannot upgrade to elite
Avira Internet Security 2013	🟩
BitDefender Total Security 2013	Cannot upgrade to elite
Comodo Internet Security Pro	Cannot upgrade to elite
ESET Smart Security	🟩
F-Secure Internet Security	🟩
Kaspersky Antivirus 2013	🟩
McAfee Antivirus 2013	🟩
Microsoft Security Essential	🟩
Norman Antivirus	🟩
Norton Internet Security 2013	🟩
Panda Internet Security 2013	🟩
PCTools Internet Security 2013	🟩
Sophos EndUser Antivirus + Firewall	Cannot upgrade to elite
Trend Micro Titanium	🟩
ZoneAlarm Antivirus + Firewall	🟩

Image Source: “Version 9.0 – Invisibility Report,” undated.

CONCLUSION

Hacking Team’s Remote Control System is a surveillance malware toolkit marketed for “lawful interception” use. We identified and analyzed RCS Android implants that had lures with a political subtext suggesting targets in the Qatif Governorate of Saudi Arabia. Analysis of these implants revealed a range of surveillance functions.

In the past several years, we have researched both Gamma Group and Hacking Team, exposing their global proliferation, and highlighting the technologies and tactics that they use. Our interest in these groups is **not because they create the most sophisticated implants**. Indeed, since the Aurora incident in 2009, there have been many public reports of sophisticated malware used by nation states. For example, Turla, a campaign attributed to Russia, was described as “one of the most complex cyber espionage programs uncovered to date.” Additional campaigns of note include Careto, linked to Spain [PDF], and Miniduke. Meanwhile, significant analysis and public discussion has gone into the Stuxnet, Duqu, and Flame, which are allegedly a product of US and Israeli collaboration.

These high-impact (and typically high development cost) implant kits all appear to be used exclusively by the countries they are attributed to, and are designed to perform targeted intrusions.

Hacking Team and Gamma Group are different for several reasons. First, their software is available to all but a few countries, provided they pay. Second, their software is marketed to target everyday criminality and "security threats," whereas the state-sponsored campaigns we outlined above are designed to support espionage operations against hardened, high-value targets.

This type of exceptionally invasive toolkit, once a costly boutique capability deployed by intelligence communities and militaries, is now available to all but a handful of governments. An unstated assumption is that customers that can pay for these tools will use them correctly, and primarily for strictly overseen, legal purposes. As our research has shown, however, by dramatically lowering the entry cost on invasive and hard-to-trace monitoring, the equipment lowers the cost of targeting *political threats* for those with access to Hacking Team and Gamma Group toolkits.

ACKNOWLEDGEMENTS

Seth Hardy, Sarah McKune, Marcia Hoffman, Sebastian Porst, Masashi Crete-Nishihata, Bill Marczak, Ron Deibert, Human Rights Watch, Abeer Allam, and several journalists and researchers who were very generous with their time.

APPENDIX A: DISTRIBUTED HACKING TEAM RCS ARCHITECTURE

This image shows a distributed Hacking Team RCS collection infrastructure, according to the documents.

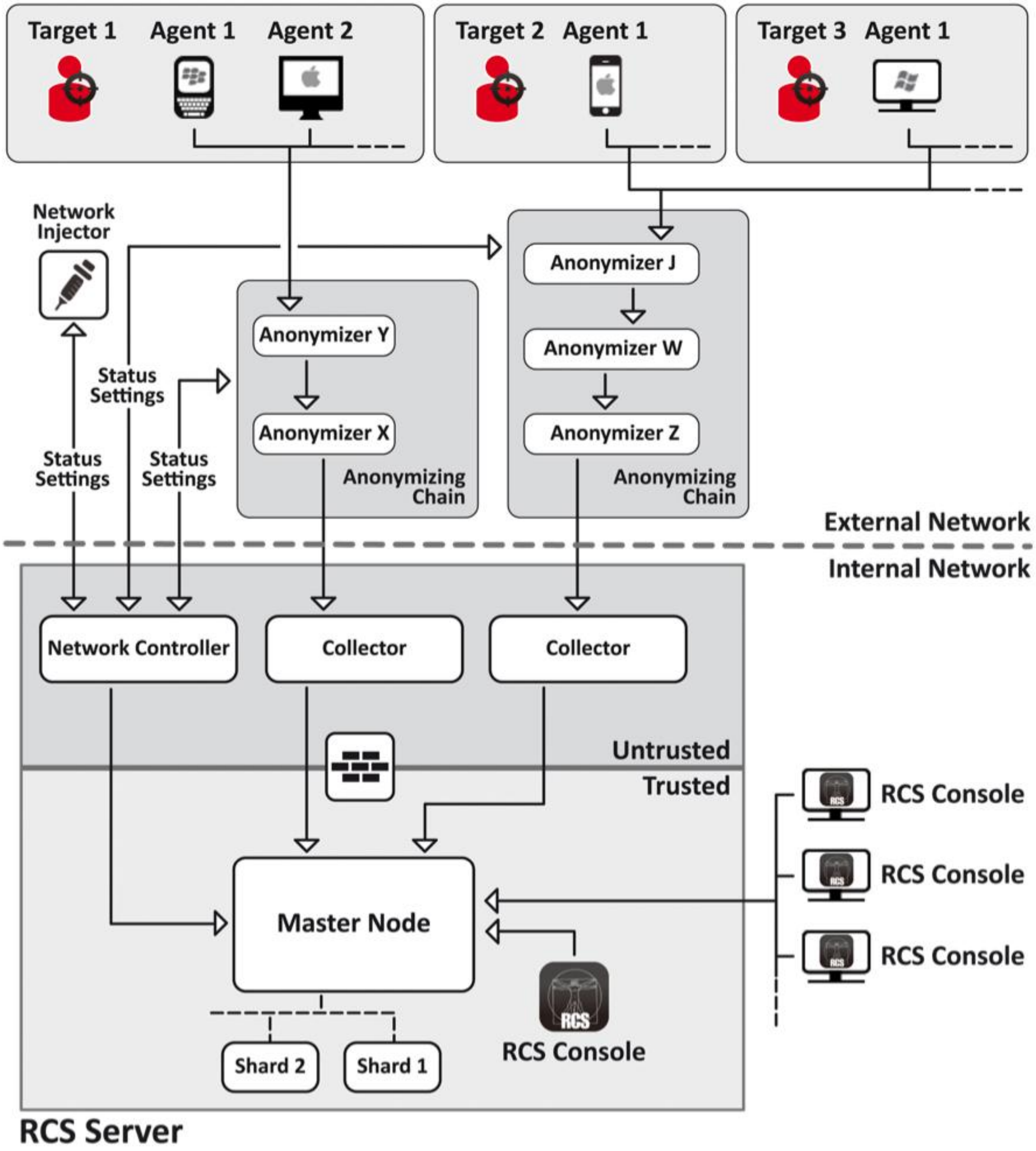


Image Source: "Hacking Team, RCS 9: The hacking suite for governmental interception, System Administrator's Guide," 2013

APPENDIX B: LIST OF 'RILCAP' COMMANDS

Commands provided by:

/system/bin/rilcap 6250af9750606a4a06ca1ec0bfa127d0e37e1c7676e37773f461a91bfe0daf93

volkill the volume daemon (vold)

blr

mount /system as read-only

blw

mount /system as read-write

rt

clone self to /system/bin/rilcap with permissions 04755

qzx [command] [args]

passes argument to system()

fhc [source] [destination]

copy file

pzm [permission flags] [file]

modify file permissions

qzs

read commands from file named qzs and execute them (execv /system/bin/sh …)

reb

reboot

adm [name]

adds [name] to device-policies.xml

ru

unlink self

fho [user] [user] [filename]

modify file owner

sd

check for and mount sd card (but it has path hardcoded as /mnt/sdcard, so won't always work)

fb [file]

copies from frame buffer to file (for screenshots)

FOOTNOTES

[1] http://developer.android.com/reference/android/Manifest.permission.html

[2] https://developer.android.com/about/dashboards/index.html