Dissecting the Hacking Team’s Operation Methods: What Security Professionals Need to Know

By: Ian Muller, Senior Security Writer
Amit Serper, Senior Security Researcher
Alex Frazer, Security Researcher
The Cybereason philosophy is that the attack methodology matters much more than the exploits and tools that the hackers leverage, which is why we focus so much on malicious operations, or Malops. Exploits will be patched and tools will evolve and change, but attack methods and hacker behavior are more likely to remain the same over time. By analyzing this angle of a cyber attack, we are able to better recognize malicious behavior and react faster to a threat.

The main idea behind this approach is that being able to identify malicious activity sooner will give you a leg up on an attacker. A zero day exploit by itself is a threat, but it’s only a method for the attackers to gain access to your system. Once they’re inside, the exploit becomes unimportant. By assessing the behavior and activity, rather than the file signatures and hashes, we can recognize the malicious operation before the attackers have enough time to start exfiltrating data.

One example of this is one of the zero day exploits released in the recent Hacking Team data leak. Because of our focus on behavior, we were able to instantly identify the privilege escalation activity within our lab when testing it against our platform, without changing anything in our own system.
“The exploits themselves, while dangerous, aren't the most interesting thing here,” says Amit Serper, Senior Security Researcher. “Any antivirus can detect using signatures, the signatures always come after the damage had happened. It’s a game of cat and mouse out there. New exploit - patch; new exploit - patch; repeat. We at Cybereason actually made a paradigm shift long ago. That is why the company was founded and that is why we are able to catch zero days so quickly.”

Penetration is inevitable, and it doesn’t matter which drivers or applications are vulnerable to an exploit. If you are able to detect abnormal activity in your environment and react to it as it occurs, you’ll be able to take a proactive stance against cyber attacks and stop hackers in their tracks.
In order to fully understand the Malop philosophy, we delve deeply into the Hacking Team data leak.

With the public release of the Hacking Team’s secrets, our researchers took advantage of the ability to dig deeply into the minds behind their operational theater. For security researchers this information is a veritable gold mine, providing us with even more clues into the latest techniques and tactics hackers are using, and how easily they are able to maintain their attacks over time. Two of our security researchers recently sat down to examine the available data, and found some amazing details about Hacking Team's activities, victims... and even the hackers that in turn brought them down.

This data dump is akin to the fall of the Soviet Union in a way. When the U.S.S.R. fell, global black markets were overflowing with Soviet weapons and, more importantly, knowledge of WMDs. This put more sophisticated weaponry and nuclear capabilities in the hands of the highest bidder, much like the Hacking Team leak has done. Except in this case the information is free, and none of the vendors whose products are exploitable, e.g. Adobe and Microsoft, were notified, amplifying the danger of the leak.

The widespread availability of this data is going to empower hacking teams across the globe, providing them with much more sophisticated techniques to launch their own attacks. These newer operations will have a completely different signature than Hacking Team’s efforts, but because of how detailed the information on their delivery server is, with perfectly readable code and extremely detailed comments, we can assess the behavior these attacks will follow and more accurately and quickly identify these operations in the future.
What we want to look more closely at is how Hacking Team targeted their attacks, and the techniques they used to maintain such large-scale operations over extended periods of time.

Hacking Team used a particular ingenious strategy for gaining access to victim machines. Firstly, the team’s operations mirrored that of the Flame malware discovered in 2012. Flame’s C&C server interface mimicked a news and adwords service, offering its “customers” - the term they used to refer to targets - a link to an “ad hosting” server, which then installed the malware. Many of its commands and protocols used news-related jargon to continue to fool detection tools and security analysts, and Hacking Team’s tactics followed the same strategy.

Note the buzzwords, “news, adwords,” used in the code.
In fact, on Hacking Team’s delivery server, we found a base64 binary file titled “news,” which we discovered was their payload. When we de-scrambled the base64 file, we found a big data blob - an AES encoded binary - containing a multi-staged payload that runs a zero day exploit for privilege escalation. The payload then executes Hacking Team’s Remote Control System (RCS) agent, which is padded with random binary data, a common anti-virus avoidance tactic.

Using a variety of standard and new techniques, such as phishing and watering hole attacks, potential targets would receive a link. Once the recipient clicked on the link, the infection server would immediately assess whether or not the machine was, in fact, a targeted recipient. If not, the script would automatically redirect them to a 404 error page or another homepage - something news or ad related (customer configurable) so as to not arouse suspicion. However, if the clicker was the intended target, the server would then profile their machine to determine their OS and browser. The server would then be able to determine if the target is using IE, Firefox, or Chrome, and what operating system they’re running, and then leverage the appropriate Adobe Flash exploit to take over the user’s machine. From there, the RCS agent was inside and able to move to the next stage of the malicious operation.
An example of a Vietnamese-targeted campaign redirecting a non-targeted individual using IE to an advertisement.

We were able to track this process by reverse engineering the files on the delivery server and JSON logs of “customer” communication. Digging deeper into the data, we were able to see when Hacking Team infiltrated a target (down to the last second), where they were located, what ISP they used, what operating system, and even which build of their browser was used to access the delivery server. For one target based in Egypt, we were able to see that they were using Chrome build 43.0.2357.130, which was released on June 22. Hacking Team infiltrated their system using the Flash exploit just six days later on June 28. This is both important, and amusing, considering Chrome is marketed as the most secure browser for the average user, but they were able to exploit it in a matter of days after the most recent update at the time.
What is also interesting about the attack is what we were able to glean about the delivery server itself, which was hosted at mynewsfeeds.info. (You may want to check your firewall and corporate proxies for this URL, in case your organization was targeted by Hacking Team!) We tracked the URL and WhoIs information to see where the team had registered it to. In fact, the registration information for their domain pointed to a rundown apartment complex in a bad neighborhood in Tel Aviv! However, the location and name associated with the WhoIs - David Cohen, the Israeli equivalent of “John Smith” - were an obvious misdirection. Not only did the team clone the techniques of Flame, which is attributed to Israel, but they also falsified their domain registration as if it were based in Israel.
One file we found related to the mynewsfeeds.info domain on VirusTotal.com was tmp_privesc, a binary which contains a privilege escalation exploit using an Adobe driver that is present on both Windows and Mac OS X operating systems. This could be the “smoking gun,” which allowed the usage of this exploit in the wild, and would allow us to identify it much more quickly on endpoints, which we will touch on more in a future write up. By leveraging Virus Total as one of our threat intelligence sources, we can apply machine learning and big data to cross examine the information from the data dump and better identify these tactics and tools when they are used again in the future.

We also found it important to note that the mynewsfeeds.info domain only had a few hashes associated with it before the Hacking Team leak. However, since then more than a dozen have cropped up, and while these weren’t found to be harmful, they all include the hash of the newsfeeds domain embedded in them - likely a result of numerous groups now downloading, compiling and running the code themselves.

The Cybereason platform was able to identify Hacking Team’s privilege escalation exploit in elevator.exe out of the box.
Taking a closer look at the Hacking Team attack operation allows us to gain a better understanding of how the existing threat landscape is going to evolve.

In part one, we discussed why the Hacking Team leak is a game-changing event for cyber security, providing a brief overview of the tools the team used and distributed to their clients and the rather sophisticated tactics they deployed in order to sustain long-term operations. Now, we’ll be focusing on their actual attack process, from the infection workflow to their RCS agent operation, and the different infection processes that they utilized.

The first thing to examine within Hacking Team’s attack process is how the infection server operates. View our flowchart on the next page for a visual of the process.

The server first runs the visitor to the infected domain through a Mod_rewrite regular expression rule on the Apache httpd server to match the six character campaign ID to the appropriate exploit kit and payload in the predesignated ID directory /var/www/files/<campaignID>. If the campaign ID doesn’t match, the server automatically redirects the visitor to a 404 error. If it does, the script moves to step two.
Hacking Team Infection Server Process

1. If the ReWrite, RegExp Matches (docs/a-zA-Z0-9(\.)?), Mod rewrite
   succeeds, continue. Otherwise, return 404.

2. If hits = 0, process is invalid. Otherwise, continue.

3. If expiry < now, process is invalid. Otherwise, continue.

4. If UserAgent Filter Match succeeds, log valid. Otherwise, log invalid.

5. If log valid, infect.
In step two, the script checks the hit counter for that campaign to ensure it equals zero - meaning that no one has been infected by the campaign yet. It also reviews the expiration date of that particular campaign. From what we have seen, all of Hacking Team’s campaigns were standardized with a one week expiration date from the time of campaign creation.

This helpdesk ticket highlights the one week expiration on the infection server.
If both the hit counter and expiration validate, the script then checks the user agent of the victim's browser against the Browscap PHP library on the server to ensure it meets the campaign requirements, eg. Windows 7, Chrome build 43.0.2357.130.

One interesting function of the infection server was Hacking Team's `xp_filter.py` Python script, which would check the victim's system to determine if they were running Windows XP or not and run a non-XP-based exploit, or a just serve a fake SWF file, `empty.swf`.

Sample of the infection server validation script from a Vietnamese attack campaign.
#!/usr/bin/env python

import os
import sys
import struct

def main():
    platform = os.environ.get('_BROWSCAP_platform')
    sys.stderr.write(platform)

    target_dir = os.path.dirname(os.path.realpath(__file__))
    if platform.lower().find('xp') == -1:
        # not xp, serve the exploit
        sys.stderr.write('\nnot xp\n')
        sys.stdout.write(open( os.path.join(target_dir, 'platform.swf')).read())
    else:
        # xp, serve fake swf
        sys.stderr.write('\nxp\n')
        sys.stdout.write(open( os.path.join(target_dir, 'empty.swf')).read())

if __name__ == '__main__':
    main()

The XP filter Python script. The comments were written by Hacking Team.

The script then “echoes” the content of the news payload into STDOUT, which is a hacky way that the script uses to send the payload through the webserver and from there to the victim. This is the base64 encoded and AES encrypted payload we referenced in our previous article, which contains the RCS agent and the team’s privilege escalation exploit. The shellcode executes the privilege escalation exploit first to gain NT AUTHORITY\SYSTEM privileges in the SYSTEM shell, then executes the agent.exe for the RCS client. Trend Micro has an excellent write-up on the privilege escalation exploit.

In addition to the Windows-based infection server, Hacking Team was also running an Android-based strategy, which utilized similar tactics but didn’t use the Flash exploit.
import os
import sys
import struct

def main():
    browser = os.environ.get('_BROWSCAP__browser')

    sys.stderr.write('[*] Browser {}
'.format(browser))
    target_dir = os.path.dirname(os.path.realpath(__file__))

    privesc = open(os.path.join(target_dir, 'news')).read()

    if 'IE' not in browser:
        article_number = os.environ.get('_REQUEST__article')

        if int(article_number) == 61441:
            sys.stdout.write(open(os.path.join(target_dir, 'news')).read())
            sys.stdout.flush()
            sys.stderr.write('[*]..server')

            sys.stderr.write('
[*] Chrome/FF News {}
'.format(article_number))
        else:
            sys.stdout.write(privesc)
            sys.stdout.flush()
            sys.stderr.write('[*] IE len {}
'.format(len(privesc)))

    The final privilege escalation and payload delivery script.

    The payload delivery process is actually impressively sophisticated, and while some may argue that the tools and exploits were utilizing were not, their actual workflow was particularly creative. In addition, the sheer variety of delivery methods provide customers with a significantly amplified ability to gain access to their intended target(s).
Once the target(s) is infected, this is when the RCS agent goes to work. There were a vast array of modules the agent would load, depending on what Hacking Team’s customer requested, from recording webcam images, Skype calls or keystrokes to tracking financial transactions (including bitcoin and other cryptocurrencies) or pinpointing the target’s geographic position. Not to mention the mobile capabilities, such as sending invisible SMS messages that leveraged exploits in the phone’s SMS stack, thus executing Hacking Team’s agent on the phone that allowed the attackers to turn the microphone on, providing a live audio stream from the target’s phone. We will cover this more in a future write-up. The actual activities of the client and the information they sought are far less interesting than the varied attack strategy that Hacking Team used.

The above is, of course, only a single attack process. Hacking Team provided a variety of solutions depending on what their customer needed, including variations better suited for nation-state level attacks. One example of this was the use of a network injector, a particularly nasty tool that would be plugged into an upstream or ISP backbone. Once active, the network injector would be able to identify the target(s) based on a customer defined rule set and wait for the victim to visit a specific URL, such as YouTube.com. Then, it would automatically redirect the victim to the team’s infection server instead. This resulted in the “the page you requested is being loaded” redirect screen.
However, Hacking Team used a wide variety of techniques to ensure infection.

Another strategy, which could be used in conjunction with the network injector, was a tool called *Melter*. This allowed the customer to silently “melt” the RCS agent into the binary of other, benign software. While not new, when combined with the network injector, this allowed campaigns to target software downloads and ensure that the target(s) installed the client’s RCS agent alongside the piece of software they were intending to get.

Of course, all of these strategies, on their own, are vulnerable to discovery, which is why Hacking Team also built an Anonymizer tool, which would randomize the attacker IP for each campaign in order to mask both the source and target(s) of the attack. The Anonymizer was Hacking Team’s own “private anonymization cloud” solution. This offered the ability for each customer to deploy their own virtual private servers (VPSs) that could be chained together for a anonymous proxy chain in order to eliminate tracing of the public-facing collectors run by each customer.
This is accomplished by passing the victim's collected data through several anonymizing machines to the collector node which then passed the data back to the master node (C&C server).

Below are a few examples of documentation on the Anonymizer tool, pulled directly from Hacking Team's *RCS 9.6 System Administrator Manual*:

**Introduction**

An Anonymizer is used to redirect data from a group of agents and Network injectors. The Anonymizer is installed on a server connected to Internet which cannot be reconnected to the rest of the infrastructure like, for example, a VPS (Virtual Private Server), needed for this purpose. Several Anonymizers can be set up in a chain to increase the level of protection. Each chain leads to one Collector.
Of course, we want to stress once again that all of this source code is accessible by anyone now, so these capabilities have entered the wild, freely usable by any hackers, whether they are experts or novices. These exploitation abilities, combined with the various reports on BGP hijacking attacks by Hacking Team (1, 2) have theoretically allowed hacking team to make everyone on the internet pass through their systems and infect them.

So, what does this mean for you?

We’ve been discussing the potential damage that the Hacking Team data dump has unleashed on the cybersecurity industry, but on an individual level it can be difficult to identify exactly what the risk factor is, and why, honestly, you should really care about it. Rather than bore you with more details of what Hacking Team was capable of doing and the tools and exploits the leak of their data released on the world, let’s delve directly into what this event means for businesses and organizations that need to protect themselves from future attacks.

In our analysis of the Hacking Team data leakage we reviewed the tools and methodologies that they were using and selling.
The example we showed earlier was the attack that mimicked Flame, the hacking operation that targeted the Iranian nuclear weapons program in 2012. This capability to imitate a nation-state attack, as well as other strategies deployed by Hacking Team, were well documented in their data dump, accompanied by full explanations of the tactics and the exploits they used. This information is now readily available to anyone who downloads the leaked data - a game changing event that breaks the fragile cyber security status quo between hackers and defenders.

The business implications here are that the overall threat landscape is already erupting with more advanced challenges and new threats to protect yourself from, with the guarantee of increased sophistication in future evolutions of cyber crime. This paints a rather bleak picture for cyber security, but all hope is not lost.

Our ability to break down the methodologies used by the Hacking Team allows us to better anticipate a Hacking Team-based threat. The way the Hacking Team attack was built will assuredly be taken, changed and redistributed in the future, but by understanding the underlying principles of their malicious operation philosophy, we can be better prepared to identify these threats as well.

This leak has provided us with the necessary information to enhance our security tools and detection platforms so that we can continue to proactively hunt for malicious activity inside any environment. While the Hacking Team leak introduces more advanced threats, it also brings a ray of hope for continuing to improve our tactics for mitigating these risks.
About Cybereason

Cybereason was founded in 2012 by a team of ex-military cybersecurity experts to revolutionize detection and response to cyber attacks. The Cybereason Malop Hunting Engine identifies signature and non-signature based attacks using big data, behavioral analytics, and machine learning. The Incident Response console provides security teams with an at-your-fingertip view of the complete attack story, including the attack's timeline, root cause, adversarial activity and tools, inbound and outbound communication used by the hackers, as well as affected endpoints and users. This eliminates the need for manual investigation and radically reduces response time for security teams. The platform is available as an on premise solution or a cloud-based service. Cybereason is privately held and headquartered in Boston, MA with offices in Tel Aviv, Israel.

© All Rights Reserved. Cybereason 2015