FOX IT
FOR A MORE SECURE SOCIETY
part of nccgroup

Lead Author: Maarten van Dantzig
Co-Authors: Danny Heppener, Frank Ruiz, Yonathan Klijnsma, Yun Zheng Hu, Erik de Jong, Krijn de Mik, Lennart Haagsma

Ponmocup

A giant hiding in the shadows

Version 1.1

November 30, 2015

Executive Summary

Ponmocup, first discovered in 2006 as Vundo or Virtumonde, is one of the most successful botnets of the past decade, in terms of spread and persistence. Furthermore, the reasons why this botnet is considered highly interesting are that it is sophisticated, underestimated and is currently largest in size and aimed at financial gain.

This underestimated botnet is still in active use and under continuous development. Having established that Ponmocup's primary goal is likely financial gain, it is interesting to look at its size. FOX-IT has determined that it has infected a cumulative total of more than 15 million unique victims since 2009. At its peak, in July 2011, the botnet consisted of 2.4 million infected systems, which as far as botnets go, is huge. Since then, the botnet has shrunk in size and is currently stable at around 500,000 active infections, as shown in Figure 1.

Compared to other botnets, Ponmocup is one of the largest currently active and, with 9 consecutive years, also one of the longest running. Ponmocup is rarely noticed though, as the operators take care to keep it operating under the radar.

Ponmocup's operators are technically sophisticated, their techniques suggest a deeper than regular knowledge of the Windows operating system. On top of that, the operators have close to 10 years of experience with malware development. Their framework was developed over time, quality tested and then improved in order to increase robustness and reduce the likelihood of discovery.

The operators are most likely Russian speaking and possibly of Russian origin. This is based on the fact that instructions to business partners and affiliates are written in Russian, and that historically, Ponmocup would not infect systems in some post-Soviet States.

Ponmocup is believed to be aimed at financial gain. Although it is difficult to quantify the exact amount of money earned with the Ponmocup botnet, it is likely that it has already been a multi-million dollar business for years now. There are multiple reasons to assume this is the case. Firstly, their infrastructure is complex, distributed and extensive, with servers for dedicated tasks. Secondly, they operate, maintain and monitor their comprehensive infrastructure with a group of operators and are quickly able to mitigate potential risks that are discovered. Thirdly, the malware itself is sophisticated and aimed at avoiding detection and analysis. FOX-IT believes, based on the earlier mentioned reasons, that they are protecting a very well run organization and infrastructure, for their main goal: financial gain.

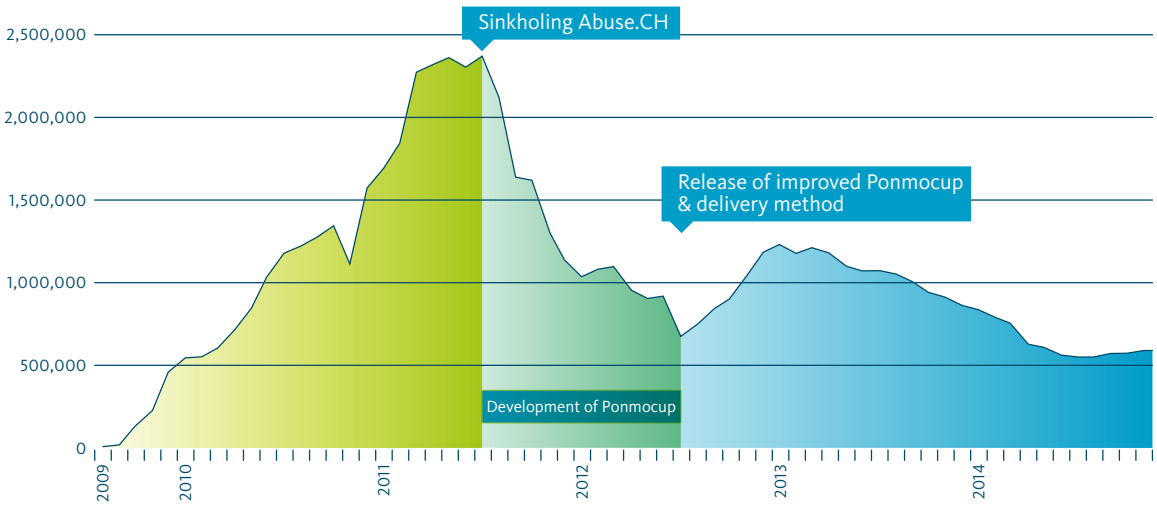

Figure 1: Number of active Ponmocup bots over time

Table of contents

1 Introduction **5**

2 Behind Ponmocup **6**
2.1 Attribution 6
2.2 Goals and impact 6
2.3 Size 7

3 Overview of the technical framework **8**
3.1 Framework components 8
3.2 Typical Ponmocup infrastructure 8

4 Delivery methods **10**
4.1 Delivery through ZIP file 11
4.2 Delivery through a signed Java applet 11
4.3 Delivery through a JAR loader 14

5 Installation, persistence and functionality **16**
5.1 The Ponmocup installer 16
5.2 Core functionality 17
5.3 Specific functionality through plug-ins 19
5.3.1 Plug-ins #14xx range – decide: finding interesting targets 22
5.3.2 Plug-in #2600 – SIP scanner: collecting information on SIP gateways 23
5.3.3 Plug-in #2610 – router scan: collecting router information 26
5.3.4 Plug-in for MS10-061 vulnerability: lateral movement 27

6 Command and control traffic **28**
6.1 Installer communication 29
6.2 Main module communication 30
6.3 Plug-in communication 30

7 Anti-analysis techniques **32**
7.1 Checks for signs of analysis 32
7.2 Delivery of fake payload 32

Appendix I – Targeted keywords 34
Appendix II – Network based indicators of compromise 36
Appendix III – Host based indicators of compromise 38

1 Introduction

Ponmocup, first discovered in 2006 as Vundo or Virtumonde, is one of the most successful botnets of the past decade, in terms of spread and persistence.

FOX-IT believes this is an underestimated botnet currently still in active use and under continuous development. Though Ponmocup has received only minimal attention from the security community and is often described as low risk, it is in fact a technically sophisticated malware framework with extensive functionality. The result of our research provides a complete time-line and unique insight into the modus operandi of the operation around Ponmocup and describes all the important details of the malware. Furthermore, this report includes currently not publicly known indicators of compromise, both on host and network level, where previous research only scratched the surface.

2 Behind Ponmocup

This chapter discusses non-technical aspects of the Ponmocup botnet: attribution, goals, impact and size.

2.1 Attribution

This section describes a number of aspects related to the operators of the Ponmocup botnet.

Based on the size of the command and control infrastructure, it is thought that the infrastructure is maintained, monitored and protected by a well-organized group of operators. This is amongst others based on the domains in use, number of proxies in use, estimated number of back-end systems, used delivery methods and limited affiliate schemes.
It was also observed that in certain cases, the operators reacted quickly to events which could impact the botnet's infrastructure, suggesting that the operators closely monitor their back-end infrastructure.
In addition, some operators were observed as active members in underground advertisement fraud forum or signed up for underground advertisement fraud schemes.

It is believed that the operators are most likely Russian speaking and possibly of Russian origin. This is based on the fact that instructions to business partners and affiliates are written in Russian, and that in the early days, Ponmocup would not infect systems in the post-Soviet States of Ukraine, Russia and Belarus. However, this specific block was later removed, for unknown reasons.

As for technical capabilities, it can be said that the operators are fairly sophisticated. The successful combination of various components to execute a stealthy malware framework, in addition to the usage of undocumented Windows APIs to reset system restore points, suggests deeper than regular knowledge of the Windows operating system. On top of that, it is certain that the operators have close to 10 years of experience with malware development, as the first variant of Ponmocop (Vundo) was discovered in 2006. This translates into features of the framework that were developed over time, quality tested with debug releases of various components, in order to increase robustness and reduce the likelihood of discovery. This includes the fact that everything in the framework (APIs, strings, etc) is obfuscated. Furthermore, the infrastructure is well thought out and load balances victims between domains and proxies as an anti-sinkhole measure. Finally, the framework uses a number of anti-analysis methods, such as the fake payload and blacklisting of IP addresses and system fingerprints once an analyst or researcher is found. Some of these techniques will be discussed in more detail further on in this paper.

2.2 Goals and impact

As with any modern malware, the Ponmocup framework is capable of supporting any objective, be it criminal or espionage-oriented in nature. These theoretical capabilities, however, aren't very useful in order to determine the operator's actual goals. Ponmocup's real goals have remained somewhat elusive over the years, primarily because FOX-IT has only rarely seen any sustained activities taking place. Based on what is known now, Ponmocup's operators are believed to be primarily interested in financial gain. However, they are currently either applying extreme restraint or carrying out their activities outside of FOX-IT's sphere of knowledge. Either way, they cherish their botnet and handle it with care. This can be supported by the following observations.

As shown in paragraph 2.3, Ponmocup is a large botnet, supporting a large amount of victims. This is a direct result of its design:

1. It is difficult for traditional anti-virus solutions to reliably detect because it uses unique encryption per infected system and locates its core components in a unique location per infected system.
2. It uses one-time domains for installation, which means that these domains cannot be used as indicators of compromise over time or across organization.

It is believed that the operators are most likely Russian speaking and/or possibly of Russian origin.

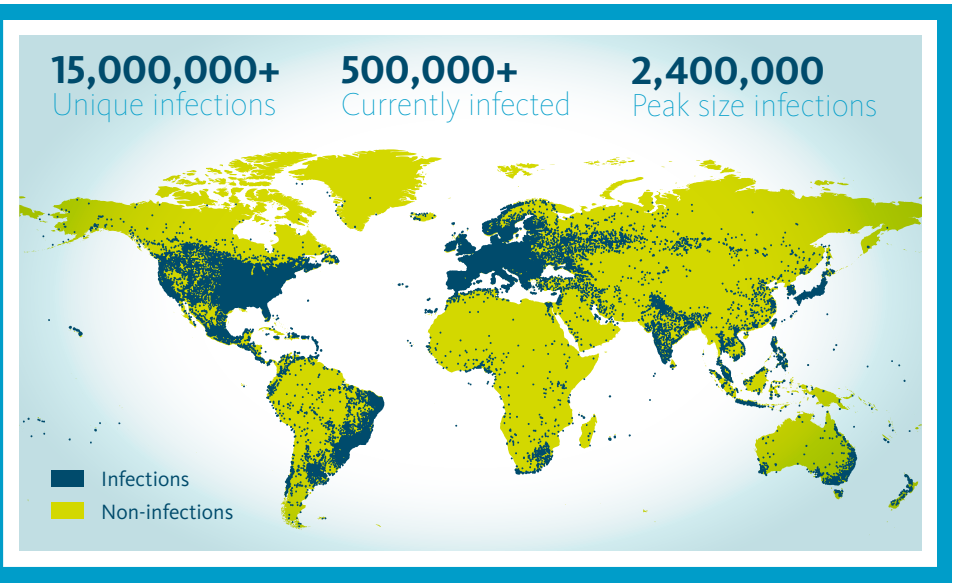

Figure 2: Ponmocup key figures

3 It supports theft of FTP and Facebook credentials out of the box, which FOX-IT believes may be used to support further spreading the botnet if needed.

Finally, Ponmocup is believed to be aimed at financial gain for the following reasons:

1 The plug-in that support advertisement fraud (ppc, abbreviation for 'pay-per-click') is the most actively developed plug-in.
2 The framework appears to target mainly wealthy and larger English speaking nations for banking, investment and trading websites that store sensitive personal information which could help in committing fraud. Its targets originally comprised mainly EU and larger English speaking countries, which later narrowed down to English speaking countries only, narrowing down even further to United Kingdom and United States only in 2012.
3 It supports theft of Bitcoin wallets.

As already outlined, Ponmocup is believed to be aimed at financial gain. Although it is hard to quantify the exact amount of money earned with the Ponmocup botnet, it is likely that it is already a multi-million dollar business for years now. There are multiple reasons to assume this is the case. Firstly, their infrastructure is complex, distributed and extensive, with servers for dedicated tasks. Secondly, they operate, maintain and monitor their comprehensive infrastructure with a group of operators and are quickly able to mitigate potential risks that are discovered. Thirdly, the malware itself is very sophisticated using a multi staged loader and sophisticated AV evasion techniques, trying to stay under the radar as much as possible, in order to avoid detection. FOX-IT believes, based on the earlier mentioned reasons, that they are protecting a very well run organization and infrastructure, for their main motivation: earning tons of money.

2.3 Size

Having established that Ponmocup's primary goal is likely financial gain, it is interesting to look at its size. FOX-IT has determined that Ponmocup has infected a cumulative total of more than 15 million unique victims since 2009. At its peak, in July 2011, the botnet consisted of 2.4 million infected systems, which as far as botnets go, is huge. The botnet has since then shrunk in size as a result of a coordinated sinkhole action and natural rotation of bots. Currently, there are still more than 500,000 victims checking in to command and control servers each month. For distribution purposes, more than 5000 websites have been compromised since 2009, using FTP credentials stolen by Ponmocup components, in order to further spread the malware.

Compared to other botnets, Ponmocup is one of the largest currently active and, with 9 consecutive years, also one of the longest running. Ponmocup is rarely noticed though, as the operators take care to keep it operating under the radar. An update that included a failed printer exploit, in 2012, provided a rare moment in the lime lights: it caused printers worldwide to start printing garbage data until they ran out of paper or ink[1].

1 *http://www.bbc.com/news/technology-18547935 and http://www.gizmodo.co.uk/2012/06/printer-virus-on-the-loose-good-day-for-paper-companies-bad-day-for-trees/*

3 Overview of the technical framework

Ponmocup is a malware framework, written in C++, designed to infect and remain persistent on a large number of victim machines. This chapter describes the components that comprise the framework.

3.1 Framework components

The Ponmocup framework employs a number of components to deliver, install, execute and control the malware, as listed in Table 1. Each component uses different anti-analysis methods to prevent the framework from being discovered.

Reverse engineering the functionalities can be a labor-intensive process, as the malware executes over various stages, where each string is decrypted in-line using various algorithms. Components that are integral to the functioning of the framework are often encrypted or stored using information specific to a victim's system.

3.2 Typical Ponmocup infrastructure

The infrastructure used to control the botnet is designed to be resilient to disruption attempts, using a separate infrastructure per component. This requires an extensive server set-up which is constantly monitored for performance issues and disruption attempts by external parties.

Ponmocup communicates to back-end servers over several proxy layers and each victim can use a specific group of proxies to communicate. Using proxy groups means that the botnet is spread over several domains, a technique that makes taking down the entire botnet difficult.

Component	Purpose
Delivery	Delivery methods used to infect victims with Ponmocup
Installer	Installs Ponmocup persistently, thoroughly checks the target machine for analysis capabilities
Initiator	DLL stored on disk, starts the loader in memory and hands over control to the loader
Loader	Finds the location of the registry key containing the encrypted main module and plug-ins, decrypts the content, starts and hands over control to the main module
Main module	Communicates with command and control server and retrieves and executes plug-ins
Plug-ins	Provide functionalities for specific tasks
Back-end infrastructure	Infrastructure used to control compromised systems

Table 1: Overview of Ponmocup components and their purpose

The operators have gone out of their way to try and defeat detection by anti-virus software, automated analysis, as well as manual analysis. Their attempts have proven very successful: over the last five years only a few analyses of Ponmocup have been published, of which none have managed to uncover the full truth of the framework.

Plug-ins carrying out specific tasks or exfiltrating specific data make use of separate proxies as well as separate back-end servers.

A simplified overview of the infrastructure between the operators and a victim is shown in Figure 3.

The operators have gone out of their way to try and defeat detection by anti-virus software, automated analysis, as well as manual analysis.

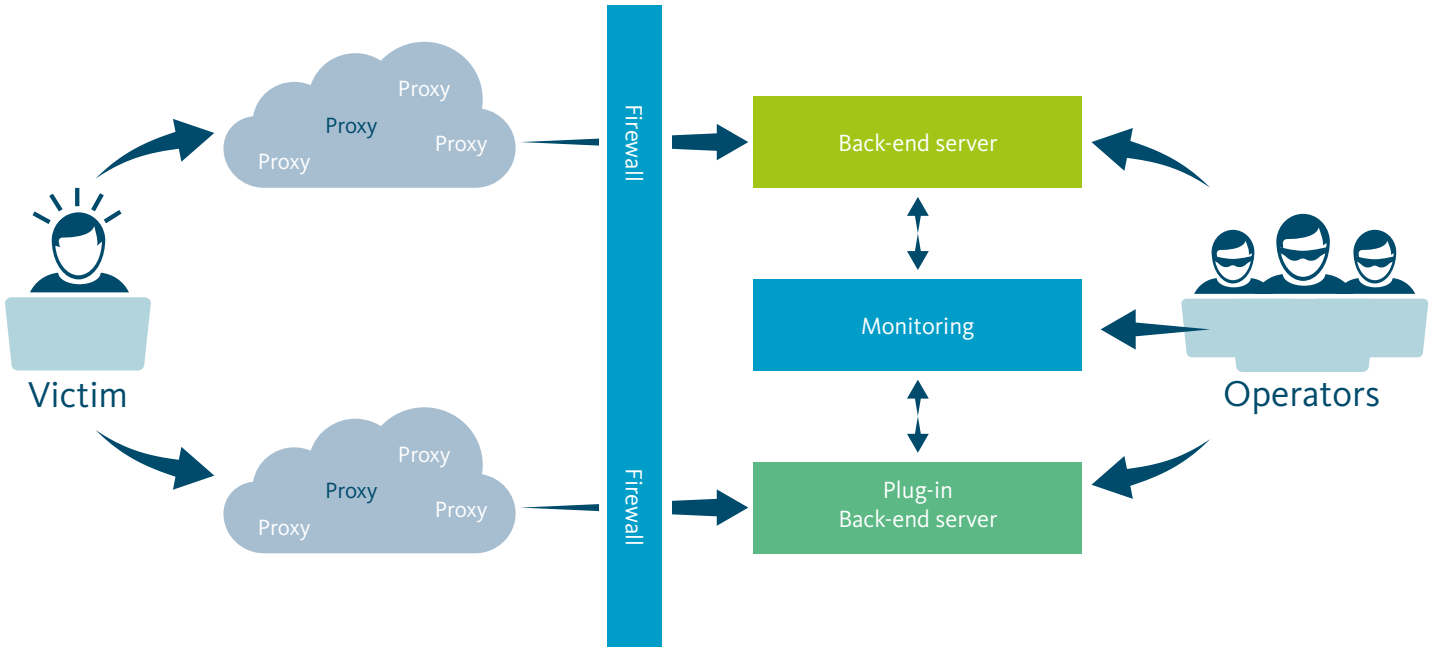

Figure 3: Typical infrastructure including victim and operators

4 Delivery methods

This chapter describes historic and current delivery methods used to distribute the Ponmocup malware

From 2009 to 2011 the two main methods used to distribute Ponmocup were fake codec packs and fake Flash Player updates. However, after a sinkhole attempt in 2011[2] the authors developed their own distribution method, publicly known as Zuponcic[3], named after the first website that was compromised for distribution purposes using this method. Although Zuponcic is commonly described as an exploit kit, that actually isn't an accurate description because it doesn't use exploits. Instead, it uses three distinct infection vectors, which in most cases depend on interaction with the victim.

Websites affected by Zuponcic are typically compromised using FTP credentials, stolen from machines infected by Ponmocup. This allows the operators behind Ponmocup to upload a carefully crafted .htaccess file to every accessible folder. It redirects visitors from the compromised website to the Zuponcic delivery mechanism. The code in the .htaccess file responsible for this redirect is placed between 500 blank lines to make it seem as if the file is empty. This code itself makes sure that not just anyone gets redirected to the malicious website; the visitor must have visited the compromised website via a search engine, social media network or webmail application. This reduces the number of potential victims, but often filters out potential victims that would more easily notice a website behaving out of the ordinary, such as the owner of the website or frequent visitors. A snippet from the .htaccess file checking for valid referrers:

```
RewriteCond %{HTTP_REFERER} ^(http\:\/\/)?([^\/\?]*\.)?(tweet|twit|linkedin|instagram|facebook\.
      |myspace\.|bebo\.).*$ [NC,OR]
RewriteCond %{HTTP_REFERER} ^(http\:\/\/)?([^\/\?]*\.)?(hi5\.|blogspot\.|friendfeed\.|friendster\.
      |google\.).*$ [NC,OR]
RewriteCond %{HTTP_REFERER}^(http\:\/\/)?([^\/\?]*\.)?(yahoo\.|bing\.|msn\.|ask\.|excite\.|altavista\.
      |netscape\.).*$ [NC,OR]
RewriteCond %{HTTP_REFERER}^(http\:\/\/)?([^\/\?]*\.)?(aol\.|hotbot\.|goto\.|infoseek\.|mamma\.
      |alltheweb\.).*$ [NC,OR]
RewriteCond %{HTTP_REFERER}^(http\:\/\/)?([^\/\?]*\.)?(lycos\.|metacrawler\.|mail\.|PINterest
      |instagram).*$   [NC]
RewriteCond %{HTTP_REFERER}!^.*(imgres).*$   [NC]
```

Once a visitor is attacked by Zuponcic, their IP-address is blacklisted and will no longer be targeted until the IP-address is cleared from the blacklist, regardless of whether the attack was successful or not. Additionally a cookie is placed on the machine to make sure that not only the IP-address, but also the actual machine itself is blacklisted. Potential victims redirected by the .htaccess file are first taken to an intermediate website (typically hijacked GoDaddy domains) using one out of 60 listed URI patterns. The conditions that a potential victim has to meet in order to be attacked are visualized in Figure 5.

Because Zuponcic requires a valid referrer chain and blacklists an IP-address after a single hit and most URL analysis tools fail to perform dynamic analysis on these compromised websites, most of these websites are falsely classified as 'safe' and remain compromised for relatively longer periods.

Once a visitor is actually redirected to the Zuponcic landing page he or she is targeted based on the used browser and presence of the Java browser plug-in. The decision chart in Figure 6 depicts the various methods used to infect visitors with the Ponmocup payload. The next paragraphs describe each method in more detail.

4.1 Delivery through ZIP file

Using a browser other than Internet Explorer 8/10 or not having Java installed, a victim will receive a ZIP file containing the payload. The name of the ZIP and embedded payload is derived from the previously used search terms, as shown inFigure 4.

Compared to the attack vectors abusing Java, described in paragraphs 4.2 and 4.3, this attack method puts in very little effort to hide the payload. Perhaps the operators felt safe enough to introduce this method to their arsenal of infection vectors at the cost of slightly more attention.

4.2 Delivery through a signed Java applet

This infection vector relies on social engineering or outdated Java software in order to execute a Java applet. These applets are typically run in a sandbox, in order to prevent them from touching the file system, so to drop the Ponmocup installer on a victim's machine this Java applet has to escape the sandbox. Ponmocup successfully does so because the Java applet is signed with a valid certificate, stolen from a legitimate organization. Older versions of Java (pre-dating Java 7 Update 21 to be specific) which blindly trust certificates issued by authorities, will run this applet outside of the sandbox without even asking for the user's permission.

Recent versions of Java always prompt for user approval before running applets, though in this case the applet might still appear legitimate to potential victims, as the applet is still signed with a valid certificate and claims to be executing an application with the name 'FlashPlayer'. The certificates listed in Table 2 have all been used by Zuponcic to infect victims with Ponmocup.

Figure 4: Using victim's own keywords for payload filename

Because the certificates used by Zuponcic are stolen from victims that have been infected by Ponmocup, it typically does not take long for a revoked certificate to be replaced by a new one. In most cases revoked certificates were replaced with a newly stolen certificate within 1–3 weeks.

The file doing all the work, FlashPlayer.class, is heavily obfuscated using control flow obfuscation. Additionally, all strings are encoded using an encoding key which consists of dynamic values from the process stack. This works if the class and calling method names are static, which has always been the case for Zuponcic; the class name is 'FlashPlayer' and calling method is 'init', which results in 'FlashPlayerinit' as the decoding key.

```
final StackTraceElement
stackTraceElement = new Exception().
getStackTrace()[1];
final String string = new
StringBuffer(stackTraceElement.
getMethodName()).insert(0,
stackTraceElement.getClassName()).
toString();
```

FlashPlayer.class will create an empty .tmp file with a random short name (2–5 characters) in the TEMP directory. It then HTTP POSTs to the Zuponcic host, using a token which is unique per victim, to retrieve an RC4 encrypted payload. This payload is then stored in the .tmp file and decrypted with a key also uniquely generated per victim (taken from the landing page). If the applet is allowed to run, it will execute the payload, once executed the browser redirects the victim to the actual website that

2 *How Big is Big? Some Botnet Statistics - https://www.abuse.ch/?p=3294*

3 *http://blog.FOX-IT.com/2013/12/19/not-quite-the-average-exploit-kit-zuponcic/*

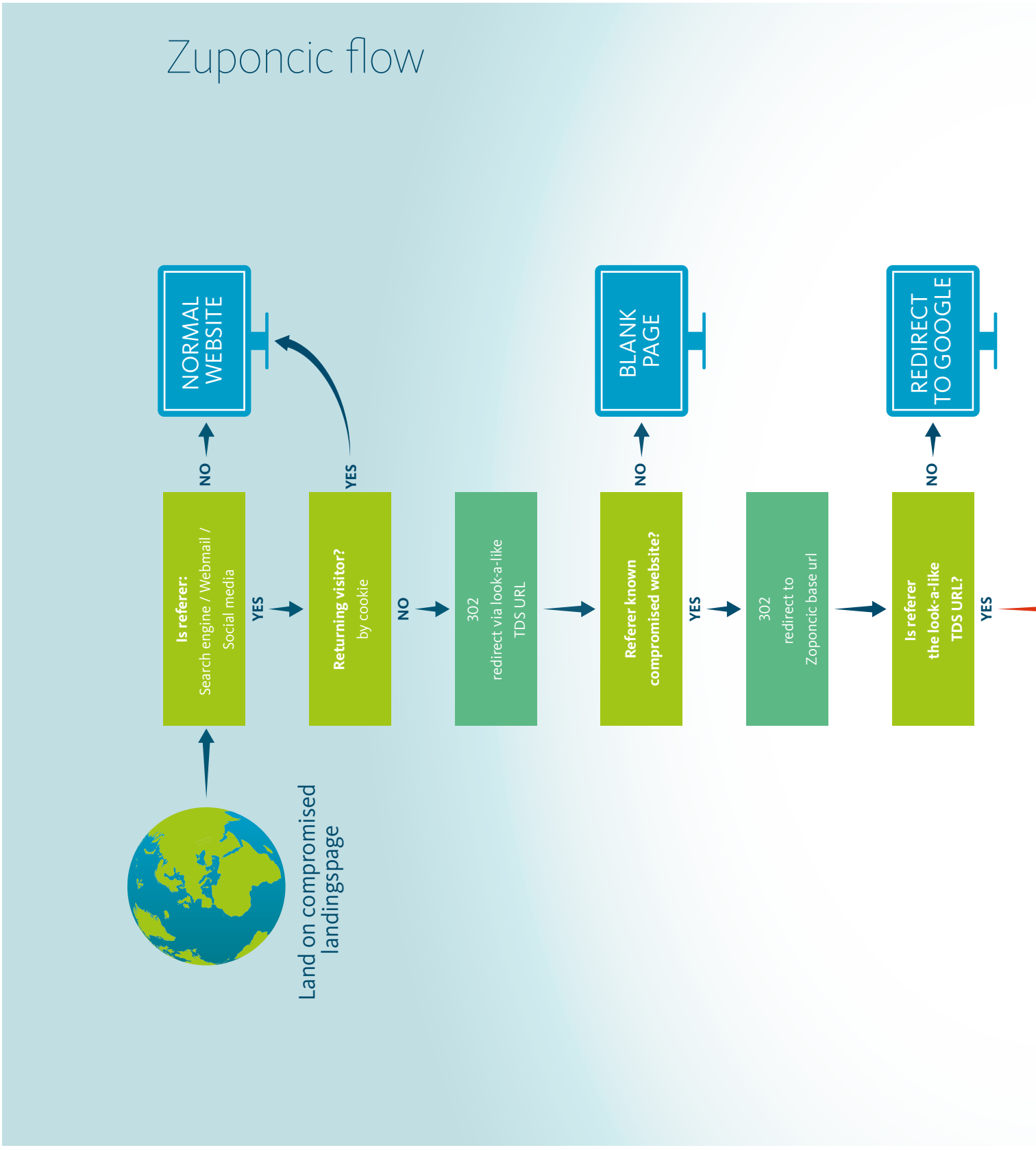

Figure 5: Zuponcic redirection flow

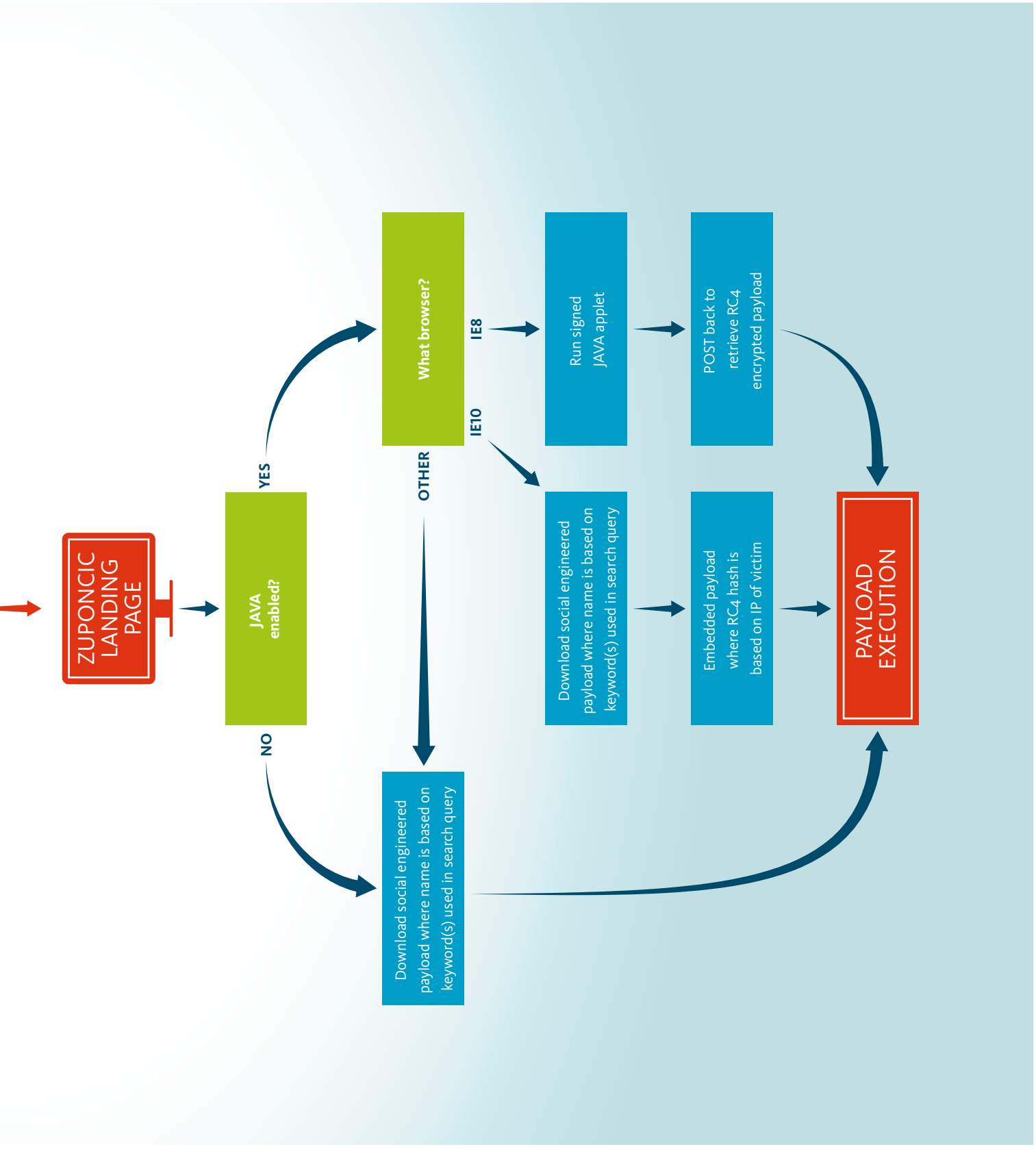

Figure 6: Zuponcic attack flow

Subject	Fingerprint (SHA1)	Issuer	Year
Kurz Instruments, Inc.	8A:DC:2D:8B:B5:3C:DC:93:C9:80:C4:F6:C0:80:59:73:8B:88:19:16	GlobalSign	2012
R P InfosystemsPvt Ltd	BB:48:74:0F:01:E6:7F:EE:A6:06:96:4B:D5:81:A7:30:BF:D0:54:D7	VeriSign	2012–2013
iLoqOy	76:90:09:5B:C3:FC:9F:9D:74:98:56:F6:E1:DD:22:C0:89:44:F7:F9	VeriSign	2013–2014
AUZSOLUTIONS.com.au	F1:39:8E:53:D1:F8:FC:06:34:F5:4E:68:72:88:5F:31:CC:09:35:23	COMODO	2014
Queen's University	73:E8:D6:F3:91:77:2A:7F:AE:81:C3:81:73:14:2E:C8:F6:28:2A:E4	UserTrust	2015

Table 2: Stolen certificates used in the delivery of Ponmocup

was visited, whether the execution is successful or not, leaving to believe nothing strange has happened, a process outlined in Figure 7.
If execution of the payload fails, FlashPlayer.class ensures that the created .tmp file is deleted from disk. This clean-up routine prevents any evidence from being left around.

4.3 Delivery through a JAR loader

A target using IE10 will receive a JAR file which has to be manually downloaded and executed. It may seem unlikely for anyone to run this manually, but through clever social engineering, Zuponcic tricks many victims into running its malicious payload. This is done by using the search terms used in the redirecting search engine in the payload name, as shown in the example in Figure 4.
This JAR file contains an RC4 encrypted payload. The key is based on the target's IP address. This means the key and the embedded encrypted binary are unique per target. Once the JAR is executed, the code below recovers the RC4 key used for decryption:

```
final URLConnection openConnection;
(openConnection = new URL("http://checkip.dyndns.com").openConnection()).
setRequestProperty("User-Agent", "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)");
final BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(openConnection.getInputStream()));
final String readLine = bufferedReader.readLine();
String group = "";
bufferedReader.close();
final Matcher matcher;
if ((matcher = Pattern.compile("([0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3})").
matcher(readLine)).find())
{
    group = matcher.group();
}
final byte[] digest = MessageDigest.getInstance("MD5").digest(group.getBytes());
```

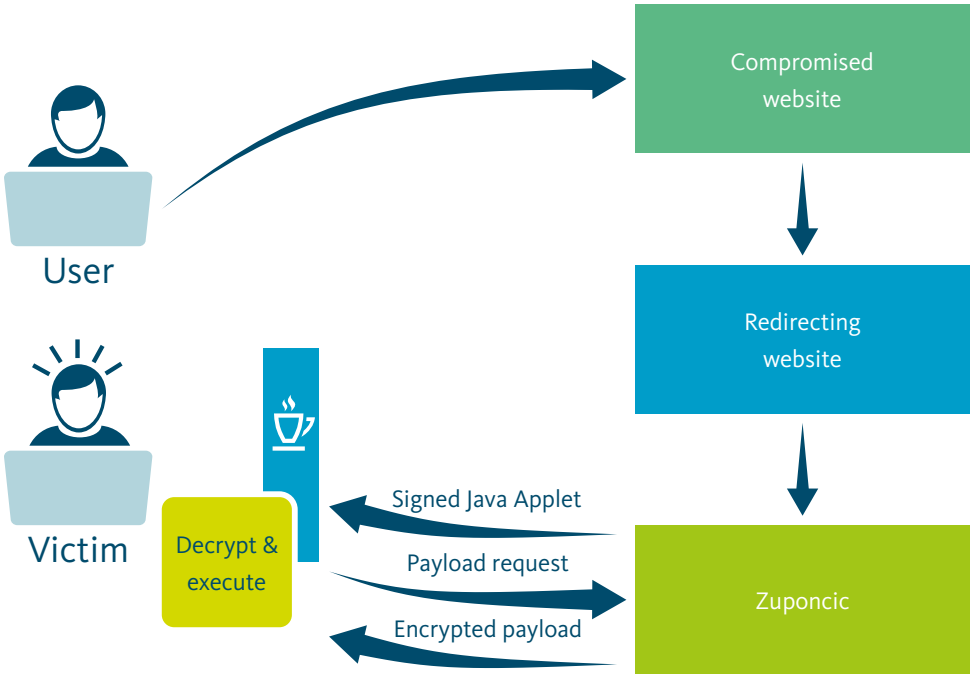

Figure 7: Java Applet retrieving and executing payload using per connection specific information

Through clever social engineering, Zuponcic tricks many victims into running its malicious payload.

5 Installation, persistence and functionality

This chapter describes the core components of Ponmocup, its method of installation, achieving persistence and its modular system of plug-ins aimed at providing a wide variety of functions on compromised systems.

5.1 The Ponmocup installer

The installer is responsible for persistently installing various Ponmocup components on a system. This paragraph focuses on components installed directly on disk. All other core components are stored and encrypted in the registry. Without these, the initiator described in this chapter cannot function.

Depending on the privileges available to the installer, the initiator can either be installed in the system directory or in the Application Data directory. When run with administrative privileges, the initiator will be named after an existing file in the system directory, with 1–2 random characters appended. For example, a randomly selected legitimate file could be:

```
C:\Windows\System32\msg711.acm
```

And the Ponmocup initiator would be named:

```
C:\Windows\System32\msg711A.DLL
```

This effectively means that the core functionalities of Ponmocup are uniquely encrypted and stored in a unique location for every victim.

If stored in the system directory the file will have the R (Read-only), S (System) and H (Hidden) attributes set. The installer adds a scheduled task to start the initiator during system boot, with the privileges of NT AUTHORITY\SYSTEM:

```
<Principals>
<Principal id="Author">
<UserId>System</UserId>
<RunLevel>HighestAvailable</RunLevel>
<LogonType>InteractiveTokenOrPassword
     </LogonType>
</Principal>
</Principals>
<Actions Context="Author">
<Exec>
<Command>C:\Windows\system32\
     runDLL32.exe</Command>
<Arguments>"C:\Windows\system32\
     msg711A.DLL",ZBADQX</Arguments>
</Exec>
</Actions>
```

Without administrative privileges the initiator is stored in the Application Data directory with a random name (6–10 characters) and initiated during system boot via a run-key in the registry.

During installation some additional tasks are carried out by the installer to ensure persistence. One of these tasks, not often seen, is to delete all system restore points and to disable the system restore option itself. To achieve this Ponmocup makes use of two APIs from the srclient.DLL library; srclient.ResetSR and srclient.DisableSR. These APIs are not documented by Microsoft, which is one of

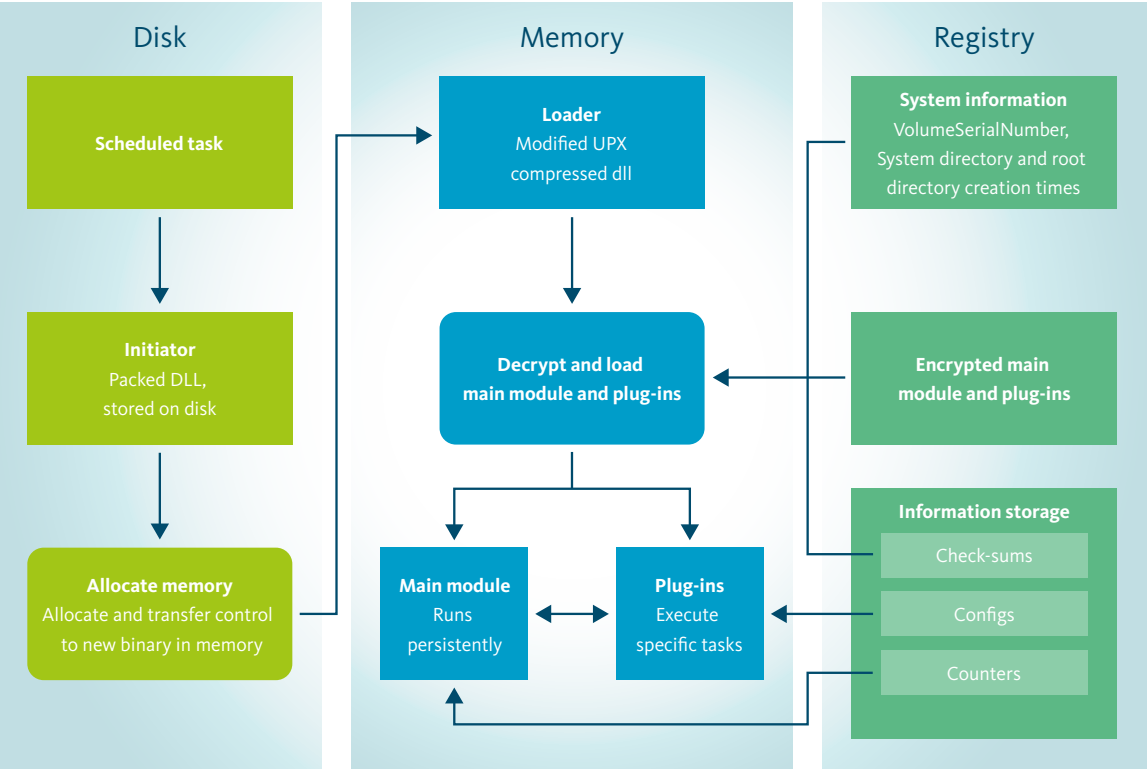

Figure 8: Overview of the process of loading Ponmocup

the indicators that the operators have a thorough understanding of windows internals, which displays a certain level of sophistication which is seen throughout the entire framework. One of these particular APIs (ResetSR) was also used by the Conficker worm [4].

5.2 Core functionality

The core of Ponmocup are the components installed on a victim's machine by default. These mainly include the components responsible for starting the main module, but also include persistent plug-ins providing specific tasks for persistence purposes.

As shown in Figure 8, loading Ponmocup is a multi-step process which is best broken down into three categories: files on disk, activity in memory and the usage of stored information in the Windows registry.

Starting the Ponmocup core is a three part process consisting of the DLL on disk (initiator), a custom UPX compressed DLL in memory (loader) and the payloads in registry (main module and plug-ins):

1. The initiator is started by the scheduled task or run key;
2. Once loaded into memory the initiator transfers control to a (modified) UPX compressed DLL;
3. This DLL acts as a loader for the main module and plug-ins.

The loader has the most important responsibilities in this process, as it has to find the main module and plug-ins, which are all encrypted and only stored in registry; both the encryption key and location are based on unique aspects of a victim's machine. This effectively means that the core functionalities of Ponmocup are uniquely encrypted and stored in a unique location for every victim.

4 *https://isc.sans.edu/forums/diary/Some+tricks+from+Confickers+bag/5830/*

The following unique information about the victim's machine is used to base the encryption on:

- Date and time of the system directory creation
- Volume serial number of the root directory of the volume Ponmocup is installing on, retrieved through the GetVolumeInformation API
- Date and time of the System Volume Information directory creation

As can be seen in the code snippet below, these unique values are each XOR'd with their combined value, forming a 16 bytes key which is used to encrypt the main module. For encryption a slightly modified version of the RC4 algorithm is used.

These values are used to uniquely generate the location of the main module and plug-ins in the Windows Registry. An example of what the registry keys might look like:

```
HKEY_CURRENT_USER\Software\wkcxjxlv\
Wjtnpgzc
```

In this case the 'Wjtnpgzc' key stores the Ponmocup main module, and persistent plug-ins. Once decrypted using the unique key, the value of this registry key is typically outlined as follows:

- Total size of decrypted content (first 4 bytes);
- Main module;
- Persistent plug-in(s).

Using this machine specific information the loader can find the location of the encrypted main module and plug-ins in the Windows Registry, decrypt their contents and execute the payloads. Before decrypting, these machine-specific values are verified against the checksums, which were stored in the registry during installation, to check if none of these values have changed. If these checks succeed, the decryption routine is started.

After decryption the loader verifies the integrity of the decrypted content by comparing the CRC32 checksum of the result with the checksum of the content before it was stored in registry during installation. Once this final check succeeds, plug-ins stored in registry are executed and control is given to the main module, which can then initiate command and control traffic. If at any stage during this process, a verification or integrity check fails, the Ponmocup execution process halts.
The main module and plug-ins can interact with each other using shared memory. This functionality is mainly used by plug-ins that only run if the main module is indeed actively running.

Both the main module and plug-ins use the registry to store configs, checksums or other relevant information storage such as counters or IDs. Information storage that may be helpful during a forensic investigation could, for example, be:

- Main PIN[5] ID
  - A PIN indicates a group of plug-ins which will be run on a victim's machine at some point in time.

```
/* setup keys */
key_1[0] = tsys32dir.dwLowDateTime ^ tsys32dir. dwHighDateTime;
key_1[1] = dwVolumeSerial;
key_1[2] = tsysvolinfodir.dwLowDateTime ^ tsysvolinfodir.dwHighDateTime;
key_1[3] = key_1[2] + key_1[0] + key_1[1] - 0x6F6F6F70;

/* xor */
key_1[0] ^= key_1[3];
key_1[1] ^= key_1[3];
key_1[2] ^= key_1[3];
```

- Some of these plug-ins that belong to the PIN can be persistent, but some plug-ins might only be executed once. Knowing the PIN ID could help identify what type of tasks may have been executed on a victim's machine.
- If, for example, this value stores 0x4A39, the PIN ID is 19001. Please see the next paragraph for more detailed information.

• Run counter
- The main module increases a counter in registry for every minute it's been active.
- If, for example, this value stores 0x0F, the main module has been active for 15 minutes.

Basic information storage can be found in separate registry keys in the following location:

```
HKEY_CURRENT_USER\Software\Microsoft\
Multimedia
```

These values can also be stored in other locations for redundancy purposes.

5.3 Specific functionality through plug-ins

The main module of Ponmocup is primarily designed to achieve persistence on a victim's machine. Plug-ins, in contrast, are used to provide functionalities for specific tasks. These tasks vary from the exfiltration of credentials and browser history, to identification of VoIP agents on the network of victims. In total FOX-IT has identified 25 plug-ins with unique identifiers, among them they share more than 4000 different versions, indicating this framework is under continuous development.
The following paragraphs describe some of the more prominent plug-ins used in the Ponmocup framework.

Plug-ins can either be present in memory only, or remain persistent by being stored and encrypted in the same registry key as the main module. Whether or not a plug-in remains persistent is dependent on the type of functionalities the plug-in provides. Once retrieved, a plug-in can make itself persistent by using the machine specific information, mentioned in chapter 5.2, to encrypt and append itself to the registry key which stores the main module and the other plug-ins. The quality of the plug-ins are thoroughly tested using specifically developed debug versions.

Identifying plug-ins is possible by analyzing their PE headers: at offset 0x20 a standard PE header contains a reserved word (WORD e_res[4]), of which **e_res[3]** is used by Ponmocup to store the ID of the corresponding plug-in. The version of the plug-in ID can be found 3 bytes further at WORD **e_oemid**.

In the example below, the PE header stores the value **0x044C** (little-endian) in WORD **e_res[3]** and **0x0BBB** (little-endian) in WORD **e_oemid**, resulting in plug-in ID '1100' and version '3003' (typically written as plug-in 1100.3003).

Table 2 provides an overview of the most important plug-ins, which are basically all DLLs given a unique identifier, version number and name by the operators.

```
00000000   4d 5a 90 00 03 00 00 00   04 00 00 00 ff ff 00 00   |MZ..............|
00000010   b8 00 00 00 00 00 00 00   40 00 00 00 00 0a 01 00   |........@.......|
00000020   4c 04 00 00 bb 0b 00 00   00 00 00 00 00 00 00 00   |L...............|
00000030   00 00 00 00 00 00 00 00   00 00 00 00 80 00 00 00   |................|
00000040   0e 1f ba 0e 00 b4 09 cd   21 b8 01 4c cd 21 54 68   |........!..L.!Th|
00000050   69 73 20 70 72 6f 67 72   61 6d 20 63 61 6e 6e 6f   |is program canno|
00000060   74 20 62 65 20 72 75 6e   20 69 6e 20 44 4f 53 20   |t be run in DOS |
00000070   6d 6f 64 65 2e 0d 0d 0a   24 00 00 00 00 00 00 00   |mode....$.......|
00000080   50 45 00 00 4c 01 04 00   00 00 00 00 00 00 00 00   |PE..L...........|
```

5 *The PIN number – a term used by the operators and of unknown origin – is one of the values stored in the registry.*

Identifier	Name	Purpose	Versions
1100	new downloader	This plug-in is known as the main module of the Ponmocup framework. Retrieves and executes additional plug-ins.	3003
1300	history tool	Collects and exfiltrates browser history from all popular browsers. This plug-in is deprecated and it's functionality are currently implemented in the 14xx plug-in range.	6
1350	avkill	Disables anti-virus related services that could potentially stop Ponmocup from functioning.	104
1400	decide	Retrieves browser history for all popular browsers, and checks if any URLs of interest to the operators were visited by the victim (only checked using a checksum). If this is the case these URLs are exfiltrated to a back-end server where this information is logged. This plug-in is only retrieved if the target is in one of the following countries of interest: • Australia, Belgium, Canada, Switzerland, Germany, Denmark, Estonia, France, United Kingdom, Mexico, Netherlands, Norway, New Zealand, Portugal, Sweden, United States	135
1402	decide_ vkusnota[6]	Retrieves browser history for all popular browsers, and checks if any URLs of interest to the operators were visited by the victim (only checked using a checksum). If this is the case these URLs are exfiltrated to a back-end server where this information is logged. This plug-in is only retrieved if the target is in one of the following countries of interest: • Australia, Canada, New Zealand, United States, United Kingdom	1
1403	decide	Retrieves browser history for all popular browsers, and checks if any URLs of interest to the operators were visited by the victim (only checked using a checksum). If this is the case these URLs are exfiltrated to a back-end server where this information is logged. This plug-in is only retrieved if the target is in one of the following countries of interest: • United States, United Kingdom	85
1507	ppc[7]	Advertisement fraud plug-in. Plug-in can inject code into the processes of Chrome, Firefox and Internet Explorer. When certain keywords are detected, a victim can be redirected to an alternative page (taken from an encrypted config in the registry). This plug-in is specifically used for the PIN[8] 3xx, 160xx and 170xx ranges.	587
1511	ppc	Advertisement fraud plug-in. Plug-in can inject code into the processes of Chrome, Firefox and Internet Explorer. When certain keywords are detected, a victim can be redirected to an alternative page (taken from an encrypted config in the registry). This plug-in is specifically used for the PIN 150xx range.	1
1512	ppc	Advertisement fraud plug-in. Plug-in can inject code into the processes of Chrome, Firefox and Internet Explorer. When certain keywords are detected, a victim can be redirected to an alternative page (taken from an encrypted config in the registry). This plug-in is specifically used for the PIN 190xx range.	26
16xx	socks	Precursor of the 18xx plug-in range, allowing the operators to connect to a victim directly, this connection is typically set-up to a specific port opened in the Windows firewall.	10
18xx	socks 2	Precursor of the 25xx plug-in range, allowing the operators to connect to a victim directly, this connection is typically set-up to a specific port opened in the Windows firewall.	24
2500	proxy	Used to directly connect to infected machines. To make sure machines behind a device providing network address translation (NAT) can still be reached individually, UPnP is used, and ports 1900 (UDP) and 2869 (TCP) are opened in the Windows firewall.	9
2550	proxy 2	Similar to the other plug-ins in the 2500 range, this plug-in can be used to directly connect to infected machines.	1

Identifier	Name	Purpose	Versions
2600	SIP scanner	Scans devices on the local subnet of the target for SIP (Sessions Initiation Protocol) agents and, if a SIP agent is present, exfiltrates information returned by the agent.	7
2610	router scanner	Scans gateway IP address of the target for common ports used by routers, and exfiltrates basic information returned by the services running on these ports.	3
2700	FTPg	1 Grabs and exfiltrates FTP and Bitcoin credentials and attempts to do so for every local user on the infected machine by bruteforcing the passwords of these accounts (using a list of commonly used passwords). 2 The specifically targeted FTP clients and Bitcoin wallets are similar to the list used by the infamous Pony Loader (2.0) Trojan [9]. 3 Stolen FTP credentials are mainly used to spread Ponmocup, with the delivery method described in chapter 4.	3
2701	FTPg_spec	1 Grabs and exfiltrates FTP and Bitcoin credentials and attempts to do so for every local user on the infected machine by bruteforcing the passwords of these accounts (using a list of commonly used passwords). 2 The specifically targeted FTP clients and Bitcoin wallets are similar to the list used by the infamous Pony Loader (2.0) Trojan. 3 Stolen FTP credentials are primarily used to further spread Ponmocup, using the delivery method discussed in chapter 4 The functionalities of this plug-in appear to be similar to plug-in #2700, but the operators are likely using this plug-in for special (spec) occasions.	1
2750	fbcookie	Grabs and exfiltrates stored Facebook credentials and cookies and attempts to do so for every local user on the infected machine by attempting to bruteforce these accounts using a list of commonly used passwords. FOX-IT has not yet observed abuse of these credentials on a large scale. It is therefore suspected that they may be used in the event of a significant loss in the number of victims, necessitating the introduction of a new spreading mechanism by the Ponmocup operators.	16
2760	sysinfo	Gathers extensive information about an infected machine, also scans for a long and diverse list of analysis and monitoring software.	5
2810	btcg	Grabs and exfiltrates generic and specific Bitcoin wallet data. Specifically targeted Bitcoin wallets: - Multibit - Electrum	2
3101	ppc	Most recently developed advertisement fraud plug-in, generates ad-fraud traffic to websites stored in a separate configs file in registry.	34

Table 3: Overview of Ponmocup's most important plug-ins

6 *vkusnota is a Russian term to describe something tasty (though typically associated with food)*

7 *ppc is short for pay-per-click*

8 *For a description of the concept of PINS, see paragraph 5.2.*

9 *https://www.damballa.com/pony-loader-2-0-steals-credentials-bitcoin-wallets-source-code-sale/*

To efficiently distribute certain functionalities to a victim, plug-ins can be grouped into a so called 'PIN' as earlier mentioned. For example, the PIN that has been most actively used since June 2012 is identified as PIN 19001 and contains 10 unique plug-ins. The table below contains an overview of the most frequently used PINs and what plug-ins they are comprised of:

PIN	Plug-ins
15001	1511, 1600, 2500, 2700, 2810
16002	1507, 1600, 2500, 2700, 2810
17001	1507, 1600, 2500, 2700, 2810
17002	1507, 1600, 2500, 2700, 2810
19001	1350, 1403, 1512, 2500, 2600, 2610, 2700, 2701, 2750, 2810
19002	1350, 1403, 1512, 2550, 2610, 2700, 2760, 2810
19010	1350, 1403, 1512, 2500, 2610, 2700, 2810

Table 4: Overview of Ponmocup's most commonly used PINs

5.3.1 Plug-ins #14xx range – decide: finding interesting targets

Through this type of plug-in, operators can retrieve the browser history for all major browsers. Not only will such a plug-in parse the history from, for example, the SQLite database of Google Chrome, but it will also retrieve the hostnames of the cookies stored on the target's machine.

This browser history is collected for the plug-in to check its content against specific keywords of interest to the Ponmocup operators. To prevent the interests of the operators from becoming known, these keywords are only checked in the form of CRC32 checksums. These keywords can be divided into 4 categories: **online banking**, investment websites, **accounting websites** and websites used to store personal information, which are used for **intelligence** purposes (for example law enforcement software or online insurance websites). An example of such keywords recovered by FOX-IT, is shown in the snippet below.

Identifying plug-ins is possible by analyzing their PE headers.

In total FOX-IT has identified 214 unique keywords, APPENDIX I contains the full list of recovered keywords. The goal of this plug-in is clear: identifying targets of interest based on their browsing behavior.

Through analysis of three specific versions of the decide plug-in, FOX-IT has identified what countries were targeted by this plug-in at certain moments in time, shown in Figure 9:

- 2009–2010: Australia, Belgium, Canada, Switzerland, Germany, Denmark, Estonia, France, United Kingdom, Mexico, Netherlands, Norway, New Zealand, Portugal, Sweden, United States
- 2010–2011: Australia, Canada, New Zealand, United States, United Kingdom
- 2012–2015: United States, United Kingdom

```
<crc_index=00 pos=00 length=13 checksum=0x03098019 keyword=us.etrade.com>
<crc_index=01 pos=06 length=01 checksum=0x01d405fc keyword=dmv.org>
<crc_index=19 pos=16 length=02 checksum=0x0167da70 keyword=drivingrecords.com>
<crc_index=02 pos=07 length=02 checksum=0x023605bb keyword=geico.com>
<crc_index=08 pos=11 length=01 checksum=0x00fe800a keyword=lppolice.com>
<crc_index=38 pos=29 length=01 checksum=0x058533bc keyword=businessonline.huntington.com/>
<crc_index=01 pos=06 length=01 checksum=0x02247d64 keyword=bbva.es>
```

5.3.2 Plug-in #2600 – SIP scanner: collecting information on SIP gateways

This plug-in attempts to identify SIP gateways on the local subnet of a victim. SIP, the Sessions Initiation Protocol, is a communications protocol used to setup and connect communications sessions, typically for voice and video calls. In the example below the plug-in identifies that the gateway IP 10.0.0.1 is running an Asterisk server; a software implementation of a telephone private branch exchange (PBX) which can be used to set-up VOIP connections.

```
00:00:00.000 ---if---flags=[UP,BROADCAST,,,MULTICAST]---addr=[10.0.0.13]---mask=[255.255.255.0]--
-bcast=[255.255.255.255]---
00:00:05.000 ---range---start=[10.0.0.0]---end=[10.0.0.254]---self=[0]---
00:00:05.000 ---ip_count=[254]---d_avr_ms=[78]---d_min_ms=[3]---d_max_ms=[153]---
     t_est_sec=[178]---
00:02:11.562 ---send_error---code=[10049]---to=[10.0.0.0:5060]---
00:02:17.281 ---known---src=[10.0.0.13]---dst=[10.0.0.1:5060]---
00:02:17.281 ---pre---begin---from=[10.0.0.1:5060]---len=[502]---crc=[4d2d01bc]---
SIP/2.0 404 Not Found
Via: SIP/2.0/UDP 10.0.0.13:5060;branch=z9hG4bK4eb9c9fb;received=10.0.0.13;rport=5060
From: "Unknown" <SIP:100@10.0.0.13>;tag=e6eaecf352f8f49ab25dcec3eba0461664bc3d70f6
To: "Unknown" <SIP:100@10.0.0.1>;tag=as7afdf3b5
Call-ID: df356de4463be948e998593728fd0d69@10.0.0.13
CSeq: 102 OPTIONS
Server: Asterisk PBX
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, INFO, PUBLISH, MESSAGE
Supported: replaces, timer
Accept: application/sdp
Content-Length: 0
00:02:17.281 ---pre---end---
```

This browser history is collected for the plug-into check its content against specific keywords of interest to the Ponmocup operators.

The plug-in then attempts to retrieve more information on the type of SIP gateway running on this IP address, by sending an OPTIONS and REGISTER, an example of an OPTIONS request:

```
00:03:15.750 ---SIP_scan_id---idx=[0]---scan_idx=[0]---pkt_type=[0]---sock_type=[0]---
     src=[10.0.0.13]--dst=[10.0.0.1:5060]---
00:03:15.750 ---send-begin---len=[556]---
OPTIONS SIP:109@10.0.0.1:5060 SIP/2.0
Via: SIP/2.0/UDP 10.0.0.13:5060;branch=z9hG4bKcac70ed4
Max-Forwards: 70
From: “Unknown” <SIP:Unknown@10.0.0.13>;tag=b899892894cb3bdc261a57219952fb0a26dea3e1cc
To: <SIP:109@10.0.0.1:5060>
Contact: <SIP:Unknown@10.0.0.13:5060>
Call-ID: 6d070c9e641bff4fb90c47f66806d6d6@10.0.0.13:5060
CSeq: 102 OPTIONS
User-Agent: Zoiper for Windows rev.1812
Date: Sat, 06 Jun 2015 12:13:12 GMT
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, INFO, PUBLISH
Supported: replaces, timer
Content-Length: 0

00:03:15.750 ---send-end----
```

The responses are summarized at the end of the log file and a CRC32 checksum is calculated over the content. In the example below the summary indicates that the response to request ID 3021 returned a 401 response:

```
00:07:27.859 ------type=[1]---code=[401]---stamp=[fbd1e9ad]---count=[155]-
-id=[3021,3002,300,4017,4026,4004,303,3001,2021,1019,103,3004,2019,3015,3014,2007,107,2010,106,4
021,4030,104,2027,4011,3030,3027,1009,4019,3006,205,2001,1007,2022,301,3005,4020,1025,309,3003,1
000,3010,3012,4008,4027,4003,2025,3029,1014,1030,2016,4022,2030,2028,207,1006,2018,4010,1011,102
1,3028,4029,1002,210,2002,3024,4014,4025,2006,108,4005,307,3026,304,3025,105,1004,2013,3018,109,
204,4023,102,1001,3019,4001,1022,2023,4012,310,4007,3011,1027,4028,1012,1010,1028,4009,1018,4015
,2026,4006,2003,1008,1015,4000,2015,308,3013,1017,202,2029,209,110,1005,1024,4002,1003,2009,1016
,4024,2000,2012,200,206,1029,3017,306,2011,1026,3016,2008,4016,201,302,4013,2005,3008,4018,2004,
3022,1020,2024,3023,2017,2020,3009,2014,1023,1013,208,3000,3007,3020,305,203]---

00:07:27.859 ---crc=[424761fc]---
```

Figure 9: Overview of countries targeted by Ponmocup's decide plug-in

All the discovered information is then encrypted and exfiltrated to a dedicated plug-in proxy, and stored in the back-end, where the operators can query the data for information of interest.

Because the processing of this data happens on back-end servers, FOX-IT does not know the exact purpose of harvesting this type of information. Having access to SIP gateways could be useful for VoIP fraud, but would also allow for the interception of voice/video communication.

In total Fox-IT has identified 25 plug-ins with unique identifiers, among them they share more than 4000 different versions, indicating this framework is under continuous development.

5.3.3 Plug-in #2610 – router scan: collecting router information

This plug-in identifies what gateway IP the victim's machine uses (for example 10.0.0.1) and then attempts to identify if, on that gateway, any services respond on the following ports:

- 22 (SSH)
- 23 (Telnet)
- 80 (HTTP)
- 443 (HTTPS)
- 8080 (Alternative HTTP port, often used by proxies)

Using an example of a typical router with an example page on the gateway IP, this plug-in would exfiltrate the following information:

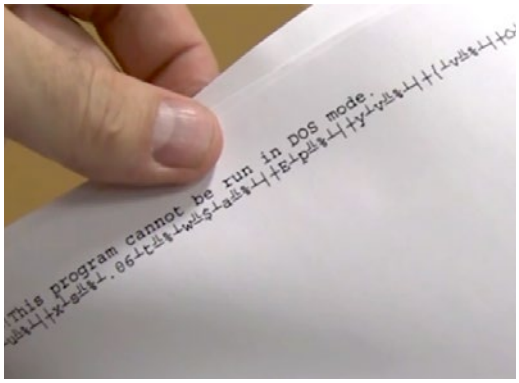

Figure 10: Ponmocup payload turning into print[11] jobs on patched systems

```
00:00:26.485 ---scan_start---
00:00:26.485 ---adapter_start---
00:00:26.485 ---name=[{4AEE80EB-8420-4950-ACDF-9C04A9D530D1}]---
00:00:26.485 ---descr=[AMD PCNET Family PCI Ethernet-adapter]---
00:00:26.485 ---addr=[0a00000d]---
00:00:26.485 ---index=[2]---
00:00:26.485 ---type=[6]---
00:00:26.485 ---dhcpenabled=[1]---
00:00:26.485 ---havewins=[0]---
00:00:26.485 ---leaseobt=[1445466127]---
00:00:26.485 ---leaseexp=[1445469727]---
00:00:26.485 ---IpAddressList(1)---ipaddr=[10.0.0.13]---ipmask=[255.255.255.0]---ctx=[2]---
00:00:26.485 ---GatewayList(1)---ipaddr=[10.0.0.1]---ipmask=[0.0.0.0]---ctx=[0]---
00:00:26.485 ---DhcpServer(1)---ipaddr=[10.0.0.1]---ipmask=[]---ctx=[0]---
00:00:26.485 ---PrimaryWinsServer(1)---ipaddr=[0.0.0.0]---ipmask=[0.0.0.0]---ctx=[0]---
00:00:26.485 ---SecondaryWinsServer(1)---ipaddr=[0.0.0.0]---ipmask=[0.0.0.0]---ctx=[0]---
00:00:26.485 ---adapter_end---
00:00:26.485 ---myhttp_start---server=[10.0.0.1]---url=[/]---
00:00:41.641 ---connect_error=[14]---
00:00:41.641 ---myhttp_end---
00:00:41.641 ---http_start---server=[10.0.0.1]---url=[/]---
00:00:43.063 ---reply=[1]---reply_code=[200]---reply_len=[113]---
00:00:43.063 ---reply_data_start---len=[113]---
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<title>Example Router Page</title>
</html>
00:00:43.063 ---http_end---
```

Additionally the plug-in will attempt to access the web-server on port 80 via the /admin URL, and exfiltrates the server's response. In the example below the /admin does not exist:

```
00:00:43.063 ---myhttp_start---
     server=[10.0.0.1]---url=[/admin]---
00:00:58.063 ---connect_error=[14]---
00:00:58.063 ---myhttp_end---
00:00:58.063 ---telnet_start---
     addr=[10.0.0.1:23]---
00:01:13.063 ---connect_error=[14]---
00:01:13.063 ---telnet_end---
00:01:19.172 ---discover_ok---
00:01:19.172 ---devcount=[1]---
00:01:19.188 ---valid_igd=
     [http://10.0.0.1:80/
     WANIPConnection]---
00:01:19.188 ---addr=[10.0.0.19]---
00:01:19.188 ---ext_ip=[0.0.0.0]---
00:01:19.188 ---time1=[1445467515]---
00:01:19.188 ---gtc1=[1573203]---
00:01:19.188 ---scan_end---
00:01:53.000 ---crc=[3aea52f2]---
```

Because the processing of this data happens on back-end servers, FOX-IT does not know the exact purpose of harvesting this type of information. Storing information on router devices could help in further propagating Ponmocup through the local network, but could also aid in identifying interesting targets.

5.3.4 Plug-in for MS10-061 vulnerability: lateral movement

12 June 2012, the day of the renewed Ponmocup release: a plug-in containing an exploit for the vulnerability in the Microsoft print spooler service [10], first used as a zero-day by Stuxnet two years earlier, is added to the group of plug-ins that are pushed to every newly infected machine, a move the operators would soon come to regret.

By impersonating the print spooler service Ponmocup could move its payload into the system directory and execute it with system privileges on unpatched Windows machines. This effectively made Ponmocup a worm and the amount of infected machines increased significantly.

A plug-in containing an exploit for the vulnerability in the Microsoft print spooler service, first used as a zero-day by Stuxnet two years earlier.

However when a Windows system was patched for the vulnerability, the payload would not be written to the system directory, but instead, to the default print spooler folder; the directory used by Windows to queue printer jobs. The print spooler service then saw the Ponmocup binary as a print job, and continued to send it to the printer, which then carried on to print the binary. Because the exploit is attempted multiple times, the printers usually just kept on printing, until they ran out of either paper or ink.

Once companies all over the world started complaining about their printers printing, what was presumably, 'garbage data', the story was soon picked up by the media, something the Ponmocup operators picked up on as well, as the plug-in was removed the very next day. To avoid even more attention the plug-in was removed from the default plug-in list and was never re-used as a separate plug-in again.

10 *https://support.microsoft.com/en-us/kb/2347290*

11 *http://www.symantec.com/connect/blogs/printer-madness-w32printlove-video*

6 Command and control traffic

All Ponmocup components that can communicate with command and control servers contain hardcoded domains. These domains are not command and control servers and are not used to send data to, but are merely used to calculate the IP addresses of the actual command and control servers.

To achieve this, Ponmocup resolves the hardcoded domain and converts the returned IP into a hex value. It then takes the CRC32 checksum of the domain and XORs these two values with each other, recovering the real command and control IP address.
This is visualized in Figure 11. In the example above the main module contains the hardcoded domain 'claims-reference.net'. As this domain has no direct relation to any actual Ponmocup infrastructure, this is a clever method of hiding the real command and control servers from any prying eyes.

Additionally, to avoid detection by intrusion detection and prevention systems, the operators behind Ponmocup have put in a lot of effort to hide its command and control traffic:

- Command and control traffic only occurs once to twice every two days and occurs at random times.
- URLs are randomly structured from a combination of two lists, both containing 50+ unique paths which are commonly used by legitimate websites.
- A command and control server typically returns a "404 Not Found" HTTP response to make it seem as if the page does not exist and isn't accepting any data.
- Data is RC4 encrypted but the key is generated differently per component.
- The encrypted data is stored in the 'Cookie' header.
  - Data is serialized into multiple fields, each Cookie name is randomly chosen out of a list of 177 keywords which are commonly used by legitimate websites.
  - Storing data in the Cookie header using popular cookie keywords is a technique that FOX-IT also observed in the highly sophisticated Regin framework.
- Domains used to calculate command and control IP only use common TLD's (.net, .com, .org).
- Plug-ins use separate proxies and back-ends to which they log exfiltrated data.

Figure 11: Conversion of a hardcoded domain to an actual command and control address

6.1 Installer communication

The installer makes use of one hardcoded domain which is only used during installation. By using a domain for installation purposes only, the other domains used for command and control traffic won't be discovered as easily. This also means that taking down this domain would not impact the current botnet size, but only cause a minor hiccup in new infected machines being able to talk back to the Ponmocup backend. This one time has only been changed a couple of times; 'fasternation.net' has been the static installation domain since 2012.

Data is RC4 encrypted using a random 16 bytes key and encoded with a custom base64 table.
Before base64 encoding, Ponmocup first places the RC4 key on top of the data, calculates the CRC32 of the data and places this value at the end of data. After decryption, every block of data has the CRC32 value of that block at the end (last 4 bytes)[15].

A typical Ponmocup check-in would look something like this:

```
Cookie: uid=referringpath=0tMNPEubad4AlBaw4aZNqYj-PphJ-NCN5PRz4F_zJTxwoqhGOhK; clogid=
mKteiILceVXJAwJ88VdDXdzjTq2zhM2HZ5kR_c_3i111MXafapXw-DLKkNvxeOqHmjfhdz4opTIoVT410xANf-
SER58Th1CP; ARSiteUser=server=UiZgtrfSoQ5zfIjHnd88G_iU4NwOFtRMT-y0m; _
thepoint=czAc0MvfrFF3sVd2vehRHAo1ZASIGD3a1PCoidT0IKjiPkjNYPB_b2lr1ntEDYk88NS1Cpc4ffDckKGtsA-
IiZ6XAkSGH0O4oXStZfojpjcSR-46IJja-cK7N9meCk1im6xTB1ocayDmZQ_yV3LKgjxK54s8EB38AQdddE6RsLx-5_
oTv8KAVlOQ0o7IWRx2ffWe6shtDPjYhYUwgTRHtFTN5_yF0T72wfLj7dflSDLUBuSKgABWqVuu0luXDwJII50zxKxbWYJ
D3n1mOOCD6u-VZbjxr0USzySQjVPEIj6CxfGcvD1nzwvX2I8r6id5EAnukW3ckSmojFCYgxdmWe_Iij7KhghqIzZmMfZ
Vmw9wrRGPrsYgkvvrn-PW_7JyEqfp7h7WjlgSuYmciCO43vKuL9VR84BS19KR5hLGntvsd5dIZQsD
```

Decrypting Ponmocup cookies, such as the one above, typically result into the following fields:

Field	Meaning
PIN	A PIN is a group of plug-ins installed on the machine. At least one PIN is pushed to every client per default.
User ID	A uniquely generated ID per system
System language	Language of operating system
Sent plug-ins	Plug-ins that have been ran on the victim machine thus far
Plug-in count	Amount of plug-ins installed on top of the default PIN
Operating system	Operating system version
Service pack build	Service pack version
Administrator privileges	Is Ponmocup running with administrative privileges?
Service	Is Ponmocup installed as a service?
Virtual Machine information	Fingerprint which consists out of the anti-virtualization checks
Connection information	Fingerprint with connection information such as the usage of a proxy
UPnP	Is Universal Plug and Play enabled?
CRC32 creation date system directory	CRC32 checksum of the creation date of the system directory
CRC32 creation date System Volume Information	CRC32 checksum of the creation date of the System Volume Information directory
CRC32 of System Volume Number volume hash	CRC32 checksum of the System Volume Number
Q2 resolving information	The domains the bot has used to calculate a proxy IP, including their CRC32 checksums

Table 5: Common fields stored in the Cookie header during command and control traffic

6.2 Main module communication

For continuous communication four hardcoded domains are, again, used to calculate the IP of the command and control server. Ponmocup resolves two domains at a time, the first calculated IP is used to send data to, the second is resolved as a back-up in case the first IP cannot be reached. To prevent the botnet from being taken down, two of the four hardcoded domains are changed every periodically[12] by updating the main module, as shown in Figure 12. By regularly rotating these domains for the past 4 years, all the Ponmocup bots are now load balanced between more than 10 domains.

To take down the botnet, one would had to have analyzed every Ponmocup payload, in intervals of 2–6 months, for 4 years straight; a time consuming task given the complexity of the malware.

The encryption routine used by the main module for command and control traffic is identical to the installer.

6.3 Plug-in communication

Plug-ins capable of communicating with command and control servers use dedicated domains which calculate IP addresses for proxies that log to separate back-end servers.

Although all plug-ins RC4 encrypt (exfiltrated) data, the RC4 key can differ. The cookie header is still used by plug-ins to exfiltrate basic information such as counters or checksums, but the exfiltration of larger data is done via (encrypted) POST requests.

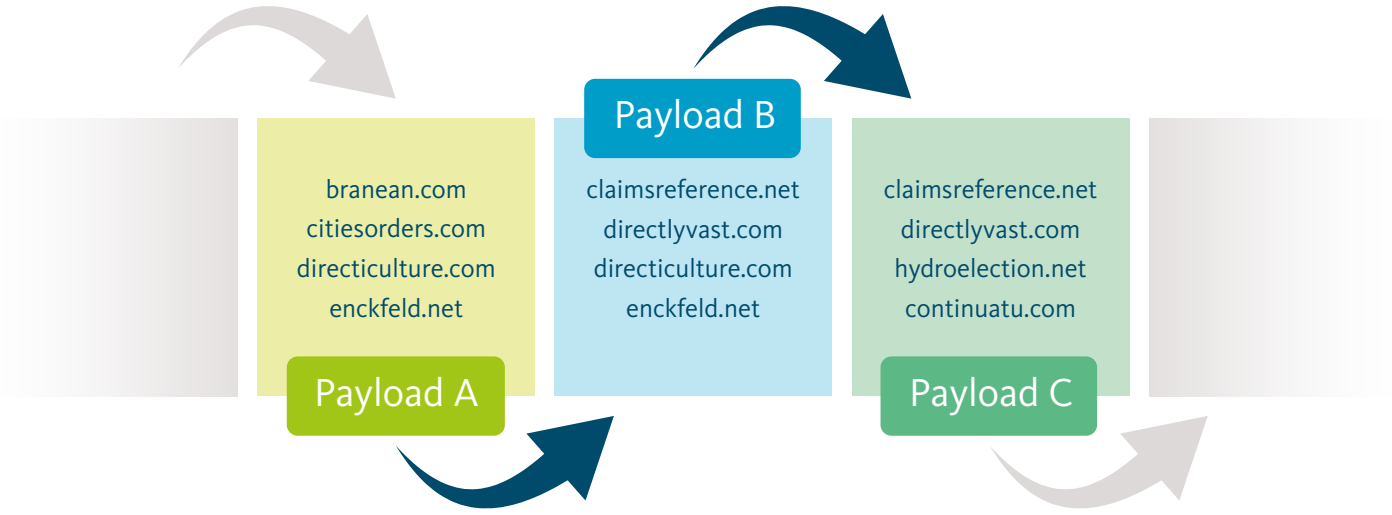

Figure 12: Separation of domains per payload

12 *In most cases we've observed domains rotating in intervals of 3–6 months*

7 Anti-analysis techniques

One of the reasons Ponmocup has been able to stay under the radar, for as long as it has, is related to the different methods the malware uses to thwart analysis attempts.

By specifically and heuristically checking for network and host based analysis tools, debuggers and virtualized environments, and then delivering a fake payload, the operators aim to prevent their malware from being detected by the security industry.

7.1 Checks for signs of analysis

Ponmocup's anti-analysis checks are performed during installation, but most Ponmocup components perform anti-analysis checks separately. The main module, for example, performs anti-analysis checks every time a victim's machine is rebooted, and plug-ins perform similar checks individually when they are executed. The table below lists a number of anti-analysis checks, some of which are less common:

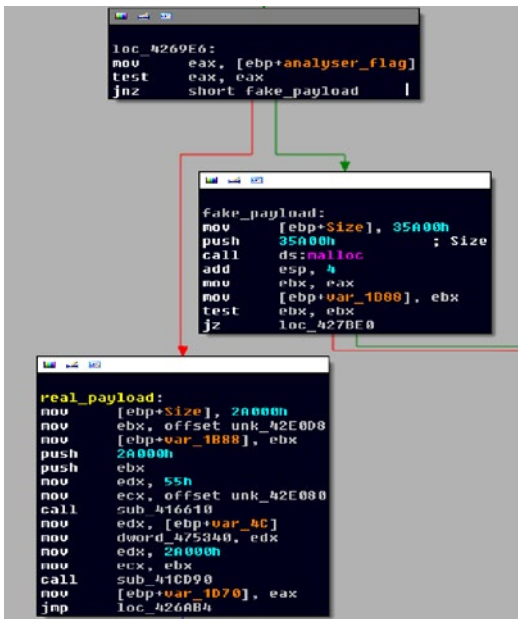

Figure 13: How the analysis flag decides between the installation of the real or fake malware

7.2 Delivery of fake payload

If one of the anti-analysis checks triggers, implying an attempt to analyze the malware, a flag is set and Ponmocup goes on to use one of its most clever tricks; delivery of a fake payload, as explained in Figure 13.

Where typical malware employing anti-analysis immediately exits if being analyzed, Ponmocup installs SanctionedMedia; a pay per install software bundle, which merely injects advertisements into webpages, commonly classified as adware. Because an analyst still observes an actual payload being dropped, the fake malware sample will often be analyzed. As this fake payload does nothing more than inject advertisements and is relatively easy to remove, it will generally not be of much interest to analysts or anti-virus companies. This fake payload is a simple, yet highly effective, disguise for a payload that poses a far more serious threat.

The fake payload is also installed onto a system in a far more obvious manner than the real payload, appearing in the process list as an exe, of which the name is derived from a random file in the system directory with 2–3 random characters appended, with the file description 'RecSave', product name 'MyPCProtect' and original file name 'Smad.exe'. This payload is written in .NET, as opposed to the traditional C++ used in the actual Ponmocup framework.

This fake payload is a simple, yet highly effective, disguise for a payload that poses a far more serious threat.

Figure 14: Homepage of SanctionedMedia.com, the fake payload used as a disguise by Ponmocup

Evasion method	Applicable to:
Blacklisted usernames	Currentuser, Sandbox, Honey, Vmware, Nepenthes, Snort, Andy, roo
Blacklisted computer names (Anubis)	TU-4NH09SMCG1HC, InsideTm
Blacklisted processes	vmware, vmount2, vmusrvc, vmsrvc, VBoxService, vboxtray, xenservice, joeboxserver, joeboxcontrol, wireshark, sniff_hit, sysAnalyzer, filemon, procexp, procmon, regmon, autoruns, atcp2log., awpta., EHSniffer., HTTP Sniffer, EtherD., geturl., HttpAnalyzer, InjectWinSock, HTTPDebugger
Blacklisted services	vmicheartbeat, vmicvss, vmicshutdown, vmicexchange, vmci, vmdebug, vmmouse, VMTools, VMMEMCTL, vmware, vmx86, vpcbus, vpc-s3, vpcuhub, msvmmouf, VBoxMouse, VBoxGuest, VBoxGuest, VBoxSF, xenevtchn, xennet, xennet6, xensvc, xenvdb
Blacklisted drivers	hgfs.sys, vmhgfs.sys, prleth.sys, prlfs.sys, prlmouse.sys, prlvideo.sys, prl_pv32.sys, vpc-s3.sys, vmsrvc.sys, vmx86.sys, vmnet.sys
Blacklisted Product ID's related to sandboxes	• Anubis (76487-337-8429955-22614) • Joe Box (55274-640-2673064-23950) • CWSandbox (76487-644-3177037-23510)
Installed software names in registry	Hyper-V, VirtualMachine
Hardware description	Vbox
VMware guest to host communication channel	A check done by executing the "IN" (x86) assembly instruction with the parameter 0x564D5868 (VMXh) to connect to the VMWare I/O port
Screen resolution, color depth (amount of colors in a single pixel), and additional monitor checks	Virtual environments often lack an actual monitor/screen or have a default resolution (e.g. 800x600).
Number of recently opened documents	A check in the registry if the machine has had at least opened 10 files.
Number of URLs in browser history	A check in the registry if there are at least 10 URLs in the browser history.
Installed software	A check in the registry if at least one software package has been installed.
Mouse movement	Consecutive calls to the 'GetCursorPos' API to determine if victim is moving the mouse.
Banned system fingerprint	If Ponmocup catches an analyst, the fingerprint of the system will immediately be blacklisted, after which this specific fingerprint and corresponding IP can never get infected with the malware again.
Banned IP	Ponmocup blacklists more than a 1000 IP ranges. Some of these ranges have been put on this blacklist preemptively and some have been blacklisted because they were observed analyzing the malware. The IP ranges on this blacklist belong to anti-virus and threat intelligence companies as well as large banks in various countries.

Table 6: List of evasion methods

Appendix I

Targeted keywords

The following is a list of keywords that the operators of Ponmocup deemed interesting. The keywords are used by the 14xx plug-in range (decide), as explained in paragraph 5.3.1.

achCreate	bancoherrero.com
/bb/logon	bankatlantic.web-cashplus.com
/bbw/	bankbahamasonline.com
/business/login	bankinter.com
/business/online/	bankline.natwest.com
/BusinessAppsHome.faces	bankline.rbs.com
/cb/servlet/cb/	bankline.ulsterbank.co.uk
/clkccm/	banklink.com
/cmachid.r	bankofamerica.com/smallbusiness
/cmserver/	banqueprivee1818.com
/cmwire	bbva.es
/Common/Admin	bbvanetoffice.com
/createWire	bbw/LogonStateMachineServlet.mibs
/cs70_banking/	blilk.com
/customerlink/	boursedirect.fr
/direct.bankofamerica.com	boursorama.com
/ebc_ebc1961/	boursorama.com
/fxim	business.co-operativebank.co.uk
/hbcash.exe/	business.hsbc.co.uk
/ibcorporate	business.netbankerplus.com
/IBWS/	business.santander.co.uk
/icm1/	businessaccess.citibank.citigroup.com
/icm2/	businessbanking.
/inets/	businessclassonline.
/onlineserv/cm/	business-eb.ibanking-services.com
/phcp/servlet/	businessonline.
/RsaGoIdAuthentication.aspx	businessonline.huntington.com/
/sbuser/	businessportal.mibank.com
/wcmfd/wcmpw/	bxs.com
/webcm/	caixacatalunya.com
/wire/confirm	caixacatalunya.es
/wireapproval	cajacanarias.es
/wireinitiation	cajamar.es
/wireManager	CashMgmt
/wiretransaction	cashmgt
/WireTransfer	cashmgt.firsttennessee.biz/
access.jpmorgan.com	Cashplus
adminamps.53.com	Cashplus
advisorchannel.com	cashproonline.bankofamerica.com
ameritrade.com	cmbnv.com

cmol.bbt.com
cmserver/login_validate.cfm
commercial.hsbc.com.hk
commercial.wamu.com
Compassconnect.compassbank.com
cortalconsors.fr
credentialdirect.com
deutsche-bank.es
direct.53.com/logon53Direct.jsp
directnet.com
dmv
dmv.org
drivingrecords.com
ebanking-services.com
enternetbank.com/exact4web/
eregal.com
etrade.com
exact4web
express.53.com
fast-trade.com
fortuneo.fr
fundsxpress.com
fxpayments.americanexpress.com
geico.com
goldleafach.com
gpsmoneymanager
hnnconhsvraps01
home1.cybusinessonline.co.uk
infoplus.
inteligator.com
internetbanking.unfcu.org
invest.ameritrade.com
itreasury.regions.com/phcp/
servlet/CustomerLoginServlet
lcl.fr
libertymutualbusinessdirect.com
linebourse.fr
lloydslink.online.lloydsbank.com
logincm.aspx
lppolice.com
metrobankdirect.com/corporate.asp
mfasa.chase.com
mybusinessbank.co.uk
myvirtualmerchant.com
nordnet.lu
nordnet.no
nordnet.se
nwolb.com
olb.ent.com
online.citibank.com
online-business.bankofscotland.co.uk
PassmarkSignIn.faces
pcsbanking.net
pcs-sd.net
phxrs-opera
quickbooks.com
rbcdain.automatedfinancial.com
rbsdigital.com
risk.nexis.com
sabadellatlantico.com
safe.bankofamerica.com
sanostra.es
scotiaitrade.com
secure.bankofamerica.com
selfbank.es
siebertnet.com
singlepoint.usbank.com
sitekey.bankofamerica.com
sitibusiness.citibank.com
srvc1.jpmorgan.com
srvc2.jpmorgan.com
ss2.experian.com
streetscape.com
svbconnect.com
tioexpress.com
trade.loginandtrade.com
trademonster.com
treas-mgt.frostbank.com
treasury.pncbank.com
treasurypathways.com
ulsterbank.co.uk
ulsterbankanytimebanking.co.uk
us.etrade.com
usaa.com
wblnk
wcmfd/wcmpw
web2.westlaw.com
web-access.com
webach
webcmpr.
webinfocus.
webinfocus.mandtbank.com
weblink.websterbank.com
websteronline.com
wellsoffice.wellsfargo.com
whitneybusinessnetwork.whitneybank.com
www.signatureny.web-access.com
www.treasury.pncbank.com
www2.citizensbankmoneymanagergps.com
www8.comerica.com

Appendix II

Network based indicators of compromise

Domains

The following domains appear hardcoded in Ponmocup instances and are used for C&C IP calculation, as described in paragraph 6.1.

abccornet.com	dogmationation.com	piclbumestream.com
adertisecorp.com	dynodns.org	postdone.com
affilipcorp.com	enckfeld.net	ratilovskoye.com
anexcorp.org	familyinteresting.com	recising.com
britishfederal.org	fasternation.net	searchforthat.net
changinessmen.com	freewayreg.com	sectionsfear.com
claimsreference.net	headedpicked.com	separtila.com
clickoptimiser.net	headedpicked.net	standardbay.net
contentdeliveryorg.net	highlytraditional.org	streamingadv.com
contextexpert.org	himmeding.com	ternations.com
continuatu.com	howeveraged.net	thomaslaid.net
culminaccessful.com	hydroelection.net	traffictradexpert.com
cybernan.net	illegedly.com	twicecitizens.com
defenciclovis.com	imagesharehost.com	veristats.net
descriptioned.com	leadwriting.com	virtualsearches.com
detroportans.com	meetinglimited.com	workerssan.net
directiculture.com	netdiscovery.org	yaltimate.com
directlyvast.com	picasootoolbar.com	

Resolving IP's

The following IPs are pointed to by the hardcoded domains listed above, as explained in chapter 6.

109.74.195.149	155.83.123.22	2.171.234.238
243.182.100.227	44.36.245.224	50.116.56.144
4.227.70.65	168.23.171.69	102.209.206.89
63.77.106.1	204.37.98.202	7.34.116.64
166.178.113.144	253.101.238.123	38.155.216.69
231.150.98.137	94.75.201.33	27.251.60.63
31.171.130.249	40.22.124.164	158.76.160.100
85.66.23.125	49.197.32.49	100.134.242.235
6.88.25.80	104.127.201.198	124.3.139.20
80.213.59.50	144.61.46.13	25.20.33.76
222.219.85.79	203.136.214.219	189.140.10.37
234.102.81.206	253.134.178.81	59.228.144.104
116.181.5.61	106.8.16.175	204.11.56.48
156.44.195.200	204.11.56.48	29.205.223.64
21.8.194.15	41.252.243.242	94.75.201.33
42.107.140.147	151.225.26.181	118.15.53.129
199.172.52.66	106.110.29.248	22.149.159.105
227.248.14.79	114.225.99.185	

IPs used for Command and Control traffic

The following IPs are actually used for command and control traffic, as explained in chapter 6.

182.62.211.45	26.252.164.23	85.17.133.194
185.17.184.249	28.16.103.211	89.172.227.240
214.66.10.71	62.212.68.230	93.115.88.220
217.23.3.243	78.109.28.248	95.211.240.193
217.23.3.244	78.109.28.249	95.211.240.194
217.23.3.249	78.109.28.250	
232.187.207.67	85.17.133.193	

Snort signatures

Typical Ponmocup command and control traffic, as explained in paragraph 6.1.

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"FOX-SRT - Trojan - Ponmocup
HTTP Request (generic)"; flow:established,to_server; content:"Accept: */*|0d 0a|";fast_
pattern;http_header; content:"Pragma|3a| no-cache|0d 0a|";http_header; content:"Cache-
Control|3a| no-cache|0d 0a|";http_header; content:!"Referer|3a|";http_header;
content:"Cookie|3a| ";http_header; pcre:"/^Host\x3A[^\r\n]+?\d{1,3}\x2e\d{1,3}\x2e\d{1,3}\
x2e\d{1,3}\r\n/Hm"; content:!"Accept-Encoding|3a| ";http_header; content:!"Accept-Language|3a|
";http_header; content:!"Content-Type|3a| ";http_header; reference:url,http://blog.Fox-IT.
com/2015/12/02/ponmocup-a-giant-hiding-in-the-shadows; threshold:type limit, track by_src,
count 1, seconds 600; classtype:trojan-activity; priority:1; sid:21001533; rev:1;)

alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"FOX-SRT - Trojan - Ponmocup
plugin-specific check-in"; content:"GET"; http_method; content:"HTTP/1.1|0d0a|Accept: */*";
distance:0; content:"Content-Type: application/x-www-form-urlencoded"; fast_pattern;
distance:0; pcre:"/Host: ([0-9]{1,3}\.){3}[0-9]{1,3}\x0d/"; distance:0; content:"User-Agent:
Mozilla/4."; distance:0; content:"Cookie: "; pcre:"/Cookie: [a-z0-9]{1,10}=[a-z0-9+/]{20,500}
(=){0,2}/i"; distance:0; urilen:<50,norm; content:!"Referer"; threshold:type limit, track
by_src, count 1, seconds 600; classtype:trojan-activity; reference:url,http://blog.Fox-IT.
com/2015/12/02/ponmocup-a-giant-hiding-in-the-shadows; sid:21001686; rev:1;)
```

Scanning activity of plug-in #2600, as explained in paragraph 5.3.2.

```
alert udp $HOME_NET $SIP_PORTS -> any any (msg:"FOX-SRT - Trojan - Ponmocup plugin #2600
(SIP scanner)"; content:"User-Agent|3a| Zoiper for Windows rev.1812|0d0a|"; threshold: type
limit, count 1, seconds 3600, track by_src; reference:url,http://blog.Fox-IT.com/2015/12/02/
ponmocup-a-giant-hiding-in-the-shadows; sid:21001493; classtype:trojan-activity; rev:1;)
```

Appendix III

Host based indicators of compromise

YARA signature

The following YARA signature can be used to scan for Ponmocup plug-ins in memory. This is based on the content of the PE headers, as explained in chapter 5.3

```
rule Ponmocup : plugins
{
        meta:
                description = "Ponmocup plugin detection (memory)"
                author = "Danny Heppener, Fox-IT"
        strings:
                $1100 = {4D 5A 90 [29] 4C 04}
                $1201 = {4D 5A 90 [29] B1 04}
                $1300 = {4D 5A 90 [29] 14 05}
                $1350 = {4D 5A 90 [29] 46 05}
                $1400 = {4D 5A 90 [29] 78 05}
                $1402 = {4D 5A 90 [29] 7A 05}
                $1403 = {4D 5A 90 [29] 7B 05}
                $1404 = {4D 5A 90 [29] 7C 05}
                $1405 = {4D 5A 90 [29] 7D 05}
                $1406 = {4D 5A 90 [29] 7E 05}
                $1500 = {4D 5A 90 [29] DC 05}
                $1501 = {4D 5A 90 [29] DD 05}
                $1502 = {4D 5A 90 [29] DE 05}
                $1505 = {4D 5A 90 [29] E1 05}
                $1506 = {4D 5A 90 [29] E2 05}
                $1507 = {4D 5A 90 [29] E3 05}
                $1508 = {4D 5A 90 [29] E4 05}
                $1509 = {4D 5A 90 [29] E5 05}
                $1510 = {4D 5A 90 [29] E6 05}
                $1511 = {4D 5A 90 [29] E7 05}
                $1512 = {4D 5A 90 [29] E8 05}
                $1600 = {4D 5A 90 [29] 40 06}
                $1601 = {4D 5A 90 [29] 41 06}
                $1700 = {4D 5A 90 [29] A4 06}
                $1800 = {4D 5A 90 [29] 08 07}
                $1801 = {4D 5A 90 [29] 09 07}
                $1802 = {4D 5A 90 [29] 0A 07}
                $1803 = {4D 5A 90 [29] 0B 07}
                $2001 = {4D 5A 90 [29] D1 07}
                $2002 = {4D 5A 90 [29] D2 07}
                $2003 = {4D 5A 90 [29] D3 07}
                $2004 = {4D 5A 90 [29] D4 07}
                $2500 = {4D 5A 90 [29] C4 09}
                $2501 = {4D 5A 90 [29] C5 09}
                $2550 = {4D 5A 90 [29] F6 09}
                $2600 = {4D 5A 90 [29] 28 0A}
                $2610 = {4D 5A 90 [29] 32 0A}
                $2700 = {4D 5A 90 [29] 8C 0A}
                $2701 = {4D 5A 90 [29] 8D 0A}
                $2750 = {4D 5A 90 [29] BE 0A}
                $2760 = {4D 5A 90 [29] C8 0A}
                $2810 = {4D 5A 90 [29] FA 0A}
        condition:
                any of ($1100,$1201,$1300,$1350,$1400,$1402,$1403,$1404,$1405,$1406,
$1500,$1501,$1502,$1505,$1506,$1507,$1508,$1509,$1510,$1511,$1512,$1600,$1601,$1700,$1800,$1801,
$1802,$1803,$2001,$2002,$2003,$2004,$2500,$2501,$2550,$2600,$2610,$2700,$2701,$2750,$2760,$2810)
}
```

Registry

Registry keys used by the Ponmocup framework for information storage.

```
HKEY_CURRENT_USER\Software\Microsoft\Multimedia\1
HKEY_CURRENT_USER\Software\Microsoft\Multimedia\2
HKEY_CURRENT_USER\Software\Microsoft\Multimedia\3
HKEY_CURRENT_USER\Software\Microsoft\Multimedia\4
HKEY_CURRENT_USER\Software\Microsoft\Multimedia\5
HKEY_CURRENT_USER\Software\Microsoft\Multimedia\6
HKEY_CURRENT_USER\Software\Microsoft\Multimedia\7
HKEY_CURRENT_USER\Software\Microsoft\Multimedia\8
HKEY_CURRENT_USER\Software\Microsoft\Multimedia\9
HKEY_CURRENT_USER\Software\Microsoft\Multimedia\10
HKEY_CURRENT_USER\Software\Microsoft\Multimedia\11
```

FOX-IT

- Was founded in 1999.
- Established one of the first Cyber Security Operations Centers in Europe.
- Is Europe's largest specialized cyber security company.
- Operates in three business areas:
  1. Cyber Threat Management: a solution portfolio aimed at reducing the risks of cyber threats, and includes: professional services, managed security services, and technology;
  2. Web and Mobile event analytics: a solution portfolio that is aimed at reducing financial risks in (online) payment transactions;
  3. High Assurance: solutions that make trusted communication possible to the highest classification levels.
- Has been involved in many high-profile Incident Response cases. Most of the cases we worked on are secret. An approved selection can be shared upon request.

FOX IT
FOR A MORE SECURE SOCIETY
part of nccgroup

Olof Palmestraat 6, Delft
PO box 638, 2600 AP Delft
The Netherlands

T +31 (0) 15 284 79 58
F +31 (0) 15 284 79 90
www.fox-it.com

IBAN NL57ABNA0554697041
KVK Haaglanden 27301624

602-001-EN