WHITEPAPER: SYMANTEC SECURITY RESPONSE

Peerbot: Catch me if you can

Elia Florio and Mircea Ciubotariu
Symantec Security Response, Ireland

Peerbot: Catch me if you can

Contents

When Petar Maymounkov and David Mazières designed the Kademlia protocol[1], they probably didn't imagine that one day it would be used to ensure the livelihood of new generation botnets. Nowadays, botnets are in a state of constant evolution and have progressed in complexity. Just three years ago, the term 'botnet' referred generically to a collection of IRC Trojans; today the term could be used merely to describe the sophistication of modern networks of malicious bots.

Malware and peer-to-peer

Research has shown that botnet development is currently proceeding in two different areas. One area of development involves the design of new bot functionalities. Malware writers continue to add new code to their bots to make them faster in propagation and invisible on the system. While older bots were created to perform distributed denial of service attacks, the new generation of bots can also send image spam, gather email addresses, make search queries on Google, log keystrokes, steal passwords and upgrade their components.

The other area of botnet development is in the design of new command and control (C&C) strategies, which is a game played at network level. A bot without control is useless, and controllers are looking for more intelligent strategies than standard IRC in order to administer their creatures without being caught. Decentralized and distributed networks, such as peer-to-peer (P2P) networks, are perfect for this purpose.

In 2006, W32.Nugache@mm represented the first concrete effort to build a malicious P2P network over TCP port 8. However, Nugache was designed with a minor flaw: the list of initial peers was hard-coded in the threat and limited to 22 servers, so it wasn't a real decentralized P2P network. But the idea was innovative, and researchers expected the next 'PeerBot' to appear soon afterwards.
In the first months of 2007 Trojan.Peacomm (a.k.a. the 'Storm Trojan') confirmed the trend and showed how legitimate P2P protocols can be used effectively to coordinate virus networks.

The 'Storm Trojan' attack

The new year's spam attack started on January 18, 2007 and was reiterated on January 21 and again later. Millions of emails were spammed to legitimate accounts with an executable attachment which turned out to be a Trojan dubbed 'Trojan.Peacomm'. It was also referred to as the 'Storm Trojan' due to the fact that some of the subject lines of the emails included news of severe storms that had hit Europe during January.

A previous attack, which occurred in the final weeks of 2006, had also triggered antivirus radars due to an elevated level of spam. This was W32.Mixor.Q@mm, and the outbreak was effective because the threat was spammed using 'postcard.exe' and similar file names.

Many similarities between the Mixor and Peacomm outbreaks led antivirus researchers to believe that

the same group was behind the two incidents. Both of the attacks were, in fact, efforts to build a wide and distributed network of compromised computers running different types of Trojans.

The set of malicious files downloaded by Mixor in 2006 included spam and mail-proxy Trojans. In the new year this initial set was enriched by new Trojans including a distributed denial of service module, a rootkit, and the peer-to-peer client.

The polymorphic packer

Peacomm, as well as all the components related to it, makes use of an improved version of the infamous packer Tibs. Some antivirus engines already detect most of the executables packed with Tibs, purely because they include detection for the packer itself.

Drawing an analogy with polymorphic viruses, the equivalent of the polymorphic decryptor in this case is the unpacking code, while the viral body equivalent is the original malicious executable. The analogy is also valid for detection: in some complex polymorphic viruses, detection relies on recognizing the polymorphic decryptor generically; in the same way, detection of Peacomm is possible by detecting the packer pattern.

Tibs by itself is nothing more than a regular packer, although its authors have put a great deal of effort into keeping it undetectable by antivirus engines. Basically, the packer adds a new section to the executable and replaces a few bytes from the entry point with a polymorphic decryptor that will restore the original executable code and data and pass control to its entry point.

In its early days Tibs used to encrypt the original executable using a simple function that was very easily bypassed using basic cryptanalysis. Subsequently, its authors decided to change the encryption algorithm to Tiny Encryption Algorithm (TEA), which gives much better protection against cryptanalysis. For further protection, older variants made heavy use of MMX and FPU instructions, in an attempt to break the emulators, since these kinds of instructions are used only in specialized applications. Recent variants make use of 'exotic' APIs such as User32!DdeQueryConvInfo, in order to trick emulators and virtual machines – which tend to stop emulation when encountering such unsupported APIs.

New, different executables are spread with the same functionality every once in a while, in the hopes that there won't be a signature-based definition recognizing the newly created files. Even though these executables look different every time, the original packed code and data and their functionality do not change unless there are changes in the source code. By using this technique, the authors ensure a pretty good chance of evading detection that relies on specific signature recognition, while the cost of establishing and maintaining such a system is minimal (i.e. given the packer, a small script could do the job in no time). The files can be refreshed as often as every download, but it has been noted that

timed intervals are preferred (for instance, the executables may be repacked every hour).

Mixor family: A strange file infector

Unfortunately, the source code for the Mixor virus has been widely available on the web since 2006. In fact, it is known as an 'open source' virus. It was released in March 2006 as 'X-Worm', a proof-of-concept virus for an underground magazine.

Mixor is a polymorphic file-infector virus with mass-mailing capabilities. The virus is also designed to carry a secondary executable file as payload, so it is the perfect threat for integration with any external Trojan or backdoor code.

The original source code of the X-Worm was modified to create new versions of the virus, which initially incorporated Trojan.Galapoper.A and later Trojan.Peacomm. In addition, the original X-Worm added a copy of itself to .rar archives, but this part of the code was removed from the variants that have been seen in the wild.

One feature that makes the Mixor virus unique is its unusual infection strategy. Traditional file infectors append the viral body at the end of the host and patch the entry-point in order to run the malicious code first. In contrast, Mixor does not append itself to the end of the file while infecting. Instead, it creates a copy of itself in the same folder but with a random file name. Next, Mixor patches the entry-point of the host program by inserting a little shell-code that will run the external copy of the virus. As a result, there will be a secondary file, which is a copy of the pure virus body, for each infected file. Figure 1 shows the differences in the infection techniques.

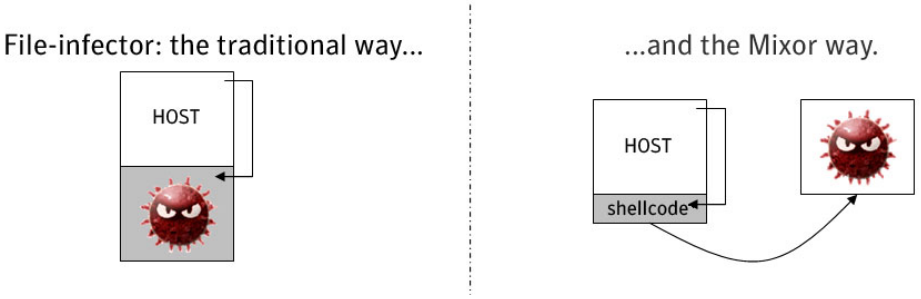

Figure 1: The file infection strategies of traditional file infectors and Mixor.

Peacomm and the P2P network

Peacomm uses a kernel mode payload injector. The Trojan drops the driver wincom32.sys and runs it as a system service. This driver injects a hidden module from the kernel mode into the user mode space of the SERVICES.EXE process via KeAttachProcess and KeInsertQueueApc. The injected executable is the component responsible for all the P2P communications and starts several threads in the SERVICES.EXE process.

The Peacomm driver was also upgraded by the authors with a full set of rootkit functionalities. In

fact, the variants released after January 21 were able to hide files, registry keys and active network connections. The rootkit uses Service Descriptor Table (SDT) hooking to hide files or keys, and hijacks IRP_MJ_DEVICE_CONTROL of '\Device\Tcp' to hide active connections of SERVICES.EXE.

The first Peacomm variant was configured to communicate over UDP port 4000, but peaks reported by network probes in the days immediately after the attack indicated that later variants also used ports 7871 and 11271.

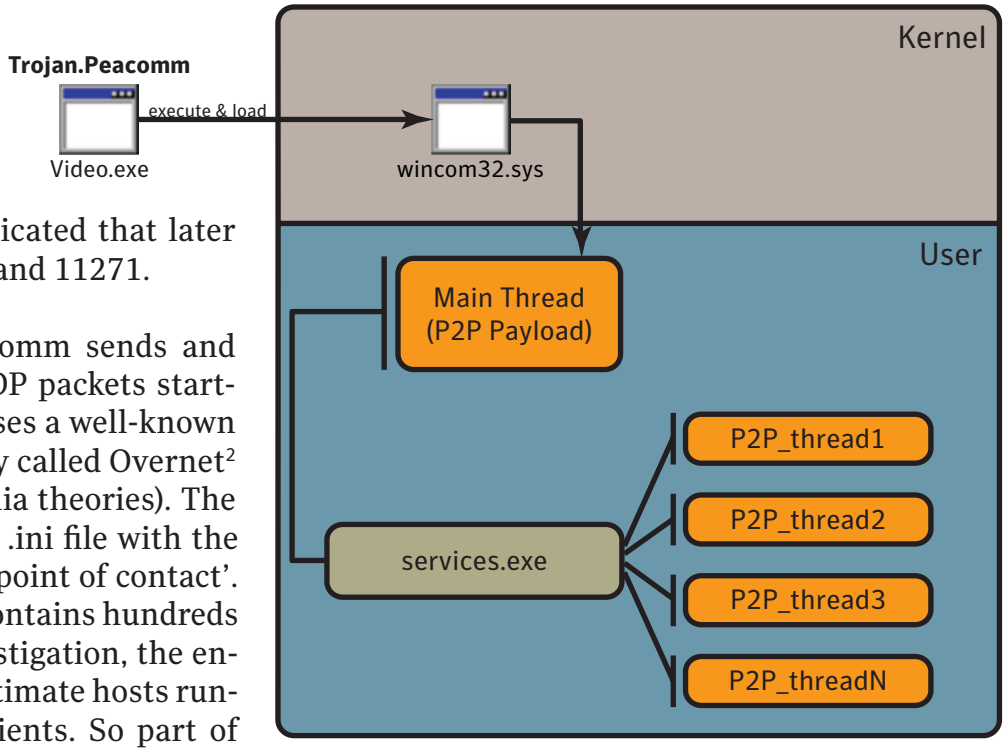

Figure 2: Module injection from kernel mode.

A computer infected by Peacomm sends and receives a large number of UDP packets starting with 0xE3 (227) bytes. It uses a well-known protocol in the P2P community called Overnet[2] (an implementation of Kademlia theories). The Trojan creates a configuration .ini file with the list of P2P hosts used as 'first point of contact'. The peers list is variable and contains hundreds of encoded entries. Upon investigation, the encoded entries proved to be legitimate hosts running Overnet or mlDonkey clients. So part of the peer-to-peer botnet is made up of legitimate

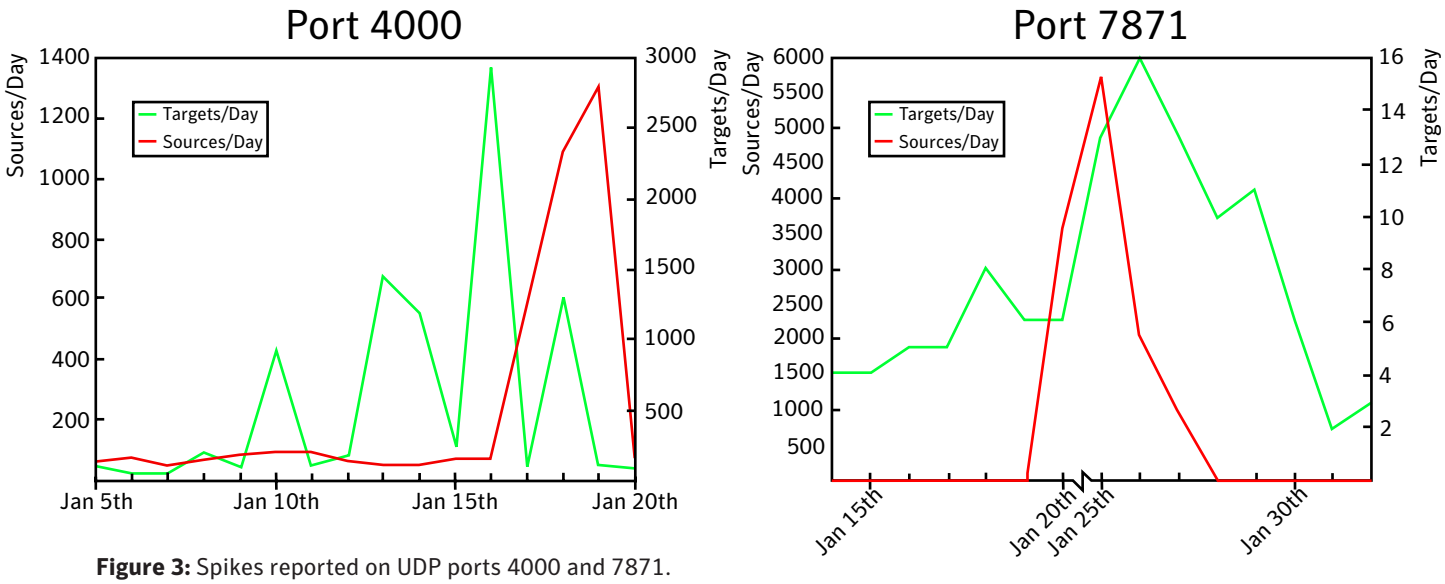

Figure 3: Spikes reported on UDP ports 4000 and 7871.

peers, which (unknowingly) support and propagate the malicious P2P traffic generated by Peacomm.

C&C over Overnet

Just as the W32.Spybot family of backdoors build up a botnet by making use of the IRC communication channels to retrieve their commands, Peacomm uses its own P2P network to retrieve information relating to which files to download and execute.

The counter section in peers.ini and in wincom32.ini for the most recent variants, denotes the current state of downloads from the network. The initial setting counter=0 specifies that no files have been downloaded yet. Using the actual counter value Peacomm computes an encrypted 16-byte search string or hash, which additionally contains checksum information that validates the search and time information, and a random part that makes the search string look different.

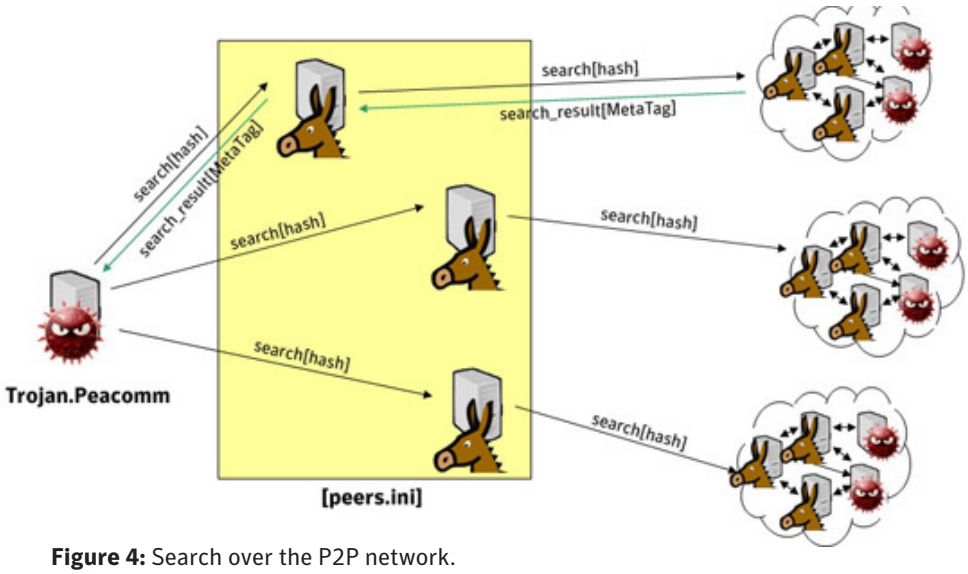

Figure 4: Search over the P2P network.

The search command for this hash uses a custom search type (0x14) and expects to retrieve an encrypted tag string as seen in the following capture:

```
User Datagram Protocol, Src Port: 4665 (4665), Dst Port: 7871 (7871)
Protocol: eDonkey (0xe3)
Message Type: Search Result (0x11)
Hash: B225564021F1B55C35FB8DA9950A6678 (search hash)
Hash: 05B3D57C0C90A3010000000000000000
Meta Tag List Size: 1
eDonkey Meta Tag
Meta Tag Type: 0x02 (TAGTYPE_STRING)
Meta Tag Name Size: 2
Meta Tag Name: id
String Length: 86
String: 6%m[f7/$'$1vo$e:9)n"!mq2[\,;jc+!2zk*g5&<
p$1cdvn"(0c="a4;xd^j'v)!,[_,'^[%"%184qh88dj'!!
```

The original string is encrypted with RSA[3] on 64-bit blocks. It is then encoded using a custom base64 algorithm that has an additional layer of light encryption which, in fact, only changes the table of translation for the base64 encoding.

In order to decrypt the string, one would need the private key pair (d, n). By analysing the Trojan's code one could easily identify the d component of the key, as it is hard-coded in plain text in the executable's data section. However, there is no sign of the modulus n.

Further analysis revealed that n is actually taken from the search result packet. More specifically, it is the second hash value, immediately after the search hash.

So far it has been observed that the private key (d, n) has been constant for different releases of Peacomm variants and it has the following value: (0x025F2D1619EF1ABD, 0x01A3900C7CD5B305).

Using the information above we can decrypt the given string and get the following structure:

```
Xor = 0x3B
Add = 0xAD
Counter = 0x5300
String = "205.209.179.112/game0.exe"
```

'Xor' and 'Add' are byte control check sums computed with the two named operations on the rest of the data. 'Counter' is a word represented in the network order that holds the next value for the counter section in the .ini file, so that the threat should know what to search for next. 'String' is the URL location of the file to be downloaded and executed.

The advantages of Peacomm's network have a great impact on the prevention of this type of attack as well as tracking the origins of the infection.

The following are a few of the properties of this network:

- It is very difficult to identify the malicious peers.
- Malicious traffic is similar to legitimate P2P traffic.
- It is serverless – even if a large number of the peers become unavailable, the network is still available.
- It is flexible – it can be easily extended with new commands and may be configured to use any port.

Served purpose

Peacomm will attempt to download and execute a variety of custom threats on the compromised computer with one ultimate purpose: to facilitate sending spam emails or instant messages. The messages are used both for advertisements and for enlarging the bot network by sending malicious applications or links.

The following are a few of the current downloads:

- The first component being installed seems invariably to be an updated variant of Trojan.Abwiz.F, which hides its presence using rootkit techniques and, together with another installed component, acts as an SMTP server in order to relay spam and send spam through the compromised computer. In doing so, it connects to a predefined location and contains lists of email addresses as well as the message to send.
- The next component that is installed will harvest recursively the email addresses found in certain file types and send them to a predefined location, where they are added to a huge database that is used for spamming.
- Another deployed component is W32.Mixor.Q@mm, which comes bundled with the most recent variant of Trojan.Peacomm. The purpose of Mixor is to infect new computers by sending infected emails containing a copy of itself. As described earlier, Mixor also infects executables, thus making it harder to remove the threat.
- The Trojan.Mespam component is also used to expand the bot network. It will send spam IM messages retrieved from a configuration server that may contain malicious URLs through some of the widely used IM clients. It does not require passwords to do so, since it registers itself in the Layered Service Provider (LSP) chain of the network interface and detects the IM connections. This way, the malicious IM messages look legitimate since they come from a known contact and therefore have a good chance of tricking the target client.
- Last, but not least, a component whose purpose was only recently discovered is installed to allow directed distributed denial of service attacks against custom IP addresses by submitting bursts of packets to them. The addresses to be attacked are retrieved from a predefined configuration server.

Conclusion

The Peacomm Trojan represents just one element in a vast scheme designed for making money. And there is a lot of money involved, since the quick release of updates and new components must be sustained by an impressive level of resources.

Peacomm also highlights the current trends in malware evolution, which seem increasingly to be profit-oriented. Whether it is achieved through sending spam or stealing personal information, we notice an increasing and concerning growth of cyber crime.

References

1. *Kademlia: A Peer-to-peer Information System Based on the XOR Metric.* Petar Maymounkov and David Mazieres.
   http://www.scs.cs.nyu.edu/~petar/kpos_iptps.pdf
2. *The Overnet Protocol.* OpenSVN.
   https://opensvn.csie.org/mlnet/trunk/docs/overnet.txt
3. *RSA.* Wikipedia.
   http://en.wikipedia.org/wiki/RSA

About Symantec
Symantec is the global leader in information security, providing a broad range of software, appliances, and services designed to help individuals, small and mid-sized businesses, and large enterprises secure and manage their IT infrastructure. Symantec's Norton™ brand of products is the worldwide leader in consumer security and problem-solving solutions. Headquartered in Cupertino, California, Symantec has operations in 35 countries. More information is available at www.symantec.com.

Symantec has worldwide operations in 35 countries. For specific country offices and contact numbers, please visit our Web site. For product information in the U.S., call toll-free 1 800 745 6054.

Symantec Corporation
World Headquarters
20330 Stevens Creek Boulevard
Cupertino, CA 95014 USA
408 517 8000
800 721 3934
www.symantec.com