Hiding in Plain Sight

Advances in malware covert communication channels

Pierre-Marc Bureau, Christian Dietrich
Black Hat 2015, Whitepaper

Introduction

Steganography, the art of concealing information in different types of medias, is a very old practice. Yet, it only recently started being used by malware operators on a large scale. Malware programmers and operators are increasing their efforts in developing covert communication channels between infected computers and their command and control servers. In addition to steganography, recent examples include hiding communication in inconspicuous network traffic
such as DNS.

When used properly, these covert communication channels can bypass automated detection mechanisms and render malware communication difficult to detect and block. From an attacker's perspective, covert communication channels are a valuable addition because they allow messages to blend in with legitimate traffic and thus significantly lower the chance of being detected even when inspected by a human analyst.

Covert Communication Channels

By definition, malware is designed to perform malicious actions on an infected host. In most cases, these actions need to be done without catching attention from the user. In order to remain undetected, communication between an attacker and the malware needs to be done in a manner that is as stealthy as possible. As a result, malware authors increasingly use covert communication channels in order to hide their communication from infected users and network analysts. From the perspective of a malware author or operator, hiding malicious command and control (C2) communication boils down to two questions:

1) How to avoid exposing information that is required to be transmitted?

2) How to design a C2 channel that avoids being blocked or monitored in typical network environments?

This paper focuses on two types of covert communication channels: using steganography and hiding in inconspicuous carrier protocols. Both types are extended with practical malware examples found in the wild.

Steganography

Steganography is the act of hiding information inside another type of media. Its usage dates back to antiquity for messages being hidden with wax on wooden tablets. Invisible ink on courrier was also very popular with the French resistance during the Second World War. In most recent cases, steganography is used to hide message inside digital images or other media files. There are three recent cases of malicious software using steganography: the Lurk malware, the Gozi Neverquest malware and the Stegoloader malware family. All of these involve image-based steganography where the hidden data is encoded in the least significant bit (LSB) of each pixel in digital images.

Lurk

The Lurk malware family was first documented by security researchers from Dell SecureWorks in 2014. This malware family is used to download additional malware on infected hosts. In most cases, Lurk downloads malware that performs click fraud. This malware family uses digital steganography to hide the URL where it fetches content from. Instead of simply downloading and executing a malicious binary, Lurk first downloads a BMP image from which it extracts a download URL using least significant bit steganography.

Gozi Neverquest

The Gozi Neverquest malware family is used for financial fraud. This malware is a modern information stealer geared at financial institutions. It can inject code into browsers to inject content displayed to users and steal credentials. In the beginning of 2015, this malware family started using steganography as a backup mechanism to retrieve URLs where it could download its configuration file. The malware downloads a favicon.ico file from a server hosted on TOR using the tor2web service. It then extracts least significant bit information from each pixel and decrypts a URL where it can download its main configuration file in case its main C2 server are not responding.

Stegoloader

Stegoloader is a modular information stealer. It was first documented in 2015 and uses steganography to download code for its main module. The deployment module used by stegoloader inspects the system it is running on to detect whether it is an analysis machine or not. If Stegoloader determines the system is not used for analysis, it downloads a PNG image file from a legitimate website and extracts its main module code using least significant bit steganography. The extracted code is decrypted using the RC4 algorithm before being executed directly from memory.

The main module for Stegoloader is responsible for establishing a profile of the infected host and send this information to its command and control server. Depending on the infected host's profile, more modules can be executed. These modules have different purposes:

- Host geolocation, sends a report on where the host is located geographically
- IDA stealer, if IDA is installed on the host, it is exfiltrated to sendspace
- Pony loader password stealer, used to steal a variety of stored credentials
- List recently opened documents

The following table summarizes the use of steganography in recent malware cases.

Malware	**Stego**	**File types**	**Compression**	**Crypto**
Gozi	LSB	ico	None	RC4
Lurk	LSB	bmp	None	Custom
Stegoloader	LSB	png	None	RC4

Inconspicuous Carrier Protocols

An alternative approach to hide C2 information consists in choosing an inconspicuous carrier protocol. A well known example is a DNS covert channel: Messages mimic regular use of DNS but in fact encode data that is to be transmitted. This principle has been picked up by malware authors and is implemented in both, cyber criminal and targeted attack malware.

Feederbot

The Feederbot malware was initially discovered and first documented in a scientific paper published in 2011. It implements a covert channel via DNS in order to transmit control information to perform advertisement fraud. In particular, it constructs artificial DNS requests and responses that are not used for DNS resolution, but solely as containers for C2 information. Messages from an infected bot to the C2 server exhibit multiple query domain name schemes, some of which mimic well known registered domains, such as google.com, while others appear to be domain names that have never been registered. An example DNS message is provided below:

```
;QUESTION
1.f16e180e9093c237ea31a4ab55ae7fac710a14e4972b30fdf4.google.com. IN ANY

;ANSWER
1.f16e180e9093c237ea31a4ab55ae7fac710a14e4972b30fdf4.google.com.  0   IN TXT
"aYpYOb/6L5NRMxDRbwQDrVfPJDw5yogih+zlfj+lQpRDPZE4n1DWB0M/l0J6YDp88Vgm"
```

In this example, the bot queries for records of any resource record type at a preconfigured DNS server which acts as the C2 server. The response contains a resource record of type TXT which consists of Base64-encoded, RC4-encrypted C2 information.

Apparently, the authors were aware of the restrictions of the DNS syntax and message format, and implement a transmission scheme that respects the defined limits such as a maximum label length of 63 characters and up to 255 characters in a TXT record string. C2 messages exceeding these limits are chunked, i.e. split up over multiple requests and responses. In short, Feederbot uses a DNS covert channel as carrier for its command and control traffic.

PlugX

Another example of DNS as carrier protocol for C2 communication can be found with the PlugX malware that is often observed in targeted attacks. PlugX is a Remote Access Tool (RAT) that provides extensive remote control and surveillance capabilities. Throughout 2014, PlugX was by far the most used malware for targeted activity and it is a prevalent tool among China-based targeted intrusion adversaries. It is known to have been used by multiple Chinese actors for several years, including e.g., AURORA PANDA and GOBLIN PANDA.

This malware family has a modular design where most of the functionality is implemented in separate plugins. The core also supports a flexible C2 carrier protocol scheme allowing C2 communication via e.g., TCP, UDP and HTTP.

A feature that has hardly been documented publicly is its ability to use DNS as carrier protocol for C2 communication. PlugX implements a DNS covert channel where C2 messages from the infected system to the C2 server are encoded in the query domain and responses are expected as TXT resource records. C2 requests leverage Base16 encoding with a custom alphabet (shifted by `0x41`). In addition, PlugX varies several fields in a C2 request, likely in order to avoid network-based detection.

Hiding commands in HTTP error pages

It is not only DNS that is used as carrier protocol for hidden C2 communication. Recently, a malicious tool was identified which provides spreading functionality and engages in distributed denial of service attacks (DDoS). In this case, the family - referred to by researchers and antivirus as *Foreign* - stores C2 commands in the body of an HTTP error page. An example is shown below:

```
HTTP/1.1 404 Not Found
Date: Mon, 9 Jul 2015 06:13:37 GMT
Server: Apache/2
X-Powered-By: PHP/5.3.29
Vary: Accept-Encoding,User-Agent
Content-Length: 357
Connection: close
Content-Type: text/html; charset=utf8

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML
2.0//EN"><HTML><HEAD><TITLE>404 Not Found</TITLE></HEAD><BODY><H1>Not
Found</H1>The requested URL /XXX/YYY.php was not found on this
server.<P><HR><ADDRESS></ADDRESS></BODY></HTML><!--
DEBUG:MTQyODUyMTUyMzcyOTk5MyNsb2FkZXIgaHR0cDovLzExMS4xNzkuMzkuODMvZ29s
ZGVuMy5leGUjMTQyODUxMjA2MTc1NDYzNSNyYXRlIDYwIwDEBUG-->
```

At first sight, this may appear as a general error page with an HTTP status code 404 which indicates that the requested resource was not found. However, the body of the response contains a C2 command, encoded using Base64 and stored between HTML comment tags. The C2 information is thus not visible to a human visitor when rendered in a browser.

Decoding the C2 information yields the following plaintext string which instructs the bot to download a file from the given URL:

```
1428521523729993#loader
hxxp://111.179.39[.]83/golden3.exe#1428512061754635#rate 60#
```

Conclusions

Increased usage of covert communication channels by malware is an evolution that was to be expected. This type of mechanism is important for malware operators needing to evade detection.

From the perspective of the attacker, evading today's detection approaches consists in two aspects:

1) A malware C2 channel should be designed to avoid characteristic elements that serve as a signature.
2) A malware C2 channel should be designed to be stealthy, possibly staying under the radar of researchers and defenders.

While the first aspect has been known for many years, and is tackled for example through encryption, it is the second aspect that motivates attackers to employ techniques presented in this paper, including steganography and covert communication channels.

As a result, covert communication channels are observed in both, cyber crime operations and targeted attacks. Several examples are highlighted in this paper: Steganography is used for banking fraud with Gozi, information theft with Stegoloader and as a distribution vector for additional malware in the case of Lurk. The PlugX malware which is often used in targeted attacks, hides its communication inside DNS queries and a recent example of a DDoS tool relies on messages hidden in HTTP error messages. All in all, these examples show that hidden communication channels have arrived in today's world of digital crime.

However, hidden communication channels are not a panacea. There is a lot of room for design errors when implemented from scratch and they need to be used in conjunction with other technologies such as cryptography to guarantee integrity, confidentiality and authenticity.

References

- http://www.secureworks.com/cyber-threat-intelligence/threats/stegoloader-a-stealthy-information-stealer/
- https://blog.malwarebytes.org/security-threat/2014/02/hiding-in-plain-sight-a-story-about-a-sneaky-banking-trojan/
- http://www.secureworks.com/cyber-threat-intelligence/threats/malware-analysis-of-the-lurk-downloader/
- https://www.sophos.com/en-us/medialibrary/PDFs/technical%20papers/sophos-vawtrak-international-crimeware-as-a-service-tpna.pdf
- http://blog.crowdstrike.com/storm-chasing/
- http://www.secureworks.com/cyber-threat-intelligence/threats/stegoloader-a-stealthy-information-stealer/
- http://www.cj2s.de/On-Botnets-that-use-DNS-for-Command-and-Control.pdf
- https://blog.fortinet.com/post/hiding-malicious-traffic-under-the-http-404-error
- http://www.crowdstrike.com/global-threat-report-2014/