(https://www.welivesecurity.com/)

New TeleBots backdoor: First evidence linking Industroyer to NotPetya

ESET's analysis of a recent backdoor used by TeleBots – the group behind the massive NotPetya ransomware outbreak – uncovers strong code similarities to the Industroyer main backdoor, revealing a rumored connection that was not previously proven

Among the most significant malware-induced cybersecurity incidents in recent years were the attacks against the Ukrainian power grid (https://www.welivesecurity.com/2017/06/12/industroyer-biggest-threat-industrial-control-systems-since-stuxnet/) – which resulted in unprecedented blackouts two years in a row – and the devastating NotPetya ransomware outbreak (https://www.welivesecurity.com/2017/06/27/new-ransomware-attack-hits-ukraine/). Let's take a look at the links between these major incidents.

The first ever malware-enabled blackout in history, which happened in December 2015, was facilitated by the BlackEnergy malware toolkit (https://www.welivesecurity.com/2016/01/04/blackenergy-trojan-strikes-again-attacks-ukrainian-electric-power-industry/). ESET researchers have been following the activity (https://www.welivesecurity.com/2014/10/14/cve-2014-4114-details-august-blackenergy-powerpoint-campaigns/) of the APT group

utilizing BlackEnergy both before and after this milestone event. After the 2015 blackout, the group seemed to have ceased actively using BlackEnergy, and evolved into what we call TeleBots (https://www.welivesecurity.com/2016/12/13/rise-telebots-analyzing-disruptive-killdisk-attacks/).

It is important to note that when we describe 'APT groups', we're drawing connections based on technical indicators such as code similarities, shared C&C infrastructure, malware execution chains, and so on. We're typically not directly involved in the investigation and identification of the individuals writing the malware and/or deploying it, and the interpersonal relations between them. Furthermore, the term 'APT group' is very loosely defined, and often used simply to cluster the abovementioned malware indicators. This is also one of the reasons why we refrain from speculations with regard to attributing attacks to nation states and such.

That said, we have observed and documented ties between the BlackEnergy attacks – not only those against the Ukrainian power grid but against various sectors and high-value targets – and a series of campaigns (mostly) against the Ukrainian financial sector by the TeleBots group. In June 2017, when many large corporations worldwide were hit by the Diskcoder.C ransomware (aka Petya and NotPetya) – most probably as unintended collateral damage – we discovered that the outbreak started spreading from companies afflicted with a TeleBots backdoor (https://www.welivesecurity.com/2017/07/04/analysis-of-telebots-cunning-backdoor/), resulting from the compromise of the financial software M.E.Doc (https://www.welivesecurity.com/2017/06/30/telebots-back-supply-chain-attacks-against-ukraine/), popular in Ukraine.

So how does Industroyer, the sophisticated malware framework used to cause the blackout of December 2016, fit into all of this? Right after we publicly reported our discovery (https://www.welivesecurity.com/2017/06/12/industroyer-biggest-threat-industrial-control-systems-since-stuxnet/), some security companies and news media outlets started to speculate that Industroyer was also architected by the BlackEnergy/Telebots group (sometimes also referred to as Sandworm). Yet, no concrete evidence has been publicly disclosed until now.

In April 2018, we discovered new activity from the TeleBots group: an attempt to deploy a new backdoor, which ESET detects as **Win32/Exaramel**. Our analysis suggests that this TeleBots' backdoor is an improved version of the main Industroyer backdoor (https://www.welivesecurity.com/wp-content/uploads/2017/06/Win32_Industroyer.pdf) – the first piece of evidence that was missing.

Links between TeleBots, BlackEnergy, Industroyer, and (Not)Petya
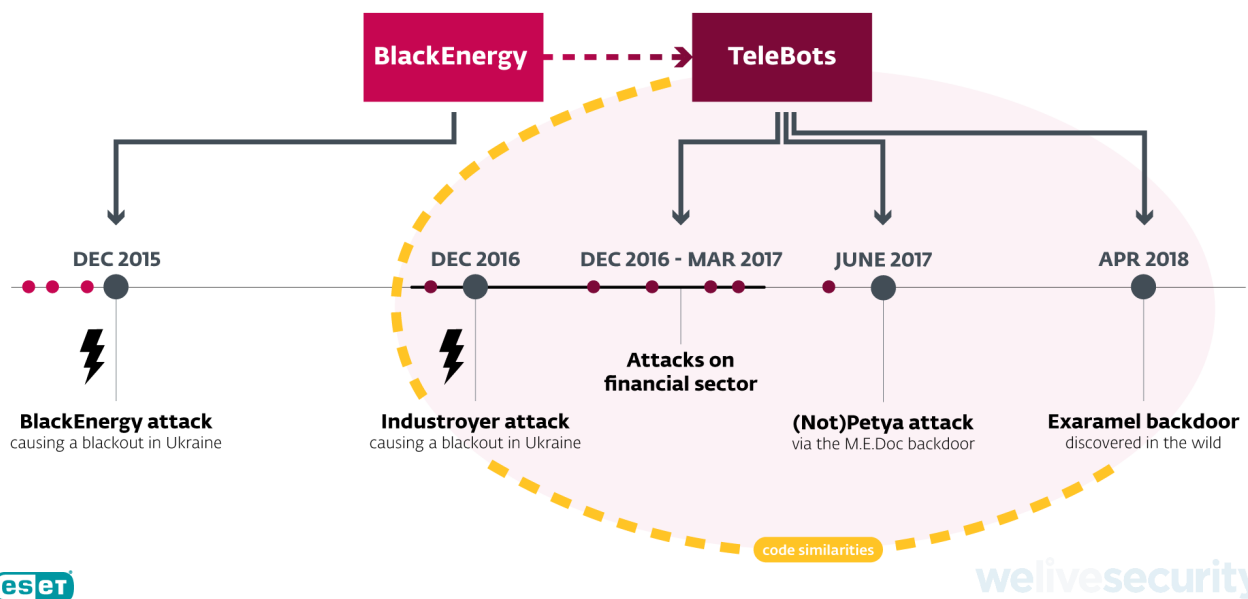
(https://www.welivesecurity.com/wp-content/uploads/2018/06/Telebots-Industroyer-Exaramel_FINAL-2.png)

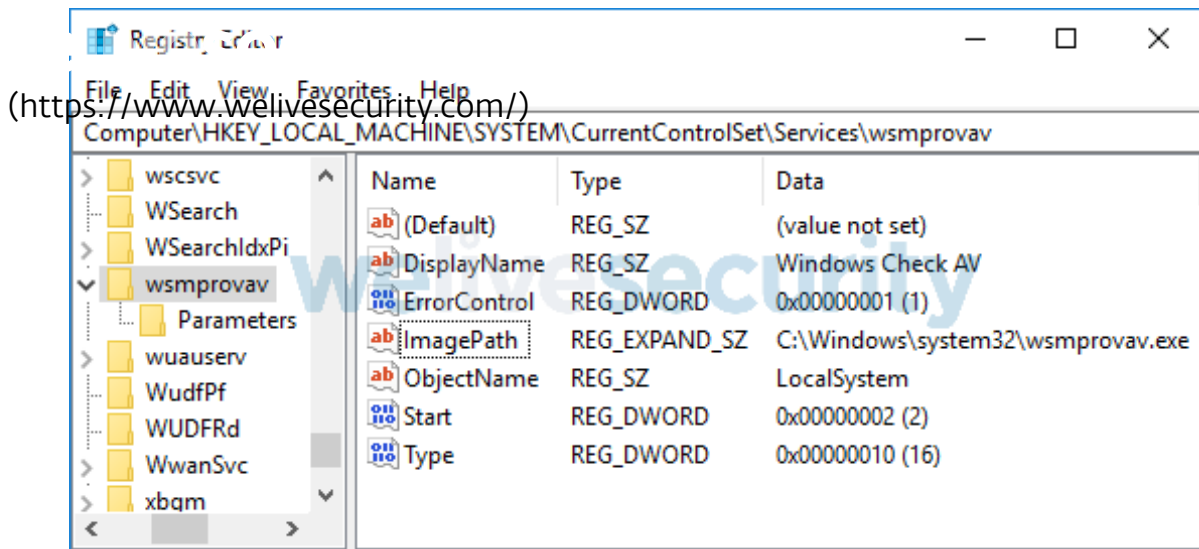## Analysis of the Win32/Exaramel backdoor

(https://www.welivesecurity.com/)

The Win32/Exaramel backdoor is initially deployed by a dropper. Metadata in this dropper suggest that the backdoor is compiled using Microsoft Visual Studio just before deployment on a particular, victimized computer.
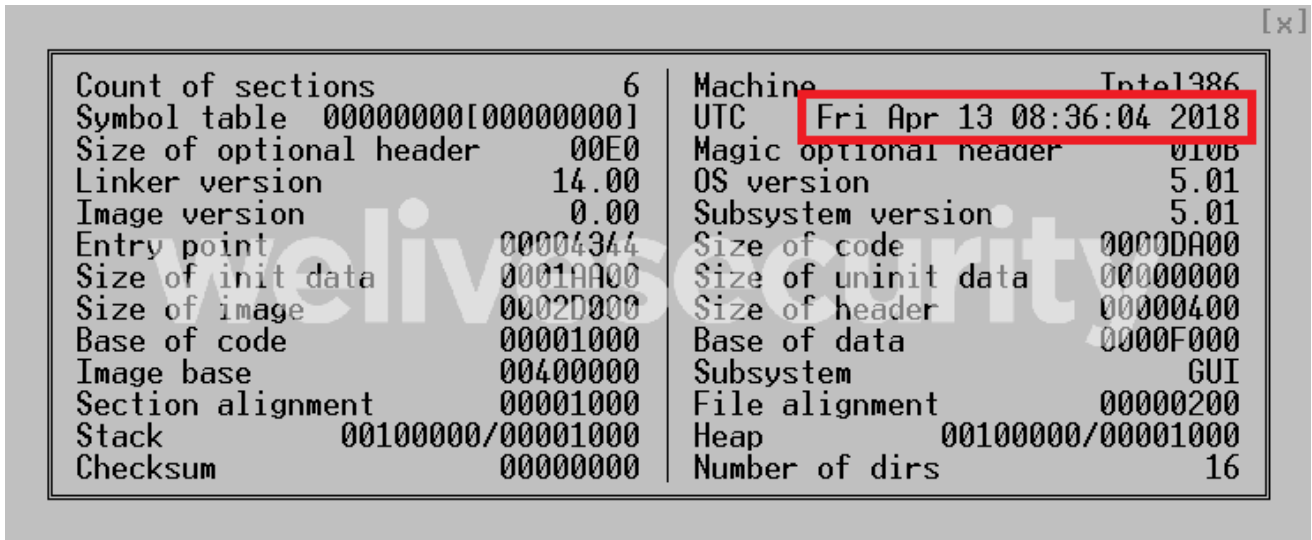
(https://www.welivesecurity.com/wp-content/uploads/2018/10/01_PE_Time_watermark.png)

Figure 1. PE timestamp in the dropper of Win32/Exaramel backdoor

Once executed, the dropper deploys the Win32/Exaramel backdoor binary in the Windows system directory and creates and starts a Windows service named `wsmprovav` with the description "Windows Check AV". The filename and the Windows service description are hardcoded into the dropper.

(https://www.welivesecurity.com/wp-
content/uploads/2018/10/02_windows_service_watermark.png)

Figure 2. Registry settings of the Windows service created by the
Win32/Exaramel backdoor

In addition, the dropper writes the backdoor's configuration into the
Windows registry, in XML format.

(https://www.welivesecurity.com/wp-
content/uploads/2018/10/03_XML_configuration_watermark.png)

Figure 3. Win32/Exaramel backdoor XML configuration

The configuration contains several blocks:

Interval – time in milliseconds used for Sleep function

Servers – list of command and control (C&C) servers

Check – website used to determine whether the host has an internet connection available (https://www.welivesecurity.com/)

Proxy – proxy server on the host network

Storage – path used for storing files scheduled for exfiltration

As can be seen from the first line of the configuration, the attackers are grouping their targets based on the security solutions in use. Similar behavior can be found in the Industroyer toolset – specifically some of the Industroyer backdoors were also disguised as an AV-related service (deployed under the name `avtask.exe`) and used the same grouping.

Another interesting fact is that the backdoor uses C&C servers with domain names that mimic domains belonging to ESET. In addition to `esetsmart[.]org` from the above mentioned configuration, we found another similar domain: `um10eset[.]net`, which was used by recently-discovered Linux version of Telebots malware. It is important to note that these attacker-controlled servers are in no way related to ESET's legitimate server infrastructure (https://support.eset.com/kb332/?locale=en_US&viewlocale=en_US). Currently, we haven't seen Exaramel use domains that mimic other security companies.

Once the backdoor is running, it connects to a C&C server and receives commands to be executed. Here is a list of all available commands:

Launch process

Launch process under specified Windows user

Write data to a file in specified path

Copy file into storage sub-directory (Upload file)

Execute shell command

Execute shell command as specified Windows user

Execute VBS code using `MSScriptControl.ScriptControl.1`

The code of the command loop and implementations of the first six commands are very similar to those found in a backdoor used in the Industroyer toolset.

```
 1 DWORD __stdcall cmd_thread(thread_param *param)
 2 {
 3   // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]
 4
 5   result1 = 0x16;
 6   v2 = init_CMD_struct(param->xml, &CMD);
 7   SetEvent((HANDLE)param->event);
 8   if ( v2 )
 9     return 1;
10   cmd_struct1 = CMD;
11   switch ( CMD->cmd_id )
12   {
13     case 1:
14       result = cmd_create_proccess(CMD);
15       goto end;
16     case 2:
17       result = cmd_create_proccess_as_user(CMD);
18       goto end;
19     case 3:
20       result = cmd_write_file(CMD);
21       goto end;
22     case 4:
23       result = cmd_copy_file_aka_upload(CMD);
24       goto end;
25     case 5:
26       result = cmd_execute_shell_cmd(CMD);
27       goto end;
28     case 6:
29       result = cmd_execute_shell_cmd_as_user(CMD);
30       goto end;
31     case 7:
32       result = cmd_eval_VBS_code(CMD);
33 end:
34       result1 = result;
35       break;
36     default:
37       break;
38   }
39   PathCombineW(&pszDest, (LPCWSTR)cmd_struct1->storage_path, L"done");
40   file_write(&pszDest, 0, 0);
41   mem_Free((LPVOID)cmd_struct1->field_0);
42   mem_Free((LPVOID)cmd_struct1->cmd_content);
43   mem_Free((LPVOID)cmd_struct1->file_content);
44   mem_Free(cmd_struct1);
45   return result1;
46 }
```

```
 1 int __cdecl run_command(cmd_internal *CMD)
 2 {
 3   int result; // eax
 4
 5   result = LOBYTE(CMD->cmd_id) - 1;
 6   switch ( LOBYTE(CMD->cmd_id) )
 7   {
 8     case 1u:
 9       result = cmd_create_proccess(CMD);
10       break;
11     case 2u:
12       result = cmd_create_proccess_as_user(CMD);
13       break;
14     case 3u:
15       result = cmd_write_file(CMD);
16       break;
17     case 4u:
18       result = cmd_copy_file_aka_upload(CMD);
19       break;
20     case 5u:
21       result = cmd_execute_shell_cmd(CMD);
22       break;
23     case 6u:
24       result = cmd_execute_shell_cmd_as_user(CMD);
25       break;
26     case 7u:
27       ExitProcess(0);
28       return result;
29     case 8u:
30       result = cmd_stop_service(CMD);
31       break;
32     case 9u:
33       result = cmd_stop_service_as_user(CMD);
34       break;
35     case 0xAu:
36       result = cmd_start_service_as_user(CMD);
37       break;
38     case 0xBu:
39       result = cmd_service_change_path_to_binary_as_user(CMD);
40       break;
41     default:
42       return result;
43   }
44   return result;
45 }
```

Figure 4. Comparison between decompiled code of the Win32/Exaramel backdoor (on the left) and the Win32/Industroyer backdoor (on the right)

Both malware families use a report file for storing the resulting output of executed shell commands and launched processes. In case of the Win32/Industroyer backdoor, the report file is stored in a temporary folder under a random filename; in case of the Win32/Exaramel backdoor, the report file is named `report.txt` and its storage path is defined in the backdoor's configuration file.

In order to redirect standard output (stdout) and standard error (stderr) to the report file, both backdoors set the `hStdOutput` and `hStdError` parameters to a handle of the report file. This is another design similarity between these malware families.

```
 1 int __cdecl cmd_execute_shell_cmd(cmd_internal *CMD)

 3   // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

 6   mem_set(&ProcessInformation, 16);
 7   SecurityAttributes.nLength = 12;
 8   SecurityAttributes.lpSecurityDescriptor = 0;
 9   SecurityAttributes.bInheritHandle = 1;
10   reportfile_handle = CreateFileW(
11                            CMD->reportfile_path,
12                            GENERIC_WRITE|GENERIC_READ,
13                            1u,
14                            &SecurityAttributes,
15                            2u,
16                            0x80u,
17                            0);
18   if ( reportfile_handle == -1 )
19     ExitCode = GetLastError();
20   StartupInfo.dwFlags |= STARTF_USESTDHANDLES|STARTF_USESHOWWINDOW;
21   StartupInfo.wShowWindow = 0;
22   StartupInfo.cb = 68;
23   StartupInfo.hStdError = reportfile_handle;
24   StartupInfo.hStdOutput = reportfile_handle;
25   ExpandEnvironmentStringsW(L"%ComSpec%", &Dst, 0x104u);
26   wsprintfW(&CommandLine, L"/c %s", CMD->cmd_content);
27   CreateProcessW(&Dst, &CommandLine, 0, 0, 1, 0x8000000u, 0, 0, &StartupInfo, &ProcessInformation);
28   if ( ProcessInformation.hProcess )
29   {
30     if ( !WaitForSingleObject(ProcessInformation.hProcess, 0xFFFFFFFF) )
31     {
32       FlushFileBuffers(reportfile_handle);
33       GetExitCodeProcess(ProcessInformation.hProcess, &ExitCode);
34       CloseHandle(ProcessInformation.hThread);
35       CloseHandle(ProcessInformation.hProcess);
36     }
37   }
38   else
39   {
40     ExitCode = GetLastError();
41   }
42   CloseHandle(reportfile_handle);
43   return ExitCode;
44 }
```

```
 1 int __cdecl cmd_execute_shell_cmd(cmd_internal *CMD)
 2 {
 3   int result; // edi
 4   WCHAR CommandLine; // [esp+8h] [ebp-A7Ch]
 5   WCHAR Dst; // [esp+828h] [ebp-25Ch]
 6   struct _STARTUPINFOW StartupInfo; // [esp+A30h] [ebp-54h]
 7   struct _PROCESS_INFORMATION ProcessInformation; // [esp+A74h] [ebp-10h]
 8
 9   mem_set(&StartupInfo, 68);
10   mem_set(&ProcessInformation, 16);
11   StartupInfo.dwFlags |= STARTF_USESTDHANDLES|STARTF_USESHOWWINDOW;
12   StartupInfo.cb = 68;
13   StartupInfo.hStdError = (HANDLE)CMD->reportfile_handle;
14   StartupInfo.hStdOutput = StartupInfo.hStdError;
15   StartupInfo.wShowWindow = 0;
16   ExpandEnvironmentStringsW(L"%ComSpec%", &Dst, 0x104u);
17   wsprintfW(&CommandLine, L"/c %s", CMD->cmd_content);
18   result = CreateProcessW(&Dst, &CommandLine, 0, 0, 1, 0x8000000u, 0, 0, &StartupInfo, &ProcessInformation);
19   if ( ProcessInformation.hProcess && !WaitForSingleObject(ProcessInformation.hProcess, 0xFFFFFFFF) )
20   {
21     FlushFileBuffers((HANDLE)CMD->reportfile_handle);
22     CloseHandle(ProcessInformation.hThread);
23     CloseHandle(ProcessInformation.hProcess);
24   }
25   return result;
26 }
```

(https://www.welivesecurity.com/wp-
content/uploads/2018/10/05_cmd_shell_EXARAMELvsINDUSTROYER_watermark.png)

Figure 5. Comparison between decompiled code of the Win32/Exaramel
backdoor (on the top) and the Win32/Industroyer backdoor (on the bottom)

If the malware operators want to exfiltrate files from the victim's
computer, they just need to copy those files into the data sub-
directory of the storage path defined in the configuration. Once the

backdoor is about to make a new connection to the C&C server it will automatically compress and encrypt all these files before sending them.

(https://www.welivesecurity.com/)

The main difference between the backdoor from the Industroyer toolset and this new TeleBots backdoor is that the latter uses XML format for communication and configuration instead of a custom binary format.

Password stealing malicious tools

Along with the Exaramel backdoor, Telebots group uses some of their old tools, including a password stealer (internally referred as CredRaptor or PAI by the attackers) and a slightly-modified Mimikatz.

The CredRaptor custom password-stealer tool, exclusively used by this group since 2016, has been slightly improved. Unlike previous versions, it collects saved passwords not only from browsers, but also from Outlook and many FTP clients. Here is a list of supported applications:

- BitKinex FTP

- BulletProof FTP Client

- Classic FTP

- CoffeeCup

- Core FTP

- Cryer WebSitePublisher

CuteFTP

(https://www.welivesecurity.com/)
FAR Manager

FileZilla

FlashFXP

Frigate3

FTP Commander

FTP Explorer

FTP Navigator

Google Chrome

Internet Explorer 7 – 11

Mozilla Firefox

Opera

Outlook 2010, 2013, 2016

SmartFTP

SoftX FTP Client

Total Commander

TurboFTP

Windows Vault

WinSCP

WS_FTP Client

This improvement allows attackers to collect webmaster's credentials for websites and credentials for servers in internal infrastructure. Once access to such servers is obtained, attackers could plant additional backdoors there. Quite often these servers are operated by OSes other than Windows, so attackers have to adapt their backdoors.

In fact, during our incident response, we discovered a Linux backdoor used by TeleBots. We named this backdoor **Linux/Exaramel.A**.

Analysis of the Linux/Exaramel backdoor

The backdoor is written in the Go programming language and compiled as a 64-bit ELF binary. Attackers can deploy the backdoor in a chosen directory under any name.

If the backdoor is executed by attackers with the string 'none' as a command line argument, then it attempts to use persistence mechanisms in order to be started automatically after reboot. If the backdoor is not executed under the root account, then it uses the `crontab` file. However, if running as root, it supports different Linux `init` systems. It determines which `init` system is currently in use by executing the command:

```
strings /sbin/init |  awk 'match($0,
/(upstart|systemd|sysvinit)/){ print substr($0,
RSTART, RLENGTH);exit; }'
```

Based on the result of this command, it uses the following hardcoded locations for its persistence:

Init system	Location
(https://www.welivesecurity.com/)	
sysvinit	/etc/init.d/syslogd
upstart	/etc/init/syslogd.conf
systemd	/etc/systemd/system/syslogd.service

During startup, the backdoor attempts to open a configuration file named `config.json`, which is stored in the same directory as the backdoor. If this configuration file does not exist, then a new file is created. The configuration is encrypted using the key **s0m3t3rr0r** via the RC4 algorithm.

(https://www.welivesecurity.com/wp-
content/uploads/2018/10/06_Linux_Exaramel_config_watermark.png)

Figure 6. Decrypted JSON configuration of the
Linux/Exaramel backdoor

The backdoor connects to the hardcoded C&C server (by default 176.31.225[.]204 in the sample we have seen to date) or to the C&C server listed in the configuration files `Hosts` value. The communication is sent over HTTPS. The backdoor supports the following commands:

Command	Purpose
App.Update	Updates itself to a newer version

Command (https://www.welivesecurity.com/)	Purpose
App.Delete	Deletes itself from the system
App.SetProxy	Sets proxy in configuration
App.SetServer	Updates C&C server in configuration
App.SetTimeout	Sets timeout value (time between connections to C&C server)
IO.WriteFile	Downloads a file from a remote server
IO.ReadFile	Uploads a file from local disk to C&C server
OS.ShellExecute	Executes a shell command

Conclusion

The discovery of Exaramel shows that the TeleBots group is still active in 2018 and the attackers keep improving their tools and tactics.

The strong code similarity between the Win32/Exaramel backdoor and the Industroyer main backdoor is the first publicly-presented evidence linking Industroyer to TeleBots, and hence to NotPetya and BlackEnergy. While the possibility of false flags – or a coincidental code sharing by another threat actor – should always be kept in mind when attempting attribution, in this case we consider it unlikely.

Of particular interest is the fact that the attackers started to use ESET-themed domain names in their operations. It should be noted that these domains were used by cybercriminals in order to hide their malicious network activity from defenders and are in no way related to ESET's server infrastructure. The list of legitimate domains used by ESET products can be found here (https://support.eset.com/kb332/?locale=en_US&viewlocale=en_US).

It should also be noted that these Win32 and Linux Exaramel
backdoors were detected at an organization that is not an industrial
(https://www.welivesecurity.com/)
facility. ESET shared its findings ahead of time with Ukrainian
investigation authorities and thanks to this cooperation the attack was
successfully localized and prevented.

ESET researchers will continue to monitor the activity of this group.

Indicators of Compromise (IoCs)

ESET detection names

Win32/Exaramel trojan

Win32/Agent.TCD trojan

Linux/Agent.EJ trojan

Linux/Exaramel.A trojan

Win32/PSW.Agent.OEP trojan

Win32/RiskWare.Mimikatz.Z application

Win64/Riskware.Mimikatz.AI application

SHA-1 HASHES

TeleBots Win32/Exaramel backdoor

65BC0FF4D4F2E20507874F59127A899C26294BC7

3120C94285D3F86A953685C189BADE7CB575091D

Password Stealer

F4C4123849FDA08D1268D45974C42DEB2AAE3377

SHA-1 HASHES

970E8ACC97CE5A8140EE5F6304A1E7CB56FA3FB8

DDDF96F25B12143C7292899F9D5F42BB1D27CB20

64319D93B69145398F9866DA6DF55C00ED2F593E

1CF8277EE8BF255BB097D53B338FC18EF0CD0B42

488111E3EB62AF237C68479730B62DD3F52F8614

Mimikatz

458A6917300526CC73E510389770CFF6F51D53FC

CB8912227505EF8B8ECCF870656ED7B8CA1EB475

Linux/Exaramel

F74EA45AD360C8EF8DB13F8E975A5E0D42E58732

Warning! All of the servers with these IP addresses were part of the Tor network, which means that the use of these indicators could result in a false positive match.

C&C servers

um10eset[.]net (IP: 176.31.225.204)

esetsmart[.]org (IP: 5.133.8.46)

○

Anton Cherepanov (https://www.welivesecurity.com/author/acherepanov/) and Robert Lipovsky (https://www.welivesecurity.com/author/lipovsky/) 11 Oct 2018 - 01:57PM

Similar Articles

(https://www.welivesecurity.com/)

CYBERCRIME
(HTTPS://WWW.WELIVESECURITY.COM/CAT

CYBERCRIME
(HTTPS://WWW.WELIVESECURITY.COM/CAT

(https://www.welivesecurity.com/2018/10
public-cybercrime-cybersecurity/)

(https://www.welivesecurity.com/2018/09
newegg-defenses-customers-credit-
card-data/)

Why ask the public about cybercrime and
cybersecurity?
(https://www.welivesecurity.com/2018/10/04/ask-
public-cybercrime-cybersecurity/)

Attackers crack Newegg's defenses, slurp
customers' credit card data
(https://www.welivesecurity.com/2018/09/2
newegg-defenses-customers-credit-card-
data/)

Discussion

Comments velivesecurity.com

(https://www.welivesecurity.com/)

♡ Recommend   🐦 Tweet   f Share

Sort by Best ▾

Start the discussion…

LOG IN WITH   OR SIGN UP WITH DISQUS ?

[D] [f] [🐦] [G]

Name

Be the first to comment.

✉ Subscribe   Ⓓ Add Disqus to your siteAdd DisqusAdd

DISQUS

🔒 Disqus' Privacy PolicyPrivacy PolicyPrivacy

welivesecurity™ BY eset®

(https://www.welivesecurity.com/)

Home (/)

About Us (https://www.welivesecurity.com/about-us/)

Contact Us (https://www.welivesecurity.com/contact-us/)

Sitemap (https://www.welivesecurity.com/sitemap/)

Our Experts (https://www.welivesecurity.com/our-experts/)

ESET (https://eset.com)

Research (https://www.welivesecurity.com/research/)

How To (https://www.welivesecurity.com/category/how-to/)

Categories (https://www.welivesecurity.com/categories/)

RSS Configurator (https://www.welivesecurity.com/rss-configurator/)

News Widget (https://www.welivesecurity.com/news-widget-generator/)

()

Privacy policy (https://www.welivesecurity.com/privacy/)

(https://www.welivesecurity.com/)

Legal Information (https://www.welivesecurity.com/legal-information/)

(https://www.welivesecurity.com/)