TURLA
LIGHTNEURON

One email away from remote code execution

Mario Leonardo Salinas

UniPi - ICT Risk Assessment

Table of contents

Table of Contents

Turla

- Turla, aka *Snake*, is Russian-based threat group active since 2004. [4]
- Victims in over 45 countries since 2004:
  - industries
  - goverments
  - military
  - education
  - research
- The group owns a large arsenal of malware and backdoors

LightNeuron

- LightNeuron [2] is a malware specifically designed to target Micrsoft Exchange servers.
- It can *spy* on emails and act as a *fully featured backdoor*
- The attack can only happen if an Exchange server is already fundamentally compromised, *e.g. root permissions*
- Hard to detect at the network level (no standard HTTP(s) communications)

Table of Contents

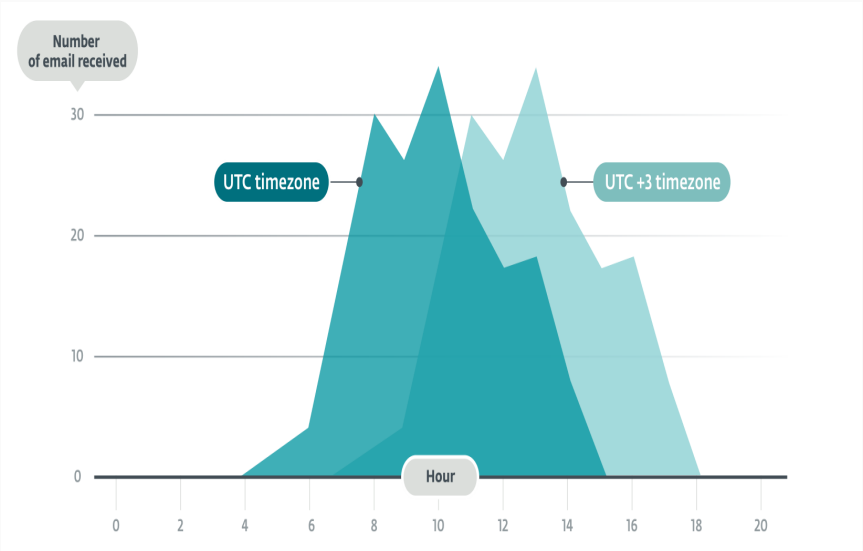
Attacker Profile i

- Turla is well known for its advanced custom tools and its ability to run highly targeted operations.
- The group is interested in collecting information from strategic people or organizations.

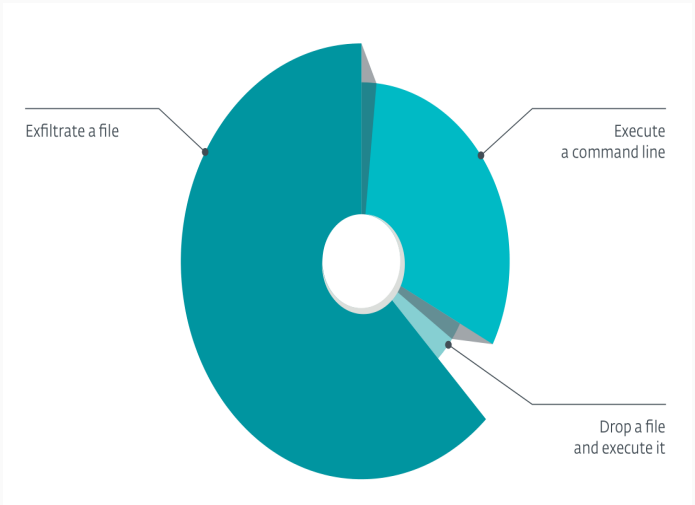
Figure 1: Timeline of important attacks attributed to Turla

Attacker Profile ii

- The operators activity matches a typical 9-to-5 workday in the UTC+3 time zone
- LightNeuron is used mostly to exfiltrate data. The remaining activity is most likely dropping and executing tools to perform lateral movements across the local network



(a) Operators working hours



(b) Distribution of the backdoor commands used by the operators

Victimology

According to ESET, LightNeuron development started before 2014; even if the development occurred several years ago, LightNeuron is still used in recent compromises. These targets are in line with traditional Turla targets:

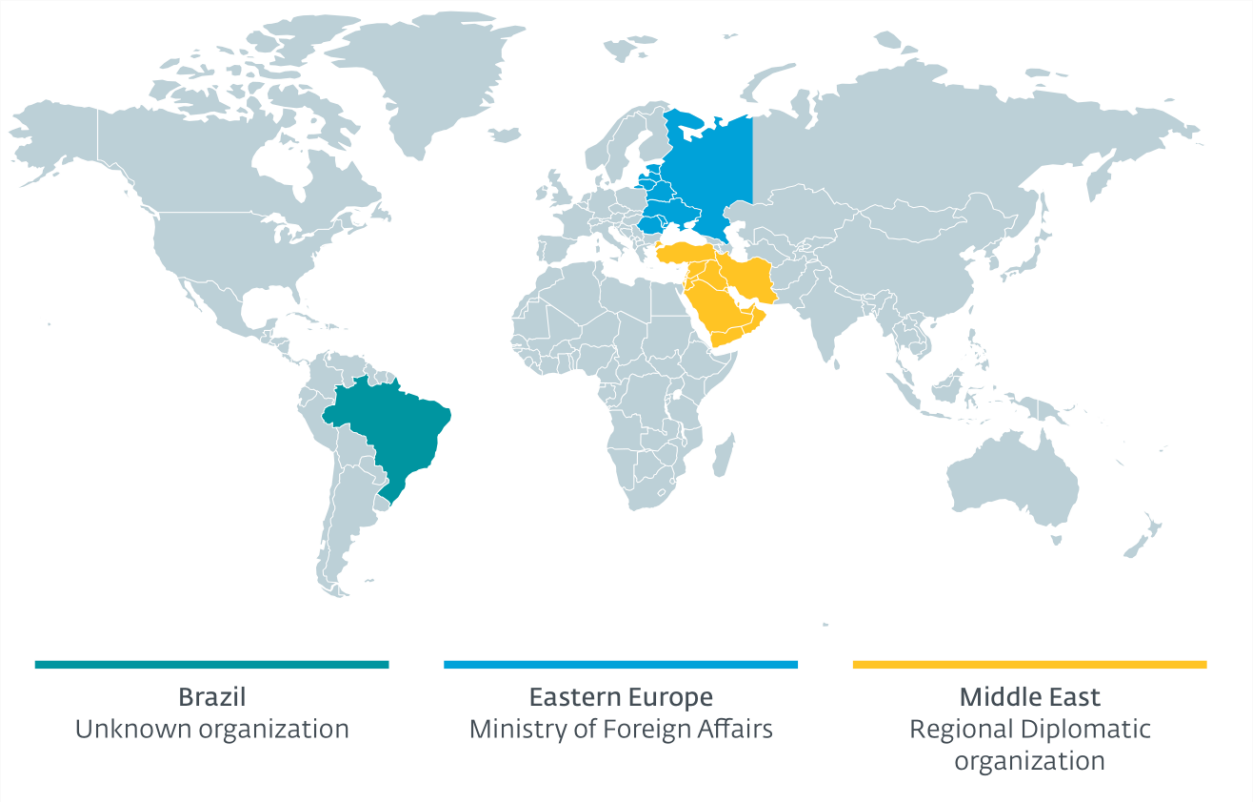


Figure 2: Map of known LightNeuron victims

Table of Contents

Microsoft Exchange Transport Agent

- Transport agents let you install custom software that is created by Microsoft, by third-party vendors, or by your organization, on an Exchange server [3]
- This software can then process email messages that pass through the transport pipeline
- Having a single pipeline means that you can have confidence that every message goes is processed in the same way; however it also means that if an attacker can introduce a transport agent, they have access to *every message*
- LightNeuron it's the first malware that uses a Transport Agent for malicious purposes

LightNeuron components

Two main components comprise LightNeuron:

- a *Transport Agent* that can process and modify all email messages going through the mail server
- a companion 64-bit *Dynamic Link Library (DLL)* containing most of the malicious code.

Turla modus operandi

A tipical Turla attack chain involves:

1. **Initial Reconnissance**, usually a basic first-stage malware or a more powerful one if they deem the victim interesting (Metasploit, Carbon or Gazer). Very specific targets
2. **Credential Gathering**, they move laterally on the network to collect accounts, using stealthy communications and periodically creating new accounts for persistence
3. **Exfiltration**, using an HTTP/email C&C channel and SATCOM IP addresses to obfuscate the traffic content and destination.

LightNeuron Attack Chain (MITRE ATT&CK)

1. **Initial Acces & Privilege Escalation**: Valid Accounts using MITM, spreadpishing emails and watering-hole attacks
2. **Execution**: PowerShell script to install Lightneuron components

3a. **Collection**: Automated Collection of both emails and files

3b. **Command & Control**: email communication using cryptography and steganography

3c. **Exfiltration**: Automated and Encrypted exfiltration via C&C interface with optional night scheduling

Initial Access & Privilege Escalation

- Valid Accounts T1078: adversaries may steal the credentials of a specific user or service account using Credential Access techniques or capture credentials earlier through social engineering [1]
- The attackers must have privileged administrative access to the server in order to start the attack chain

Execution

- PowerShell T1086: a powershell script is executed
- The malicious Transport Agent is a 32-bit Windows DLL developed in .NET
- The attackers drop this executable in the Exchange folder located in the Program Files folder.

Execution: Transport Agent Installation

Once admin priviledge have been obtained, a PowerShell script is executed to register the DLL as a Transport Agent

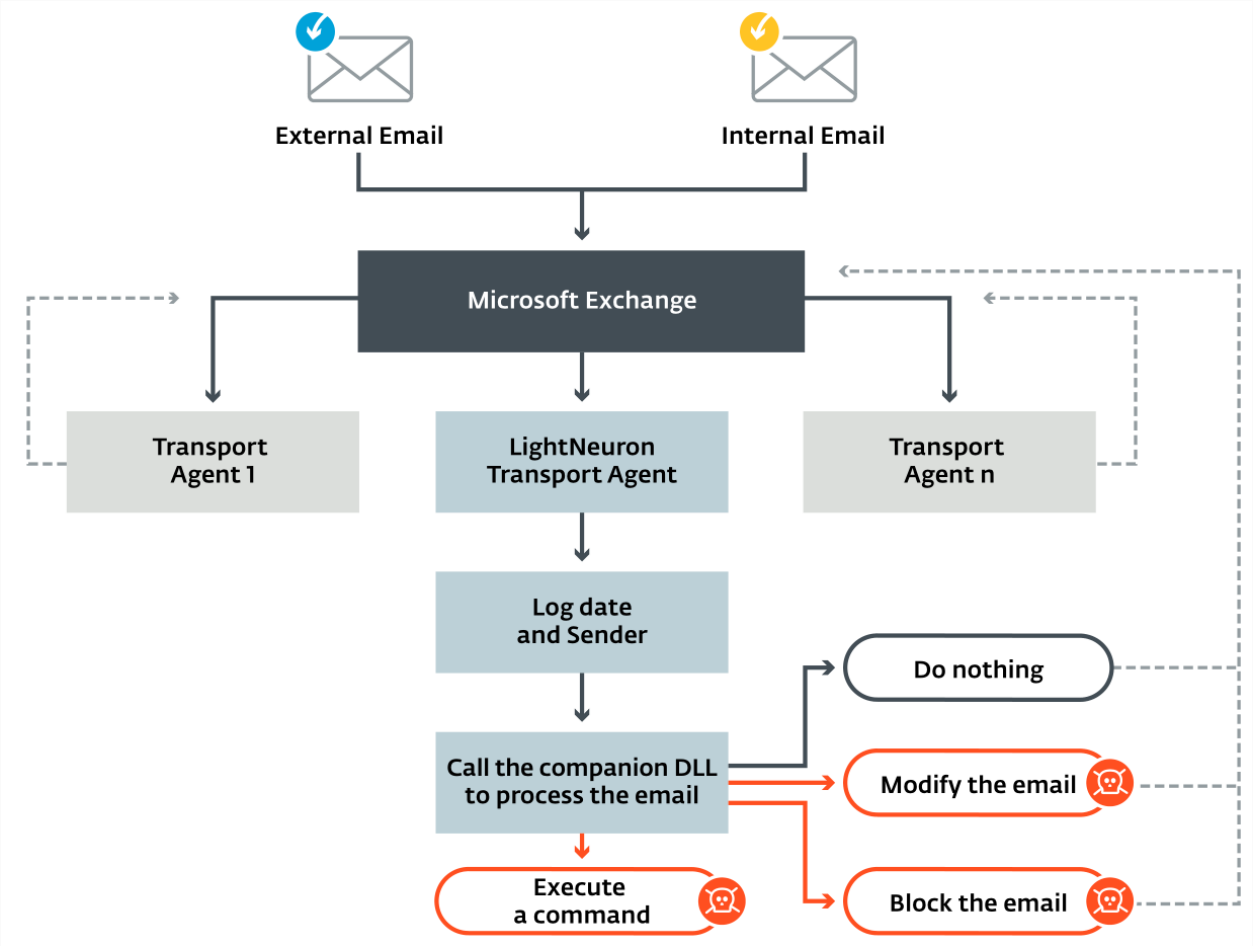


Figure 3: LightNeuron Transport Agent

Execution: Companion Dynamic Link Libray

- The companion DLL is a 64-bit Windows DLL developed in C
- When the Transport Agent loads the DLL, the DLL's main function performs various initialization tasks.
- After initail decription operations, it decrypts the configuration file stored in `%tmp%/winmail.dat`; this filename has been chosen to hide their configuration file in plain sight

The Configuration Files

- The configuration file `winmail.dat` contains various parameters
- An interesting one is `CONFIG_FILE_NAME`
- Once decrypted, this second configuration file contains the rules used to process the emails.

Configuration File Rule System i

- The second configuration file contains several class nodes, each one corresponding to a different function (aka handler) implemented in the DLL.
- Each class node contains a set of rules describing conditions using the logical operators AND and OR.
- At the end of the file is the mapping of the class names with the name of the functions in the DLL.

```
<class name="zip" metric="30" id="1" dllName="ZipMe" type="dll" include="1">
    <rule metric="10" id="1" include="1">
        <and>
            <or>
                <To condition="cnt" value="email1@[redacted]" />
                <From condition="cnt" value="email1@[redacted]" />
                <To condition="cnt" value="email2@[redacted]" />
                <From condition="cnt" value="email2@[redacted]" />
                [...]
            </or>
            <and>
                <To condition="!cnt" value="email3@[redacted]" />
                <From condition="!cnt" value="email3@[redacted]" />
                [...]
            </and>
        </and>
    </rule>
</class>
<class name="command" metric="40" id="1" dllName="ZipMe" type="dll" include="1">
    <rule metric="10" id="1" include="1">
        <attachment_Content-Type condition="cnt" value="image/jpeg" />
    </rule>
</class>

log:logHandler
zip:zipHandler
changeSubject:changeSubjectHandler
changeBody:changeBodyHandler
create:createHandler
command:commandHandler
block:blockHandler
replace:replaceHandler
stat:statHandler
```

Configuration File Rule System ii

- These rules are applied to every email processed by the DLL
- This configuration is highly flexible
- There are eleven different handlers implemented in the DLL

Figure 4: Description of the handlers implemented in the DLL

Collection i

- Since these rules are applied to every email passing through the transport pipeline, the behavioral characteristics of LightNeuron reside in this second configuration file.
- MITRE Techniques:

  Automated Collection T1119: depending on the configuration, LightNeuron can collect the files in a specific path

  Email Collection T1114: LightNeuron collects all the emails matching one rules specified in its configuration.

Collection ii

- In every handler definition, the email is in the form of a *linked-list* with the different fields parsed (From, To, body, etc.)
- The handler can modify this linked-list and will return a code corresponding to the action it performed
- Then, the Transport Agent interprets this return code to know if it should modify the email, block it or execute .NET assembly code

Return Value	Description
0	No modification
1	Email modified
2	Block the email
3	Error
4	Contains .NET assembly

Figure 5: Handler return codes and their descriptions

Command & Control and Exfiltration: the backdoor

- The command handler is actually the implementation of a backdoor controlled by email *(T1071 Standard Application Layer Protocol)*.
- It has the following properties:
  - Depending on the rules, the commands are hidden in a PDF or a JPG attachment.
  - It uses steganography to hide data in PDF documents or JPG pictures. *(T1001 Data Obfuscation)*
- One decrypted, the attachment is passed to the routine that reads the blob of data containing the command, aka *container*

Command & Control and Exfiltration: containers i

- The first four bytes are the size of the container and the following bytes are encrypted with AES-256 with a key hardcoded in the binary.
- Once decrypted, we see the different fields used to store information about the commands to be executed.
  - At offset 0x08, the email address to which the result of the command is sent.
  - At offset 0x1D, the *instruction code*.
  - At offset 0x25, the first argument of the function that will be called.

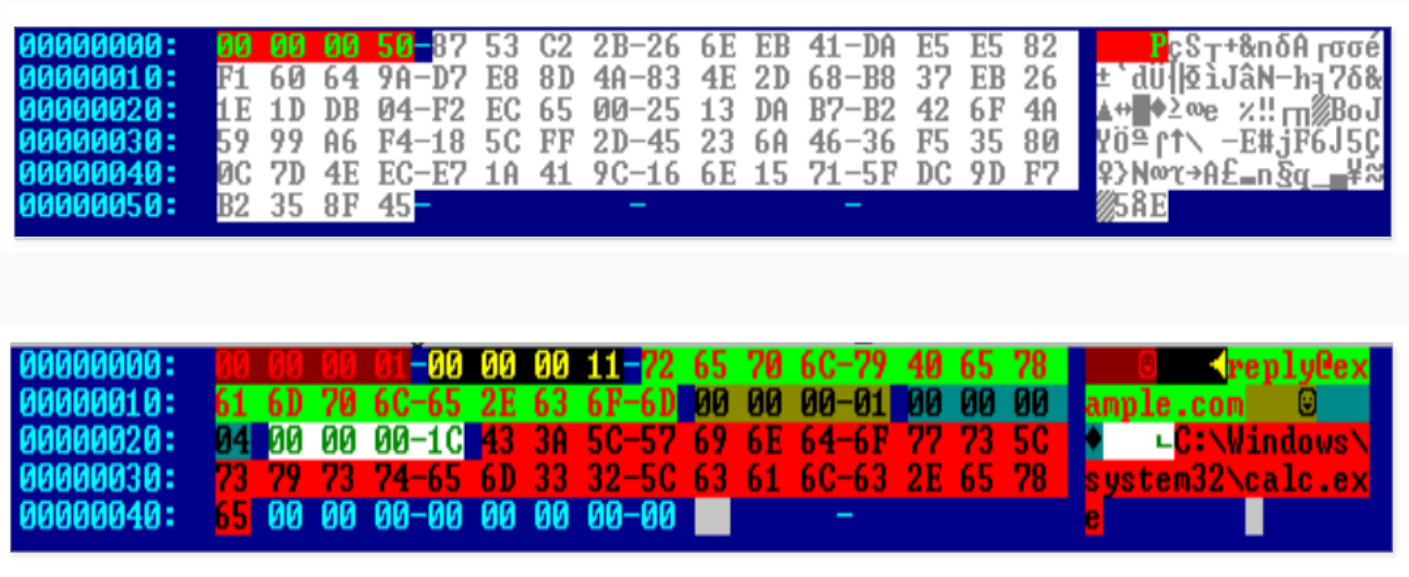


Figure 6: Hexadecimal dump of the encrypted (top) decrypted (bottom) container.

Command & Control and Exfiltration: containers ii

- When processing a container, the backdoor writes the `CmdId` value to a log file (anti-replay mechanism)
- Finally, the command output is encrypted with AES and a PDF document or a JPG image
- An email is then created.

Instruction Code	Description	Arg 1	Arg 2	Arg 3
0x01	Write a file. Execute it if it is an executable.	Exe path	N/A	File data
0x02	Delete a file	File path	N/A	N/A
0x03	Exfiltrate a file	File path	Set to "1" to delete the file	N/A
0x04	Execute a process (CreateProcess)	Command line	N/A	N/A
0x05	Execute a command line (cmd.exe /c)	Command line	N/A	N/A
0x06	Return 0	N/A	N/A	N/A
0x07	Disable backdoor for N minutes.	Minutes	N/A	N/A
0x09	Exfiltrate a file (duplicate function of 0x03)	File Path	Set to "1" to delete the file	N/A
0x65	No-op	N/A	N/A	N/A

(a) List of instruction codes

```
struct encrypted_container {
    int size; //clear text
    container[]; //encrypted with AES-256
}

struct container { //Can contain multiple commands
    int CmdId; //Unique ID to identify the container
    int rcptl; //Recipient address length
    char rcpt[rcptl]; //Recipient address (address to which the output data will be sent)
    command[]; //list of commands
}

struct command {
    int InstId; //Unique ID to identify this command
    int InstrCode; //The instruction that will be executed
    int fpl; //First parameter length
    char fp[fpl]; //First parameter
    int spl; //Second parameter length
    char sp[spl]; //Second parameter
    int bpl; //Third parameter length
    char bp[bpl]; //Third parameter
}
```

(b) Structure of the command container (C-like syntax)

Command & Control and Exfiltration: sending the email

- To send the email, it simply drops it in the folder `<ExchangeInstallFolder>/TransportRoles/PickUp/` and the filename starts with `msg` followed by the result of the `GetTickCount` function. According to the Microsoft documentation [6]
- Moreover exchange does not perform any secutity check on the email sent via this folder
- Thus, security solutions will not see the data exfiltrated through LightNeuron, which makes this malware very **stealthy**

Command & Control and Exfiltration: Automated and Scheduled Exfiltration

- If the configuration parameters `SEND_TIME` and `SEND_AT_NIGHT` are set, a thread launched by the DLL main function will loop indefinitely.
- The exfiltration function loops over the files that match the `SEND_FILE` specification
- It is possible to include wildcards in the filename specification to match several different files
- Then, for each file, it will send an email containing the file in either a JPG or a PDF attachment

Exfiltration	T1020	Automated Exfiltration	Depending on the configuration, LightNeuron can exfiltrate files located in a specific path.
	T1022	Data Encrypted	Data is encrypted using AES.
	T1041	Exfiltration Over Command and Control Channel	Data is exfiltratated using an email C&C channel.
	T1029	Scheduled Transfer	Depending on the configuration, automatic exfiltration can happen during the night or during working hours.

Figure 7: Exfiltration MITRE techniquesd used by LightNeuron

Table of Contents

Countermeasures i

- The cleaning of LightNeuron is not an easy task. **Simply removing the two malicious files will break Microsoft Exchange**, preventing everybody in the organization from sending and receiving emails.
- Before actually removing the files, the malicious Transport Agent should be disabled
- open `<ExchangeInstall-Folder>/TransportRoles/Agents/agents.config` and check every DLL signature.
- disable the malicious Transport Agents, and after that it is possible to safely remove the infected files.

Countermeasures ii

Given that attackers have gained administrative privileges on the Exchange server, there are no bulletproof mitigations against this threat; however there are some recommendations worth to mention:

- Use dedicated accounts for the administration of Exchange servers with strong, unique passwords and, if possible, 2FA.
- Monitor closely the usage of these accounts (DLL uses a lot of log files)
- Restrict PowerShell execution
- Regularly check all the installed Transport Agents

Microsoft's View

"an attacker would need to have administrative access on an Exchange server as a member of the Exchange Administrator group in an organization's Active Directory. Exchange Administrator accounts are tightly controlled, and membership cannot be obtained by persuading a victim to click through a security warning or elevation request." [5]

Table of Contents

Conclusions

- LightNeuron is another example that Turla operators have a large set of sophisticated, custom malware at their disposal
- This is the first time a malicious actor has leveraged a Microsoft Exchange Transport Agent to enable persistence on a mail server
- This technique is very interesting as it allows them to receive commands and exfiltrate data without any filtering.
- Probably not many servers have been infected in this manner, but it's certainly wise to review Exchange server configurations to look for any unexplained transport agent.

References i

[1] The MITRE Corporation. MITRE ATT&CK. https://attack.mitre.org/.

[2] Matthieu Faou. Turla LightNeuron, one email away from remote code execution. Technical report, ESET Research.

[3] Microsoft. Transport agents. https://docs.microsoft.com/en-us/exchange/transport-agents-exchange-2013-help.

[4] Edward Millington. Turla. https://attack.mitre.org/groups/G0010/.

[5] Tony Redmond. Exchange and the Turla LightNeuron Attack. https://www.petri.com/exchange-turla-lightneuron-attack.

[6] D. Strome. Configure the Pickup directory and the Replay directory. https://docs.microsoft.com/en-us/exchange/configure-the-pickup-directory-and-the-\replay-directory-exchange-2013-help.