StealthWare – Social Engineering Malware

Running malware for Social Engineering and Covert Operations

By: Joey Dreijer

Social Engineering and Covert Operations

Security companies provide specialised Social Engineering services

A few examples:

- (Spear) Phishing attacks: Sending falsified e-mails to individuals and/or entire companies
- USB Drop campaigns: Who doesn't want free USB sticks?
- Advanced pentest campaigns: From gathering intel to physical penetration at client facilities

Social Engineering and Covert Operations

So your client asks you to perform a social engineering test / covert ops assignment to gain access to their network, what now?

- ***How far can you go?***
- *What methodology will you use?*
- *What is your entry point?*
- ***What overly priced framework will you use?***

Having the right framework

Is it possible to 'stealthy' (and effectively) use social engineering malware for specialized security assessments?

- *W*hat existing tools are out there?
- What network/security policies will you often find on company premises?
- Can these policies be bypassed?
- Can the researched tools effectively cope with the different network architectures?

Having the right toolkit

Research focus on the limitations of existing tools

vs.

vs.

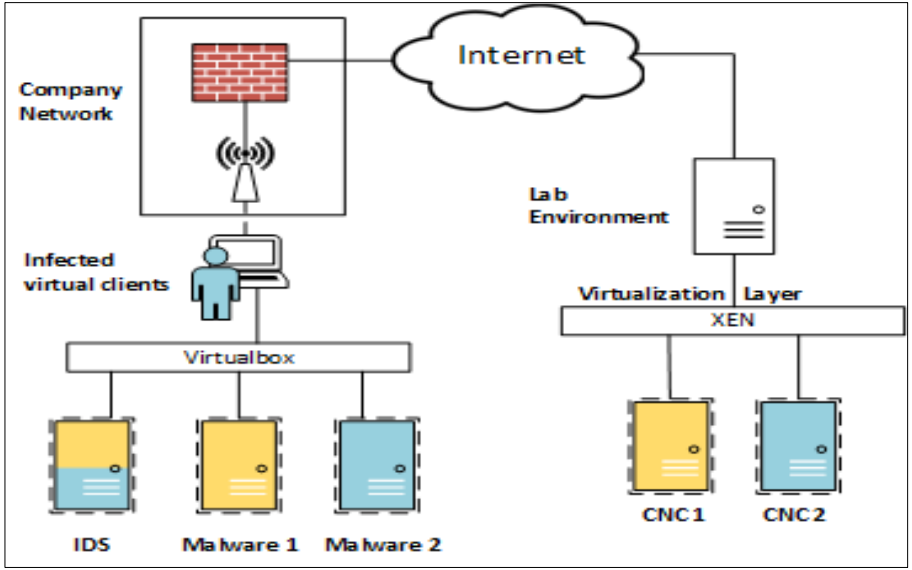
vs.

NO FOCUS ON EXPLOITATION*

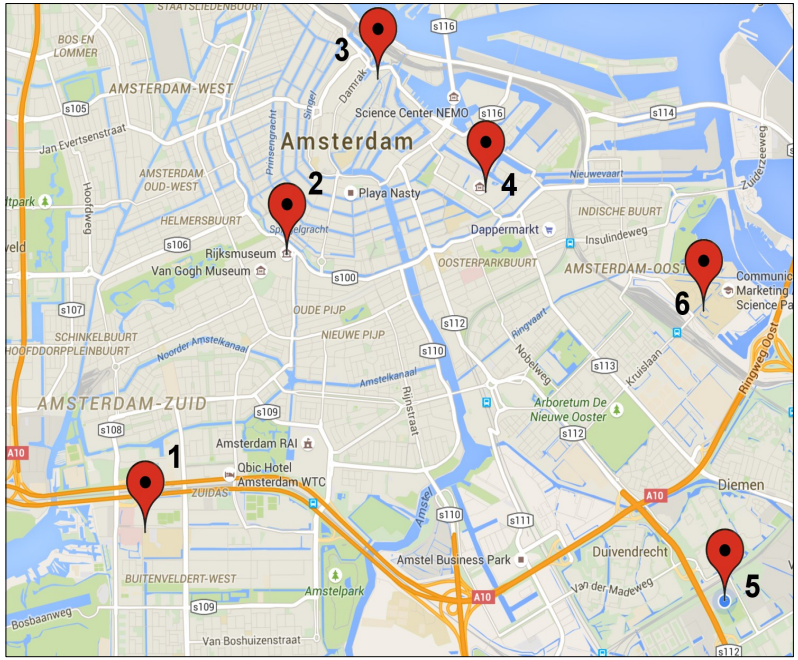
*At least, only at minimal level

Testing environment

- Infect virtual client
- Communicate with CnC server
- On-site locations with different network configurations



Testing environment



Field testing reachability

Campus networks

University labs (Proxy networks)

Open Wifi points (captive portals)

Restaurants (semi-open networks)

Company networks (ie. unauth proxies

Common network configurations

Testing different network configurations:

- Clients behind a captive portal
- Clients behind an unauthorized proxy
- Clients behind an authorized proxy

And different firewall policies:

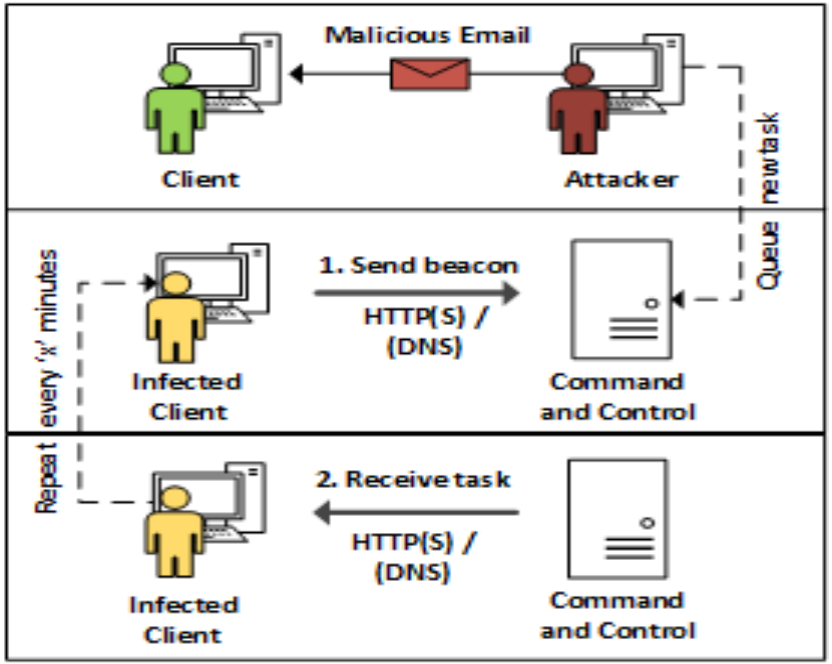
- **Open Internet:** Everything is allowed (out)
- **Limited access:** Port 80/443 (Web), 53 (DNS) and IMAP/SMTP (143, 25) are allowed. Everything else is blocked
- **Web-Only:** Only allowing 80/443 for 'daily' browsing and internal DNS

Command and control

1. Client infected via e-mail social engineering campaign

2. Client 'beacons' command and control server to ask for queued commands

3. Server replies with task or 'None'



Command and control channels

	Cobalt Strike*	ThrowBack ~Nyan**	ThrowBack
HTTP	Yes	No	No
HTTPS	Yes	Yes	Yes
DNS	Yes (TXT+A Records)	Yes (RRSIG+A Records)	No
Social Media	No	Yes (Twitter Stego)	No

* Only taking current default channels into account
** Proof-of-concept malware client based on ThrowBack backend.

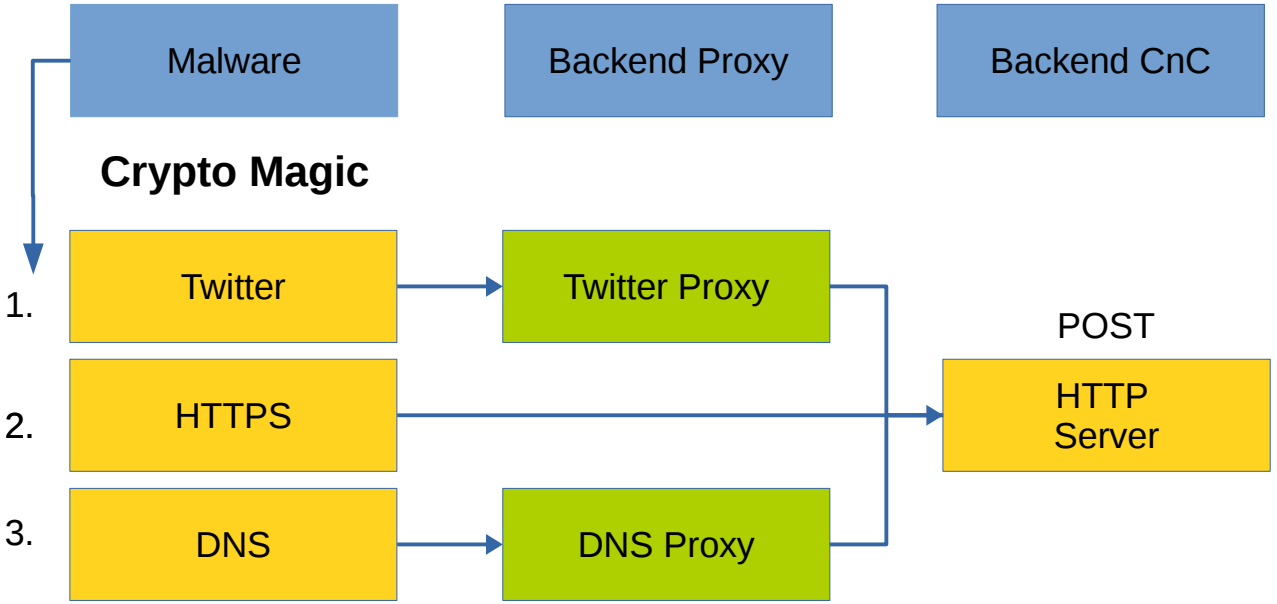
Effectiveness

None of the default clients have 'fallback' methods :(
Ie. No HTTP access? Try HTTPS. No HTTPS? Try DNS. No DNS? Try smoke signals

Requires prior knowledge of the network and/or 'HTTP is probably open anyway' statistical knowledge

Current proof-of-concept attempts to find a way out autonomously

Effectiveness proof-of-concept



Automatically attempt channel 1 and increment after failed attempts

Effectiveness (with prior-knowledge)

Network Config	Cobalt Strike	ThrowBack ~Nyan	ThrowBack
Unauth Proxy	Yes	Yes	Yes
Auth Proxy	Yes	Yes	Yes (but buggy)
Captive Portal (with DNS allowed)	No	Yes	No

Both Cobalt Strike and Throwback (Nyan) are able to get the current Windows configured proxy settings.

TODO: Still creating/visiting environments to test reachability. Full 'documented' details in report later

Detectability

Beacon detection in PCAP Files – L. van Duijn (OS3, 2014): Proof of Concept code, beacon detection still not 'ready' for realtime analysis

SSL Stripping + DPI (a la Blue Coat): *R*unning appliances as Blue Coat with SSL stripping

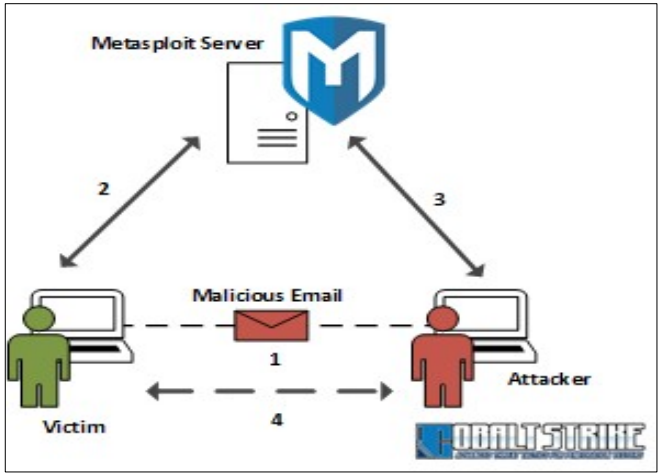
Domain 'trust' index: Monitor 'trusted' domains and analyse domain structures (ie. Runforestrunabcd.omgthisunique1928481.ru)

Anomaly detection: Ex. Beacons during the night, lunch and/or Fussball session

Static Signatures: Only available for 'known' malware. But not for ThrowBack and Cobalt Strike yet?!

Detectability

'Hindsight' methodolody: Virus Scanners / IDS systems don't detect standard beaconing. MetaSploit interpreter sessions on the other hand...



Develped SNORT (2.9+3.0Alpha) IDS Signatures for Cobalt Strike and ThrowBack HTTPS

1. Specific traffic behaviour
2. Standard response sizes

Available in the report

Detectability – Simple IDS example

Cobalt Strike HTTPS channel:

- Server response size always the same
- Client always RESETS connection (instead of ack/fin)

172.20.10.14	145.100.105.99	TCP	66	49173→443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=25…
145.100.105.99	172.20.10.14	TCP	66	443→49173 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MS…
172.20.10.14	145.100.105.99	TCP	54	49173→443 [ACK] Seq=1 Ack=1 Win=66048 Len=0
172.20.10.14	145.100.105.99	TLSv1.2	244	Client Hello
145.100.105.99	172.20.10.14	TCP	54	443→49173 [ACK] Seq=1 Ack=191 Win=30336 Len=0
145.100.105.99	172.20.10.14	TLSv1.2	140	Server Hello
172.20.10.14	145.100.105.99	TCP	54	49173→443 [ACK] Seq=191 Ack=87 Win=66048 Len=0
145.100.105.99	172.20.10.14	TLSv1.2	161	Change Cipher Spec, Encrypted Handshake Message
172.20.10.14	145.100.105.99	TCP	54	49173→443 [ACK] Seq=191 Ack=194 Win=65792 Len=0
172.20.10.14	145.100.105.99	TLSv1.2	161	Change Cipher Spec, Encrypted Handshake Message
172.20.10.14	145.100.105.99	TLSv1.2	523	Application Data
145.100.105.99	172.20.10.14	TCP	54	443→49173 [ACK] Seq=194 Ack=767 Win=31360 Len=0
145.100.105.99	172.20.10.14	TLSv1.2	251	Application Data
145.100.105.99	172.20.10.14	TLSv1.2	139	Encrypted Alert
172.20.10.14	145.100.105.99	TCP	54	49173→443 [ACK] Seq=767 Ack=477 Win=65536 Len=0
172.20.10.14	145.100.105.99	TCP	54	49173→443 [FIN, ACK] Seq=767 Ack=477 Win=65536 Len=0
172.20.10.14	145.100.105.99	TCP	54	49173→443 [RST, ACK] Seq=768 Ack=477 Win=0 Len=0
145.100.105.99	172.20.10.14	TCP	54	443→49173 [ACK] Seq=477 Ack=768 Win=31360 Len=0

Bypassing limited detection

Improving ThrowBack and creating NYAN Edition

1. Randomize content (length) request and response

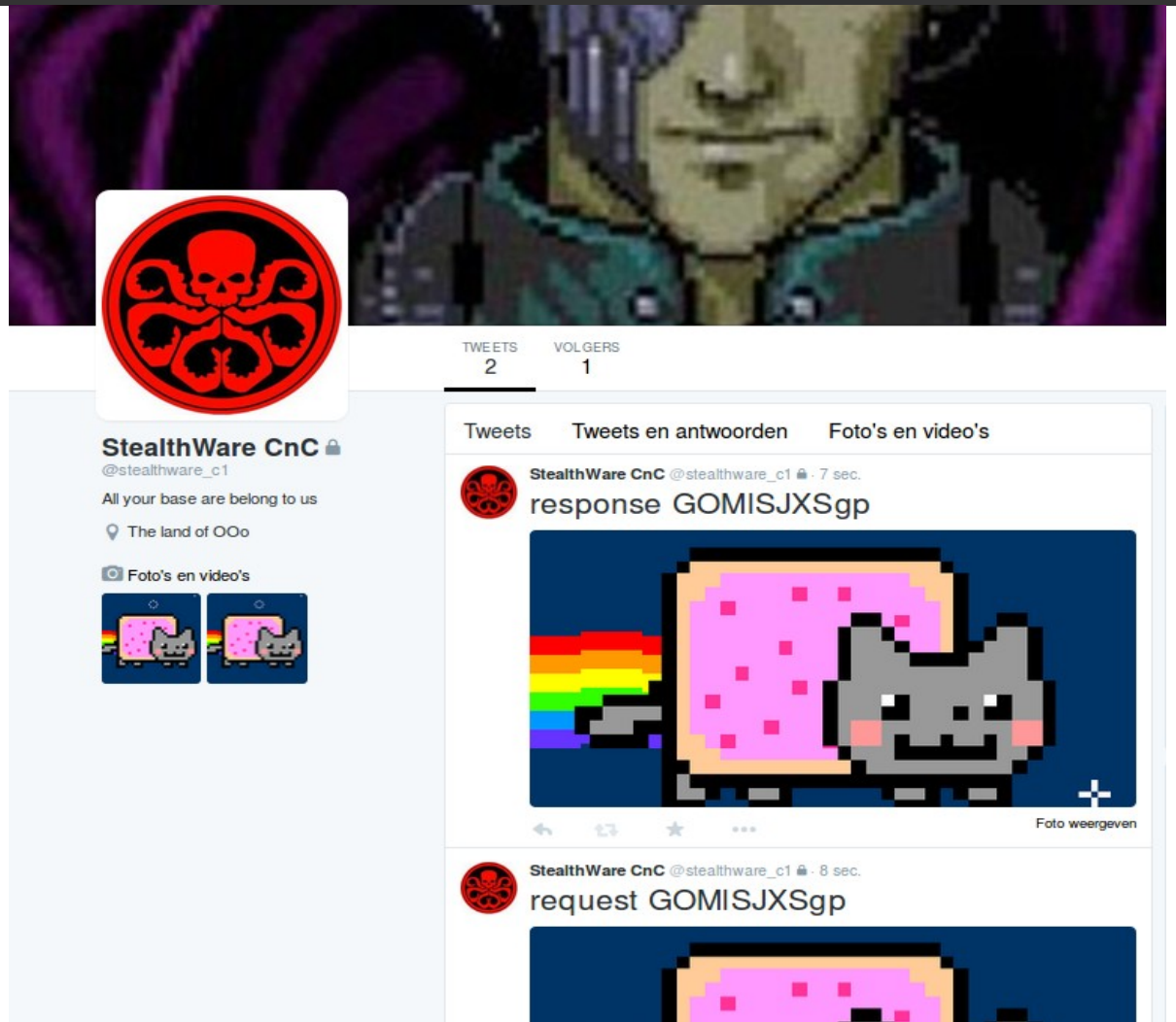
2. Random beacon timers (ie. Set time + 1% - 80%)

3. Multiple 'bogus' sessions to prevent specific behavior signatures

4. DNS: Base64 in TXT records is an old trick. Put your data in a valid RRSIG format for compliancy!

5. Using trusted channels/domains for Command and Control

StealthWare – Social Engineering Malware



Conclusion

Not many frameworks available (and commercial)

Cobalt Strike works in most scenarios (with prior-knowledge)

Network detection can be very easy, depending on the monitoring tools made available (remember hindsight?)

Current proof-of-concept bypassing common detection and network limitations. ***Good*** anomaly detection still rare

WIP code available on GitHub to test real-life monitoring capabilities