OFFENSIVE HUNTING

Using Blue Team techniques in Red Team Ops

Mark Bergman
Marc Smeets
June 2022

HACK IN PARIS
CYBERSECURITY EVENT

OUTFLANK
clear advice with a hacker mindset

ABOUT YOUR SPEAKER

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

- Infosec class of 1998 (hobby) / 2006 (professionally)
- Red Team operator, tool builder, trainer, some blue Threat Hunting experience

Outflank - @OutflankNL

- Specialised in Red Teaming, trainings and offensive security tooling
- Public tools and blogs via:
  - https://outflank.nl/blog
  - https://github.com/OutflankNL

OFFENSIVE INFRA – TYPICAL SETUP FOR 1 OPERATION

Command and Control

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

Fake identities

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

Tracking

- Tracking pixels (10+)

Delivery

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

Generic backend components

- Communication channels (2+)
- Test environments (1+)
- Log aggregation (1+)

OFFENSIVE INFRA - TYPICAL CHALLENGES

Oversight

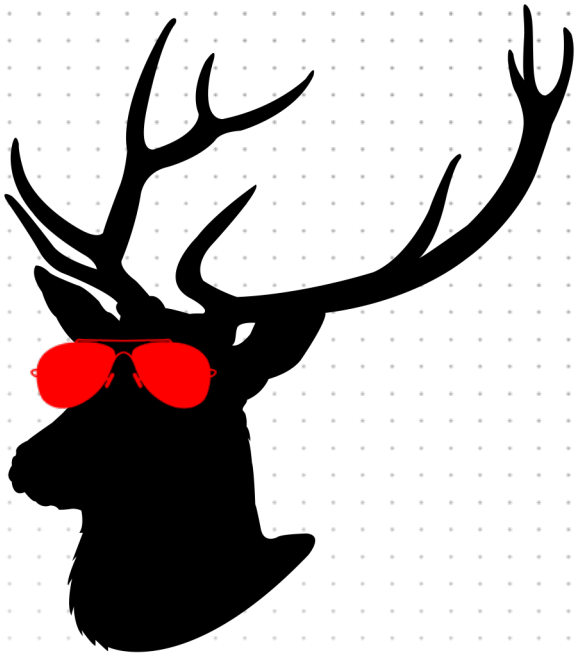
Insight

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

TOOLING -> REDELK

+

=

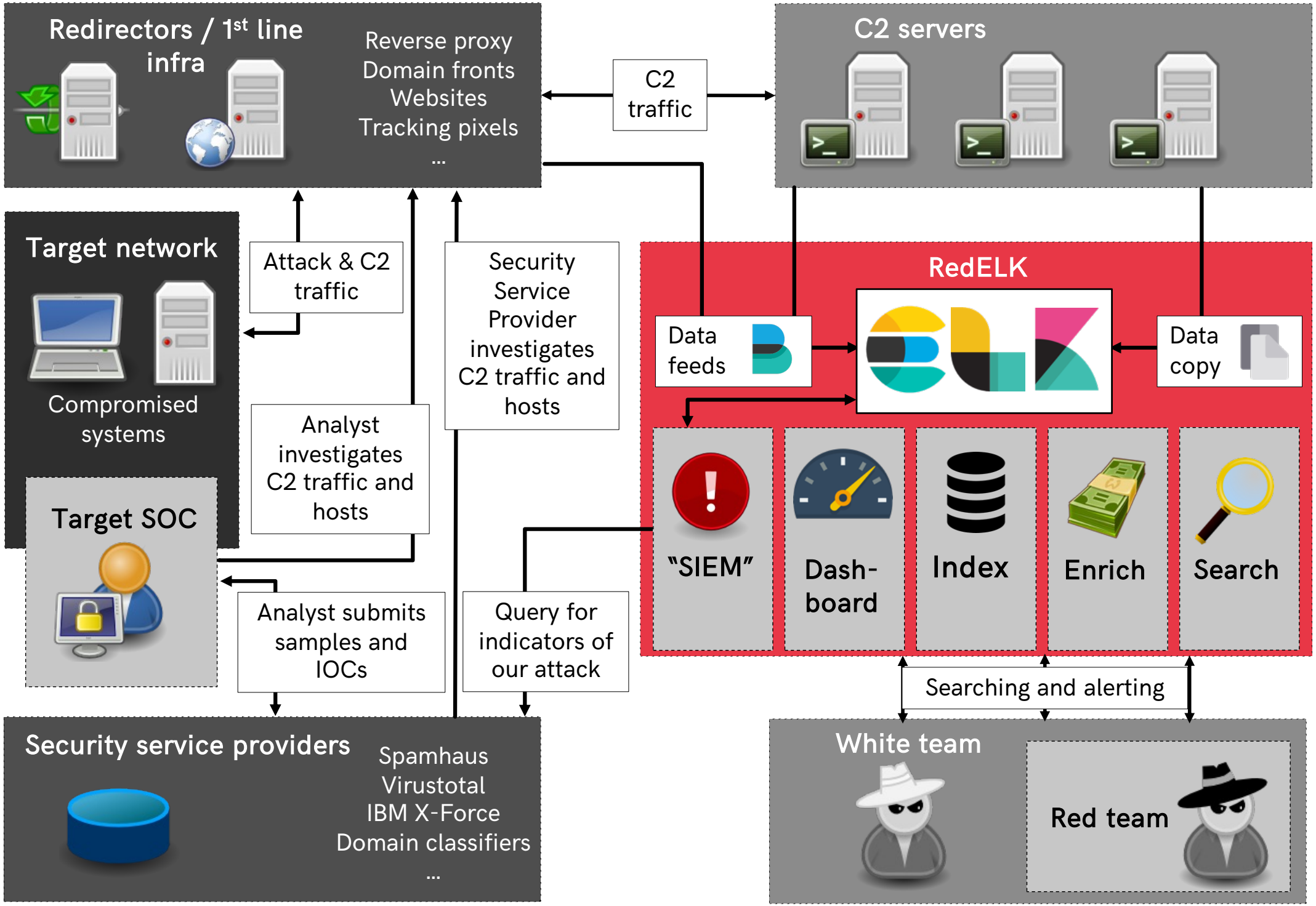


https://github.com/outflanknl/RedELK/

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

https://outflank.nl/blog/2020/02/28/redelk-part-2-getting-you-up-and-running/

https://outflank.nl/blog/2020/04/07/redelk-part-3-achieving-operational-oversight/


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

INDICATORS OF INVESTIGATION

Direct actions to our offensive infrastructure	
Rogue user-agents	Traffic from analyst based on user-agent, e.g. python, curl, Slack, WhatsApp, etc
Rogue IP traffic	Any traffic going to C2 backend but not known by RedELK config
Deflected traffic	Traffic deflection decisions made by smart redirector logic
Known blue	Any access to your infra from known Blue Team IP ranges

Indirect actions to our offensive infrastructure	
AV hash	Hash of our malware is known at Virus Total, Hybrid Analyses or IBM X-Force
Infra blacklist	IP, URL of TLS cert of our infra gets on a blacklist
Domain classification	Domain of our infra gets classified as bad, or classification changes

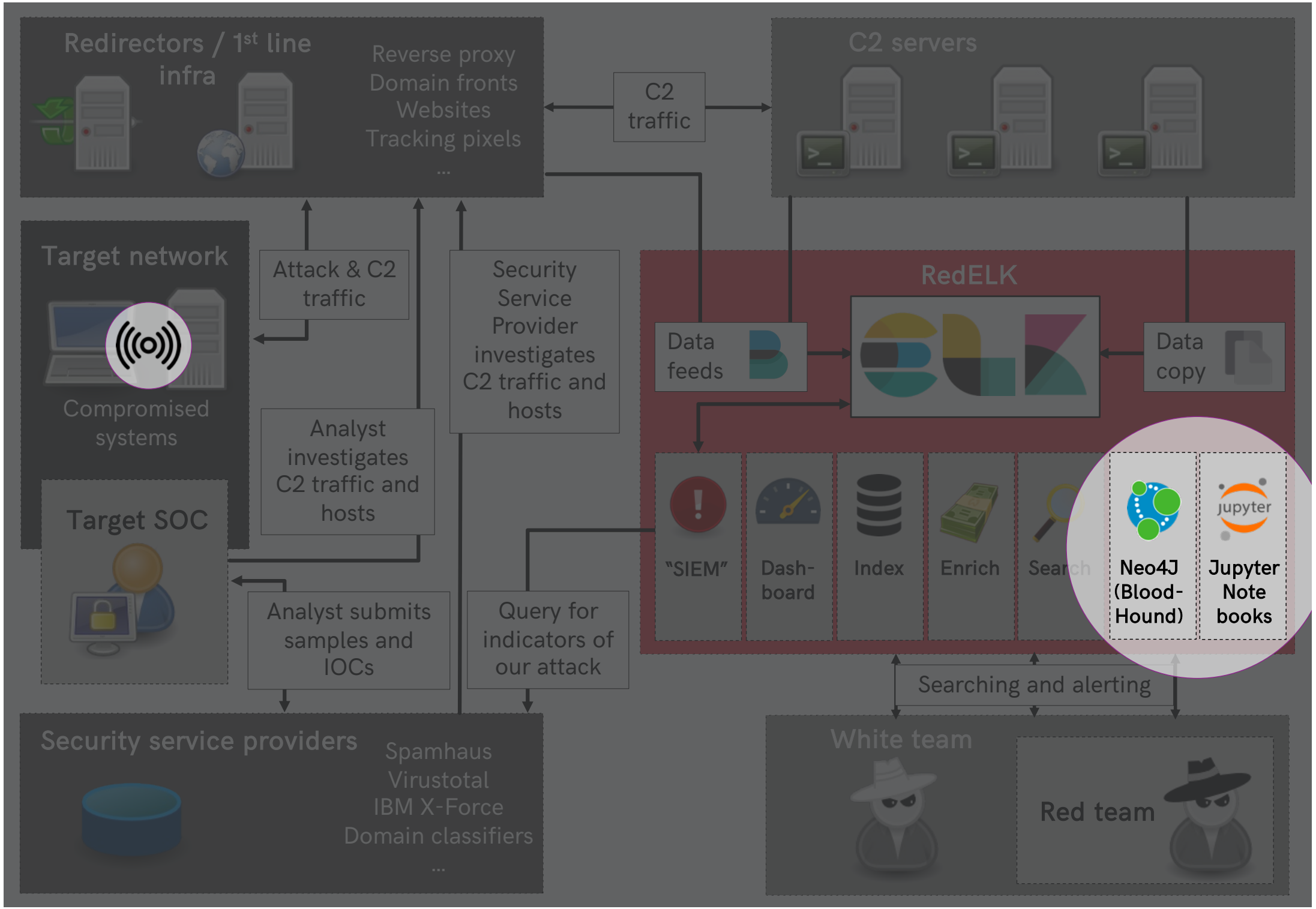
NEXT STEPS

We are interested in optimising our operation and in detecting blue

Blue generally progresses over two axes:

1. More data and rule based detections
   - Detection Engineering & Threat Detection
   - Easy for us with current RedELK

2. Improve quality and usability of existing data sources
   - Threat Hunting
     - Hypotheses based approach to catch attacks that slipped through existing detections and controls
     - Requires free format tools to interact with data
   - Hard for us, needs different approach


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
Neo4J (Blood-Hound)
Jupyter Note books
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

EXAMPLE 1

OPTIMISING THE OPERATION

FINDING PATH TO DA

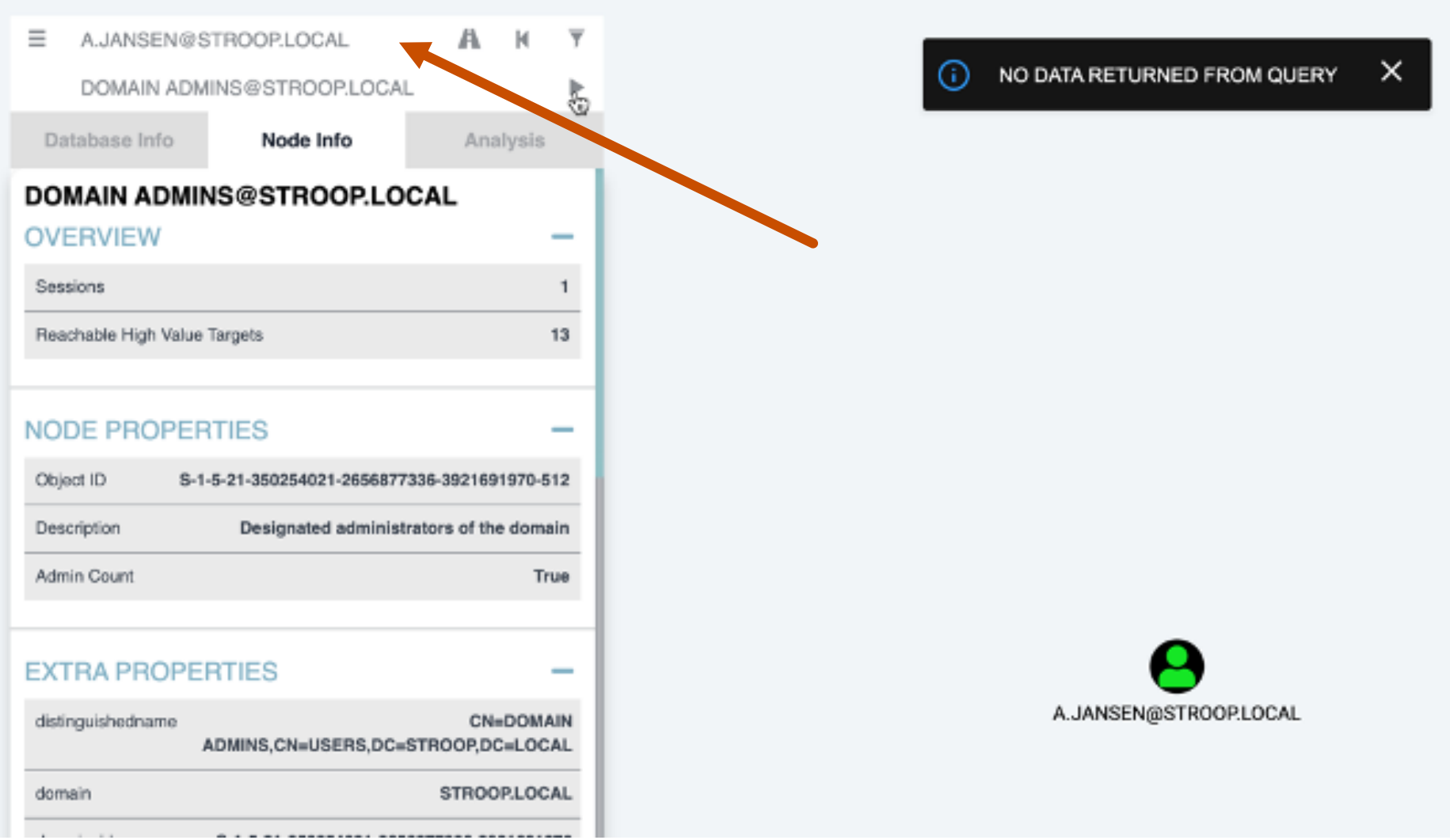
OPERATION PRE REDELK

Look at a PS

06/22 09:49

```
17763; beacon arch: x64 (x64)
06/22 09:49:25 UTC [input] <Outflank> ps
06/22 09:49:25 UTC [task] <T1057> Tasked beacon to list processes
06/22 09:49:25 UTC [checkin] host called home, sent: 12 bytes
06/22 09:49:25 UTC [output]
[System Process]        0       0
System  0       4       x64     NT AUTHORITY\SYSTEM     0
Registry        4       88      x64     NT AUTHORITY\SYSTEM     0
smss.exe        4       376     x64     NT AUTHORITY\SYSTEM     0
csrss.exe       476     492     x64     NT AUTHORITY\SYSTEM     0
wininit.exe     476     568     x64     NT AUTHORITY\SYSTEM     0
csrss.exe       560     576     x64     NT AUTHORITY\SYSTEM     1
winlogon.exe    560     664     x64     NT AUTHORITY\SYSTEM     1
services.exe    568     680     x64     NT AUTHORITY\SYSTEM     0
rdpclip.exe     872     1160    x64     STROOP\A.Jansen 2
sihost.exe      1760    2032    x64     STROOP\A.Jansen 2
svchost.exe     680     1744    x64     STROOP\A.Jansen 2
svchost.exe     680     4944    x64     STROOP\A.Jansen 2
taskhostw.exe   1492    148     x64     STROOP\A.Jansen 2
svchost.exe     680     5348    x64     NT AUTHORITY\SYSTEM     0
ctfmon.exe      5348    5408    x64     STROOP\A.Jansen 2
explorer.exe    5660    [illegible]     x64     STROOP\A.Jansen 2
smartscreen.exe 876     5528    x64     STROOP\A.Jansen 2
svchost.exe     680     6656    x64     STROOP\A.Jansen 2
ShellExperienceHost.exe 876     7072    x64     STROOP\A.Jansen 2
SearchUI.exe    876     2228    x64     STROOP\A.Jansen 2
RuntimeBroker.exe       876     6292    x64     STROOP\A.Jansen 2
RuntimeBroker.exe       876     6408    x64     STROOP\A.Jansen 2
SecurityHealthSystray.exe       5720    5764    x64     STROOP\A.J
SecurityHealthService.exe       680     4508    x64     NT AUTHORI
RuntimeBroker.exe       876     3972    x64     STROOP\A.Jansen 2
svchost.exe     680     7208    x64     NT AUTHORITY\SYSTEM     0
cmd.exe 5720    7804    x64     STROOP\A.Jansen 2
conhost.exe     7804    7812    x64     STROOP\A.Jansen 2
svchost.exe     680     7908    x64     STROOP\A.Jansen 2
```

COPY AND PASTE TO BLOODHOUND



LETS MAKE REDELK AND BLOODHOUND PLAY (1/4)

Load Jupyter notebooks

Connect to ElasticSearch

Connect to Neo4J

```
# define the connection to our ES instance

urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)
ssl_context = create_ssl_context()
ssl_context.check_hostname = False
ssl_context.verify_mode = ssl.CERT_NONE

es = Elasticsearch(
    ['redelk-elasticsearch'],
    http_auth=(CredESUsername, CredESPassword),
    scheme="https",
    port=9200,
    ssl_context=ssl_context
)

# Now test the connection to ES -- this should give output

if es.ping():
    print("ES connection successful")
else:
    raise ValueError("ES Connection failed")
```

```
ES connection successful
```

```
from py2neo import Graph
#g = Graph("bolt://206.189.85.93:7687", auth=("neo4j", "BloodHound"))
g = Graph("bolt://redelk-bloodhound:7687", auth=(CredNeo4jUsername, CredNeo4jPassword))

CurrentDomain = getCurrentDomain()
print("[ ] Got Domain %s"%CurrentDomain)
SUFFIX = CurrentDomain
```

```
[ ] Got Domain STROOP.LOCAL
```

LETS MAKE REDELK AND BLOODHOUND PLAY (2/4)

WRITE SOME UGLY PYTHON

- Query all PS commands
- Parse the lines
- Format a datastructure
- Query neo4j

```python
def processListTable_from_queryRes(linesResult):
    ## now for each resultline parse processes
    overallListProcesses = []  # combine all PS in timeframe to one object
    for line in linesResult:
        x=line
        processes = get_value('_source.implant.output',line)
        if processes == None:
            print("[e] %s"%(line))
            #continue
        else:
            #print(processes)
            for proc in processes.split('\n'):
                pItem = proc.split('\t')
                processD = {}
                processD['proc_Name'] = pItem[0] if 0 < len(pItem) else None
                processD['proc_PPID'] = pItem[1] if 1 < len(pItem) else None
                processD['proc_PID'] = pItem[2] if 2 < len(pItem) else None
                processD['proc_arch'] = pItem[3] if 3 < len(pItem) else None
                processD['proc_user'] = pItem[4] if 4 < len(pItem) else None
                processD['proc_session'] = pItem[5] if 5 < len(pItem) else None
                processD['time'] = None
                try:
                    processD['target_user'] = get_value('_source.user.name',line)
                    processD['target_hostname'] = get_value('_source.host.name',line)
                    processD['target_os'] = get_value('_source.host.os.family',line)
                    processD['target_osversion'] = get_value('_source.host.os.version',line)
                    processD['redelk_id'] = get_value('_id',line)
                    processD['redelk_timestamp'] = get_value('_source.@timestamp',line)
                    processD['timestamp'] = datetime.datetime.strptime(processD['redelk_timestamp'], "%Y-%m-%dT%H:%M:%S
                except KeyError:
                    pass
                try:
                    if int(processD['proc_PID']) > 0: overallListProcesses.append(processD)
                except TypeError:
                    pass

    pd_processes = pd.DataFrame(json_normalize(overallListProcesses))
    return(pd_processes)

# day by day
#today = datetime.datetime.now()
#testDate = today - datetime.timedelta(hours=24)
testDate = datetime.datetime(2022,6,22,9,45)
testEndDate = datetime.datetime(2022,6,22,11,0)

mail = False
mailtext = ""
debug = ""
while testDate <= testEndDate:
    templines = getRedELKProcessesForDT(testDate,1)
    if len(templines) > 0:
        print("\n[ELK] Testing PS around %s"%testDate)
        tempproc = processListTable_from_queryRes(templines)
        unique_usernames = list(set(tempproc['proc_user'].to_list()))
        for u in unique_usernames:
            if not u: continue
            #if u[:len(PREFIX)] == PREFIX:
            if '\\' in u:
                if u.split('\\')[0] in SUFFIX:
                    bh_username = "%s@%s"%(u.split('\\')[1].upper(),SUFFIX.upper())
                    print("[NEO]  looking for %s"%bh_username)
                    bh_path = bh_SetUserCompromised_GetUserPathTo(g,bh_username,compromiseToggle=False)
                    if bh_path:
                        debug = tempproc
                        print("[###]  PATH TO DOMAIN ADMINS FOUND!!!!!!")
                        #print(bh_path)
    testDate = testDate + datetime.timedelta(minutes=1)
```

LETS MAKE REDELK AND BLOODHOUND PLAY (3/4)

GET PS OUTPUT FROM REDELK AND PARSE THAT

	proc_Name	proc_PPID	proc_PID	proc_arch	proc_user	proc_session	time	target_user	target_hostname	target_os	targ
0	System	0	4	x64	NT AUTHORITY\SYSTEM	0	None	SYSTEM *	I-win224	Windows	
1	Registry	4	88	x64	NT AUTHORITY\SYSTEM	0	None	SYSTEM *	I-win224	Windows	
2	smss.exe	4	376	x64	NT AUTHORITY\SYSTEM	0	None	SYSTEM *	I-win224	Windows	
3	csrss.exe	476	492	x64	NT AUTHORITY\SYSTEM	0	None	SYSTEM *	I-win224	Windows	
4	wininit.exe	476	568	x64	NT AUTHORITY\SYSTEM	0	None	SYSTEM *	I-win224	Windows	
...	...	...	...	...	...	...	...	...	...	...	
125	mmc.exe	5720	5952	x64	STROOP\A.Jansen	2	None	SYSTEM *	I-win224	Windows	
126	Taskmgr.exe	5720	7460	x64	STROOP\A.Jansen	2	None	SYSTEM *	I-win224	Windows	
127	procexp64.exe	5720	6764	x64	STROOP\A.Jansen	2	None	SYSTEM *	I-win224	Windows	
128	svchost.exe	680	7824	x64	NT AUTHORITY\SYSTEM	0	None	SYSTEM *	I-win224	Windows	
129	WmiPrvSE.exe	876	6468	x64	NT AUTHORITY\SYSTEM	0	None	SYSTEM *	I-win224	Windows	

LETS MAKE REDELK AND BLOODHOUND PLAY (4/4)

FOR EACH USER

- Set owned in BloodHound
- Query for path to DA

```
[ELK] Testing PS around 2022-06-22 09:49:00
[NEO]  looking for A.JANSEN@STROOP.LOCAL

[ELK] Testing PS around 2022-06-22 09:51:00
[NEO]  looking for J.DROSTE@STROOP.LOCAL
[NEO]  looking for A.JANSEN@STROOP.LOCAL

[ELK] Testing PS around 2022-06-22 10:54:00
[NEO]  looking for A.JANSEN@STROOP.LOCAL
[NEO]  looking for ADMIN-A.JANSEN@STROOP.LOCAL
[###]  PATH TO DOMAIN ADMINS FOUND!!!!!!
```

EXAMPLE 2

DETECTING BLUE

SUSPICIOUS CHANGING OF PASSWORDS

KRBTGT RESET

```
[redacted] 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
```

PASSWORD RESET OF SPECIFIC ACCOUNTS

```
beacon> help BlueCheck
Synopsis: BlueCheck

Use Active Directory Service Interfaces (ADSI) to query for user password changes.

beacon> BlueCheck krbtgt
[*] Tasked beacon to spawn BlueCheck
[+] host called home, sent: 103479 bytes
[+] received output:
[+] BLUECHECK: stroop.local\krbtgt password last changed at: 1/27/2020 8:41:40 AM, accoun
beacon> BlueCheck admin-w.trommel
[*] Tasked beacon to spawn BlueCheck
[+] host called home, sent: 103488 bytes
[+] received output:
[+] BLUECHECK: stroop.local\admin-w.trommel password last changed at: 1/27/2020 8:53:19 A

[L-WIN223] w.tax/6340
beacon>
```

prepare a query and get all bluecheck items which are security tools

In [18]:
```
import datetime
from elasticsearch.helpers import scan
from pandas import json_normalize

def getRedELKLinesForDT(day,delta=5):
    QUERY = "bluechecktype:\"sectools\""
    INDEX = "bluecheck-*"
    py_timestamp = day
    fromtime =  (py_timestamp - datetime.timedelta(days=0)).strftime("%Y-%m-%dT%H:%M:%S.%fZ")
    totime   =  (py_timestamp + datetime.timedelta(minutes=delta)).strftime("%Y-%m-%dT%H:%M:%S.%fZ")
    jsonQuery = guiQueryWindow(QUERY,fromtime,totime)
    cnt = 0
    linesResult = []
    for line in scan(es,query=jsonQuery,index=INDEX):
        linesResult.append(line)
        cnt += 1
    return(linesResult)

# day by day
#today = datetime.datetime.now()
#testDate = today - datetime.timedelta(hours=24)
testDate = datetime.datetime(2022,6,22,9,45)
testEndDate = datetime.datetime(2022,6,22,11,0)

l = getRedELKLinesForDT(testDate,99999999)
```

Use the first line as baseline

In [19]:
```
BaseLine = get_value("_source.implant.output",l[0])
```

In [20]:
```
print(BaseLine)
```

```
received output:
[+] BLUECHECK Security Tools Check:
[+] Security products found: 2
    ProcessID:   2724
    Vendor:      Sysinternals - www.sysinternals.com
    Product:     System activity monitor

    ProcessID:   5528
```

In [21]:
```
def parseBlueCheck_stc(valIn):
    content = valIn.split('\n')
    content[:] = [x for x in content if not x.startswith('[')]
    content2 = '\n'.join(content).strip().split("\n\n")
    proclist = []
    procdlist = []
    for content2item in content2:
        proc = {}
        worklist = content2item.split('\n')
        for workitem in worklist:
            if ":\t " in workitem:
                k,v = workitem.split(":\t ",1)
                proc[k.strip()] = v.strip()
        procdlist.append(proc)
        proclist.append(proc.get('Product'))
    return(procdlist,proclist)

# set baseline
BaseLineprocdlist, BaseLineproclist = parseBlueCheck_stc(BaseLine)
```

In [22]:
```
BaseLineproclist
```

Out[22]:
```
['System activity monitor', 'Windows Defender SmartScreen']
```

In [23]:
```
BaseLineprocdlist
```

Out[23]:
```
[{'ProcessID': '2724',
  'Vendor': 'Sysinternals - www.sysinternals.com',
  'Product': 'System activity monitor'},
 {'ProcessID': '5528',
  'Vendor': 'Microsoft Corporation',
  'Product': 'Windows Defender SmartScreen'}]
```

now loop over the rest and alarm when we find something changing

```
In [24]: print("[%s] baseline set to: %s"%(get_value("_source.c2.timestamp",l[0]),BaseLineproclist))


         for line in l:
             line_output = get_value("_source.implant.output",line)
             procdl,procl = parseBlueCheck_stc(line_output)
             newSProc = False
             newSProcName = ""
             sleep(1)
             for proc in procl:
                 #print(proc)
                 if proc not in BaseLineproclist:
                     newSProc = True
                     newSProcName = proc
             if newSProc:
                 print("[%s] found new product: %s"%(get_value("_source.c2.timestamp",line),newSProcName))
             else:
                 print("[%s] nothing new "%get_value("_source.c2.timestamp",line))

         [06/22 10:44:48] baseline set to: ['System activity monitor', 'Windows Defender SmartScreen']
         [06/22 10:44:48] nothing new
         [06/22 10:46:04] nothing new
         [06/22 10:52:01] found new product: Sysinternals Process Explorer

In [25]: newSProcName

Out[25]: 'Sysinternals Process Explorer'

In [ ]:
```

EXAMPLE 3

DETECTING BLUE

SSL INTERCEPTION ENABLED

START OF SSL INTERCEPTION

```
beacon> help CertCheck
Synopsis: CertCheck https://www.example.com

Use WinHTTP to query SSL certificate information from a specified url.

beacon> CertCheck https://www.outflank.nl
[+] host called home, sent: 6150 bytes
[+] received output:
[+] BLUECHECK SSL Certificate: https://www.outflank.nl
[+] Subject Information:
*.outflank.nl
[+] Issuer Information:
US
Let's Encrypt
R3
[+] Check Finished
```

START OF SSL INTERCEPTION

```
beacon> help CertCheck
Synopsis: CertCheck https://www.example.com

Use WinHTTP to query SSL certificate information from a specified url.

beacon> CertCheck https://www.outflank.nl
[+] host called home, sent: 6150 bytes
[+] received output:
[+] BLUECHECK SSL Certificate: https://www.outflank.nl
[+] Subject Information:
*.outflank.nl
[+] Issuer Information:
NL
Limburg
R3
[+] Check Finished
```

INDICATORS OF INVESTIGATION - INTERNAL

TYPE OF CHECK	DETAIL
Password reset	Password of critical accounts reset around same time
TLS interception	Unexpected change of TLS cert of a domain
Security tool	Unexpected change of AV / EDR tools installed
Log forwarding	Unexpected change of log forwarding config
Security config	Unexpected change of specific security parameters, e.g. change of accounts or (GPO change) of specific settings
Accounts login	Unexpected change of (type) of accounts logging in

WHAT IS BLUE DOING?

COPY!

LEANING FROM BLUE

- Send sample to cloud for analysis
- Heuristic analysis
- Application whitelisting
- Artificial Intelligence

- Report back EDR versions and 'know how to defeat this'
- Continues monitoring on compromised hosts
- Alerting or serving payloads on whitelist
- WHAHAHA

SUMMARY

Goal of Red Teaming is to make Blue Teams better

Dear red, RedELK is here to help you

Dear blue, think of your OPSEC

https://github.com/OutflankNL/RedELK

https://outflank.nl/blog/

OUTFLANK

clear advice with a hacker mindset

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

@MarcOverIP

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

@xychix