
black hat®
ARSENAL 2022
DECEMBER 7-8
EXCEL LONDON / UK
#BHEU @BlackHatEvents

Invoke-DNSteal:

Exfiltrating DNS information "Like a Boss"

Whoami

Joel Gámez Molina // @JoelGMSec

- Security Consultant @ Deloitte - Red Team Operations
- System adminitrator for more than 10 years
- Ex-CTO of the Cyberguard startup (2 years)
- Professor of Ethical Hacking, Pentesting and PowerShell
- Speaker at national & internacional cibersecurity conferences
- Creator and writer of the blog darkbyte.net
- Hacking tools programmer (AutoRDPwn, Cloudtopolis, EvilnoVNC, PyShell, PSRansom..)

Prologue

During Red Team exercises, there are different ways to exfiltrate sensitive information to the outside.

One of the most underrated *covert channels* is the domain name system (DNS).

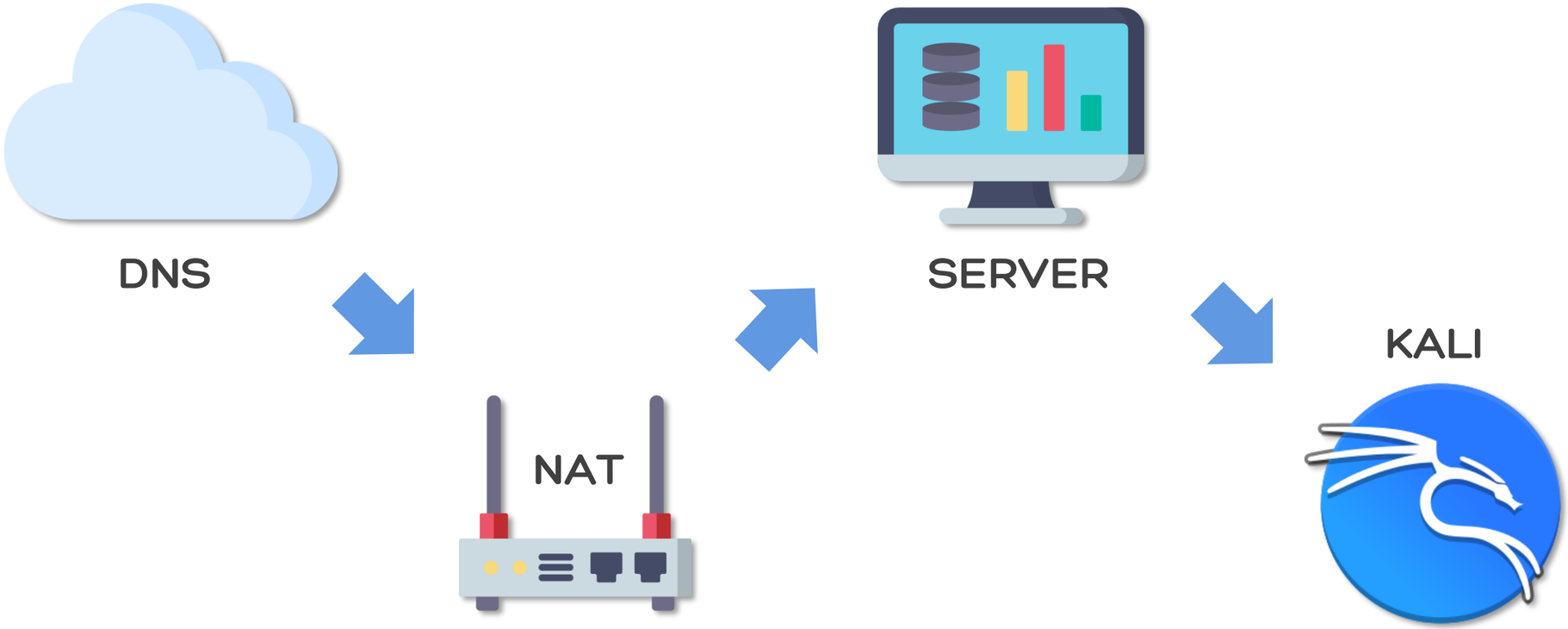
Currently, a wide variety of tools exist for this purpose in different programming languages.

Unlike all other legitime channels, most firewalls allow DNS traffic.

History

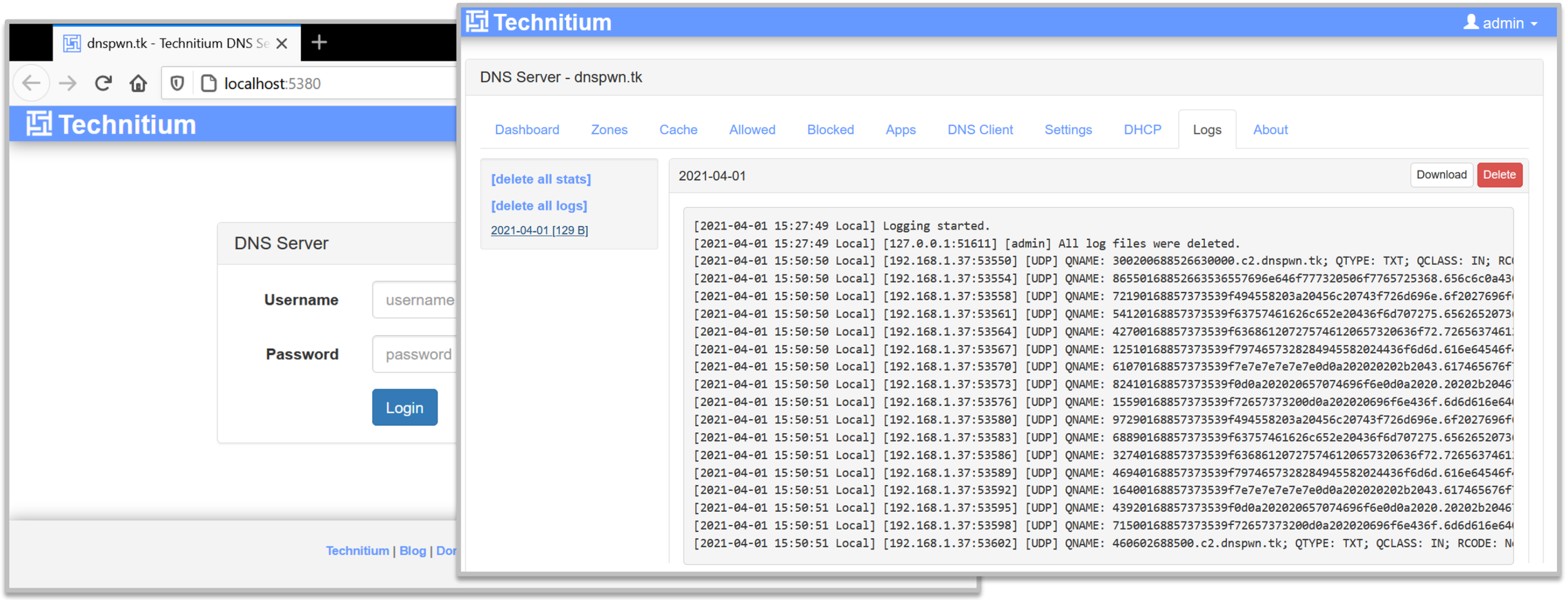
- A customer requests to check the security of its DNS anti-exfiltration system
- The tests should cover the whole protocol spectrum (TCP/UDP)
- The origin of the connection would be made from a secure environment (Citrix)
- An intermediate firewall will filtrate all outbound connections
- All security measures will be enabled (DLP, EDR, etc)
- All possible DNS records would be used (A, TXT, SOA, MX..)

Laboratory



Public DNS

Fake Server



PoC Tools

- https://github.com/iagox86/dnscat2
- https://github.com/Arno0x/DNSExfiltrator
- https://github.com/IncideDigital/Mistica
- https://github.com/cpl/exodus
- https://github.com/1N3/PowerExfil
- https://github.com/ytisf/PyExfil

Comparison

DNSCat2

- Good for receiving a reverse shell
- Default encrypted communication (SHA3)
- Ability to receive files and create tunnels
- High flexibility (time between queries, key, query type, port)

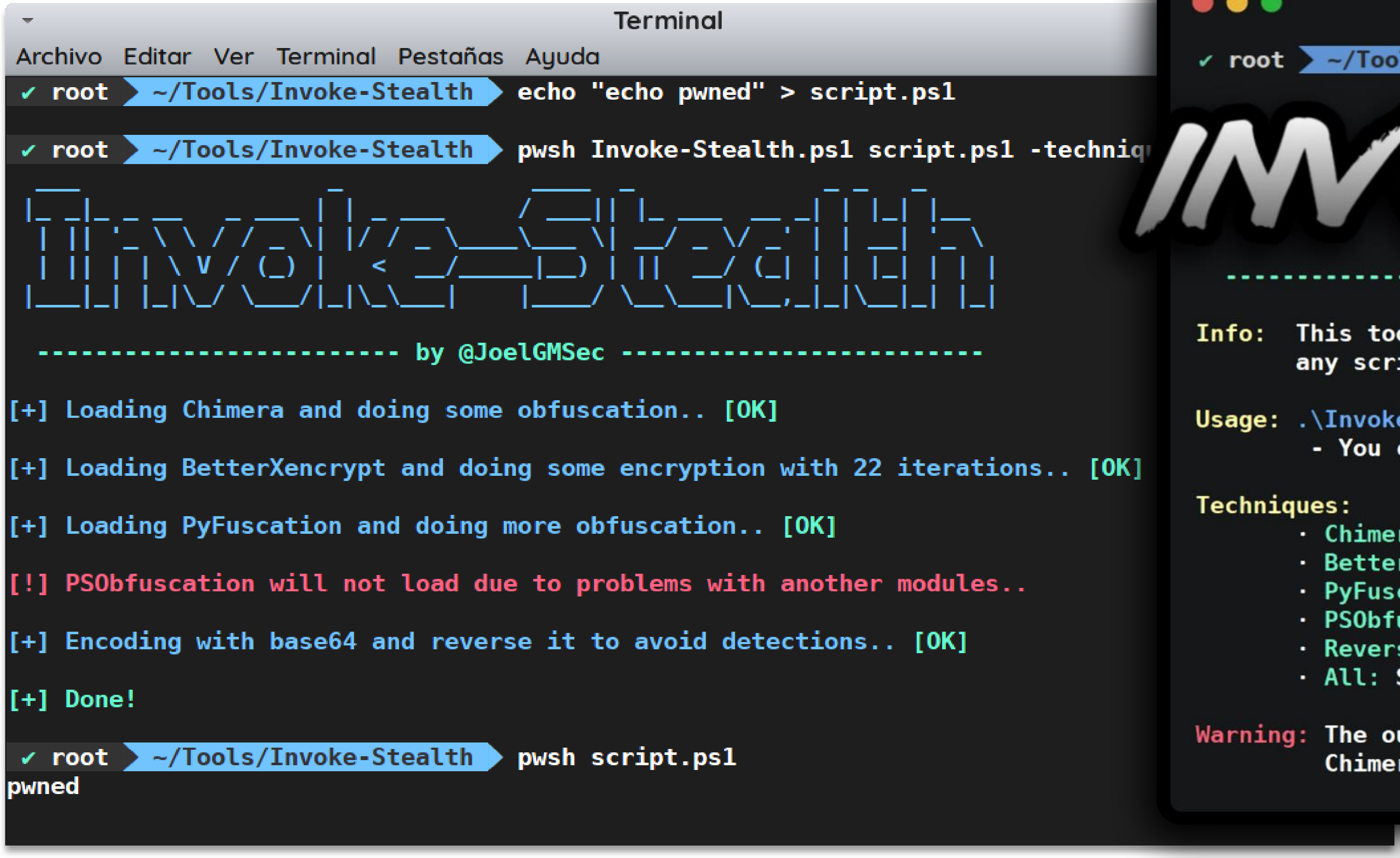
DNSExfiltrator

- Good for sending files over DNS
- Default encrypted communication (RC4)
- Can only be used for sending and receiving files
- Quite flexible (time between queries, key, encryption)

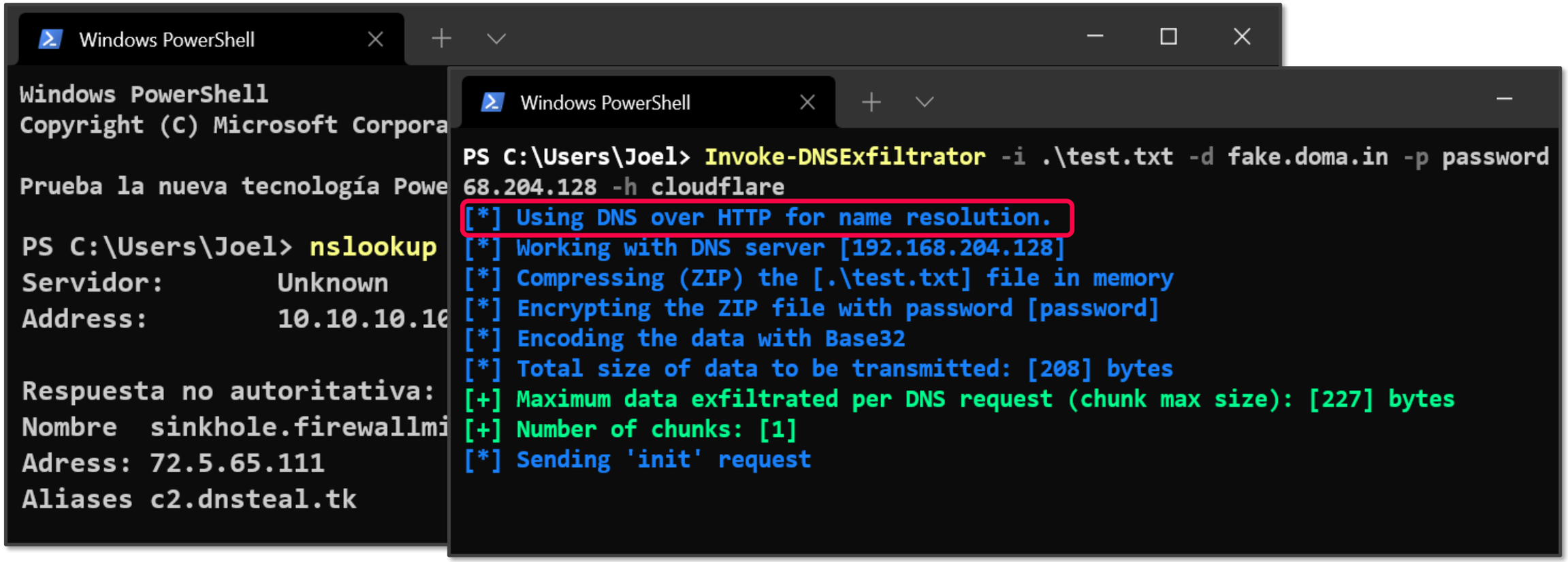
Mística

- Good for port forwarding through DNS
- Encrypted communication by default (RC4)
- Possibility to receive files and command & control
- Lots of flexibility (key, query type, port)

Next Try



Next Try



Do it yourself

- Lightweight, high compatibility and with as few dependencies as possible
- Preferably use native system functions like nslookup
- Support for DNS on both UDP and TCP
- Customisable query length to control overall size
- Randomised timeouts to avoid behavioural detections
- Possible evasion techniques using random elements

Do it yourself

```
def request(self, ip):
 if self.datatxt:
  packet=''

  if "-udp" in mode:
   packet+=self.data[:2] + "\x81\x80"
   packet+=self.data[4:6] + self.data[4:6] + '\x00\x00\x00\x00'
   packet+=self.data[12:]
   packet+='\xc0\x0c'
   packet+='\x00\x01\x00\x01\x00\x00\x00\x3c\x00\x04'

  if "-tcp" in mode:
   hexdata= ord(self.data[1]) + 0x10
   packet+=self.data[0] + chr(hexdata)
   packet+="\x00\x01\x85\x80\x00\x01\x00\x01"
   packet+=self.data[10:]
   packet+='\xc0\x0c'
   packet+='\x00\x01\x00\x01\x00\x00\x00\x00\x00\x04'

  packet+=str.join('',[chr(int(x)) for x in ip.split('.')])
 return packet
```


PS C:\Windows\System32\Pwned> .\Invoke-DNSteal.ps1 -h
INVOKE-DNSTEAL
---------- by @JoelGMSec ----------
Info: This tool helps you to exfiltrate data through DNS protocol
and lets you control the size of queries using random delay
Usage: .\Invoke-DNSteal.ps1 -t target -p payload -l lenght
-s server -tcponly true/false -min 3000 -max 5000
Parameters:
• Target: Domain target to exfiltrate data
• Payload: Payload to send over DNS chunks
• Lenght: Lenght of payload to control data size
• Server: Custom server to resolve DNS queries
• TcpOnly: Set TcpOnly to true or false
• Delay Min: Min delay time to do a query in ms
• Delay Max: Max delay time to do a query in ms
• Random: Use random domain name to avoid detection
Warning: The lenght (payload size) must be between 4 and 240
The process time will increase depending on data size

Things to do

- Improve the payload compression system
- Encrypt information by default (RC4, AES)
- Use additional records (TXT, SOA, NS..)
- Support for sock tunnelling and port forwarding
- Command & Control via PowerShell
- Linux client and Python 3 support

Thank you!

Questions?

@Joel6MSec