Engenharia reversa e análise de malware

Cristian Souza
@cristianzsh

https://cristian.sh

AGENDA

1. Engenharia reversa
2. Análise de malware
3. Análise estática
4. Análise dinâmica
5. Detecção de malware
6. Técnicas de evasão
7. Por onde começar?

Engenharia reversa

Uma visão prática

Engenharia reversa

- Processo de identificar o funcionamento de um dispositivo ou sistema por meio da análise de sua estrutura interna e operação.
- Não se aplica apenas ao mundo computacional.
- Técnica altamente utilizada no meio militar.
- Também utilizada por diferentes empresas para estudar produtos concorrentes.

Engenharia reversa: Tupolev Tu-4

- Em 1945, três bombardeiros americanos modelo B-29 foram forçados a aterrisar em território russo.
- Os soviéticos os desmontaram e utilizaram engenharia reversa para copiar o bombardeiro.
- O resultado foi o Tupolev Tu-4.

Engenharia reversa: Operação Diamante

- Operação realizada pelo serviço de inteligência de Israel.
- O objetivo foi capturar um caça Mikoyan-Gurevich MiG-21, de origem soviética.
- A operação foi bem-sucedida em 1966.
- Os americanos e israelenses estudaram o MiG-21, entendendo o seu funcionamento, fraquezas e pontos fortes.

Engenharia reversa na computação

- **Wine:** implementação open-source da API do Windows. Permite a execução de programas compilados nativamente para Windows em ambientes Linux.

- **Samba:** permite o compartilhamento de arquivos entre sistemas Windows e Linux.

- **OpenOffice:** conjunto de softwares de produtividade open-source. Compatível com o Microsoft Office e disponível para diferentes plataformas.

Técnicas de engenharia reversa de software

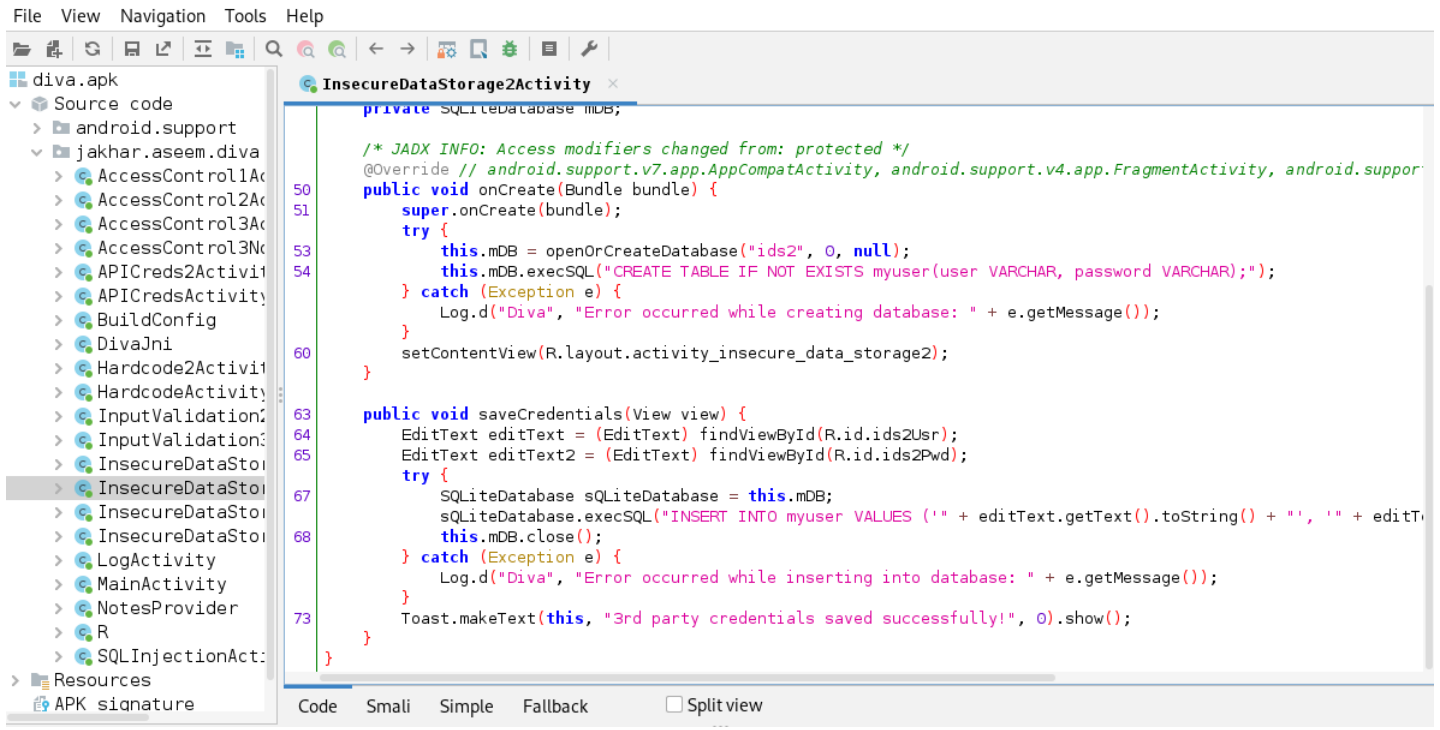
- **Disassembly:** podemos utilizar ferramentas conhecidas como **disassemblers** para obtermos o código do artefato em linguagem de máquina (**Assembly**).

```
00000000        push    ebp
00000001        mov     ebp, esp
00000003        movzx   ecx, [ebp+arg_0]
00000007        pop     ebp
00000008        movzx   dx, cl
0000000C        lea     eax, [edx+edx]
0000000F        add     eax, edx
00000011        shl     eax, 2
00000014        add     eax, edx
00000016        shr     eax, 8
00000019        sub     cl, al
0000001B        shr     cl, 1
0000001D        add     al, cl
0000001F        shr     al, 5
00000022        movzx   eax, al
00000025        retn
```

- Essa técnica funciona para qualquer artefato. Entretanto, o tempo para se entender o funcionamento pode ser desfavorável ao analista, especialmente se técnicas de anti-análise forem empregadas.

Técnicas de engenharia reversa de software

- **Decompilação:** utilizamos ferramentas conhecidas como decompiladores para tentar recriar o código-fonte em uma linguagem de alto nível.



- Nem sempre é possível reverter com sucesso o código de máquina.

Técnicas de engenharia reversa de software

- **Análise do fluxo de dados:** consiste no monitoramento de recursos criados ou modificados pelo programa em tempo de execução. Por exemplo: tráfego de rede, arquivos criados/alterados/apagados.

```
┌──(root㉿kali)-[~]
└─# tcpdump -vvv
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262
144 bytes
11:11:05.515425 IP (tos 0×0, ttl 4, id 51287, offset 0, flags [none], proto
UDP (17), length 165)
    10.10.10.1.52729 > 239.255.255.250.1900: [udp sum ok] UDP, length 137
11:11:05.590788 IP (tos 0×0, ttl 64, id 1747, offset 0, flags [DF], proto UD
P (17), length 74)
    10.10.10.129.44487 > 10.10.10.2.domain: [bad udp cksum 0×28de → 0×41f2!
] 62227+ PTR? 250.255.255.239.in-addr.arpa. (46)
11:11:06.434503 IP (tos 0×0, ttl 128, id 32281, offset 0, flags [none], prot
o UDP (17), length 131)
    10.10.10.2.domain > 10.10.10.129.44487: [udp sum ok] 62227 NXDomain q: P
TR? 250.255.255.239.in-addr.arpa. 0/1/0 ns: 239.in-addr.arpa. [5s] SOA sns.d
ns.icann.org. noc.dns.icann.org. 2022091119 7200 3600 604800 3600 (103)
11:11:06.434909 IP (tos 0×0, ttl 64, id 13979, offset 0, flags [DF], proto U
DP (17), length 69)
    10.10.10.129.52873 > 10.10.10.2.domain: [bad udp cksum 0×28d9 → 0×a580!
] 32476+ PTR? 1.10.10.10.in-addr.arpa. (41)
```

Engenharia reversa: Fontes de informação

- **Análise estática do código**
  - Permite a extração de informações básicas sobre o artefato (importações, exportações, variáveis, classes, etc).

- **Análise dinâmica (trace de execução)**
  - Permite identificar comportamentos do artefato, monitoramento dos valores de variáveis, funções chamadas, etc.

- **Bancos de dados**
  - Permite identificar algumas das estruturas de dados utilizadas pelo artefato.

- **Documentações**
  - Em alguns casos é possível analisar comentários deixados pelos desenvolvedores e aprender mais sobre o funcionamento do binário.

Engenharia reversa: Algumas ferramentas

Engenharia reversa: Dificuldades

- **Técnicas de ofuscação:** consiste em dificultar a análise do código-fonte por meio da ofuscação dele (e.g., alteração dos nomes de variáveis e de métodos). Dessa forma, a estrutura do código fica sem sentido ao analista.

- **Junk code:** código sem sentido adicionado propositalmente para dificultar a análise.

- **Anti-disassembly:** utilização de técnicas para dificultar o entendimento do código de máquina.

- **Anti-debugging:** técnicas para detecção de ferramentas de debug (normalmente o artefato muda o seu comportamento na presença de tais ferramentas).

- **Anti-VM:** técnicas utilizadas para verificar se o artefato está sendo executado em uma máquina virtual, pois são ambientes típicos de análise.

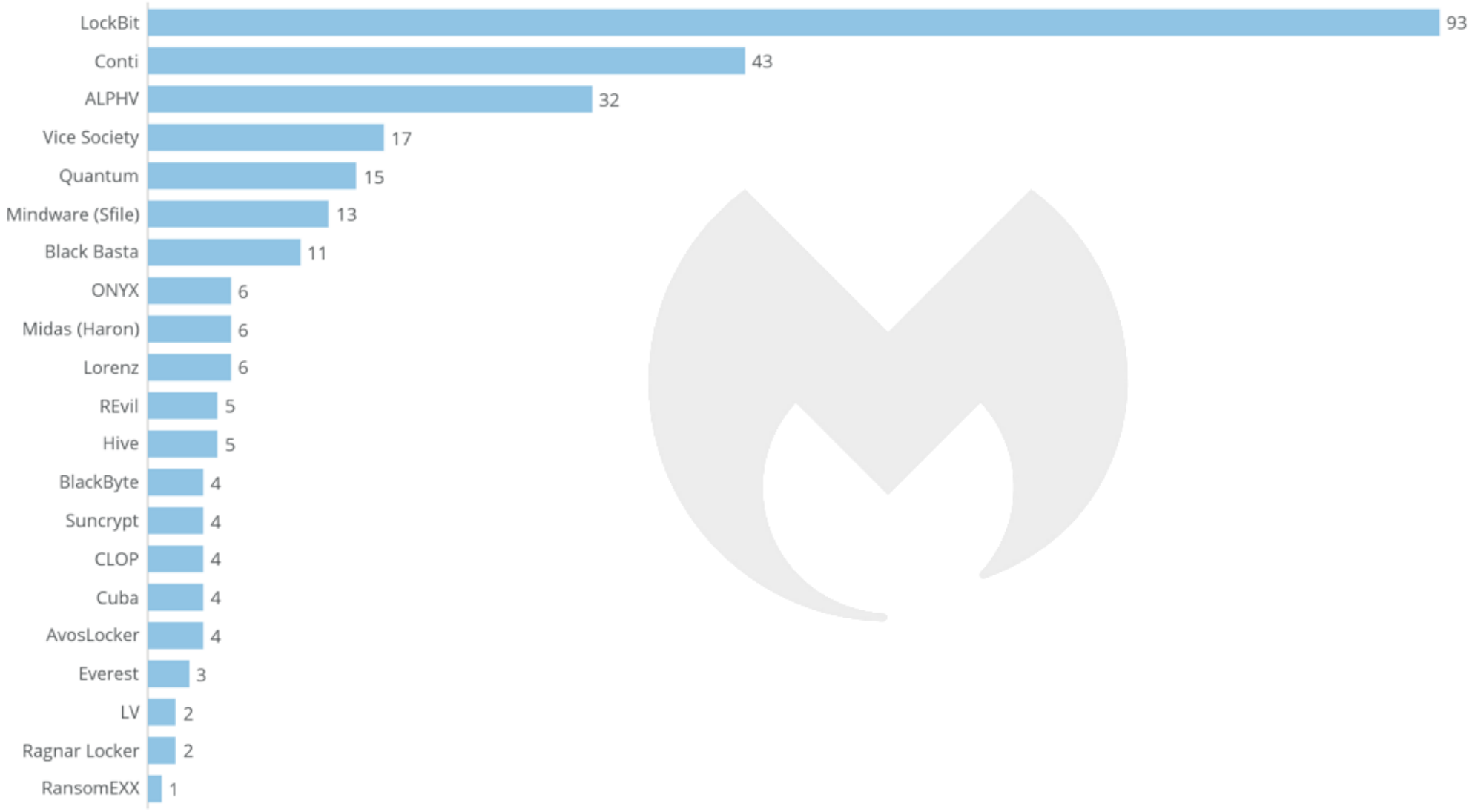
Análise de malware

Uma importante área para a segurança da informação

Cenário atual de malware

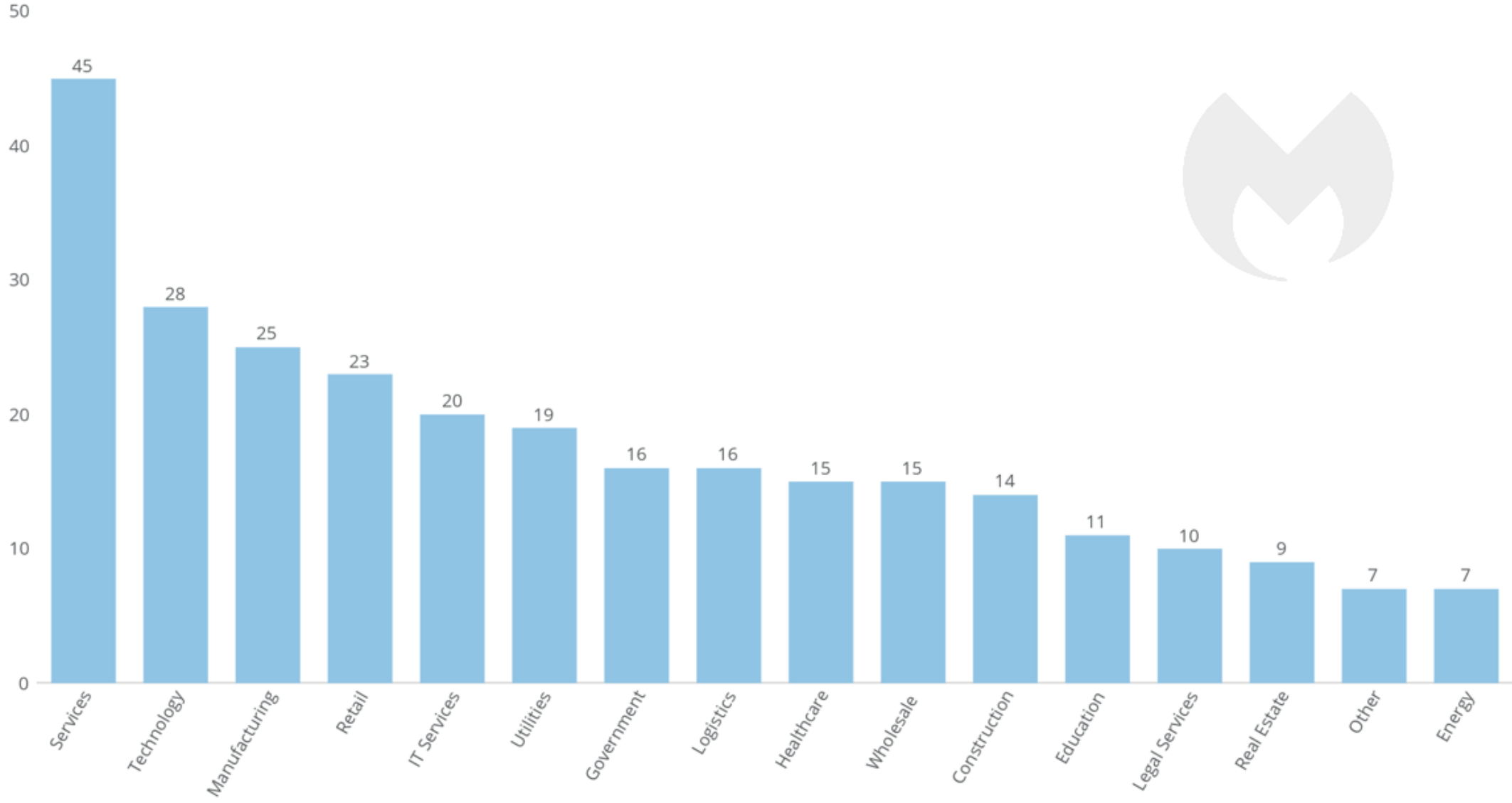
- Malware == **mal**icious soft**ware**.
- Infecções por malware continuam sendo uma das principais ameaças aos sistemas computacionais.
- Esse cenário se agrava com o crescimento das famílias de ransomware.
- Lucrativo comércio de Ransomware-as-a-Service (RaaS).
- Segundo a CrowdStrike, infecções por ransomware aumentaram 82% em 2021 em relação a 2020.

Principais famílias de ransomware – 2022



Fonte:
MalwareBytes

Setores mais afetados – 2022



Fonte:
MalwareBytes

Análise de malware

- A análise de malware é de fundamental importância para a segurança da informação.
- A partir do entendimento comportamental e do código Assembly do artefato, são produzidas ferramentas para detecção e mitigação eficientes.
- Porém, o tempo que um analista leva para dissecar um binário malicioso e implantar regras na solução desenvolvida pode ser desfavorável ao usuário final.
- A diversidade de ferramentas disponíveis e que a escolha errada de uma pode atrasar o trabalho de análise.

Análise de malware

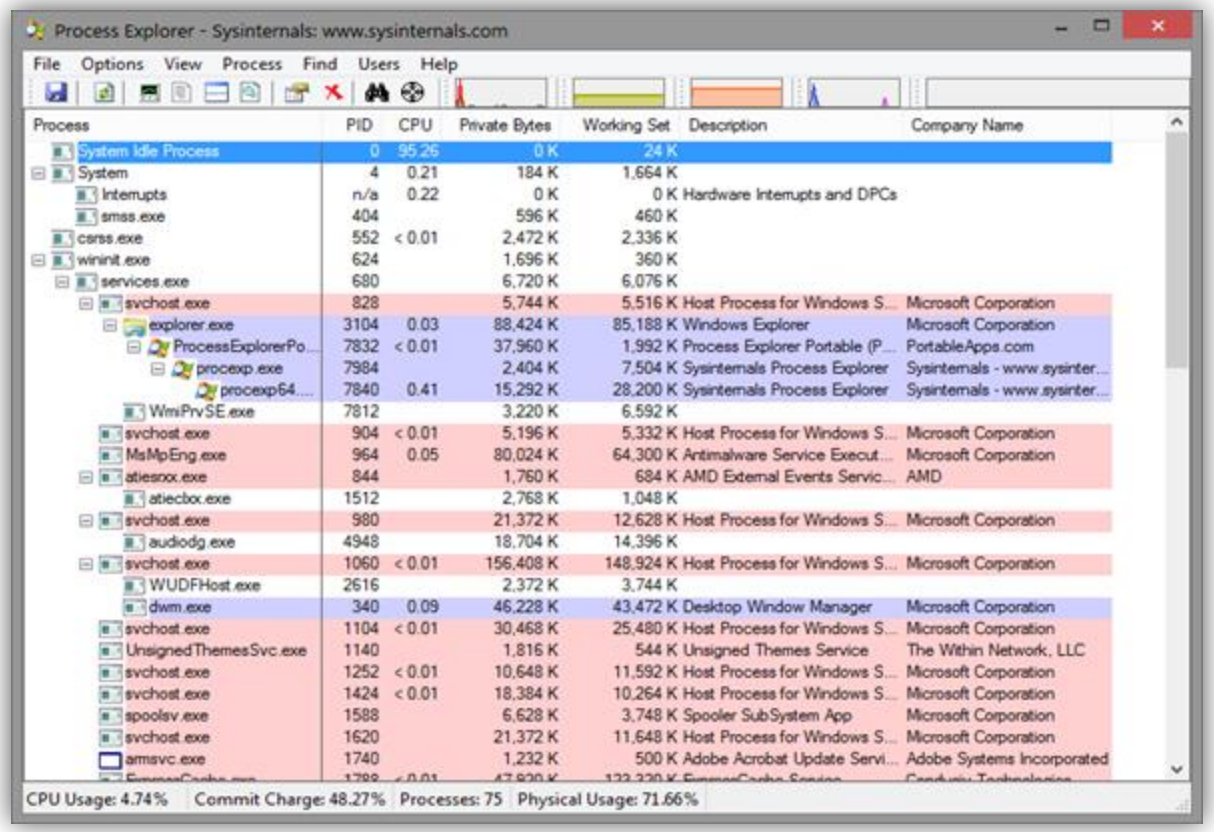
- A análise de um programa malicioso pode ser realizada de forma estática ou dinâmica.
- **Análise estática:** coleta de informações sobre o binário sem executá-lo.
- **Análise dinâmica:** execução do artefato em um ambiente controlado.
- É importante notar que uma estratégia não substitui a outra e ambas devem ser empregadas para um completo entendimento sobre o malware em questão.

Análise de malware

Análise estática:

```
$ sha256sum mimikatz.exe
92804faaab2175dc501d73e814663058c78c0a042675a8937266357bcfb96c50  mimikatz.exe
$ xxd mimikatz.exe | head -n 20
00000000: 4d5a 9000 0300 0000 0400 0000 ffff 0000  MZ..............
00000010: b800 0000 0000 0000 4000 0000 0000 0000  ........@.......
00000020: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000030: 0000 0000 0000 0000 0000 0000 2001 0000  ............ ...
00000040: 0e1f ba0e 00b4 09cd 21b8 014c cd21 5468  ........!..L.!Th
00000050: 6973 2070 726f 6772 616d 2063 616e 6e6f  is program canno
00000060: 7420 6265 2072 756e 2069 6e20 444f 5320  t be run in DOS
00000070: 6d6f 6465 2e0d 0d0a 2400 0000 0000 0000  mode....$.......
00000080: 34c8 d822 70a9 b671 70a9 b671 70a9 b671  4.."p..qp..qp..q
00000090: 79d1 2371 72a9 b671 79d1 3571 4fa9 b671  y.#qr..qy.5qO..q
000000a0: 79d1 3271 60a9 b671 79d1 2571 72a9 b671  y.2q`..qy.%qr..q
000000b0: 6b34 2a71 72a9 b671 1647 7d71 74a9 b671  k4*qr..q.G}qt..q
000000c0: eb42 7d71 72a9 b671 0634 db71 72a9 b671  .B}qr..q.4.qr..q
000000d0: 6efb 3271 72a9 b671 0634 cd71 5fa9 b671  n.2qr..q.4.q_..q
000000e0: 70a9 b771 0aab b671 576f c871 71a9 b671  p..q...qWo.qq..q
000000f0: 79d1 3f71 13a9 b671 79d1 2271 71a9 b671  y.?q...qy."qq..q
00000100: 79d1 2771 71a9 b671 5269 6368 70a9 b671  y.'qq..qRichp..q
00000110: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000120: 5045 0000 6486 0600 6339 5a5e 0000 0000  PE..d...c9Z^....
00000130: 0000 0000 f000 2200 0b02 0900 002c 0c00  ......".......,..
```

Análise dinâmica:



Análise de malware

- Algumas ferramentas de mercado:

Análise de malware

- Algumas ferramentas propostas pela academia:

Trabalho	Nome da solução	Tipo de análise	Plataforma / Tipo de arquivo	Descrição
(WILLEMS; HOLZ; FREILING, 2007)	CWSandbox	Dinâmica	Windows / PE	Solução capaz de injetar códigos no processo malicioso para acesso a regiões da memória pertencentes a ele.
(SEIFERT et al., 2007)	Capture	Dinâmica	Windows / PE	Ferramenta para monitoramento do registro do Windows e processos do sistema.
(RHEE et al., 2010)	LiveDM	Dinâmica	Linux / ELF	Solução para análise de novas alocações de memória no sistema operacional.
(MOHAISEN; ALRAWI; MOHAISEN, 2015)	AMAL	Dinâmica	Windows / PE	Ferramenta que permite a definição de diferentes propriedades operacionais para análise dinâmica de *malware*.
(SOUZA; SILVA, 2021)	Freki	Estática	Windows / PE	Plataforma gráfica para análise estática de *malware*. Permite que os dados sejam consultados via API REST.

Análise estática

Conhecendo o artefato

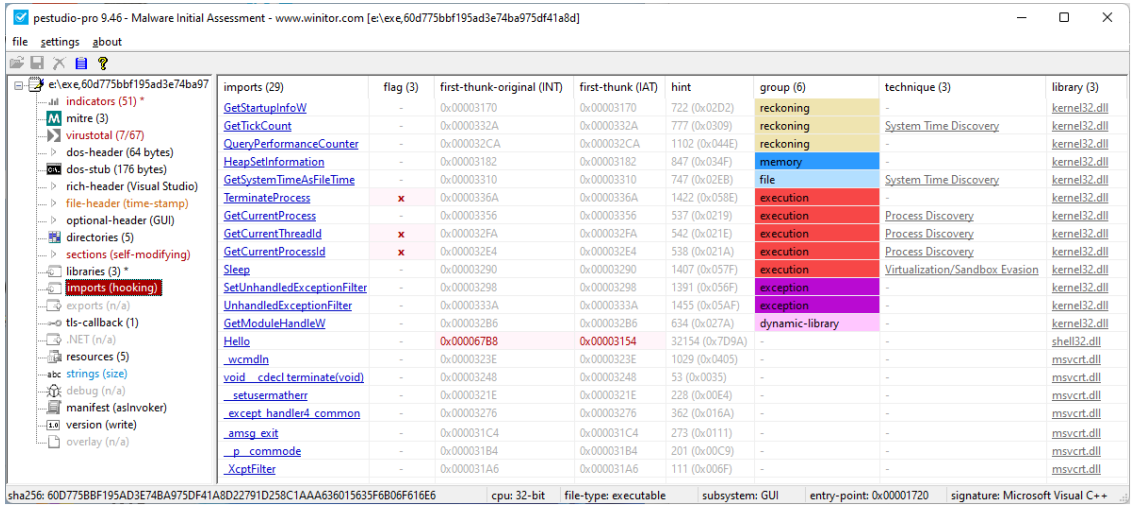
Análise estática

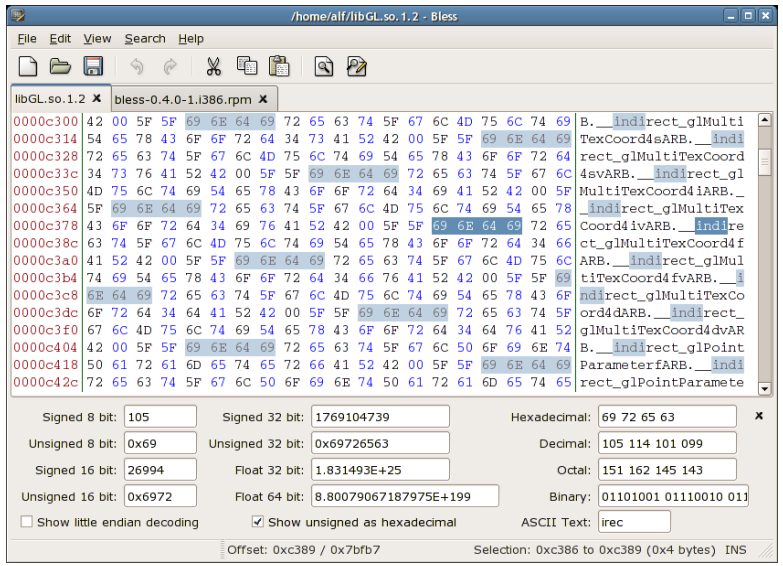
- Normalmente não executamos o artefato nesse tipo de análise.
- Conduzimos o processo de forma mais cautelosa, apenas buscando indícios da presença de instruções maliciosas.
- Muito útil para reconhecermos a estrutura do artefato (e.g., bibliotecas utilizadas, arquivos que serão criados, endereços IP de C&C servers, etc).
- Ou seja, utilizamos técnicas de engenharia reversa para chegarmos a tais resultados.

Análise estática: Algumas técnicas

- **Cálculo de hashes**
  - MD5, SHA-1, SHA-256, SHA-384, SHA-512, SSDEEP.
- **Análise da estrutura do arquivo**
  - Cabeçalhos, seções, importações, capacidades, strings.
- **Reconhecimento de padrões**
  - YARA.

Análise estática: Algumas ferramentas





Análise estática: Project Freki

Yara matches

Check_Qemu_Description, Check_Qemu_DeviceMap, Check_VBox_Description, Check_VBox_DeviceMap, Check_VBox_Guest_Additions, Check_VBox_VideoDrivers, Check_VMWare_DeviceMap, Check_VmTools, Check_Wine, vmdetect, Check_FilePaths, Check_OutputDebugStringA_iat, anti_dbg, antisb_sandboxie, antivm_virtualbox, antivm_bios, vmdetect_misc, network_dns, win_registry, Str_Win32_Winsock2_Library

Sections

Section	Virtual Address	Virtual Size	Pointer to raw data	Characteristics	Entropy	MD5
.text	0x1000	0x4f04	0x400	0x60500060	5.84	813a67ce69f969f8c4be506419caf754
.data	0x6000	0x30	0x5400	0xc0300040	0.52	9dbdb234d933061400ea88f48a1ba184
.rdata	0x7000	0x32b8	0x5600	0x40300040	5.84	05e6f678e9582460f7dd26374ef720b9
.bss	0xb000	0x400	0x0	0xc0600080	0.0	d41d8cd98f00b204e9800998ecf8427e
.idata	0xc000	0xd24	0x8a00	0xc0300040	4.76	aab2bb441a368f3c6952b605daa56c47
.CRT	0xd000	0x34	0x9800	0xc0300040	0.27	deb78d57f8d139ba832e60e440297d99
.tls	0xe000	0x20	0x9a00	0xc0300040	0.2	4d0db56ecaa4036333d178cca8b31a98
.rsrc	0xf000	0x8ef0	0x9c00	0xc0300040	7.85	6d313e884adc1e033a70c0fda038ba55

Imports

+ ADVAPI32.dll
+ IPHLPAPI.DLL
+ KERNEL32.dll

Análise estática: Project Freki

Imports

+ ADVAPI32.dll
+ IPHLPAPI.DLL
+ KERNEL32.dll
+ MPR.DLL
+ msvcrt.dll
+ ole32.dll
+ OLEAUT32.dll
+ SHELL32.dll
+ USER32.dll
– WS2_32.dll

Name	Address
freeaddrinfo	0x40c468
getaddrinfo	0x40c46c

Capabilities

– allocate RWX memory

Namespace	Scope	Matches
host-interaction/process/inject	basic block	0x405121

+ check for OutputDebugString error
+ check for time delay via GetTickCount

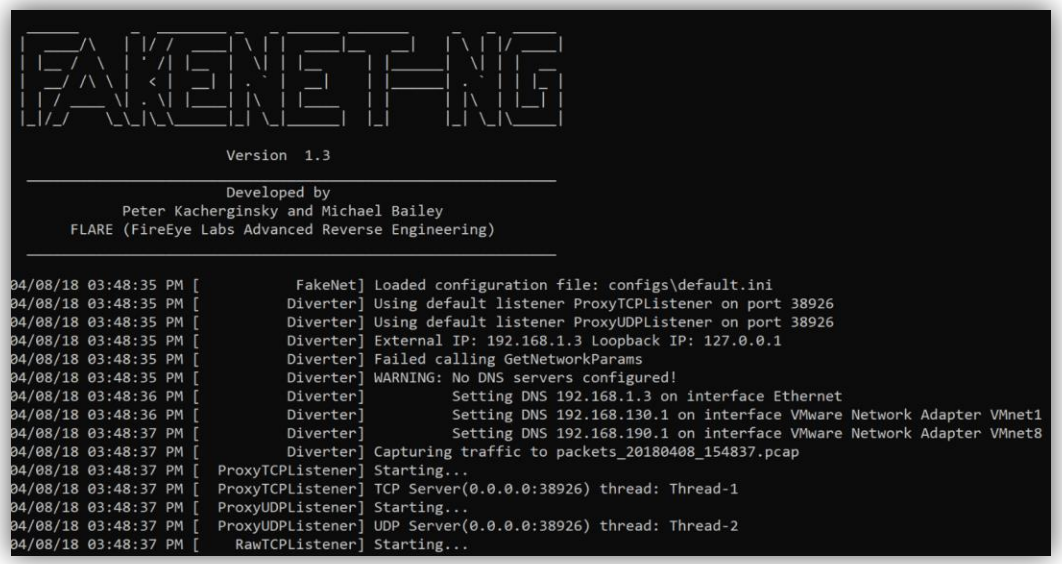
Análise dinâmica

Executando o artefato

Análise dinâmica

- Nesse tipo de análise, executamos o artefato em um ambiente controlado.
- Esse ambiente é conhecido como sandbox.
- A sandbox normalmente é uma máquina virtual totalmente isolada do sistema hospedeiro (sem rede ou compartilhamento de arquivos).
- A partir da execução do malware, podemos identificar ações que normalmente não poderiam ser identificadas em uma análise estática, especialmente se boas técnicas de anti-análise forem empregadas pelo atacante.

Análise dinâmica



cuckoo

DEBUGGER

Detecção de malware

Entendendo como o seu AV funciona

Detecção de malware

- Diversas técnicas foram aprimoradas ao longo dos anos para identificação e contenção automatizada de artefatos maliciosos.

- Novas formas de identificação a partir de estudos relacionados ao processamento de imagens e visão computacional estão em constante evolução.

- A escolha de uma ferramenta para detecção depende de fatores como a sua acurácia e performance.

Detecção de malware

- Detecção baseada em assinaturas.
  - Melhor que o uso de uma função de *hash*.
  - Método padrão e muito rápido.
  - Regras são configuradas previamente (e.g., YARA).

- Detecção baseada em heurística.
  - Não precisa de uma correspondência exata de bytes para classificação.
  - Possibilita a detecção de malware de forma mais genérica.
  - Pode gerar alguns falsos positivos.

- Detecção baseada em comportamentos.
  - Detecção em tempo real a partir da análise comportamental do artefato.
  - Permite detectar ameaças não catalogadas previamente.

Detecção de malware

KASPERSKY lab

CROWDSTRIKE

Técnicas de evasão

Burlando as defesas existentes

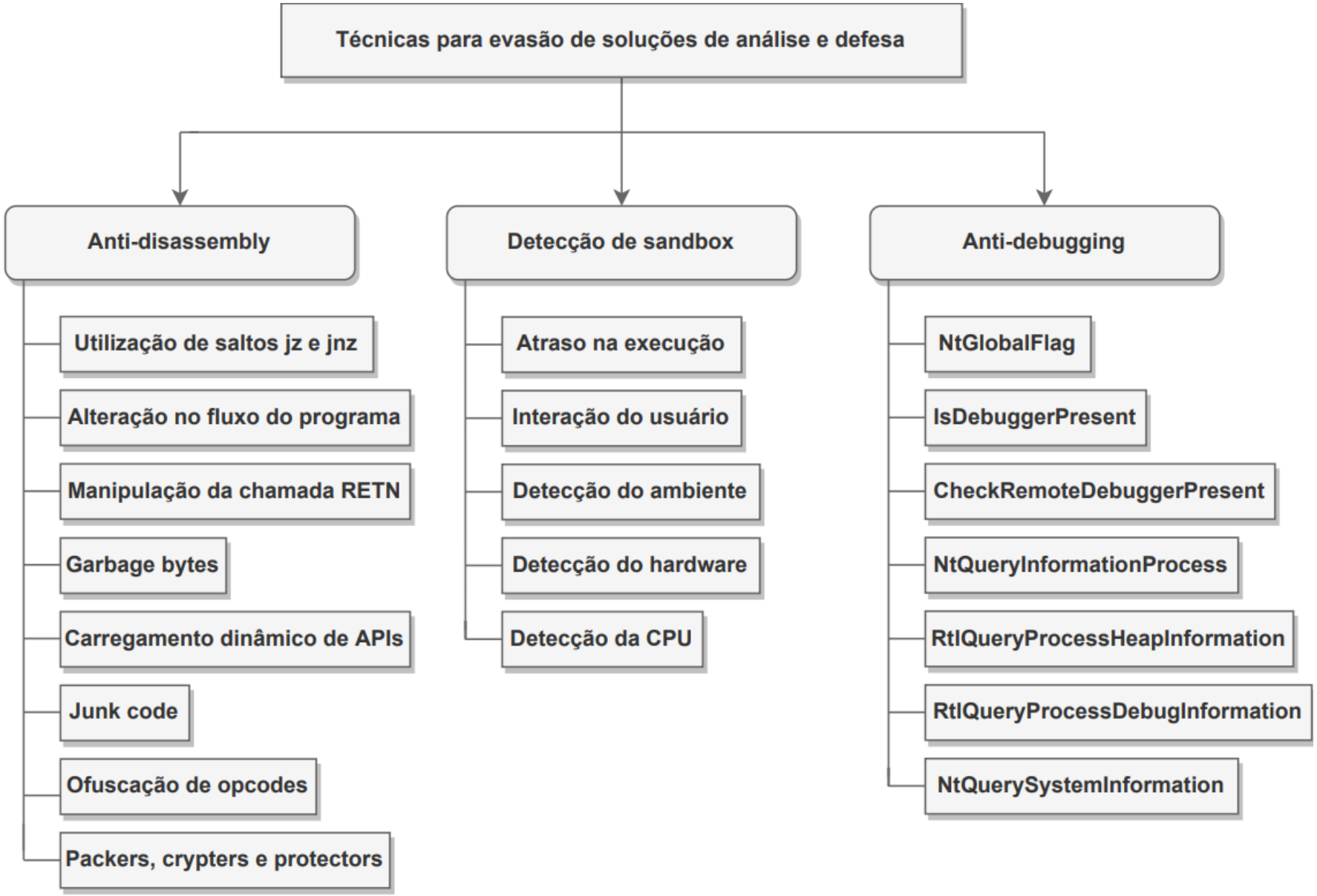
Técnicas de evasão

- Técnicas de evasão, também conhecidas como técnicas de anti-análise, são comumente empregadas por atacantes com o objetivo de burlar mecanismos de análise e defesa existentes, dificultando o desenvolvimento de mitigações eficientes.

- Diversos mecanismos podem ser empregados para esse fim, como técnicas para detecção de debuggers, anti-disassembly e detecção de sandboxes.

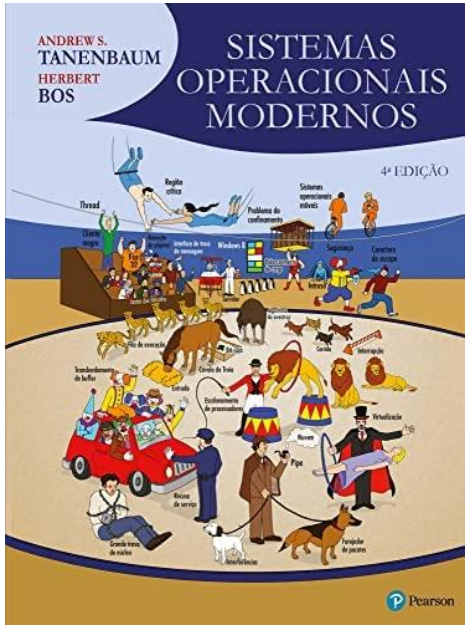
- Exemplo notório: Wannacry.

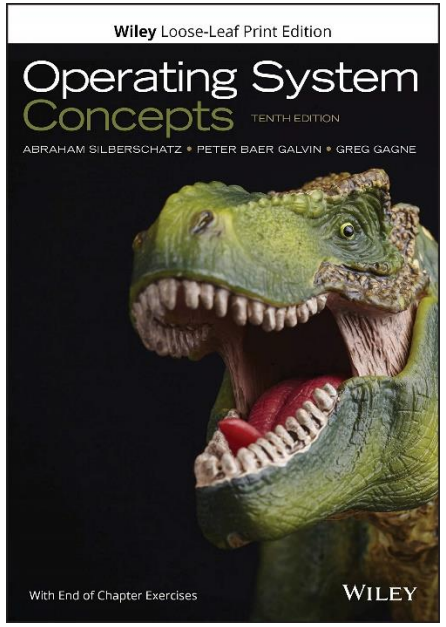
Técnicas de evasão

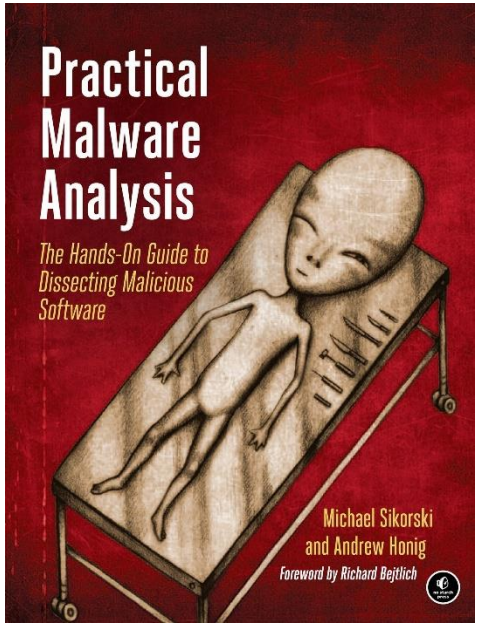


Por onde começar?

Por onde começar?







Engenharia reversa e análise de malware

Cristian Souza
@cristianzsh

https://cristian.sh