INTRODUÇÃO BÁSICA A ANALISE DE MALWARE 1

Joas Antonio

SOBRE O LIVRO

- Um livro conceitual, não possui prática;
- Feito para iniciantes que querem conhecer um pouco sobre Analise de Malware;
- Os principais conceitos e técnicas utilizadas na Analise de Malware;
- Não é um livro profundo, mas apresenta uma base para se aprofundar nos estudos e claro, vou deixar conteúdos para você estudar;
- Apresentar algumas ferramentas utilizadas na Analise de Malware;
- É necessário ter um bom nível de inglês para começar à brincar;

SOBRE O AUTOR

- **Nome:** Joas Antonio;
- Apaixonado por Segurança da Informação;

CONCEITOS

ANALISE DE MALWARE

O QUE É ANALISE DE MALWARE?

- A análise de malware é o estudo ou processo de determinação da funcionalidade, origem e impacto potencial de uma determinada amostra de malware , como vírus, worm, cavalo de Troia, rootkit ou backdoor. Malware ou software malicioso é qualquer software de computador destinado a prejudicar um sistema operacional ou a roubar dados confidenciais de usuários, organizações ou empresas. O malware pode incluir software que reúne informações do usuário sem permissão.

BENEFICIOS DA ANALISE DE MALWARE

O principal benefício da análise de malware é que ela ajuda respondedores de incidentes e analistas de segurança:

- Triagem de incidentes pragmaticamente por nível de gravidade
- Descubra indicadores ocultos de comprometimento (COI) que devem ser bloqueados
- Melhorar a eficácia dos alertas e notificações do COI
- Enriqueça o contexto ao procurar ameaças

TIPOS DE ANALISE DE MALWARE

- A análise pode ser realizada de maneira estática, dinâmica ou híbrida (Junção dos dois)

Análise estática

- A análise estática básica não requer que o código seja realmente executado. Em vez disso, a **análise estática examina o arquivo em busca de sinais de intenção maliciosa** . Pode ser útil identificar infraestrutura maliciosa, bibliotecas ou arquivos compactados.
- Os indicadores técnicos são identificados como nomes de arquivo, hashes, seqüências de caracteres, como endereços IP, domínios e dados de cabeçalho do arquivo, para determinar se esse arquivo é malicioso. Além disso, ferramentas como desmontadores e analisadores de rede podem ser usadas para observar o malware sem realmente executá-lo, a fim de coletar informações sobre como o malware funciona.
- No entanto, **como a análise estática não executa o código, o malware sofisticado pode incluir um comportamento malicioso em tempo de execução que pode não ser detectado** . Por exemplo, se um arquivo gera uma sequência que baixa um arquivo mal-intencionado com base na sequência dinâmica, ele pode não ser detectado por uma análise estática básica. As empresas se voltaram para a análise dinâmica para uma compreensão mais completa do comportamento do arquivo.

TIPOS DE ANALISE DE MALWARE

Análise dinâmica

- **A análise dinâmica de malware executa o código malicioso suspeito em um ambiente seguro chamado** sandbox **.** Esse sistema fechado permite que os profissionais de segurança assistam o malware em ação sem o risco de deixá-lo infectar seu sistema ou escapar para a rede corporativa.
- A análise dinâmica fornece aos caçadores de ameaças e respondedores de incidentes uma visibilidade mais profunda, permitindo que eles descubram a verdadeira natureza de uma ameaça. Como benefício secundário, o sandbox automatizado elimina o tempo necessário para fazer a engenharia reversa de um arquivo para descobrir o código malicioso.
- O desafio da análise dinâmica é que os adversários são inteligentes e sabem que existem caixas de areia por aí, e se tornaram muito bons em detectá-las. Para enganar uma sandbox, os adversários ocultam códigos dentro deles que podem permanecer inativos até que certas condições sejam atendidas. Somente então o código é executado.

TIPOS DE ANALISE DE MALWARE

Análise hibrida

- A análise estática básica não é uma maneira confiável de detectar códigos maliciosos sofisticados e, às vezes, malwares sofisticados podem ocultar a presença da tecnologia sandbox. **Ao combinar técnicas básicas e dinâmicas de análise, a análise híbrida fornece à equipe de segurança o melhor de ambas as abordagens -** principalmente porque ele pode detectar código malicioso que está tentando ocultar e, em seguida, pode extrair muito mais indicadores de comprometimento (IOCs) por código estatístico e anteriormente não visto. . A análise híbrida ajuda a detectar ameaças desconhecidas, mesmo as do malware mais sofisticado.
- Por exemplo, uma das coisas que a análise híbrida faz é aplicar a análise estática aos dados gerados pela análise comportamental - como quando um pedaço de código malicioso é executado e gera algumas alterações na memória. A análise dinâmica detectaria isso e os analistas seriam alertados para voltar e executar a análise estática básica nesse despejo de memória. Como resultado, mais COI seriam gerados e explorações de dia zero seriam expostas.

ESTÁGIOS DA ANALISE DE MALWARE

A análise de software malicioso envolve vários estágios, incluindo, entre outros, o seguinte:

- Reversão manual de código
- Análise interativa do comportamento
- Análise de propriedades estáticas
- Análise totalmente automatizada

ESTÁGIOS DA ANALISE DE MALWARE

Análise de propriedades estáticas

- As propriedades estáticas incluem seqüências de caracteres incorporadas no código do malware, detalhes do cabeçalho, hashes, metadados, recursos incorporados etc. Esse tipo de dado pode ser tudo o que é necessário para criar IOCs e pode ser adquirido muito rapidamente porque não há necessidade de executar o programa para vê-los. As informações coletadas durante a análise de propriedades estáticas podem indicar se é necessária uma investigação mais profunda usando técnicas mais abrangentes e determinar quais etapas devem ser seguidas.

ESTÁGIOS DA ANALISE DE MALWARE

Análise interativa do comportamento

- A análise comportamental é usada para observar e interagir com uma amostra de malware em execução em um laboratório. Os analistas procuram entender as atividades de registro, sistema de arquivos, processos e rede da amostra. Eles também podem realizar análises forenses de memória para aprender como o malware usa a memória. Se os analistas suspeitarem que o malware tem uma certa capacidade, eles podem configurar uma simulação para testar sua teoria.

ESTÁGIOS DA ANALISE DE MALWARE

Análise totalmente automatizada

- A análise totalmente automatizada avalia rápida e simplesmente os arquivos suspeitos. A análise pode determinar possíveis repercussões se o malware se infiltrar na rede e produzir um relatório de fácil leitura que fornece respostas rápidas para as equipes de segurança. A análise totalmente automatizada é a melhor maneira de processar malware em escala.

ESTÁGIOS DA ANALISE DE MALWARE

Reversão manual de código

- Nesse estágio, os analistas fazem engenharia reversa de código usando depuradores, desmontadores, compiladores e ferramentas especializadas para decodificar dados criptografados, determinam a lógica por trás do algoritmo de malware e entendem os recursos ocultos que o malware ainda não exibiu. A reversão de código é uma habilidade rara, e a execução de reversões de código leva muito tempo. Por esses motivos, as investigações de malware geralmente ignoram essa etapa e, portanto, perdem muitas informações valiosas sobre a natureza do malware.

TÉCNICAS DE DETECÇÃO DE MALWARE - ANTIVÍRUS

Assinatura:

- A detecção através de **assinatura** é a técnica mais antiga e básica, aonde o antivírus procura por trechos de código ou combinação de caracteres que já foram utilizados em outros malwares, pois se eles também estiverem ali, certamente esse programa é um malware.

Heurística:

- A **heurística** analisa as características do programa e procura por padrões internos que possam indicar que o programa é um malware. Cada vez que um padrão é encontrado, é atribuída uma pontuação para ele, e se na soma total essa pontuação for superior a um determinado valor, o antivírus considera esse programa sendo um malware desconhecido.
- O melhor exemplo disso acontece quando **você baixa um arquivo para o seu computador, e assim que ele é baixado o seu antivírus já te avisa que ele é um malware e apaga ele**.

TÉCNICAS DE DETECÇÃO DE MALWARE - ANTIVÍRUS

Cloud:

- Há algum tempo os antivírus utilizam proteção na nuvem, garantindo que isso aumenta a taxa de detecção de novos malwares, ao mesmo tempo que fornecem uma proteção mais rápida para seus usuários. Como isso funciona?
- **Exemplo simples:** imagine que um internauta acabou de encontrar em um site um novo ativador de Windows para ativar o Windows pirata dele, e aí ele baixa esse ativador e dá um duplo-clique nele para usá-lo. Quando o internauta executa esse arquivo, o antivírus que ele estiver utilizando vai fazer uma análise rápida desse programa para decidir se ele é um programa confiável ou não.
- **Essa análise que o antivírus faz é multi-layer, ou seja, ele não faz uma ÚNICA análise: ele realiza simultaneamente diversas análises, verificações e comparações.**

TÉCNICAS DE DETECÇÃO DE MALWARE - ANTIVÍRUS

Inteligência Artificial:

- O principal motivo do uso da Inteligência Artificial em um antivírus é vencer o desafio do antivírus não errar NUNCA, ou seja, ele precisa detectar 100% dos malwares e ter 0% de falso positivo (que é quando um programa comum é detectado como malware quando ele não é).
- E da mesma forma que nós só ficamos experientes em algo depois de muita prática, acertando e errando, a Inteligência Artificial também precisa de muito treino, com erros e acertos. E o treino da Inteligência Artificial para identificar malwares exige que ela tenha acesso ao maior número possível de arquivos (maliciosos ou não) para ela treinar a sua detecção - e como o Windows Defender vem pré-instalado no Windows e é utilizado por centenas de milhões de usuários, ele é perfeito para isso.
- Toda vez que o Windows Defender detecta um malware no computador de algum usuário, as informações desse malware (incluindo o próprio malware, os trechos suspeitos do seu código, o comportamento que ele teve, etc.) são imediatamente repassadas para servidores da Microsoft que estão na nuvem.

CASOS DE USOS DA ANALISE DE MALWARE

Detecção de malware

- Os adversários estão empregando técnicas mais sofisticadas para evitar os mecanismos de detecção tradicionais. Ao fornecer análise comportamental profunda e identificar código compartilhado, funcionalidade maliciosa ou infraestrutura, as ameaças podem ser detectadas com mais eficácia. Além disso, uma saída da análise de malware é a extração de IOCs. Os COI podem então ser alimentados com SEIMs, plataformas de inteligência de ameaças (TIPs) e ferramentas de orquestração de segurança para ajudar a alertar as equipes sobre ameaças relacionadas no futuro.

Alertas e triagem de ameaças

- As soluções de análise de malware fornecem alertas de alta fidelidade no início do ciclo de vida do ataque. Portanto, as equipes podem economizar tempo priorizando os resultados desses alertas em relação a outras tecnologias.

CASOS DE USOS DA ANALISE DE MALWARE

Resposta a Incidentes

- O objetivo da equipe de resposta a incidentes (IR) é fornecer análise de causa raiz, determinar o impacto e obter êxito na correção e recuperação. O processo de análise de malware ajuda na eficiência e eficácia desse esforço.

Caça à Ameaça

- A análise de malware pode expor comportamentos e artefatos que os caçadores de ameaças podem usar para encontrar atividades semelhantes, como acesso a uma conexão, porta ou domínio de rede específico. Pesquisando logs de firewall e proxy ou dados SIEM, as equipes podem usar esses dados para encontrar ameaças semelhantes.

Pesquisa de malware

- Pesquisadores de malware acadêmicos ou do setor executam análises de malware para entender as mais recentes técnicas, explorações e ferramentas usadas pelos adversários.

REGISTRADOR X86

Os registradores de uso geral do x86

- São eles: **AX, BX, CX, DX, BP, SI e DI.**
- Tais registradores são de 16 bits, ou sejam, podem receber ou enviar 16 bits de dados de uma só vez. Porém, alguns destes (AX, BX, CX e DX) são comumente usados como registradores de 8 bits, ou seja, são divididos em dois blocos de 8 bits.
- O bloco com os 8 primeiros dígitos são identificados por L, de *LOW*, e os outros bits são identificados pela letra H, de *HIGH*.
- Assim, o registrador AX é formado por AH e AL.
- BX é formado por BH e BL.
- CX é formado por CH e CL.
- DX é formado por DH e DL.
- Eles são ditos de uso geral pois podem ser usados livremente pelo programador Assembly para armazenar e trocar informações em seus códigos da maneira que mais lhe convier.
- Vale salientar que, embora sejam de uso geral, eles possuem uso específicos, que serão vistos na seção a seguir, onde daremos mais detalhes sobre cada um deles, suas importâncias e usos especiais.

REGISTRADOR X86

As características específicas de cada registrador desses, de uso geral, são as seguintes:

- **AX, AH e AL** → Registrador Acumulador, muito usado em operação aritméticas, para armazenar o endereço de *offset*, interrupção e operações com I/O (entrada/saída)
- **BX, BH e BL** → Registrador de Base, que guarda o *offset* do ponteiro de dados
- **CX, CH e CL** → Registrador contador, usado para armazenar contagens, com nas instruções LOOP e REP. Esse registro vai incrementando/decrementando automaticamente, conforme sua utilização (em strings, por exemplo)
- **DX, DH e DL** → Registrador de dados, usado para instruções de divisão e multiplicação, como o acumulador, além de também ser usado para armazenar o *offset* de um ponteiro de dados
- **BP** → Ponteiro de base, usado para indicar uma posição na transferência de dados na memória
- **SI** →*Source Index*, usado em instruções que envolvem strings para armazenar o ponteiro da origem
- **DI** → *Destination Index*, também usado em strings, onde armazena um ponteiro da base do destino

REGISTRADOR X86

Flags

- São registros especiais que servem para fazer coisas específicas, como indicar, alterar e controlar algumas características dos microprocessadores.
- **C (*Carry*)** → Além de indicar situações de erro, é usado é operações de adição e subtração (é o 'vai 1' e 'empresta 1' aqui no Brasil)
- **P (*Parity*)** → Se determinado número é ímpar, a *flag* P assume valor lógico 0, ou valor lógico 1 caso o número seja par
- **A (*Auxiliary Carry*)** → Usada nas instruções DAA e DAS em adição e subtração de números BCD
- **Z (*Zero*)** → Testa se determinada operação terá valor lógico 0. Retorna 1 se o resultado é 0, e 0 se não for
- **S (*Sign*)** → Indica o sinal de um número. Tem valor 1 se o número for negativo, e 0 se for positivo
- **T (*Trap*)** → Habilita o *trap* do microprocessador, fazendo-o interromper as instruções de acordo com o que houver nos registros de controle e debug
- **I (*Interrupt*)** → Faz as interrupções serem habilitadas quando o pino INTR é 1. Pode ser modificado pelas instruções STI e CLI
- **D (*Direction*)** → Incrementa ou decrementa os registros SI e DI em operações com strings. Se igual 1, o registro automaticamente decrementa, se 0 ele incrementa. Modificado através das instruções STD e CLD
- **O (*Overflow*)** → Usado em operações de adição e subtração para indicar o extrapolamento de endereço

REGISTRADOR X64

- A principal característica do AMD64 é a disponibilidade de registros 64-bit de uso geral, ou seja, Rax, rbx etc, operações de número inteiro 64-bit aritméticas e lógicas, e endereços virtuais 64-bit. Os projetistas tiveram a oportunidade de fazer outras melhorias também. As alterações mais significativas são:
- **Manipular inteiros de 64-bit:** Todos os registradores de uso geral (GPRS) são expandidos de 32 bits para 64 bits, e todas as operações aritméticas e lógicas, memória para registro e registro para memória etc, podem agora operar diretamente sobre inteiros de 64-bit. Adições ao endereço de pilha estão sempre em 8 bytes, e ponteiros são 8 bytes de tamanho.
- **Registradores adicionais:** Além de aumentar o tamanho dos registradores de uso geral, o número deles é aumentada de oito (ou seja, EAX, EBX, ECX, EDX, EBP, ESP, ESI, EDI) em x86-32 para dezesseis (isto é, RAX, RBX, RCX, RDX, RBP, RSP, RSI, RDI, R8, R9, R10, R11, R12, R13, R14, R15). É conseqüentemente possível manter mais variáveis locais nos registradores do que na pilha, e constantes frequentemente acessadas; os argumentos para sub-rotinas pequenas e rápidas podem igualmente ser passados nos registradores em maior medida. Entretanto, o AMD64 ainda tem poucos registradores do que muitos processadores comuns do RISC (que têm tipicamente 32-64 registradores) ou VLIW-como máquinas tais como o IA-64 (que tem 128 registradores).
- **Registradores adicionais do MMX (SSE):** Similarmente, o número de registradores de 128 bit do MMX (usados em instruções Streaming-SIMD) é aumentado igualmente de 8 a 16.

REGISTRADOR X64

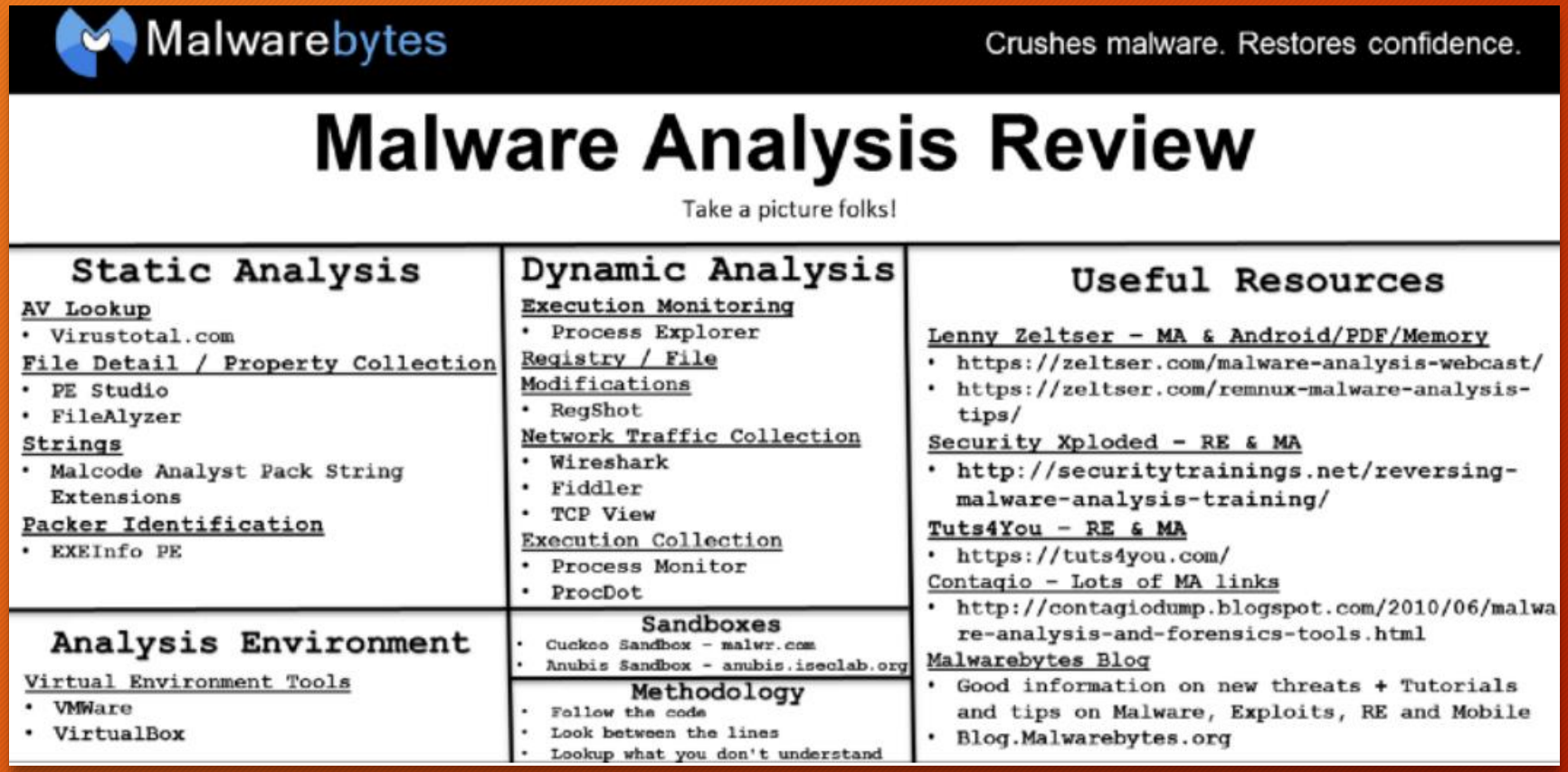
- **Espaço de endereço virtual maior:** Os modelos de processadores atuais que executam a arquitetura AMD64 podem endereçar até 256 TB(2^48 ou 281.474.976.710.656 bytes) do espaço de endereço virtual. Este limite pode ser aumentado nas implementações futuras a 16 EB (2^64 ou 18.446.744.073.709.551.616 bytes). Isto é comparado a apenas 4 GB (2^32 ou 4.294.967.296 bytes) para x86 de 32 bits. Isto significa que arquivos muito grandes podem ser operados pelo mapeamento de todo o arquivo para o processo de endereçamento de espaço (que por vezes é mais rápido do que trabalhar com chamadas de leitura/gravação do arquivo), em vez de ter de mapear regiões do arquivo para dentro e fora do espaço de endereçamento.
- **Espaço de endereço físico maior:** As execuções atuais da arquitetura AMD64 podem endereçar até 1 TB (2^40 ou 1.099.511.627.776 bytes) do RAM; a arquitetura permite estender esta a 4 PB (2^52 ou 4.503.599.627.370.496 bytes) no futuro (limitado pelo formato da entrada da tabela de páginas). No modo legado, a extensão do endereço físico (PAE) é incluída, enquanto está na maioria de processadores x86 de 32 bits atuais, permitindo o acesso a um máximo de 64 GB (2^36 ou 68.719.476.736 bytes).
- **Ponteiro de Instrução de acesso a dados relativos:** As instruções podem agora referenciar os dados relativos à instrução ponteiro (registrador RIP). Isto faz a posição do código ser independente, como é usado frequentemente em bibliotecas compartilhadas e em código carregados no tempo de execução, e mais eficiente.

REGISTRADOR X64

- **Instruções SSE:** A original arquitetura AMD64 adotou da Intel as instruções SSE e SSE2 como núcleo de instruções. As instruções SSE3 foram adicionadas em abril 2005. O SSE2 substitui o conjunto x86 instrução set precisão do bocado de s IEEE 80 com a escolha de uma ou outra matemática flutuante de 32 bits ou 64-bit de IEEE. Isto fornece as operações de vírgula flutuante compatíveis com muitos outros processadores centrais modernos. As instruções SSE e SSE2 foram estendidas igualmente para operar sobre os oito registos novos do MMX. SSE e SSE2 estão disponíveis na modalidade de 32 bits nos processadores x86 modernos, entretanto, se são usado em programas de 32 bits, aqueles programas trabalhará somente em sistemas com processadores que têm a característica. Esta não é uma edição em programas 64-bit, porque todos os processadores AMD64 têm SSE e SSE2, assim que usar as instruções SSE e SSE2 em vez das instruções x86 não reduz o jogo das máquinas em que os programas x64 podem ser funcionados. SSE e SSE2 são geralmente mais rápidos do que, e duplicam a maioria das características das instruções x86 tradicionais, de MMX, e de 3DNow!.
- **Não Executa-bit:** O bit "NX" (63º bit da entrada da tabela de páginas) permite que o sistema operacional especifique quais páginas no espaço de endereçamento virtual podem conter códigos executáveis e quais não podem. Uma tentativa para executar código a partir de uma página etiquetada de "não executar" irá resultar em uma violação de acesso à memória, semelhante a uma tentativa de escrever para uma memória ROM. Isto deve tornar mais difícil, para um código malicioso, assumir o controle do sistema através de ataques de "buffer overflow" ou "buffer não checado". A mesma funcionalidade está disponível em processadores x86 desde o 80286 como um atributo dos descritores de segmento, no entanto, isso só funciona em um segmento inteiro de uma vez. O endereçamento segmentado há muito tempo é considerado um modo obsoleto de funcionamento, e todos os sistemas operacionais nos PC atuais burlam isso, configurando todos os segmentos para um endereço base de 0 e um tamanho de 4 GB (4.294.967.296 bytes). A AMD foi a primeira fornecedora da família x86 a implementar o Não-Executa no modo de endereçamento linear. O recurso também está disponível no modo legado sobre processadores AMD64, e os recentes processadores Intel x86, quando PAE é utilizado.

REGISTRADOR X64

- **Remoção das características mais velhas:** Uma série de "sistema de programação" características da arquitetura x86 não são utilizados nos modernos sistemas operativos e não estão disponíveis em AMD64 em longas (64-bit e compatibilidade) modalidade. Estes incluem endereçamento segmentado (embora a FS e GS segmentos são mantidas em forma vestigial extra para utilização como base ponteiros para estruturas do sistema operacional), a tarefa estado mudar mecanismo e modo virtual 8086. Estas características fazem do curso continuem a ser totalmente implementado em "legacy mode", permitindo assim a correr estes processadores 32-bit e 16-bit sistemas operacionais sem modificação.



REVIEW MALWARE ANALYSIS

ASSEMBLY

- **A Linguagem Assembly é uma linguagem de baixo nível, ou seja, próximo à linguagem de máquina, mas não basta apenas conhecer um conceito para entende-la, você precisa aprender profundamente, segue alguns materiais.**
- https://www.youtube.com/watch?v=2giDgeXpJE0 **(Papo Binário)**
- http://www.inf.furb.br/~maw/arquitetura/aula16.pdf
- https://medium.com/@FreeDev/entenda-o-que-%C3%A9-assembly-ed64526cab49
- http://www.dsc.ufcg.edu.br/~pet/jornal/maio2014/materias/historia_da_computacao.html
- https://www.eximiaco.tech/pt/2019/12/26/entendendo-a-stack-em-sua-forma-mais-primitiva-em-assembly/
- https://www.youtube.com/watch?v=zlVP_RsxeXs
- http://www.ece.utep.edu/courses/web3376/Notes_files/ee3376-assembly.pdf
- https://www2.southeastern.edu/Academics/Faculty/kyang/2009/Fall/CMPS293&290/ClassNotes/CMPS293&290ClassNotesChap03.pdf
- https://www.youtube.com/watch?v=tyl9H0Wnsqc (Papo Binário)

WINDOWS API

- Uma API (*Application Programming Interface*) é uma interface para programar uma aplicação. No caso do Windows, consiste num conjunto de funções expostas para serem usadas por aplicativos, inclusive em *user mode*.
- O suporte ao desenvolvedor está disponível na forma de um kit de desenvolvimento de software , Microsoft Windows SDK , fornecendo a documentação e as ferramentas necessárias para criar o software com base na API do Windows e nas interfaces associadas do Windows.
- A API do Windows (Win32) está focada principalmente na linguagem de programação C , em que suas funções e estruturas de dados expostas são descritas nessa linguagem nas versões recentes de sua documentação. No entanto, a API pode ser usada por qualquer compilador ou montador de linguagem de programação capaz de lidar com as estruturas de dados de baixo nível (bem definidas), juntamente com as convenções de chamada prescritas para chamadas e retornos de chamada . Da mesma forma, a implementação interna da função da API foi desenvolvida em várias linguagens, historicamente. Apesar de C não ser uma programação orientada a objetoslinguagem, a API do Windows e o Windows foram historicamente descritos como orientados a objetos. Houve também muitas classes de mensagens publicitárias e extensões (de Microsoft e outros) para linguagens orientadas a objetos que fazem essa estrutura orientada a objeto mais explícito (Biblioteca Microsoft Foundation Classes (MFC), Visual Biblioteca de Componentes (VCL), GDI + , etc.) . Por exemplo, o Windows 8 fornece a API do Windows e a API do WinRT , que são implementadas em C ++ e são orientadas a objetos por design.
- https://docs.microsoft.com/en-us/windows/win32/apiindex/windows-api-list
- https://mentebinaria.gitbook.io/engenharia-reversa/windows-api
- https://docs.microsoft.com/en-us/windows/win32/api/

DLL

- Dynamic-link library (biblioteca de vínculo dinâmico) ou DLL, é a implementação feita pela Microsoft para o conceito de bibliotecas compartilhadas nos sistemas operacionais Microsoft Windows e OS/2. Essas bibliotecas geralmente tem as extensões DLL, OCX (para bibliotecas que contêm controles ActiveX), ou DRV (para drivers de sistema legados).
- Os formatos de arquivos para DLL são os mesmos dos arquivos executáveis para Windows. Assim como os executáveis (EXE), as DLL podem conter códigos, dados, e recursos (ícones, fontes, cursores, entre outros) em qualquer combinação.
- No sentido amplo do termo, qualquer arquivo de dados com esse mesmo formato pode ser chamado de DLL de recursos. Exemplos dessas DLL incluem bibliotecas de ícones, podendo ter a extensão ICL, e os arquivos de fontes, que têm as extensões FON e FOT.
- O propósito original das DLL era economizar espaço em disco e memória necessária para aplicativos, armazenando-os localmente no disco rígido. Em uma biblioteca padrão não-compartilhada, trechos de código são adicionados ao programa que faz a chamada; se dois programas usam a mesma rotina, o código deve ser incluído em ambos. Assim como, códigos que vários aplicativos compartilham podem ser separados em uma DLL que existe como apenas um único arquivo, carregado apenas uma vez na memória durante o uso. Devido ao uso extensivo de DLL, as versões iniciais do Windows puderam rodar em máquinas com pouca memória.
- As DLL proveem os benefícios comuns de bibliotecas compartilhadas, como a modularidade. Esta modularidade permite que alterações sejam feitas no código ou dados em uma DLL auto-contida, compartilhada por vários aplicativos, sem que qualquer modificação seja feita nos aplicativos em si. Essa forma básica de modularidade permite a criação de *patches* e *service packs* relativamente pequenos para grandes aplicativos, como Microsoft Office, Microsoft Visual Studio, e mesmo o próprio Microsoft Windows.

Hooking and Injection

- Hooking usa um recurso de sistema operacional para monitorar eventos enviados ao processo, como mensagens de teclado e mouse de baixo nível. Os aplicativos podem utilizar um gancho direcionado ou global para registrar keyloggers (malévolos) ou escutar pressionamentos de teclas para executar uma funcionalidade de agregação de valor, como executar macros ou outras funcionalidades de teclas de atalho. (benevolente)
- A injeção de DLL é o que parece, uma biblioteca vinculada dinâmica é injetada no processo de destino, forçando o processo a carregar a DLL. Uma vez carregada, a DLL injetada pode agir como uma API que pode ser acessada externamente a partir do processo (pense na API backdoor) e pode interagir com os internos públicos do processo que, de outra forma, seriam impossíveis.
- A injeção de DLL é principalmente usada com benevolência por depuradores de software e software de acessibilidade para pessoas com deficiência. No entanto, também é usado para trapacear nos videogames para um jogador, por meio de truques e treinadores.
- Um treinador utiliza essas duas técnicas. O aplicativo carregado (o treinador) conecta o aplicativo de destino e escuta as teclas associadas à funcionalidade das teclas de atalho. Geralmente, ao mesmo tempo, o DLL injeta o aplicativo com sua API backdoor. O treinador escuta as teclas digitadas associadas à funcionalidade das teclas de atalho e, quando esse evento é detectado, executa a funcionalidade associada por meio da API de backdoor.
- No que diz respeito à segurança, você não deseja que nenhum software não confiável faça nenhum deles. Você deve ter muito cuidado com quem confia, e é por isso que treinadores não respeitáveis são tão perigosos para instalar e usar. Ambas as técnicas requerem permissões elevadas do sistema operacional, o que lhes dá acesso a outros recursos de alto nível do sistema que você nunca desejaria ter software não respeitável.

Tipos de Malware

- Existem muitos tipos de malwares, seja variantes ou até mesmo malwares que atualmente não são utilizados, por isso, em vez de listar vou deixar alguns conteúdos para vocês conhecerem o máximo possível.
- https://www.veracode.com/blog/2012/10/common-malware-types-cybersecurity-101
- https://www.crowdstrike.com/epp-101/types-of-malware/
- https://www.uscybersecurity.net/malware/
- https://www.ijser.org/researchpaper/Types-of-Malware-and-its-Analysis.pdf
- https://ftms.edu.my/v2/wp-content/uploads/2019/02/csca0101_ch08.pdf
- https://backend.orbit.dtu.dk/ws/portalfiles/portal/4918204/malware.pdf
- https://www.sciencedirect.com/topics/computer-science/logic-bomb#:~:text=Logic%20bombs-,A%20logic%20bomb%20is%20a%20malicious%20program%20that%20is%20triggered,a%20specific%20date%20and%20time.
- https://en.wikipedia.org/wiki/Category:Types_of_malware
- https://www.kaspersky.com/resource-center/threats/malware-classifications

APTs (Ameaças Persistentes Avançadas)

- Uma ameaça persistente avançada (APT) é um ator furtivo de ameaças à rede de computadores, geralmente um estado nacional ou um grupo patrocinado pelo estado, que obtém acesso não autorizado a uma rede de computadores e permanece despercebido por um longo período. Nos últimos tempos, o termo também pode se referir a grupos patrocinados não estatais que conduzem intrusões direcionadas em larga escala para objetivos específicos.
- As motivações desses atores de ameaças são tipicamente políticas ou econômicas. Todos os principais setores empresariais registraram casos de ataques de atores avançados com objetivos específicos que procuram roubar, espionar ou interromper. Esses setores incluem governo, defesa, serviços financeiros, serviços jurídicos, industriais, telecomunicações, bens de consumo e muito mais. Alguns grupos utilizam vetores de espionagem tradicionais, incluindo engenharia social , inteligência humana e infiltração para obter acesso a um local físico para permitir ataques à rede. O objetivo desses ataques é colocar código malicioso personalizado em um ou vários computadores para tarefas específicas.
- A mediana "tempo de permanência", a hora em que um ataque do APT passa despercebida, difere amplamente entre as regiões. O FireEye relata que o tempo médio de permanência para 2018 nas Américas é de 71 dias, EMEA é de 177 dias e APAC é de 204 dias. Isso permite que os atacantes passem uma quantidade significativa de tempo no ciclo de ataque, se propagem e atinjam seus objetivos.

https://www.kaspersky.com/resource-center/definitions/advanced-persistent-threats

MITRE ATTACK

- A MITRE introduziu o ATT&CK (Adversarial Tactics, Techniques & Common Knowledge - Táticas, técnicas e conhecimento comum dos inimigos) em 2013 como uma forma de descrever e classificar os comportamentos dos inimigos com base em observações do mundo real. O ATT&CK é uma lista estruturada de comportamentos conhecidos do agressor, que foram compilados em táticas e técnicas e expressos em várias matrizes, bem como via STIX/TAXII. Como essa lista é uma representação abrangente dos comportamentos dos agressores ao comprometer as redes, ela é útil para várias análises ofensivas e defensivas, representações e outros mecanismos.
- **Entendendo as matrizes do ATT&CK**
- A MITRE dividiu o ATT&CK em várias matrizes diferentes: Enterprise, Mobile e PRE-ATT&CK. Cada uma dessas matrizes contém várias táticas e técnicas associadas ao tema da matriz.
- A matriz Enterprise é formada por técnicas e táticas que se aplicam aos sistemas Windows, Linux e/ou MacOS. A Mobile contém táticas e técnicas que se aplicam a dispositivos móveis. A PRE-ATT&CK contém táticas e técnicas relacionadas às ações dos agressores *antes* de tentar explorar uma rede ou sistema em particular.
- https://www.anomali.com/pt/what-mitre-attck-is-and-how-it-is-useful

O ATT&CK é valioso em vários ambientes cotidianos. Qualquer atividade de defesa que faz referência aos agressores e seu comportamento pode se beneficiar da taxonomia do ATT&CK. Além de oferecer um léxico comum para os defensores cibernéticos, o ATT&CK também fornece uma base para testes de penetração e red teaming. Isso proporciona aos defensores e red teamers uma linguagem comum para se referir ao comportamento dos inimigos.

Exemplos de quando pode ser útil aplicar a taxonomia do ATT&CK:

- **Mapeamento de controles de defesa**
  - Os controles de defesa podem ter um significado bem claro ao fazer referência às táticas e técnicas do ATT&CK às quais se aplicam.
- **Busca de ameaças**
  - Mapear as defesas para o ATT&CK gera um mapa de falhas defensivas que fornecem aos caçadores de ameaças os locais perfeitos para encontrar atividades de agressores que passaram despercebidas.
- **Detecções e investigações**
  - A central de operações de segurança (SOC) e a equipe de resposta a incidentes podem fazer referência às técnicas e táticas do ATT&CK detectadas ou descobertas. Isso ajuda a entender onde se encontram os pontos fortes e fracos de defesa, valida das mitigações e os controles de detecção e pode revelar configurações incorretas e outros problemas operacionais.
- **Agendes de referenciação**
  - Os agentes e grupos podem ser associados a comportamentos específicos e definíveis.
- **Integrações de ferramentas**
  - Diferentes ferramentas e serviços podem padronizar-se em táticas e técnicas do ATT&CK, oferecendo uma coesão para a defesa que não costuma existir.
- **Compartilhamento**
  - Ao compartilhar informações sobre um ataque, um agente ou grupo ou controles de defesa, os defensores podem assegurar um entendimento comum utilizando as técnicas e táticas do ATT&CK.
- **Red Team/Atividades do teste de penetração**
  - O planejamento, execução e geração de relatórios do red team, purple team, e as atividades do teste de penetração podem usar o ATT&CK para falar a mesma língua dos defensores e destinatários dos relatórios, bem como entre eles.

MITRE ATTACK - VANTAGENS

MITRE ATTACK e APTS

- O Mitre é um bom Framework para entender como os APTs trabalham principalmente para disseminar um malware, seja na exploração de uma vulnerabilidade ou utilizando técnicas de Engenharia Social avançada para que um usuário execute seu programa malicioso.
- Entender o Mitre não é apenas útil para saber como funciona os ataques, mas em como se proteger e tomar as medidas necessárias.
- https://www.mitre.org/publications/project-stories/cybersecurity-defending-against-advanced-persistent-threats
- https://attack.mitre.org/techniques/enterprise/
- https://www.mitre.org/capabilities/cybersecurity/overview/cybersecurity-blog/7-steps-for-an-apt-detection-playbook-using
- https://www.peerlyst.com/posts/using-mitre-att-and-ck-to-defend-against-advanced-persistent-threats-chiheb-chebbi

CYBER KILL CHAIN

- O termo Kill Chains foi originalmente usado como um conceito militar relacionado à estrutura de um ataque ; consistindo na identificação do alvo, forçar o despacho no alvo, decisão e ordem para atacar o alvo e, finalmente, a destruição do alvo. Por outro lado, a idéia de "quebrar" a cadeia de mortes de um oponente é um método de defesa ou ação preventiva. Mais recentemente, a Lockheed Martin adaptou esse conceito à segurança da informação , usando-o como um método para modelar intrusões em uma rede de computadores . O modelo de cadeia de assassinatos cibernéticos já foi adotado na comunidade de segurança da informação. No entanto, a aceitação não é universal, com os críticos apontando para o que eles acreditam serem falhas fundamentais no modelo.
- https://en.wikipedia.org/wiki/Kill_chain
- https://www.proof.com.br/blog/o-que-e-cyber-kill-chain/
- https://blogbrasil.westcon.com/cyber-kill-chain-5-etapas-para-eliminar-um-ciberataque
- https://www.varonis.com/blog/cyber-kill-chain/
- https://computerworld.com.br/2018/07/16/voce-conhece-o-conceito-de-cyber-kill-chain/

CYBER KILL CHAIN – F2T2EA

F2T2EA

Um modelo militar de Kill Chain é o "F2T2EA", que inclui as seguintes fases:

- Localizar: identifique um alvo. Encontre um alvo nos dados de vigilância ou reconhecimento ou por meios de inteligência.
- Correção: fixa a localização do alvo. Obtenha coordenadas específicas para o destino a partir dos dados existentes ou coletando dados adicionais.
- Faixa: Monitore o movimento do alvo. Acompanhe o alvo até que seja tomada uma decisão de não engajá-lo ou que o engajamento seja bem-sucedido.
- Alvo: Selecione uma arma ou ativo apropriado para usar no alvo para criar os efeitos desejados. Aplique os recursos de comando e controle para avaliar o valor do alvo e a disponibilidade de armas apropriadas para envolvê-lo.
- Envolver: Aplique a arma ao alvo.
- Avaliar: avalie os efeitos do ataque, incluindo qualquer inteligência coletada no local.
- Este é um processo integrado, de ponta a ponta, descrito como uma "cadeia", porque uma interrupção em qualquer estágio pode interromper todo o processo.

https://www.airforcemag.com/article/0700find/

MALWARE – OBFUSCATION

Ofuscação de malware

- Compreender a ofuscação é mais fácil do que pronunciá-la. A ofuscação de malware torna os dados ilegíveis. Quase todo malware usa-o.
- Os dados incompreensíveis geralmente contêm palavras importantes, chamadas "strings". Algumas seqüências contêm identificadores como o nome do programador de malware ou a URL da qual o código destrutivo é extraído. A maioria dos malwares tem ofuscado seqüências de caracteres que ocultam as instruções que informam à máquina infectada o que fazer e quando fazer.
- A ofuscação oculta tão bem os dados do malware que os analisadores de código estático simplesmente passam. Somente quando o malware é executado é que o código verdadeiro é revelado.

Técnicas simples de ofuscação de malware

- Técnicas simples de ofuscação de malware como OR (XOR) exclusivo, Base64, ROT13 e codepacking são comumente usadas. Essas técnicas são fáceis de implementar e ainda mais fáceis de ignorar. A ofuscação pode ser tão simples quanto o texto interposto ou o preenchimento extra dentro de uma string. Mesmo olhos treinados muitas vezes perdem o código ofuscado.
- O malware imita os casos de uso diários até ser executado. Após a execução, o código malicioso é revelado, espalhando-se rapidamente pelo sistema.

Técnicas avançadas de ofuscação de malware

- A ofuscação de malware de próximo nível é ativa e evasiva. Técnicas avançadas de malware, como conscientização ambiental, ferramentas automatizadas confusas, evasão baseada em tempo e ofuscação de dados internos, permitem que o malware ocorra nos ambientes operacionais e voe sob o radar de um software antivírus respeitável.
- Alguns malwares prosperam com o clique de isca dos usuários no download de arquivos de malware ou na abertura de páginas maliciosas, enquanto outros interceptam o tráfego e injetam malware, obtendo um impacto vasto e rápido.

MALWARE – ENCODING

- O termo *codificação de dados* se refere a todas as formas de modificação de conteúdo com o objetivo de ocultar a intenção. O malware usa técnicas de codificação para mascarar suas atividades maliciosas e, como analista de malware, você precisará entender essas técnicas para entender completamente o malware.
- Ao usar a codificação de dados, os invasores escolhem o método que melhor atende às suas metas. Às vezes, eles escolhem cifras simples ou funções básicas de codificação que são fáceis de codificar e fornecem proteção suficiente; outras vezes, usarão cifras criptográficas sofisticadas ou criptografia personalizada para dificultar a identificação e a engenharia reversa.
- https://www.oreilly.com/library/view/practical-malware-analysis/9781593272906/ch14.html
- https://medium.com/@bromiley/malware-monday-obfuscation-f65239146db0

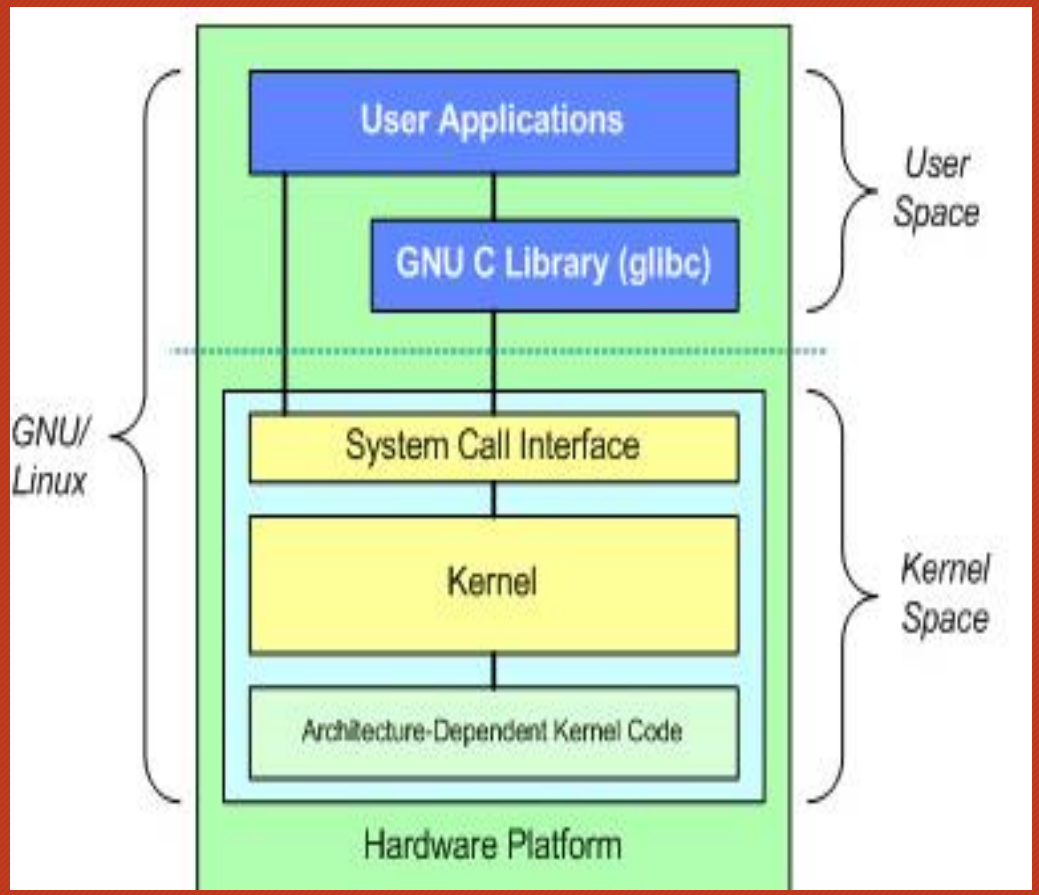
MALWARE – CRYPTO

- No sentido tradicional, a criptografia de malware é o processo de codificação de informações, para que apenas as partes autorizadas possam acessar os dados em um formato legível.
- Quando a criptografia de malware é usada para intenções maliciosas, ela é chamada ransomware. O Ransomware mantém os arquivos como reféns usando criptografia. Quando o pagamento do resgate é recebido, os arquivos são descriptografados e o usuário recupera o acesso. Os criadores de malware de hoje normalmente solicitam pagamento na forma de criptomoeda ou cartão de crédito.
- O malware geralmente infecta os sistemas quando esquemas de phishing ou outras táticas de e-mail, posando como e-mail legítimo, convencem o usuário a clicar em um link ou baixar um arquivo.
- https://blog.malwarebytes.com/threat-analysis/2018/02/encryption-101-malware-analysts-primer/

KERNEL

- Todo sistema operacional possui um kernel. Windows, macOS, iOS, Android, SO Chrome e Linux têm, cada um, um sistema de baixo nível que é responsável pela interface de todos os softwares com o hardware físico do computador. Sem o kernel, nenhum dos seus aplicativos seria capaz de usar o computador físico; aplicativos como Firefox, Chrome, LibreOffice, MS Office ou Outlook não funcionariam. O kernel também é responsável por permitir que os processos troquem informações usando o que é chamado Comunicação entre Processos (IPC).
- **Existem três tipos de kernels:**
- **Kernels monolíticos**: Esses kernels abrangem a CPU, a memória, o IPC, os drivers de dispositivos, o gerenciamento do sistema de arquivos e as chamadas do servidor do sistema. Também é responsável por distribuir memória do sistema para os aplicativos. Esses tipos de kernel geralmente são melhores para acessar o hardware e multitarefa.
- **Microkernels**: Microkernels adotam uma abordagem minimalista e gerenciam apenas a CPU, a memória e o IPC.
- **Kernels Híbridos:** Os Kernels Híbridos têm a capacidade de decidir o que eles querem executar no Modo Usuário ou Kernel. Embora isso forneça o melhor dos dois mundos, exige muito mais dos fabricantes de hardware para criar drivers que sirvam para fazer interface entre a execução de código e hardware.

KERNEL LINUX



https://www.ibm.com/developerworks/br/library/l-linux-kernel/index.html
https://www.escolalinux.com.br/blog/kernel-do-linux-o-que-e-e-para-que-serve
https://www.significados.com.br/kernel/
https://ic.unicamp.br/~islene/1s2009-mc514/Kernel_Linux.pdf
http://tiagoconceicao.pt/resources/userfiles/articles/files/works/pirlc/asl_kernel_linux.pdf
http://www.staroceans.org/kernel-and-driver/Understanding_Linux_Kernel_-_en.pdf

ROOTKITS

- Um rootkit é um programa ou, mais frequentemente, uma coleção de ferramentas de software que fornece a um agente de ameaças acesso remoto e controle sobre um computador ou outro sistema. Embora tenha havido usos legítimos para esse tipo de software, como fornecer suporte remoto ao usuário final, a maioria dos rootkits abre um backdoor nos sistemas das vítimas para introduzir software malicioso, como vírus, ransomware, programas de keylogger ou outros tipos de malware, ou para usar o sistema para mais ataques à segurança da rede. Os rootkits geralmente tentam impedir a detecção de software malicioso pelo software antivírus do terminal.
- https://www.incibe-cert.es/en/blog/kernel-rootkits-en
- https://securelist.com/rootkit-evolution/36222/
- https://resources.infosecinstitute.com/rootkits-user-mode-kernel-mode-part-2/
- https://www.imperva.com/learn/application-security/rootkit/
- https://www.lume.ufrgs.br/bitstream/handle/10183/26345/000757798.pdf?sequence=1

SANDBOXES

- Na segurança do computador , um sandbox é um mecanismo de segurança para separar programas em execução, geralmente em um esforço para atenuar a disseminação de falhas no sistema ou vulnerabilidades de software. É frequentemente usado para executar programas ou códigos não testados ou não confiáveis, possivelmente de terceiros, fornecedores, usuários ou sites não verificados ou não confiáveis, sem arriscar danos à máquina host ou ao sistema operacional. Normalmente, uma caixa de areia fornece um conjunto de recursos rigidamente controlado para a execução de programas convidados, como espaço de armazenamento e espaço de memória . O acesso à rede, a capacidade de inspecionar o sistema host ou ler os dispositivos de entrada geralmente não são permitidos ou são fortemente restringidos.
- No sentido de fornecer um ambiente altamente controlado, as caixas de areia podem ser vistas como um exemplo específico de virtualização . O sandboxing é frequentemente usado para testar programas não verificados que podem conter vírus ou outro código malicioso, sem permitir que o software danifique o dispositivo host.

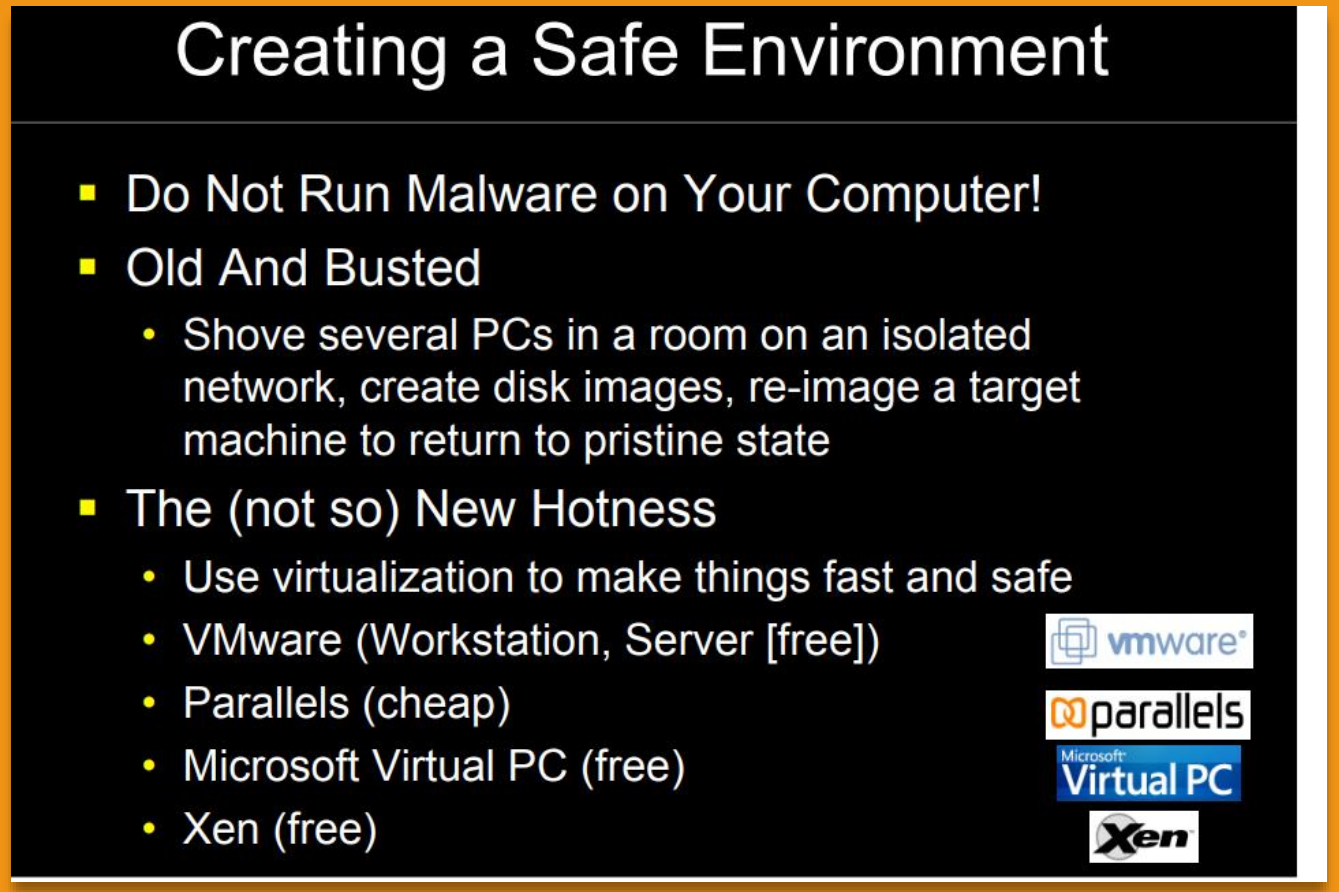
Engenharia Reversa

- A engenharia reversa é o processo de levar um binário compilado e tentar recriar (ou simplesmente entender) a forma original do programa. Um programador inicialmente escreve um programa em uma linguagem de alto nível, como C ++, C#, Java ou Visual Basic (A.k.a Delphi, Pascal, Assembly). Como o computador não fala essas línguas, o código que o programador escreveu é montado de uma maneira que a maquina consiga traduzir, ao qual um computador fala. Este código é chamado de binário, ou o idioma da maquina. Eles não são muito amigáveis, e muitas vezes requer uma grande quantidade de poder cerebral para descobrir exatamente o que o programador tinha em mente.
- https://medium.com/@leonardomarciano/engenharia-reversa-1-in%C3%ADcio-de-uma-grande-aventura-9526447ee50e
- https://www.youtube.com/watch?v=IkUfXfnnKH4
- https://www.mentebinaria.com.br/forums/topic/90-caminho-para-os-iniciantes-em-engenharia-reversa/

PRÁTICA

ANALISE DE MALWARE

AMBIENTE DE TESTES



https://www.blackhat.com/presentations/bh-dc-07/Kendall_McMillan/Presentation/bh-dc-07-Kendall_McMillan.pdf

AMBIENTE DE TESTES

Caso você queira montar seu laboratório de testes, além das dicas anteriores, você tem outros meios:

https://www.malwaretech.com/2017/11/creating-a-simple-free-malware-analysis-environment.html

https://www.sans.org/reading-room/whitepapers/threats/paper/1841

https://resources.infosecinstitute.com/environment-for-malware-analysis/#gref

https://www.youtube.com/watch?v=gFxImi5t37c

https://www.youtube.com/watch?v=oPsxy9JF8FM

https://www.youtube.com/watch?v=iU3WZBWuYO4 (Papo Binário)

BlackStorm Security

Importante: se houver algum equipamento visivelmente atacado, é fundamental seguir alguns procedimentos para preservar a memória RAM e os dados que serão analisados.

1. Jamais desligue o equipamento, pois é importante termos acesso à memória RAM onde os malwares deixam rastros de sua presença e ação.
2. Não passe antimalwares ou antivírus, pois podem destruir informação relevante.
3. Isole a máquina da rede local imediatamente – isso pode ser feito tirando o cabo de rede ou, caso esteja usando WiFi, desligando o dispositivo via hardware (ALT-F2 em alguns casos).
4. Não formate o disco ou apague arquivos – novamente, isso pode apagar evidências importantes para nossa análise.
5. Nunca, jamais, restaure backup sem estar 100% seguro da desinfecção. Seu risco é contaminar seu backup e novamente o equipamento infectado/atacado.
6. E por fim, nunca considerar especialistas em Segurança da Informação como especialistas de Análise de Malware. Nesses casos é comum que, quanto mais o pessoal insista em reparar o problema, maior é a disseminação do código malicioso pela rede e/ou equipamento.

MEDIDAS PREVENTIVAS EM UM ATAQUE

FONTE: http://www.blackstormsecurity.com/bs/malware.html

Ferramentas de Analise de Malware 1

- Python for Malware Analysis
- X64dbg
- IDA Pro
- Radare2
- WinDBG
- ILSpy
- DIE
- Yara Rules
- Remnux
- Bro
- Any.run
- MISP
- strace

Ferramentas de Analise de Malware 2

- RegShot
- Pstools
- Process Monitor, Explorer and Hacker
- Pharos
- OllyDGB
- Mac-a-mal
- Immunity Debugger
- GDB
- Binwalk
- Binnavi
- BAP
- BARF

Ferramentas de Analise de Malware 3

- Capstone
- Codebro
- Cutter
- Dnspy
- Evans EDB
- Fport
- Fibratus
- Ghidra
- Triton
- String Sniffer
- SMRT
- Angr
- xortool

Ferramentas de Analise de Malware 4

- Floss
- NoMoreXOR
- VirtualDeobfuscator
- Hachoir3
- Bulk_extractor
- PDF Tools
- Spidermonkey
- Box-js
- SWF Investigator
- Firebug
- Krakatau
- Anlyz.io
- DeepViz

Ferramentas de Analise de Malware 5

- Firmware.re
- IRMA
- Intezer
- Joe Sandbox
- Virus Total
- Chkrootkit
- Hashdeep
- AnalyzePE
- https://github.com/alexandreborges

Etc…

Python for Malware Analysis

- Existem muitas ferramentas de análise de malware baseadas em Python que você pode usar hoje. Abaixo estão alguns exemplos que considero úteis para a análise estática de arquivos:
- pyew
- AnalyzePE
- pescanner
- peframe
- pecheck (eu discuti este anteriormente)
- Essas ferramentas produzem resultados úteis e servem como excelentes pontos de partida para entender o Python. Simplesmente visualizando o código fonte e realizando pesquisas conforme necessário, você pode aprender com o que os autores escreveram e modificar o código para servir a seu próprio objetivo. No entanto, ao adquirir experiência em análise técnica, você provavelmente encontrará cenários em que as ferramentas existentes não atendem às suas necessidades e uma solução personalizada deve ser desenvolvida. Tenha certeza, esses casos não exigem que você escreva código do zero. Em vez disso, você pode confiar nas bibliotecas Python existentes para extrair dados e manipular a saída de uma maneira específica para suas necessidades.
- https://malwology.com/2018/08/24/python-for-malware-analysis-getting-started/
- https://www.andreafortuna.org/2017/06/29/python-for-malware-analysis/
- https://www.youtube.com/watch?v=2gyAemhbxnE
- https://www.youtube.com/watch?v=tNxJzx754BI

Python for Malware Analysis

- https://pdfs.semanticscholar.org/774c/d0dfc5e674356b3d8453f3c89b949ec1a811.pdf
- https://www.oreilly.com/library/view/machine-learning-and/9781491979891/ch04.html
- https://www.malwaretech.com/2018/03/best-programming-languages-to-learn-for-malware-analysis.html

ANTI-VMs

- Analistas e investigadores de malware geralmente usam ambientes isolados, como máquinas virtuais (VMs) ou sandboxes, para analisar códigos desconhecidos de malware. Da mesma maneira, os produtos de segurança geralmente usam VMs e caixas de proteção para executar códigos potencialmente maliciosos antes de serem aprovados para entrar na rede organizacional.
- Na tentativa de evitar a análise e ignorar os sistemas de segurança, os autores de malware geralmente projetam seu código para detectar ambientes isolados. Depois que esse ambiente é detectado, o mecanismo de evasão pode impedir a execução do código malicioso ou alterar o comportamento do malware para evitar a exposição de atividades maliciosas durante a execução em uma VM.
- Por exemplo: ao executar em hardware real, o malware se conectará ao servidor de Comando e Controle (C&C), mas quando uma VM for detectada, ela se conectará a um domínio legítimo, fazendo com que o analista ou o sistema de segurança acredite que esse é um código legítimo.
- https://www.cyberbit.com/blog/endpoint-security/anti-vm-and-anti-sandbox-explained/
- https://www.deepinstinct.com/2019/10/29/malware-evasion-techniques-part-2-anti-vm-blog/
- https://resources.infosecinstitute.com/category/certifications-training/malware-analysis-reverse-engineering/anti-disassembly-anti-debugging-anti-virtual-machine/#gref
- https://github.com/ExpLife0011/anti-anti-vm-detection-dll-1

ANTI-Debuggers

- Os autores de malware sempre procuraram novas técnicas para permanecerem invisíveis. Isso inclui, é claro, ser invisível na máquina comprometida, mas é ainda mais importante ocultar indicadores e comportamentos maliciosos durante a análise. Os autores de malware assumem que, uma vez que o arquivo malicioso esteja fora do ar, seu relógio vitalício está correndo e, eventualmente, ele será detectado pelos pesquisadores e analisado minuciosamente.
- Para tornar a análise pós-detecção mais difícil, os atores de ameaças usam várias técnicas anti-análise, uma das mais comuns é a Anti-Depuração. Os atores de ameaças provaram ser mais inovadores, não apenas no malware que estão criando, mas também nas técnicas que estão empregando para evitar a detecção e análise por analistas e produtos de segurança cibernética. A anti-depuração, portanto, representa um obstáculo para os analistas de malware, pois pode prolongar o processo de engenharia reversa do código e, assim, dificultar a decifração de como ele funciona. Depois que o malware percebe que está sendo executado em um depurador, ele pode ajustar seu caminho de execução de código usual ou modificar o código para provocar uma falha, o que dificulta as tentativas dos analistas de decifrá-lo, ao mesmo tempo em que acrescenta tempo e sobrecarga adicional ao seu esforços.

ANTI-Debuggers

- https://resources.infosecinstitute.com/category/certifications-training/malware-analysis-reverse-engineering/anti-disassembly-anti-debugging-anti-virtual-machine/#gref
- https://resources.infosecinstitute.com/anti-debugging-and-anti-vm-techniques-and-anti-emulation/#gref
- https://securityaffairs.co/wordpress/88546/malware/anti-debugging-techniques-malware.html
- https://www.lasca.ic.unicamp.br/paulo/papers/2017-SBSeg-marcus.botacin-anti.anti.analysis.evasive.malware.pdf
- http://antukh.com/blog/2015/01/19/malware-techniques-cheat-sheet/

ANTI-Disassembly

- As técnicas anti-disassembly funcionam tirando proveito das suposições e limitações dos disassemblers. Por exemplo, disassemblets podem representar apenas cada byte de um programa como parte de uma instrução por vez. Se o disassemblers for levado a disassembling no offset errado, uma instrução válida poderá ser ocultada.
- https://medium.com/@preetkamal1012/anti-disassembly-techniques-e012338f2ae0
- https://pt.slideshare.net/SamBowne/practical-malware-analysis-ch-15-antidisassembly
- https://j33m.net/reversing/2018/05/07/analyzing-malware-with-anti-disassembly.html
- https://www.malwinator.com/2015/11/22/anti-disassembly-used-in-malware-a-primer/

Reverse Engineering Malware

- A engenharia reversa de malware envolve desmontar (e às vezes descompilar) um programa de software. Por meio desse processo, as instruções binárias são convertidas em mnemônicas de código (ou construções de nível superior) para que os engenheiros possam ver o que o programa faz e quais sistemas ele afeta. Somente conhecendo seus detalhes, os engenheiros são capazes de criar soluções que podem atenuar os efeitos maliciosos pretendidos pelo programa. Um engenheiro reverso (também conhecido como "reversor") usará uma variedade de ferramentas para descobrir como um programa está se propagando através de um sistema e o que ele foi projetado para fazer. E, ao fazer isso, o inversor saberia quais vulnerabilidades o programa pretendia explorar.
- Os engenheiros reversos são capazes de extrair dicas reveladoras quando um programa foi criado (embora se saiba que os autores de malware deixam rastros falsos), quais recursos incorporados eles podem estar usando, chaves de criptografia e outros detalhes de arquivos, cabeçalhos e metadados. Quando o **WannaCry** foi **submetido** a engenharia reversa, as tentativas de encontrar uma maneira de rastrear sua propagação levaram a descobrir o que hoje é conhecido como seu "interruptor de interrupção" - um fato que provou ser incrivelmente importante para impedir sua disseminação.

Reverse Engineering Malware

Para reverter o código de malware, os engenheiros costumam usar muitas ferramentas. Abaixo uma pequena seleção dos mais importantes:

- Desmontadores (por exemplo, IDA Pro). Um desmontador desmontará um aplicativo para produzir código de montagem. Os descompiladores também estão disponíveis para a conversão de código binário em código nativo, embora não estejam disponíveis para todas as arquiteturas.
- Depuradores (por exemplo, x64dbg, Windbg, GDB). Os inversores usam depuradores para manipular a execução de um programa, a fim de obter informações sobre o que está fazendo quando está em execução. Eles também permitem que o engenheiro controle certos aspectos do programa enquanto ele estiver em execução, como áreas da memória do programa. Isso permite mais informações sobre o que o programa está fazendo e como está impactando um sistema ou rede.
- Visualizadores de PE (por exemplo, CFF Explorer, PE Explorer). Os visualizadores de PE (para formato de arquivo executável portátil do Windows) extraem informações importantes dos executáveis para fornecer, por exemplo, a visualização de dependências.
- Analisadores de rede (por exemplo, Wireshark). Os analisadores de rede informam a um engenheiro como um programa está interagindo com outras máquinas, incluindo quais conexões o programa está fazendo e quais dados ele está tentando enviar.
- https://zeltser.com/reverse-engineering-malware-methodology/
- https://www.youtube.com/watch?v=EMJr_B0mWUY
- https://www.youtube.com/watch?v=raoO1tYDpIo
- https://www.youtube.com/watch?v=yZ6D4-Jz3Hc
- https://www.youtube.com/watch?v=3NTXFUxcKPc

Reverse Engineering Malware

https://www.youtube.com/watch?v=35teUHnZNGU

https://www.youtube.com/watch?v=VOTnt2PdqNM

https://www.youtube.com/watch?v=VOTnt2PdqNM

https://www.youtube.com/watch?v=aQqMXujUBhE

https://www.youtube.com/watch?v=NVP4X12BVcc

https://www.youtube.com/watch?v=gjCKKLuDoP8

https://www.youtube.com/watch?v=rcA2tPp4nSU (Alexandre Borges)

https://www.youtube.com/watch?v=UB3pVTO5izU

https://www.youtube.com/watch?v=bCaMuHAJcHw

https://medium.com/secjuice/the-road-to-reverse-engineering-malware-7c0bc1bda9d2

https://cybersecurity.att.com/blogs/labs-research/reverse-engineering-malware

https://www.infosecinstitute.com/skills/learning-paths/malware-analysis-reverse-engineering/

PRÁTICA: ANALISE DE MALWARE ESTÁTICA

https://www.theta432.com/post/malware-analysis-part-1-static-analysis

https://enterprise.comodo.com/forensic-analysis/static-malware-analysis.php

https://hackercombat.com/static-malware-analysis-vs-dynamic-malware-analysis/

https://medium.com/bugbountywriteup/malware-analysis-101-basic-static-analysis-db59119bc00a

https://www.youtube.com/watch?v=SIem8ZIe1xk

https://www.youtube.com/watch?v=FpcDdlL0Y1E

https://www.youtube.com/watch?v=H3BJ4gmPEm4

https://scholarworks.sjsu.edu/cgi/viewcontent.cgi?referer=https://www.google.com/&httpsredir=1&article=1391&context=etd_projects

https://arxiv.org/abs/1808.01201

http://www.ozgurcatak.org/files/02-static_analysis.pdf

https://www.lasca.ic.unicamp.br/paulo/papers/2010-SBSEG-dario.fernandes-andre.gregio-vitor.afonso-rafael.santos-mario.jino-analise.malware.pdf

https://periciacomputacional.com/analise-de-malware-na-forense-computacional/

https://github.com/sully90h/practical-malware-analysis

PRÁTICA: ANALISE DE MALWARE DINÂMICA

https://www.youtube.com/watch?v=XgtG9p_Pr9E

https://www.youtube.com/watch?v=J-nYVgHZqjo

https://www.youtube.com/watch?v=5cvpGSSUZI0

https://www.youtube.com/watch?v=qtoS3CG6ht0

https://www.youtube.com/watch?v=Sv8yu12y5zM&t=286s

https://www.youtube.com/watch?v=XgtG9p_Pr9E&list=PLUFkSN0XLZ-kqYbGpY4Gt_VATd4ytQg-Z (Playlist)

https://www.youtube.com/watch?v=6ObusWC-Qcs

https://www.youtube.com/watch?v=Q5AnGdMP4Tw

https://www.youtube.com/watch?v=tgJR1-mkvzY

http://www.rennes.supelec.fr/diwall/talks/kruegel.pdf

https://publications.sba-research.org/publications/malware_survey.pdf

https://informationsecurity.report/Resources/Whitepapers/51e831f9-aeef-41a4-b2e9-5162a2ac5f65_How%20Malware%20Analysis.pdf

https://pdfs.semanticscholar.org/7efb/d1be5679425631e204ce06d1ab3c28a2e281.pdf

https://github.com/sully90h/practical-malware-analysis (LABS)

PRÁTICA: ADVANCED MALWARE ANALYSIS

https://www.youtube.com/watch?v=aYQ4TIcGD2o

https://www.youtube.com/watch?v=67vesKcxQOQ

https://www.youtube.com/watch?v=9L9I1T5QDg4

https://www.youtube.com/watch?v=YF-o0ig5NQo

https://www.youtube.com/watch?v=WlE8abc8V-4&t=3s

https://www.youtube.com/watch?v=UZzd096Z850

https://www.youtube.com/watch?v=7zKHxL99ytQ

https://www.youtube.com/watch?v=SFaDTQiiiww

https://www.youtube.com/watch?v=EBVhX_1vaoE

https://www.blackhat.com/presentations/bh-usa-08/Laspe_Raber/BH_US_08_Laspe_Raber_Deobfuscator.pdf

https://www.youtube.com/watch?v=ZsUx__lny2Q

https://www.youtube.com/watch?v=6gYZdS-co7s

https://www.youtube.com/watch?v=l2P5CMH9TE0

PRÁTICA: ADVANCED MALWARE ANALYSIS

https://www.youtube.com/watch?v=CiJocXXMXK4

https://www.youtube.com/watch?v=Sv8yu12y5zM&t

https://www.youtube.com/watch?v=C6vsdyZnPPo

https://www.youtube.com/watch?v=PBvMnjQPDac

https://www.vmray.com/cyber-security-blog/paymen45-ransomware-malware-analysis-spotlight/

https://medium.com/@knownsec404team/lucky-ransomware-analysis-and-file-decryption-1581a7180c1c

https://blog.malwarebytes.com/threat-analysis/2018/06/malware-analysis-decoding-emotet-part-2/

https://blog.malwarebytes.com/threat-analysis/2018/05/malware-analysis-decoding-emotet-part-1/

https://www.youtube.com/watch?v=E_70CSwcU7E

https://www.youtube.com/watch?v=-0DEEbQq8jU

BÔNUS: Evade Any.run?

https://www.nagenrauft-consulting.com/2019/11/14/app-any-run-anti-evasion-bypass/

https://securityaffairs.co/wordpress/105830/malware/any-run-sandbox-evasion.html

https://www.financialcert.tn/2020/07/13/malware-adds-any-run-sandbox-detection-to-evade-analysis/

https://cyware.com/news/malware-authors-develop-new-method-to-evade-analysis-by-anyrun-sandbox-29cbe32f

BÔNUS: Impactos de um Malware

http://techgenix.com/malwares-impact-serious-long-lasting/#:~:text=Effects%20malware%20can%20have%20on%20your%20computer&text=Malware%20causes%20your%20computer%20to,cause%20your%20computer%20to%20crash.&text=Malware%20could%20be%20used%20for,to%20sites%20for%20its%20purposes.

https://usa.kaspersky.com/resource-center/threats/malware-damage

https://www.nist.gov/system/files/documents/itl/BITS-Malware-Report-Jun2011.pdf

https://www.digintrude.com/malwares-and-its-impact-on-business.html

https://www.trendmicro.com/vinfo/us/security/research-and-analysis/threat-reports

https://www.youtube.com/watch?v=I06wFfgn5eE (Palestra Mercês)

https://resources.malwarebytes.com/files/2020/02/2020_State-of-Malware-Report.pdf

https://blog.emsisoft.com/en/36303/ransomware-statistics-for-2020-q1-report/

BÔNUS: Impactos de um Malware

https://cybermap.kaspersky.com/pt

https://www.crowdstrike.com/resources/reports/2020-crowdstrike-global-threat-report/

Filtre no Google Noticias pesquisas como “Ransomware”, “Malware”, “Viruses”, “Advanced Persistent Threats”, “Threats Advanced” e etc.

Cursos para estudar

https://www.youtube.com/c/PapoBin%C3%A1rio/playlists (Papo Binário)

http://www.blackstormsecurity.com/bs/treinamento.html

https://esecurity.com.br/cursos/security-specialist/

https://hackersec.com/treinamentos/curso-analise-de-malware-fundamentals/

https://www.udemy.com/

https://hakin9.org/

https://www.fireeye.com/services/training/courses/malware-analysis-master-course.html

https://www.amazon.com.br/Practical-Malware-Analysis-Hands-Dissecting/dp/1593272901

REFERENCE

https://www.crowdstrike.com/epp-101/malware-analysis/
https://en.wikipedia.org/wiki/Malware_analysis
https://www.baboo.com.br/artigos/como-antivirus-funcionam/
https://www.programacaoprogressiva.net/2020/05/registradores-dos-microprocessadores-Intel-x86.html
https://pt.wikipedia.org/wiki/AMD64
https://www.ic.unicamp.br/~rodolfo/Cursos/mo401/2s2005/Trabalho/041438-x86-apresentacao.pdf
https://en.wikipedia.org/wiki/Windows_API
https://pt.wikipedia.org/wiki/DLL
https://security.stackexchange.com/questions/74631/what-is-the-difference-between-dll-hooking-and-dll-injection
https://en.wikipedia.org/wiki/Advanced_persistent_threat
https://resources.infosecinstitute.com/category/certifications-training/malware-analysis-reverse-engineering/malware-obfuscation-encoding-encryption/#gref
https://searchsecurity.techtarget.com/definition/rootkit
https://www.oficinadanet.com.br/linux/24969-o-que-e-o-kernel-do-linux
https://github.com/rshipp/awesome-malware-analysis
https://www.lastline.com/blog/reverse-engineering-malware/
https://en.wikipedia.org/wiki/Sandbox_(computer_security)