Fichamento dos Artigos

João Roberto da Silva Porto - 222217111

Abril de 2024

Atividade 1 do curso de PSB lecionado pela Prof. Mirlei Moura.

Primeiro serão feitas descrições e análises individuais acerca de cada texto de uma série de artigos, então seguidas por um breve estudo comparativo entre às respectivas obras com enfâse em critérios específicados na descrição da atividade pela docente.

1 Análise individual

1.1 Assembly Reverse Analysis on Malicious Code of Web Rootkit Trojan

1.1.1 Título

Assembly Reverse Analysis on Malicious Code of Web Rootkit Trojan, do português, Análise de Assembly Reverso sobre Código Malicioso em Trojan de RootKit Web.

1.1.2 Ano

2009.

1.1.3 País

China.

1.1.4 Autores

Yong Wang, Dawu Gu, Janping Xu e Fenyu Zen.

1.1.5 Objetivo

Entender o functionamento e, preferivelmente, encontrar vias de imunização para agentes maliciosos que atuem através de rootkit's, o que habilitaria até uso da interface ACPI para armazenamento persistente do malware.
Mais especificamente são analisados os vírus *web downloader machine dog Trojan* e *BIOS Trojan*.

1.1.6 Ferramentas utilizadas

Turbo Assembler (TASM), mais especificamento as versões 5.0 e 4.0, incluindo também linkers, debuggers, montadores e compiladores de recursos, tanto nas iterações 16 e 32 bits para análise do virús CIH, um BIOS Trojan, para entender seu funcionamento.
Microsoft Macro Assembler (MASM) para análise dos hooks de interrupção usados pelos rootkit's para obterem informações necessárias para escalação de privilégio. OllyDbg, debugger de 32 bits, é usado para análise do Machine Dog. Award Bios Editor, ferramenta que permite extrair e alterar configurações da BIOS.

1.1.7 Uso da linguagem Assembly

É utilizado assembly para compreensão do processo de infiltração silenciosa do CIH na BIOS, o que ocorre através dos primeiros megabytes da partição 0 do disco de boot, associado à manipulação de hooks de interrupção (syscall's) que são 'sequestradas' pelo agente malicioso, o qual as nota a fim de deduzir os endereços de memória sensíveis e modifica-los, por fim alterando os hooks seguintes incluindo o procedure do vírus na rotina.
Também é descrito o uso de assembly para localização da string da chamada de importação do Machine Dog e outras informações relevantes associadas, como o IP para requisição do Trojan. Essas informações são centrais para o último

segmento do artigo onde é proposta um tratamento ao Trojan Web baseado nas informações, como nomes de processos e arquivos visíveis no código de montagem. Além disto, é também analisada ACPI Machine Language (AML) para entendimento do funcionamento do Trojan de BIOS, observando o código assembly que performa a infiltração por meio de configurações da interface ACPI para acessar espaços ROM desprotegidas em módulos como placas de som e rede.

1.1.8 Principais resultados e conclusões

O uso de estratégias de desassembly e análise reversa são centrais no entendimento do funcionamento desses malwares, e de suas respectivas imunizações. Ainda assim, foi notada a facilidade relativa para implementação especialmente de Web Trojan's através do uso de protocolos que habilitem criptografia, como HTTPS, dificultando analise de pacotes enviados pela própria maquina.
O autor destaca também a necessidade de mais ferramentas voltadas para analises do tipo, desde mecanismos para melhor monitoramento de registros do sistema até observação de tabelas da BIOS.

1.2 Detection of Malware using Machine Learning based on Operation Code Frequency

1.2.1 Título

Detection of Malware using Machine Learning based on Operation Code Frequency, traduzível ao português como, Detecção de Malware usando Aprendizado de Maquina baseado em Frequência de Opcodes.

1.2.2 Ano

2021

1.2.3 País

Índia

1.2.4 Autores

Pavitra Mohandas, M J Shankar Raman, Vasan V S, Sudesh Kumar Santhosh Kumar, Sandeep Pai Kulyadi e Balaji Venkataswami.

1.2.5 Objetivo

O principal objetivo é de validar o uso de Machine Learning para diagnóstico acurado de malware sem necessidade de execução do possível vetor de transmissão, impedindo possíveis vias colaterais de ataque, tomando como nota as limitações dos modelos de detecção por assinatura, que se baseiam apenas em agentes maliciosos já conhecidos, e comportamentais, atuam durante execução do vírus, expondo o sistema até a detecção e comumente produzindo falsos-positivos.
Outros alvos intermediários relevantes são: a inspeção e avaliação de técnicas de detecção de malware e construir uma tabela de frequência de sequências de opcodes presentes comumente em vírus.

1.2.6 Ferramentas utilizadas

Flask, web server operável em python, usado para envio de pdf's violados.
ObjDump, uma ferramenta de desassembly para sistemas unix, utilizada para conversão de executáveis gerados a partir de pdf's infectados com malware em arquivos .asm (com instruções assembly).
3 modelo de aprendizagem, sendo o principal deles do tipo Random Forest, para realizar as predições da natureza do executável.
Cuckoo SandBox para analises preliminares do caráter de arquivos potencialmente infectados.

1.2.7 Uso da linguagem Assembly

O código assembly obtido a partir dos executáveis embutidos em pdf's é encaminhado para analise quantitiva da frequência de seus opcode's após tratamento para remoção dos operandos, uma vez higienizados é feita mensuração a fim de obter tabelas de frequências úteis como *features* que alimentam os modelos de aprendizagem testados no estudo.
Essa abordagem é favorecida em função da natureza metamórfica, diferente de polimórfica, do malware estudado, que impossibilitam análises de assinatura direta.

1.2.8 Principais resultados e conclusões

É aferida e assegurada a eficiência do modelo de aprendizagem baseado nos dados extraídos a partir do desassembly para analise automatizada do código de montagem com taxa de falso-positivos e de acertos consideravelmente competitiva, notavelmente, em casos de vírus metamórficos, sendo essa uma majoritária vantagem frente às abordagens tradicionais de mitigação de ataques.

1.3 Opcode and API Based Machine Learning Framework For Malware Classification

1.3.1 Título

Opcode and API Based Machine Learning Framework For Malware Classification, em português, Framework de Machine Learning Baseado em Opcode e API para Classificação de Malware.

1.3.2 Ano

2022

1.3.3 País

Índia

1.3.4 Autores

Hrishabh Soni, Pushkar Kishore e Durga Prasad Mohapatra

1.3.5 Objetivo

A meta central do artigo é a produção de um framework suficiente e versátil de classificação de malware com uma performance superior aos modelos baseados em Deep Learning.
Esse objetivo tem como premissa as notadas deficiências dos modelos mais tradicionais baseados em Machine Learning, onde ainda há necessidade de conjuntos de 'features' de tamanho considerável e operacionalidade restrita a contexto específicos, resultando em baixa adaptabilidade real, e o alto custo energético e temporal associado ao treinamento de soluções baseadas Deep Learning.
Assim é proposta uma nova abordagem voltada para classificação procedural de 'features' baseadas em duas categorias de elementos Opcodes e chamadas de API.

1.3.6 Ferramentas utilizadas

Microsoft Malware Classification Challenge, um conjunto de arquivos para benchmark de ferramentas anti-vírus.
IDA PRO, desassembler usado para extração do código asm dos arquivos.
Modelo de aprendizagem do tipo Random Forest
Biblioteca Python Pyparsing

1.3.7 Uso da linguagem Assembly

Feito o desassembly, foi executada análise do código de montagem buscando identificar chamadas de funções e diretrizes de montagem que sinalizassem chamadas de DLL's (a serem unidos pelo linker), sendo estas chamadas de API's utilizadas como parâmetros a serem acrescidos aos datasets de treinamento. Em seguida também são extraídos os opcodes para integrarem mais features de treinamento.

1.3.8 Principais resultados e conclusões

O framework de classificação proposto apresenta performance superior à todos os demais sistemas descritos no corpo do trabalho, com margem considerável em parte das métricas verificadas. É concluído no estudo a eficiência de um modelo baseado no uso de n-engrams como camada seguinte a validação de decisões tomadas pelo modelo Random Forest. O autor também sublinha ao fim, o potencial de possíveis abordagens baseadas em redes neurais e outras técnicas de inteligência voltadas para interpretação dos resultados do framework proposto.

2 Análise Comparativa

Os artigos diferem um tanto na finalidade e metodologia mas são conectados pelo tema recorrente de análise e mitigação de danos e infecção por malware.
Apesar das soluções propostas diferirem em suas respectivas naturezas, partindo de tratamentos específicos voltados ao combate de agentes maliciosos singulares e já conhecidos, enquanto outro texto abrange metodologias focadas em agentes metamórficos propondo técnicas de Machine Learning como uma via de reação e prevenção, por fim o terceiro texto refinando essas metodologias baseadas em aprendizagem de maquina com um tratamento mais robusto destinado aos dados usados na composição dos datasets.
Sobretudo, o elemento chave comum às obras é a análise à nível de código de montagem como sendo meio mais eficiente para análise reversa de software malicioso, especialmente no contexto de observação estática e prevenção. Cada autor fazendo valer diferentes ferramentas tomando nota não só do entendimento das instruções como dos fluxos e etapas que compõem o processo de montagem desde o linker à transformação em binário, associado à conhecimento relativo ao fluxo de processamento de system calls para estudo atento às interrupções e chamadas do kernel. Tudo voltado ao entendimento das diferentes estratégias adotadas por infiltradores. Sendo notável a necessidade comum aos estudos mais contemporâneos de automatizar em alguma escala o processo de predição de comportamento danoso à medida que a produção de malware alcança velocidade industrial e constrói vetores de ataque com identidades cada vez mais mutáveis.