Exploration of Solutions for Malware Recognition in Files Prone to Phishing Techniques

(Strategic Heuristic Recognition & Lateralization of Cyber Knowledge – S.H.R.L.C.K.)

Projeto Final de Curso

Aluno: Adelino Daniel da Rocha Vilaça - a16939
Orientador: Professor Luis Gonzaga Martins Ferreira

Licenciatura em Engenharia de Sistemas Informáticos (PL)

3º ano

Barcelos | Novembro, 2024

Resumo

Palavras-Chave:

Lista de Abreviaturas e Siglas (Validar)

Índice de Figuras

Não foi encontrada nenhuma entrada do índice de ilustrações.

Índice

1. Introdução e Contextualização

Vivemos num mundo onde as ameaças cibernéticas evoluem diariamente, colocando em risco a segurança da informação pessoal, empresarial e governamental. Uma das formas mais recorrentes de ataque é o phishing, que utiliza técnicas de Engenharia Social para enganar utilizadores a abrirem ficheiros aparentemente inofensivos, como PDFs, documentos de Word ou folhas de cálculo do Excel, que muitas vezes estão carregados de malware. Estes ficheiros, ao serem executados ou abertos, podem comprometer sistemas inteiros, roubar dados sensíveis ou criar acessos remotos para atores maliciosos (Command and control / Botnets).

O projeto **SHRLCK** surge como uma resposta à necessidade de analisar, identificar e neutralizar essas ameaças através de uma abordagem baseada em heurística e análise de padrões recorrendo possivelmente a machine learning. O foco do projeto é explorar formas de deteção e classificação de código malicioso em ficheiros frequentemente utilizados em campanhas de phishing, analisando-os de forma sistemática e eficiente. O objetivo é compreender melhor as técnicas utilizadas para ocultar código malicioso, como a utilização do null padding (preenchimento de nulos), compactação e disfarces das assinaturas digitais, além de investigar como estas ameaças podem ser detetadas.

2. Contexto e Motivação

A motivação por trás deste projeto decorre da constante ameaça cibernética representada por ficheiros maliciosos. Estes ficheiros podem ser distribuídos em massa através de campanhas de spam ou por meios direcionados a grupos específicos (Spear Phishing), causando danos significativos e de longo alcance. Embora existam ferramentas comerciais para detetar malware, muitas delas dependem de bases de dados de assinaturas sinalizadas ou de uma análise superficial, o que nem sempre é eficaz contra novas ameaças, como no caso dos "Zero-Days" ou variantes ocultas por técnicas avançadas de ofuscação.

O **SHRLCK** visa explorar soluções avançadas que possam superar estas limitações, através da integração de técnicas de deteção heurística e da aplicação de inteligência artificial para inspeção de padrões comportamentais. O objetivo não é apenas detetar a presença de malware, mas também identificar o seu comportamento, assinatura e origem. Desta forma, espera-se contribuir

para o avanço na área da análise de malware, gerando conhecimento prático e explorando os desafios associados a este problema crítico na segurança digital.

3. Objetivos do Projeto

Explorar processos de identificação e de análise de malware em ficheiros regularmente utilizados para phishing, como PDFs, Excel e documentos Word, que evadem a detecção através de técnicas como null padding e dados inúteis. O projeto incluirá inspeção da estrutura dos ficheiros em causa, análise heurística/algorítmica e classificação de malware baseada em machine learning para identificar ameaças em blocos de código. (A ser revisto devido ao nível de complexidade/tempo disponível)

4. Fases e Plano do Projeto

a. Fase 0: Analisar o estado da arte (Investigar Teses)

A necessidade deste projeto deve-se à falta de soluções substitutas ao impedimento de execução de ficheiros envolvidos em phishing, utilizados diariamente no nosso dia a dia.

Soluções Atuais para Análise Estática / Dinâmica:

- **Antivírus e Anti-malware**: Softwares como VirusTotal, Norton, McAfee, Malwarebytes e outros são habitualmente usados para detecção e prevenção de malware. No entanto, muitos dependem principalmente de assinaturas/hashes já sinalizadas para identificar ameaças podendo não ser tão eficazes contra-ataques mais sofisticados, como zero-days.
- **Sandboxes**: Plataformas como Cuckoo Sandbox ou Hybrid Analysis analisam o comportamento de ficheiros executando-os em ambientes controlados. Esta abordagem pode identificar comportamentos maliciosos, como abertura de portas (Reverse Shell), downloads suspeitos ou manipulação de ficheiros do sistema.

- **Soluções EDR (Endpoint Detection and Response)**: Ferramentas como CrowdStrike e Carbon Black focam-se na detecção baseada em comportamentos, análise de processos e eventos, o que oferece uma camada extra de segurança, especialmente contra ameaças mais recentes.
- **Outras Soluções Específicas para Documentos**: Existem ferramentas dedicadas para analisar ficheiros PDF, Word, etc, em busca de macros maliciosas, como o Peepdf, DocBleach, PDFid, Oletools e o PDFParser.

b. Fase 1: Processamento Inicial dos Ficheiros e Remoção de Null Padding/Padrões de Dados inúteis

***Objetivo*:** Identificar e remover o null padding ou outros dados inúteis para reduzir o tamanho do ficheiro e revelar o conteúdo potencialmente malicioso para análise das signatures/hashes através das API's (Possível análise do tipo de ficheiro inserido + Owner, etc) (Muitas vezes apenas alteram o ícone do ficheiro e mantém o tipo de ficheiro facilmente visualizável como ou .exe ou como por exemplo .scr)

Passos:

Carregar ficheiros (PDFs, Word, Excel) no programa.

Utilizar técnicas de heurísticas/algoritmia para detectar e remover excessos de nulls ou padrões de dados inúteis, como “âââââ” ou “çççççç”.

Ferramentas:

Linguagem: Python

Bibliotecas: pandas para manipulação de dados, PyPDF2, python-docx, openpyxl para processamento de ficheiros (Pesquisar mais)

Análise Conjunta de Algoritmo de padrões com Heurística (Caso um dos casos seja ç . . . ç . . . . ç . . . . . ç . . ç . .) (Tornando-se dificil verificar padrões dinâmicos) (Então usando um algoritmo de, por exemplo, contador de caracteres seguidos ou separados por "." + Heuristica de Pattern Recognition (KMP (Knuth-Morris-Pratt), Rabin-Karp, ou Finite Automata) torna-se viável) (Investigar mais) (Verificar Turing Paradox)

Resultado: Uma versão limpa e padronizada do ficheiro pronta para análise de malware (Fase 2).

c. Fase 2: Análise das assinaturas digitais = Hashes com flag, com base em APIs

Objetivo: Após a limpeza, analisar o ficheiro utilizando APIs de antivírus para verificar assinaturas digitais de malware conhecidas (MD5 / SHA256)

Passos:

Enviar ficheiro para APIs selecionadas (ex: API do Malwarebytes) para análise.

Utilizar as respostas da API (JSON) para determinar se alguma assinatura de malware conhecida é detectada.

Ferramentas:

API do VirusTotal (ou similar, como Malwarebytes, caso sejam necessárias restrições de privacidade) ("The Public API is limited to 500 requests per day and a rate of 4 requests per minute" Viável) (Investigar API Privada do VirusTotal, confidencialidade do ficheiro) (Para Python/Golang) (API Pública já assegurada, prosseguir com testes)

Bibliotecas: requests para chamadas API, JSON para parsing dos resultados

Resultado: Gerar um relatório de risco para cada ficheiro, sinalizando malware conhecido, caso seja detectado.

d. Fase 3: Exploração de IA/algoritmos de ML

Objetivo: Para ficheiros que passaram pela verificação das APIs sem detecção, aplicar machine learning para analisar possíveis blocos de código de malware com base em anomalias de comportamento, tipo e estrutura do ficheiro (Exemplo: Código em Ficheiro

PDF abre porta 8080 para uma comunicação FTP/TCP, visto que não faz sentido, sinaliza como possível malware embutido) (Verifica ligação C2)

Passos:

Pré-processar os ficheiros (embutindo ou convertendo o conteúdo) para análise com ML.

Usar um modelo ML com datasets de detecção de malware ou ajustá-lo para padrões específicos de phishing/malware.

Ferramentas:

Modelo de Machine Learning: Modelo transferido/Cloud (ex: baseado nos datasets da EMBER ou similar) para identificar padrões.

Bibliotecas: scikit-learn, TensorFlow ou PyTorch para aplicação do modelo, pandas para gestão de datasets.

Dados:

Datasets: EMBER, MalNet, e outros sets públicos para treino de malware

Resultado: Resultado da classificação baseada em ML com pontuação de risco potencial para detecção de ameaças de malware / Verificação de compactação/obscuração de ficheiro

5. Ferramentas e Plataformas (Validar)

6. Etapas do projeto (Validar)

Para efeitos de organização de trabalho, estruturação do desenvolvimento e avaliação frequente dos resultados e do progresso geral do projeto, foram definidas 4 etapas e as respetivas datas-limite.

Estas etapas estão sintetizadas na seguinte tabela.

Tabela 1 - Etapas do projeto

Etapa		Descrição	Prazo limite

7. Anexos

8. Conclusão

9. Bibliografia

Não existem origens no documento atual.