ESCOLA
SUPERIOR
DE TECNOLOGIA
E GESTÃO
POLITÉCNICO
DO PORTO

M

MESTRADO
ENGENHARIA INFORMÁTICA

WE-SAND – A Sandbox for Web Security

Joaquim Cristiano Sampaio Carvalho

WE-SAND – A Sandbox for Web Security

Joaquim Cristiano Sampaio Carvalho
ESTG-IPP
8110253@estg.ipp.pt

Tese apresentada à Escola Superior de Tecnologia e Gestão
para obtenção do grau de Mestre em Engenharia Informática
realizada sob a orientação do
Prof. Doutor João Paulo Ferreira de Magalhães
Professor Adjunto na Escola Superior de Tecnologia e Gestão

2019

Agradecimentos

Manifesto o meu agradecimento a todas as pessoas que contribuíram de algum modo na elaboração deste trabalho.

Em especial aos professores João Paulo Magalhães e Dorabela Gamboa pelo apoio e orientação para a realização deste trabalho.

Resumo

Os ciberataques já se verificam há alguns anos, mas foi nestes últimos anos que ganharam maior visibilidade. Num mundo interligado pela via digital, os atores maliciosos encontraram novas formas de ganhar dinheiro e notoriedade prejudicando organizações de vários tipos. A exploração de vulnerabilidades em sistemas informáticos e aplicações, o envio de *e-mail* com conteúdos aparentemente legítimos/normais mas que de facto escondem *payload* maliciosos, a cifra de dados e pedidos de resgate para reaver os mesmos, são alguns exemplos de ameaças cibernéticas que para muitas organizações e pessoas já se traduziu em ciberataque de facto, e consequentemente em impactos de foro económico, de reputação entre outros. Lidar com o problema não é fácil. À medida que se vão criando tecnologias de deteção e proteção, refinando processos e treinando pessoas para o fenómeno, os atores maliciosos procuram novas ferramentas, tecnologias e procedimentos para contornar as barreiras levantadas. Uma das tecnologias na área são os sistemas de *sandbox*. Tratam-se de sistemas isolados que permitem a análise de artefactos com o intuito de averiguar se os mesmos são ou não maliciosos.

Neste trabalho analisámos vários sistemas de *sandbox* e apresentamos um protótipo funcional (cujo nome é **WE-SAND**) de uma aplicação que agrega vários sistemas de *sandbox* com o intuito de facilitar o processo de análise e de identificação de *payload* malicioso. Para a análise foi feito um levantamento dos sistemas de *sandbox* existentes e foram selecionados cinco destes sistemas. Sobre estes foram realizados testes utilizando artefactos propositadamente maliciosos e artefactos não maliciosos. A análise de eficácia revelou a existência de diferenças significativas entre os sistemas de *sandbox.* Apesar de alguns apresentarem boas taxas de deteção (acima dos 90%), existem artefactos maliciosos que não são detetados pelos sistemas de *sandbox* bem como existem artefactos não maliciosos a ser identificados como tal. Outra análise efetuada foi a medição do tempo requerido para a análise por cada sistema de *sandbox*. Com base nos resultados obtidos pela análise, procedeu-se ao desenho e implementação do **WE-SAND**. Trata-se de uma aplicação que agrega vários sistemas de *sandbox* (atualmente três, mas possível de ser estendido) que facilita o envio de artefactos para análise e recolha dos resultados da análise. O protótipo está funcional e a sua maturidade e adoção permitirá às organizações analisar o *payload* dos acessos com o exterior a fim de detetar e evitar a ocorrência de ciberataques realizadas com recurso a este tipo de técnica.

Palavras-chave: ciberataques, ciberameaças, *phishing*, *malware*, *sandbox*, análise artefactos.

Abstract

Cyber-attacks have been around for some years, but in recent years they have gained greater visibility. In a digital-interconnected world, malicious actors have found new ways to make money and notoriety. Exploiting vulnerabilities in computer systems and applications, sending emails with such as legitimate/normal content but that actually hide malicious payload, encrypting data on computers and servers and request for ransoms to retrieve them are some examples of current threats, which unfortunately are affecting many organizations and individuals leading to huge economic, political and reputational impacts. Dealing with the problem is not easy. New detection and protection technologies are being developed, the refining of processes and training people for the phenomenon is already occurring but the malicious actors are continuously looking for new tools, technologies and procedures to get around the barriers raised. One of the technologies that exist in the area and that have been evolving are the sandbox systems. These are isolated systems that allow the analysis of artifacts in order to ascertain whether they are malicious or not.

In this work we analyze several sandbox systems and present a functional prototype of an application (named WE-SAND) that aggregates several sandbox systems in order to facilitate the process of analysis and identification of malicious payload. A survey of existing sandbox systems was conducted and five of these systems were selected. Tests over the five sandbox systems were conducted, using both purposely malicious artifacts and non-malicious artifacts. The efficacy analysis showed that there are significant differences between sandbox systems and that although some have good detection rates (over 90%), there are malicious artifacts that are not detected by sandbox systems as well as there are non-malicious artifacts being identified as malicious. Another analysis was the measurement of the time required for the analysis by each sandbox system. Such analysis is important to known how timely an organization can detect and mitigate the occurrence of malicious artifacts. Based on the results obtained from the analysis, the WE-SAND prototype was designed and implemented. It is an application that aggregates several sandbox systems (currently three, but possible to be extended) and facilitates the sending of artifacts for analysis and the collection of results. The prototype is functional, and its maturity and adoption will allow organizations to analyze the payload of incoming data (e.g., emails, files, web pages) in order to detect and mitigate the occurrence of cyber-attacks conducted using these types of techniques.

Keywords: cyber-attacks, cyber-threats, phishing, malware, sandbox, artifact analysis.

Índice

Índice de Figuras

Índice de Tabelas

Índice de Gráficos

Índice de Listagens

Acrónimos

BAT	Batch
BIN	Binary
BIOS	Basic Input/Output System
BOT	Robot
CAB	Microsoft cabinet file
CD	Compact Disc
CPL	Description of Control Panel
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DDOS	Distributed Denial of Service
DNS	Domain Name System
DOS	Disk Operating System
EXE	Executable File
FTP	File Transfer Protocol
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure
IDS	Intrusion detection System
IOC	Indicator Of Compromise
IOE	Indicators Of Exposure
IP	Internet Protocol
ISP	Internet Service Provider
JS	JavaScript
MBR	Master Boot Record
P2P	Peer-to-Peer
PDF	Portable Document Format
PHP	Hypertext Preprocessor
PIF	Postcode Information File
PL	Perl Script
PNG	Portable Network Graphics
RAM	Random-Access Memory
ROM	Read-Only Memory
SSD	Solid-State Drive
TTP	Tools Techniques Procedures

TXT	Plain Text File
URL	Uniform Resource Locator
USB	Universal Serial Bus
VPN	Virtual Private Network
WWW	World Wide Web
XLS	Excel Spreadsheet
XLSM	Excel Open XML Macro-Enabled Spreadsheet
ZIP	Zipped File

Capítulo 1 – Introdução

Neste Capítulo apresentamos a motivação subjacente ao trabalho realizado. O tema do trabalho é enquadrado na motivação e são apresentados os objetivos principais do projeto. A estrutura da dissertação é igualmente apresentada.

1.1.Motivação e objetivos a atingir

Num mundo em que as tecnologias estão cada vez mais presentes, as empresas sentem a necessidade de terem uma presença na *World Wide Web* para fortalecer o contacto com o exterior. Uma página Web é o melhor cartão-de-visita e por vezes uma vantajosa ferramenta de trabalho. Dependendo do negócio das empresas, a presença *online* permite também aumentar a exposição de produtos, reduzir os custos em publicidade, expandir atividades de comércio por 24 horas por dia sem que haja sobrecarga excessiva nos custos operacionais e facilitar comunicações rápidas entre empresas, clientes e fornecedores. No entanto, a par das vantagens, a *World Wide Web* abriu portas a um mercado paralelo, muitas vezes definido por *underground* ou mercado negro. Trata-se de um mercado conotado com atividades ilícitas onde operam vários negócios. Venda de estupefacientes, armas, órgãos humanos, documentos sensíveis, dados pessoais, dados empresariais, informação bancária, são exemplos de atividades realizadas nestes ditos mercados. É também do conhecimento geral que a obtenção dos dados que circula nesse mercado está relacionada com a ocorrência de ciberataques, dividindo-se os ciberataques por vários tipos. Assim, e entre os tipos de ciberataques mais frequentes temos: cibercrime (roubo de dinheiro), *phishing*, *hacktivismo* e ciberespionagem [1]. Quer as ações, quer os resultados são difundidos através dos ditos mercados negros.

O número de ciberataques e a sua complexidade tem vindo a crescer de ano para ano. São vários os registos que se encontram publicamente relacionados com ciberataques e os seus impactos. De acordo com o estudo publicado em [2], 95% de ataques de *phishing* podem estar associados a espionagem, tendo estes como objetivos principais atacar organizações financeiras de bens e serviços. Ainda, e de acordo com a Google, são criados diariamente entre 9500 a 10000 Websites com conteúdos maliciosos como é indicado em [3] [4]. A título de exemplo, os autores em [5] descrevem um ataque à gigante Apple que consistiu na instalação de um *software* malicioso que controla e rouba dados dos dispositivos. No artigo publicado em [6] é referida a WhatsApp, tendo esta sido atingida por um ciberataque que se estima ter comprometido os dados pessoais de 200 mil utilizadores. O artigo publicado em [7] revela que os *hackers* (por vezes atores maliciosos) atacaram os servidores da Experian e roubaram os nomes e os números de segurança social de mais de 15 milhões de pessoas. No contexto nacional em [8], é referido que em Portugal entre 2014 e 2015 23% das empresas analisadas por uma corretora de seguros foram alvo de ataques informáticos. A título de exemplo, em 2013 surgiu o *malware* Anunak, que no ano seguinte evoluiu para o *malware* designado por Carbanak. Os autores deste *malware* atacaram mais

de 100 instituições bancárias. O ataque atingiu bancos em mais de 40 países e resultou em perdas de mais de mil milhões de euros. O ataque era iniciado com o acesso ao computador de um dos colaboradores através de um *e-mail* do tipo *spear phishing* (envio de *e-mail* dirigido ao setor bancário) contendo um ficheiro anexo malicioso. A simples abertura do *e-mail* e anexo levava à instalação do *malware*. Após instalação, o *software* malicioso fornecia aos atores maliciosos *printscreens* dos ambientes de trabalho dos colaboradores, capturava as teclas pressionadas e abria um porto para acesso remoto. Com estes recursos foi possível conhecer o funcionamento dos sistemas em cada banco e ter acesso aos sistemas de transferência de dinheiro. Na Figura 1, é ilustrado o fluxo de ataque tendo este passado por três etapas distintas: envio de *malware* via *e-mail* (1), entrada nos sistemas informáticos (2), recolha e transferência de dinheiro (3).

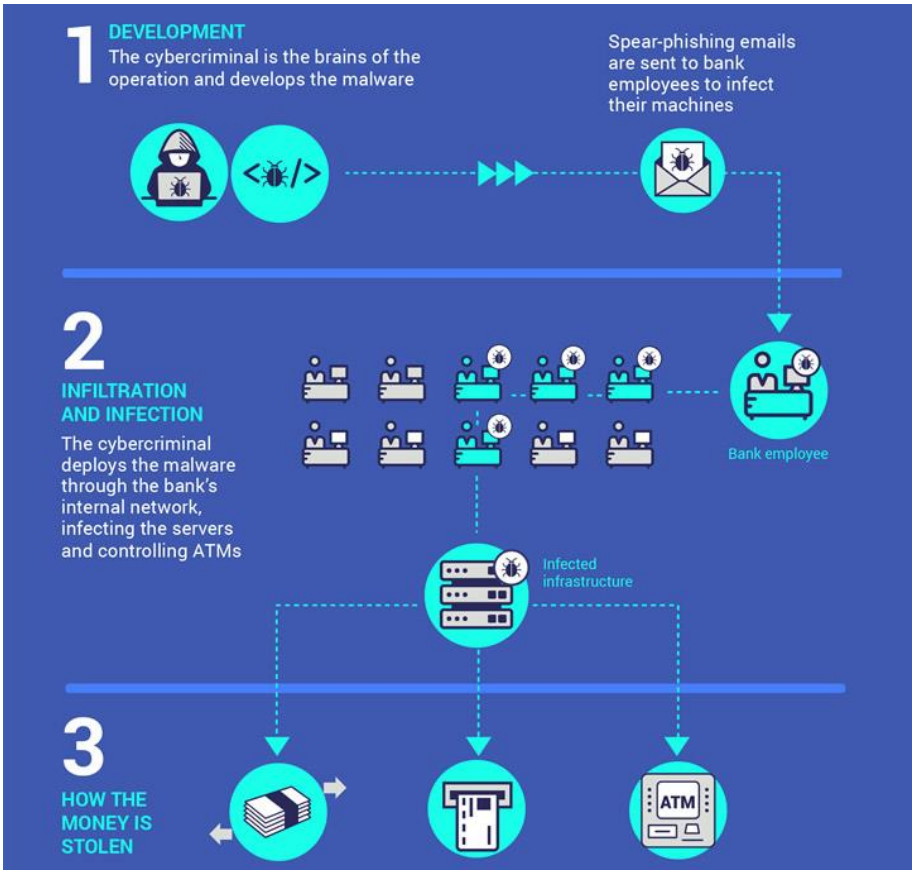


Figura 1 – Ataque malware Carbanak [9]

De acordo com o indicado em [9], o dinheiro foi obtido através dos seguintes meios:

- As caixas multibanco foram programadas para fornecer o dinheiro, numa hora específica. Os criminosos apenas tiveram que esperar junto das caixas multibanco e recolher o dinheiro, que as caixas iam disponibilizando;
- A conta dos clientes foi inflacionada num determinado montante e esse montante foi depois transferido para a conta dos atores maliciosos.

Com base nos dados publicados em [10], no primeiro trimestre de 2016 houve um aumento significativo no número de *e-mails* não solicitados que contêm anexos maliciosos. O gráfico apresentado na Figura 2, ilustra a evolução desde 2015, até ao primeiro trimestre de 2016. Tipicamente no corpo ou anexos dos *e-mails* existem *links* para/ou ficheiros maliciosos que ao serem acionados, podem descarregar ficheiros maliciosos ou redirecionar o utilizador para Websites com conteúdo malicioso.

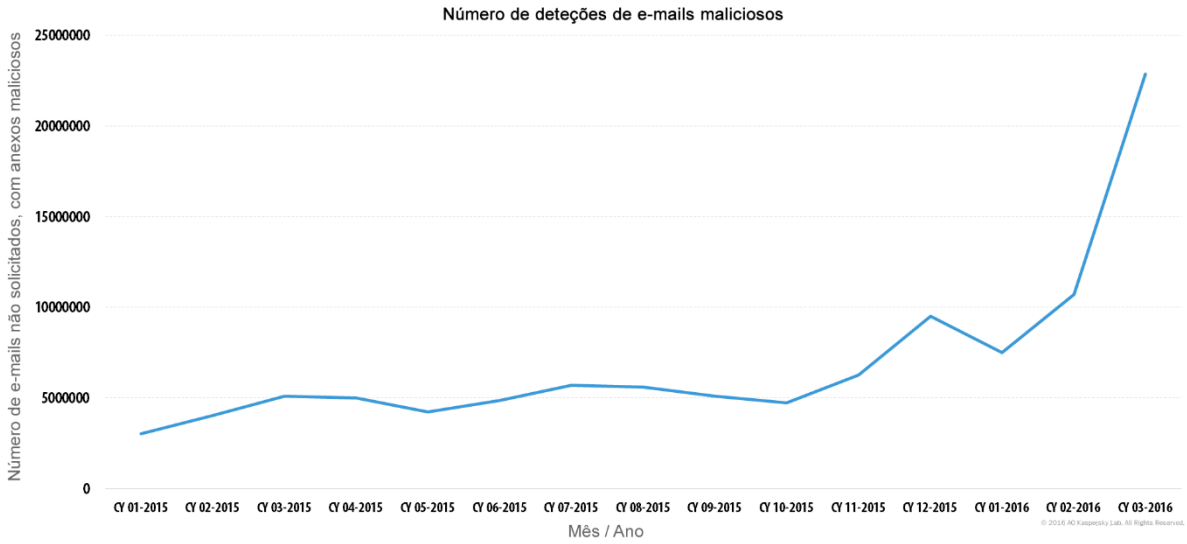


Figura 2 – Número de e-mails com conteúdo malicioso (deteção Kaspersky) [10]

A Kaspersky indica em [11] que a quantidade de *spam* em 2017 diminuiu em média 1.68% face a 2016. Ao mesmo tempo, o número de ataques de *phishing* aumentou. Um dos casos referenciados prende-se com os serviços de mineração de criptomoeda, nos quais os atacantes têm Websites disfarçados de trocas de criptomoeda ou serviços de mineração, mas que na maioria dos casos só leva a que as vítimas percam dinheiro. Os estudos referem que a maioria destes serviços é distribuída via *e-mail*. O *e-mail* é também o meio mais utilizado para ataques de *phishing* e de distribuição de *malware*. A inclusão de *links* no *e-mail* que desencadeiam o *download* de *software* malicioso (*drive-by-download*) é também frequente. Neste âmbito torna-se cada vez mais importante para as organizações analisar os ficheiros recebidos e as páginas Web acedidas pelos seus colaboradores. O conteúdo descarregado/acedido caso contenha conteúdo malicioso, pode comprometer a organização sob o ponto de vista económico e de reputação. Atualmente existem ferramentas que facilitam a análise do conteúdo do *e-mail* e anexos. Os autores em [12] apresentam algumas ferramentas que alertam quando um Website acedido é malicioso. No entanto estas ferramentas são orientadas para o utilizador e não para o administrador de rede.

A par do crescimento em número também se verifica uma crescente complexidade nos ciberataques. Se no passado os conhecimentos dos atores maliciosos teriam que ser sofisticados, hoje em dia com a criação de *kits* e com o acesso mais fácil à informação, torna-se mais simples para os

atacantes criar e disseminar um ataque, ou mesmo recrutar novos membros para a sua organização. Esta organização pode ser constituída por uma comunidade *online* descentralizada, que coopera com um fim comum, que pode ser social, político ou, financeiro, possibilitando assim a partilha de conhecimento e execução de ataques de forma organizada, rápida e eficaz.

Neste trabalho propomos uma solução de *sandbox* focada em análise de artefactos Web **(WE-SAND)** para reduzir o acesso a conteúdo ilegítimo/perigoso. O **WE-SAND** é uma *appliance* projetada para funcionar ao nível da rede *core* com dois modos de funcionamento: *inline* e *out-of-band*. No modo *inline* ele irá intercetar os pedidos Web e analisar o conteúdo do Website destino por forma a apurar se o mesmo apresenta risco de acesso ou não. No modo *out-of-band* irá replicar o acesso à Web feito por uma organização (a partir dos *logs*) e analisar o conteúdo desses mesmos acessos para a deteção de conteúdos potencialmente perigosos transmitidos de e para o utilizador/organização. Considerando o desafio e complexidade em colocar a solução no modo *inline*, ao longo deste projeto o foco será o desenvolvimento e teste da solução *out-of-band*. Desta forma o **WE-SAND** tem como objetivo principal detetar o acesso a conteúdo malicioso transmitido via Web e alertar os administradores de rede para o sucedido. A implementação do **WE-SAND** é importante na medida em que permitirá:

- Identificar acesso a Websites maliciosos (por análise de *URL* e de conteúdo);
- Reduzir o risco de ataques às organizações, com recurso à monitorização contínua do conteúdo acedido pelos colaboradores;
- Sensibilizar para a formação dos colaboradores, sobre os perigos que correm ao acederem a determinado Website;
- Facultar a análise de artefactos sem necessidade de elevado *know-how* ou tempo para a análise. Funcionando como uma interface, o **WE-SAND** encarregar-se-á de submeter os artefactos aos sistemas de *sandbox* considerando o tipo de artefacto sob análise.

Sendo o *drive-by-download* uma das formas mais populares de difusão de *malware*, e o Web *phishing* de roubo de informação, a criação de uma ferramenta como o **WE-SAND** irá auxiliar os administradores de rede a ter uma visão global e atempada sobre os acessos a páginas Web que representam potencial perigo. Desta forma o administrador terá um melhor conhecimento sobre o acesso a Websites potencialmente perigosos, podendo adotar mais rapidamente medidas de resposta aos incidentes.

1.2. Estrutura da dissertação

Este documento está organizado em capítulos. No segundo capítulo são apresentados os conceitos referentes ao mundo dos ciberataques, descrevendo de forma breve as ameaças, motivação para as ameaças, tipos de ameaças e a anatomia de um ataque. Para além disso, este capítulo apresenta um enquadramento histórico do *malware* e introduz o conceito de sistemas de *sandbox*. Nos sistemas de *sandbox*, podemos ter vários tipos de análise de código malicioso (análise estática / dinâmica). No

Capítulo 3 são apresentadas as especificações do **WE-SAND** e toda a estrutura envolvida no seu desenvolvimento. No quarto capítulo são apresentados os resultados obtidos durante a submissão de artefactos. Estes testes deram origem a um conjunto de resultados que permitiram medir a *performance* de cada sistema de *sandbox*. No Capítulo 5 é apresentada a implementação do sistema, arquitetura, base de dados e apresentação de resultados. Finalmente, no Capítulo 6 são apresentadas as conclusões finais e trabalho futuro a realizar.

Capítulo 2 – Revisão da Literatura

O segundo Capítulo tem o objetivo de clarificar e definir os conceitos envolvidos no trabalho apresentado. O mesmo enquadra "o mundo" dos ciberataques e apresenta uma perspetiva histórica dos mesmos. No decorrer do capítulo é apresentado o conceito de *sandbox*, conceito este que é a base de todo o trabalho. O objetivo é apresentar de forma pertinente o problema em estudo e fundamentar a abordagem do trabalho.

Surgem a cada dia novas ameaças, cada vez mais sofisticadas. Os cibercriminosos estão constantemente a desenvolver técnicas inovadoras para contornar as tecnologias e os métodos de segurança existentes. Estas alterações constantes são apresentadas sobre o acrónimo *TTP (Tools, Techniques e Procedures)*, isto é, as alterações constantes que os atores maliciosos executam para evitar/contornar os sistemas de segurança. A este respeito, e como se indica em [13], as soluções de segurança tradicionais como *firewall*, antivírus e sistemas de prevenção de invasão já não são suficientes para garantir uma proteção adequada. Organizações como por exemplo seguradoras, empresas de retalho, entidades bancárias, apesar de disporem de boas soluções tradicionais de segurança, continuam a ser alvo de ataques por parte de atores maliciosos.

Segundo o artigo [14] o setor bancário é um dos mais visados pelos atores maliciosos, cujo objetivo é obter dados e dinheiro. Os dados são considerados uma fonte rentável, pois existem outras organizações a comprar esses mesmos dados com o intuito de oferecer empréstimos e garantias de crédito às pessoas. No mesmo artigo é referido o ataque ao banco central de Bangladesh, que culminou no roubo de aproximadamente 71 milhões de euros. Este ataque deveu-se às falhas nos equipamentos de rede e ausência de *firewall*, além disso, uma investigação conduzida pela BAE Systems indica que os atacantes usaram um *malware* personalizado para "enganar" o sistema bancário. Outros exemplos de ataques são apresentados no artigo [15]. Entre os vários ataques destaca-se o ataque ao banco central Grego e ao banco central de Chipre.

Outro caso emblemático tem a ver com os ataques a infraestruturas críticas (e.g. energia, transportes, abastecimento de água, silos, hospitais). A organização Chatham House referenciada no artigo [16], divulgou um relatório onde afirma que a indústria nuclear é um alvo fácil, uma vez que a mesma está atrasada na prevenção de ameaças. Foram identificadas no estudo a existência de instalações nucleares cujos sistemas informáticos não se encontram isolados da rede de comunicações pública, ou seja, parte das instalações nucleares fazem uso de ligações através de *VPN* sobre a Internet. Mesmo quando isoladas da rede pública, os mecanismos de segurança podem ser contornados com uma simples *pen drive*. Também foi evidenciada uma falta de treino e falhas de comunicação entre os engenheiros que trabalham nos reatores e a equipa de segurança. Esta falha de comunicação e treino identificou um grave problema, ou seja, um ataque só é descoberto após ocorrer e não existem mecanismos preventivos, tornando desse modo uma central nuclear vulnerável a ataques externos e internos. Um exemplo de um

ataque a centrais nucleares é conhecido por *Stuxnet*. No relatório apresentado em [17] a Microsoft afirma que a primeira versão do *Stuxnet* foi criado em janeiro de 2009. De acordo com o relatório, os Estados Unidos da Améria, direcionaram um ataque contra o Irão, que conseguiu deixar inutilizado parte do programa de enriquecimento de urânio. Com o *Stuxnet*, foi possível atacar de forma remota e faseada as várias estruturas, deixando inutilizado grande parte do plano nuclear. O ataque envolveu a utilização de *pen drives* que manipularam os sistemas de controlo das centrifugadoras, reportando valores normais de utilização das mesmas, quando na prática estavam a ser manipuladas por forma a reduzir a capacidade de enriquecimento de urânio. Mais tarde soube-se que tentaram, um ataque similar contra a Coreia do Norte. O objetivo seria com o passar do tempo atacar outros estados e controlar os vários sistemas, uma vez que o objetivo não era o roubo de informação, mas sim o controlo de equipamentos industriais. O caso *Stuxnet* é conhecido como o primeiro ataque do tipo ciberguerra.

No gráfico ilustrado na Figura 3, são apresentados os tipos de ciberataques mais comuns: cibercrime, ciberespionagem, ativismo e ciberterrorismo. Estes dados foram recolhidos durante o mês de Dezembro de 2017 e fazem referência aos incidentes conhecidos apresentados em [18], [19] e [20].

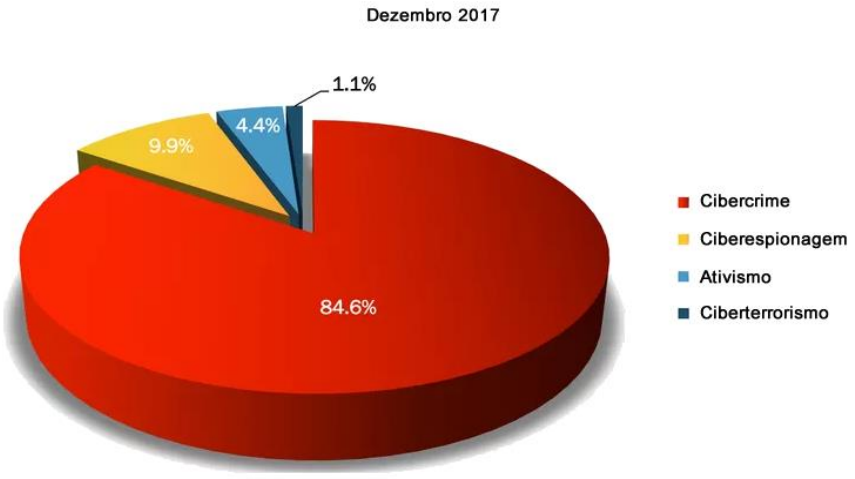


Figura 3 – Principais motivações que levam a ataques informáticos [20]

De entre as ameaças mais comuns, a que predomina é o cibercrime (84,6%). O estudo apresentado em [21], indica que no âmbito do cibercrime, destaca-se o furto de identidade. O furto de identidade tem como objetivo a obtenção não consentida dos dados pessoais e sua utilização para a prática de crimes. Este tipo de crime pode ser praticado pelos métodos tradicionais (e.g. furto de documentos, roubo de correio, observar a vítima enquanto utiliza a informação pessoal) ou recursos tecnológicos (e.g. anúncios de emprego, redes sociais, *phishing*), tendo a finalidade de utilizar os dados da vítima em proveito próprio.

2.1. Tipos de Ameaças

Ao longo desta seção são descritos vários tipos de ameaças. Estas ameaças podem derivar de pessoas internas ou externas à organização e são exploradas através de vulnerabilidades que possam existir ao nível dos sistemas, processos ou pessoas. Neste âmbito enquadra-se a denominada engenharia social, a exploração de vulnerabilidades em aplicações informáticas e a instalação de *malware* tirando partido de diversos meios. A engenharia social é amplamente utilizada e visa os colaboradores da organização, onde alguém faz uso da ingenuidade ou confiança do utilizador, para obter informações que podem ser utilizadas para ter acesso não autorizado a computadores e informações associadas à organização. O *malware* é um termo genérico que cobre várias ameaças de segurança. O *malware* assume várias formas: **vírus**, ***worms***, **cavalos de troia (*trojan*), *backdoors*** e ***rootkits***. Também na categoria de *malware* inclui-se o ***spyware***, ***botnets***, ***keystroke loggers (keylogger)*** [22].

- ***Vírus*** – o vírus de computador é constituído por partes de *software* que pode replicar-se e facilmente contaminar um ou vários computadores. Este pode ser disseminado pela rede/Internet, ou por dispositivos externos (*CD*, dispositivos *USB*).

- ***Worms e Mass-Mailers*** – *worms* são programas autorreplicantes, ou seja, criam cópias funcionais de si mesmos e disfarçadamente infetam outros computadores, através da Internet. Outro tipo de *worm* é o *mass-mailers*, esta divisão deve-se pelo facto dos *mass-mailers* serem projetados com o objetivo de a sua difusão realizar-se via *e-mail* e quando executado pelo recetor o mesmo propaga-se automaticamente, ou seja, reencaminha um *e-mail* com um ficheiro malicioso em anexo para todos os contactos da lista do recetor.

- ***Cavalos de troia (Trojan Horses)*** – os cavalos de troia incluem *software* malicioso embebido em *software* não malicioso. Desta forma o utilizador pensa que esta a executar um *software* confiável, no entanto dentro deste existe o código malicioso. Na realidade o *software* não realiza só a tarefa a que se destina, na maioria das vezes descarrega e instala um vírus de computador. Outras vezes os cavalos de troia são inofensivos, pois os mesmos carecem de autorreplicação, eles dependem dos utilizadores para causar danos no computador. Estes são geralmente espalhados através da técnica de engenharia social. Embora não haja nenhuma classificação, os cavalos de troia podem ser divididos de acordo com o seu comportamento:
    - ***Cavalo de troia de acesso remoto*** – permite o controlo remoto por parte de terceiros. Após ligados remotamente os mesmos podem, por exemplo, instalar outros programas e realizar ligações para transferência de ficheiros.

- ***Cavalo de troia de destruição de dados*** – têm a capacidade de apagar por completo ou corromper os dados armazenados num computador, quer sejam ficheiros do sistema ou do utilizador.
- ***Downloader Trojans*** – é um tipo de aplicação que de forma isolada é inofensiva, ou seja, a mesma necessita que o computador tenha acesso à Internet para instalar outros aplicativos como *adware*, *spyware*, *Remote Access Trojans* e desse modo permitir o ataque.
- ***Security Software Disablers*** – é um tipo de *trojan* que quando instalado no computador da vítima tenta desativar o *software* de segurança instalado, tal como antivírus ou *firewalls*, sem consentimento dos utilizadores.
- ***Denial-of-Service (DoS) Trojans*** – consiste num conjunto de vários pedidos realizados a um servidor. O objetivo é deixar o servidor sem recursos (*CPU*, memória *RAM*) e levar o mesmo ao seu colapso em termos de recursos computacionais.
- ***Dialers*** – este tipo de *malware* utiliza linhas *dial-up* (linha modem/telefone) para realizar chamadas de valor acrescido.
- ***Keyloggers*** – é *trojan* malicioso, que quando instalado, captura o texto digitado e envia essas informações para o atacante. Existem *keyloggers* que são *opensource*/*freeware* classificados como legítimos, no entanto parte deles são utilizados principalmente para fins maliciosos, tais como roubo de credenciais de acesso a contas bancárias, contas de *e-mail*, contas em redes sociais e contas em lojas *online*.

- ***Backdoors*** – são falhas de segurança que permitem aos atacantes acederem remotamente ao computador. Este tipo de aplicação pode por exemplo ser incluído num cavalo de troia ou na substituição de um determinado serviço de assistência remota por uma versão alterada. Estes programas podem ser facilmente explorados por terceiros não autorizados. Para reduzir o risco, estes programas devem ser instalados sempre do Website do prestador do serviço, na tentativa de minimizar a instalação de um programa adulterado. Ao contrário do cavalo de troia, os *backdoors* abrem um canal de comunicação sem que o utilizador se aperceba.

- ***Exploits*** – são vulnerabilidades existentes no *software*, que permitem aos atacantes executar ações maliciosas no computador de destino. O objetivo, neste caso, é causar um comportamento acidental ou imprevisto na execução do *software* ou *hardware*, tanto em computadores como em aparelhos eletrónicos acessíveis através da rede. Ao contrário dos outros meios de disseminação, um *exploit* não necessita que o utilizador faça *download* de

um ficheiro e o instale. Neste tipo de vulnerabilidade o ator malicioso apenas depende de si para realizar o ataque, sendo deste modo uma arma poderosa para o mesmo.

- ***Rootkits*** – é um conjunto de ferramentas utilizadas para esconder processos, ficheiros ou dados do sistema operativo, atacar e infetar outros computadores, roubar informações e interagir em tempo real com o sistema comprometido. Com o *rootkit* o atacante pode controlar o computador na sua totalidade e utilizar o mesmo para a realização de outros ataques. Os autores em [22] [23] apresentam exemplos de ataques baseados em *rootkits.* Os *rootkits* podem ser divididos em diferentes categorias:
  - ***Rootkits nível do utilizador*** – são os mais básicos e são subdivididos em dois tipos:
    - ***Rootkits binários*** – são compostos por programas binários maliciosos, que substituem os programas originais do sistema. São utilizados para aceder remotamente e esconder evidências de acessos.
    - ***Rootkits do tipo biblioteca*** – são substituídas as principais bibliotecas, do sistema por versões maliciosas.
  - ***Rootkits Kernel*** – são aplicados nos módulos do núcleo do sistema operativo, que permitem ocultar e modificar informação. Este permite carregar dinamicamente módulos maliciosos e alterar a informação diretamente na memória do sistema operativo.
  - ***Rootkits de Master Boot Record*** – reescrevem o *MBR* com código malicioso, sendo o original guardado e o *rookit* instalado no primeiro setor do disco. Este é carregado no *MBR*, ou seja, fora do sistema operativo o que torna invisível para o sistema operativo e para o *software* de segurança instalado (e.g. *firewall*, antivírus).
  - ***Rootkits em máquina virtual*** – a virtualização permite simular as caraterísticas físicas de uma arquitetura informática. Estes *rootkits* são capazes de se alojar na máquina virtual e tentarem de alguma forma atingir o sistema que suporta todo o ambiente de virtualização.
  - ***Rootkit de firmware*** – é um *firmware* alterado, que quando instalado na *BIOS* resiste à formatação do sistema.

- ***Botnet*** – os atacantes distribuem um *software* malicioso, que transforma um determinado computador num *bot*. Ao conjunto de *bots* dá-se o nome de *botnet*. O *bot* executa tarefas sem o consentimento do utilizador. Esta estratégia é utilizada para infetar uma grande quantidade de computadores colocando-os ao serviço de um *command and control server*.
- ***Spyware*** – *software* que permite monitorizar a interação dos utilizadores com o computador afetado e transmitir essa informação para uma entidade externa na Internet, sem o conhecimento e autorização do utilizador.

- ***Adware*** – as aplicações de *adware* são associadas a um risco médio. No entanto este método incomoda os utilizadores dos Websites, quando lança inúmeras janelas de anúncios indesejados (*pop-up*). Normalmente estes anúncios possuem *URLs* para *software* de terceiros. Este *software* por vezes é malicioso e quando descarregado e instalado pode provocar danos no computador do utilizador.
- ***Phishing*** – *phishing* é um tipo de ação fraudulenta que recorre ao uso de mensagens de *e-mail* que aparentam ter origem fidedigna, como por exemplo um banco, mas efetivamente provêm de pessoas mal intencionadas com o objetivo de obter informação ou redirecionar os utilizadores para páginas fraudulentas [24] [22].

A Tabela I apresenta um resumo, entre os *softwares* maliciosos e as suas ações. Esta tabela contem informações sobre como o código malicioso é obtido, instalado, propagado e outras ações que realiza após alojado no computador hospedeiro. A informação apresentada na Tabela I foi obtida a partir do relatório publicado em [22].

Tabela I – Software malicioso por tipo e suas ações – quadro resumo [22]

	Vírus	***Worm***	***Bot***	***Trojan***	***Spyware***	***Backdoor***	***Rootkit***
Como é obtido:							
Recebido automaticamente pela rede		✓	✓				
Recebido por *e-mail*	✓	✓	✓	✓	✓		
Descarregado de Websites na Internet	✓	✓	✓	✓	✓		
Partilha de ficheiros	✓	✓	✓	✓	✓		
Uso de unidades removíveis infetadas	✓	✓	✓	✓	✓		
Redes sociais	✓	✓	✓	✓	✓		
Mensagens instantâneas	✓	✓	✓	✓	✓		
Inserido por um invasor		✓	✓	✓	✓	✓	✓
Ação de outro código malicioso		✓	✓	✓	✓	✓	✓
Como ocorre a instalação:							
Execução de um ficheiro infetado	✓						
Execução explícita do código malicioso		✓	✓	✓	✓		
Via execução de outro código malicioso						✓	✓
Exploração de vulnerabilidades		✓	✓			✓	✓
Como se propaga:							
Insere uma cópia de si próprio nos ficheiros do sistema	✓						
Envia uma cópia de si próprio automaticamente pela rede		✓	✓				

Envia uma cópia de si próprio automaticamente por *e-mail*		✓	✓				
Não se propaga				✓	✓	✓	✓
Ações maliciosas mais comuns:							
Altera e/ou remove ficheiros	✓			✓			✓
Consome grande quantidade de recursos		✓	✓				
Rouba informações sensíveis			✓	✓	✓		
Instala outros códigos maliciosos		✓	✓	✓			✓
Possibilita o retorno do invasor						✓	✓
Envio *spam* e *phishing*			✓				
Aplica ataques na Internet		✓	✓				
Procura manter-se escondido	✓				✓	✓	✓

Com base no relatório publicado em [25] são apresentadas, na Figura 4, as estatísticas relativas ao grupo de *malware* que mais impacto teve no ano de 2015.

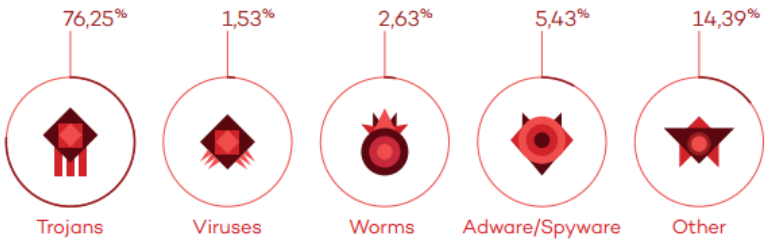


Figura 4 – Percentagem dos principais tipos de malware referente a 2015 [25]

De forma comparativa, na Figura 5, é apresentado um gráfico relativo a abril 2017 onde se verifica que o número de *trojans* continua a ser o mais representativo.

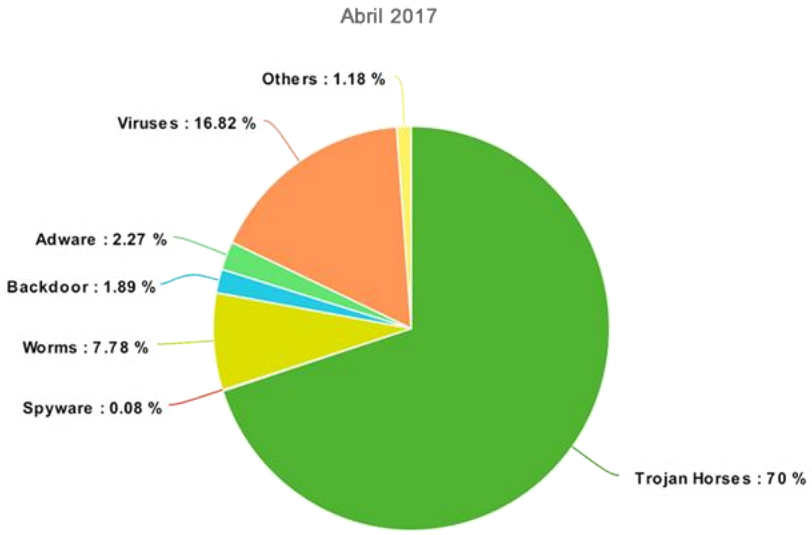


Figura 5 – Percentagem dos principais tipos de malware referente a abril de 2017 [26]

Com base nos números apresentados no artigo [27], na Figura 6 é apresentado um gráfico que representa a evolução de *malware*. Pela mesma pode-se verificar que o crescimento de *malware* é exponencial ao longo dos últimos 12 anos.

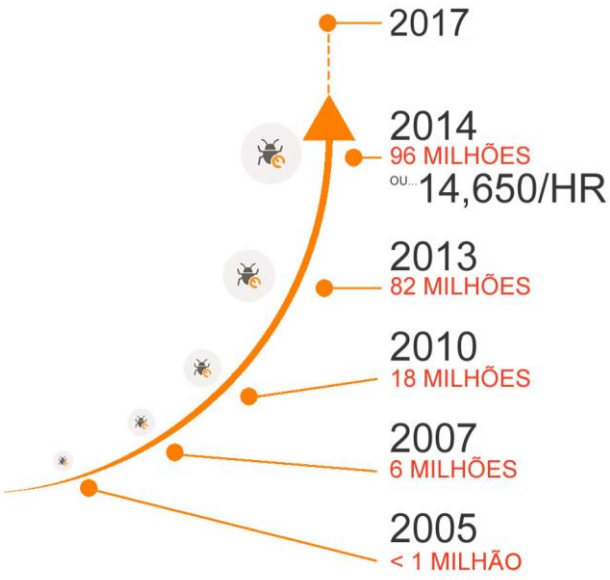


Figura 6 – Evolução do malware nos últimos 12 anos [27]

Na Figura 7 e de acordo com a publicação realizada pelos analistas da empresa de *software* de segurança G Data em [28], é ilustrado o número de novas variantes de *malware*, atingindo mais de 7,4 milhões em 2017.

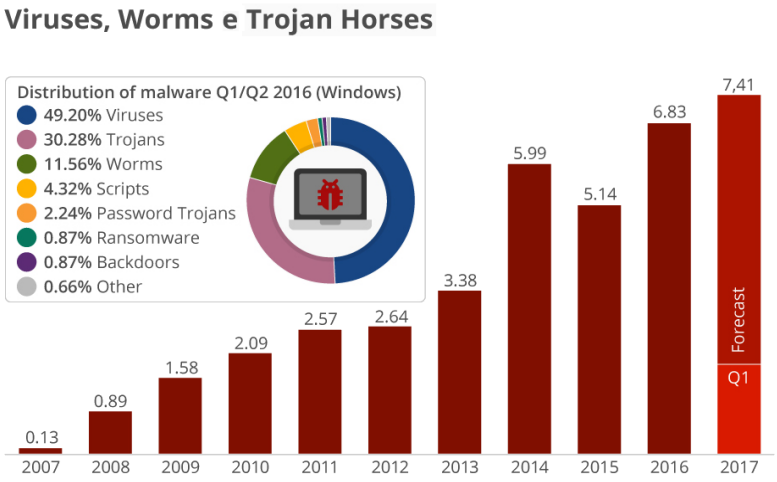


Figura 7 – Especificação de malware detetado (em milhões) [29]

Com tanta variedade e volume de *malware* e técnicas avançadas de engenharia social, evitar a ocorrência de um ataque não é tarefa fácil. Resta às organizações a adoção de medidas preventivas e boas práticas englobando pessoas, processos e tecnologias. A prossecução das boas práticas inumeradas nas normas ISO (e.g. 27001:2006 [30] e 17799:2005 [31]) são fundamentais para reduzir riscos. A adoção de soluções de monitorização continua e avaliação das ciberameaças deverá igualmente ser

considerada para melhor lidar com o fenómeno. O treino das pessoas para reconhecer e evitar ciberataques é também um caminho a percorrer.

2.2.Anatomia de um ataque

Para lidar com ciberataques é importante entender como um ataque funciona e distinguir os diferentes tipos de ataques que podem ocorrer. A discussão em torno do uso de ciberinteligência ou informações para detetar e prevenir ataques é especificamente utilizada para ataques planeados e direcionados. Com o decorrer do tempo, cada vez mais os ataques são direcionados tanto a empresas específicas como a entidades públicas. Para que as equipas de segurança consigam lidar com as ciberameaças e ciberataques devem entender o processo subjacente ao ataque e identificar os pontos frágeis onde o ataque pode ser realizado. Lockheed Martin desenvolveu o modelo de cadeia de ataque apresentado na Figura 8, que se divide em sete partes. O primeiro passo, em qualquer tipo de ataque direcionado é o reconhecimento da organização e infraestrutura da mesma. Há cinco anos atrás, o reconhecimento da rede alvo envolvia a análise dos endereços *IP* públicos e a partir deles eram procurados sistemas vulneráveis. Hoje em dia, esta estratégia não é tão utilizada. Com o aparecimento das redes sociais, os atacantes tiram partido das mesmas para as ações de reconhecimento das organizações. Na verdade, a maioria das operações de reconhecimento pode ser conduzida sem intrusão física na rede da organização alvo, todas as informações pertinentes ao reconhecimento estão disponíveis através de motores de pesquisa, redes sociais, Websites de análise de risco e classificação de organizações.

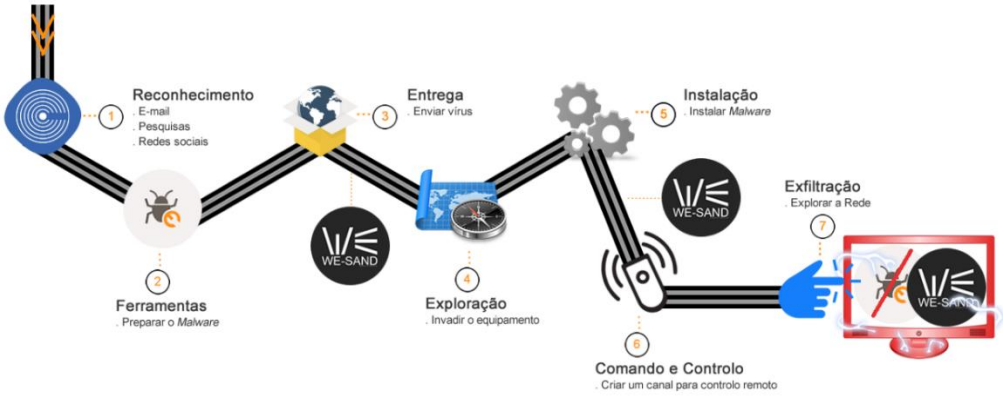


Figura 8 – Modelo de ataque

Uma vez terminada a fase de reconhecimento da operação, a próxima fase é "ferramentas" (*weaponization*). Nesta fase, o atacante utiliza uma ferramenta de acesso remoto, com o objetivo de explorar a rede da organização. Na maioria dos casos, esta ferramenta é distribuída via *e-mail* e a exploração ocorrerá tipicamente através de um Website ou documento comprometido. O Website pode ser criado especificamente para esse fim, ou é utilizado um Website legítimo que está comprometido e sob alçada do atacante. O terceiro passo no ataque é a entrega da ferramenta através dos métodos

apresentados anteriormente. Este processo poderá levar entre alguns dias até várias semanas. Durante este tempo os atacantes continuam a refinar e a melhorar o método de entrega, para poderem aumentar a taxa de sucesso da operação. Este processo pode ser equiparado a um processo de *marketing*, pois os atacantes analisam as mensagens que foram lidas e as que não foram e tentam ajustar as próximas mensagens para incentivar as vítimas a abrirem e descarregarem os ficheiros em anexo. O próximo passo é denominado por exploração. Após entrega da ferramenta, caso a mesma seja executada pela vítima é iniciado o processo de exploração por parte do atacante. Este pode explorar algumas ferramentas do sistema ou o sistema na sua totalidade. A maioria dos atacantes utilizam ficheiros maliciosos que ficam alojados na memória *RAM*. A instalação é a quinta etapa da cadeia de um ataque. A maioria dos programas de acesso remoto utilizados, apenas comunica para fora da rede e desse modo permite aos invasores instalar *key loggers*, gravadores de ambiente de trabalho, *scanner* de rede e através deles obterem informações existentes nas organizações. O próximo passo na sequência de um ataque é o comando e controlo. A aplicação de acesso remoto tenta ganhar o acesso privilegiado à máquina. Os atacantes normalmente utilizam a porta 80 (*HTTP*), a 443 (*HTTPS*) e a 53 (*DNS*), para realizarem estas comunicações pois estas portas são difíceis de bloquear uma vez que são necessárias para o funcionamento dos serviços Web, *e-mail*, entre outros. Uma vez obtido o acesso, o atacante examina o sistema, bem como a rede interna e desse modo tenta procurar pontos vulneráveis para espalhar o ataque. O ponto de entrada inicial pode não ser de particular interesse para o atacante, este apenas vai usá-lo para fazer escalada de privilégios e a partir daí comprometer outras máquinas e remover os sinais da sua entrada. Desta forma, mesmo que o ataque seja descoberto posteriormente, o ponto de entrada original e os meios de acesso estão escondidos da equipa de segurança e pode ser usado novamente com êxito, caso o atacante necessite. Este processo faz parte do último passo da sequência, ou seja, exfiltração. Uma vez realizada a descoberta da rede, é fácil para o atacante ter acesso aos dados e através deles atingir o objetivo final.

Um estudo da Ponemon no artigo [32], revelou que 63% das organizações já sofreu um ataque informático. Em média são necessários de 146 a 170 dias para detetar um ataque, 39 dias para colmatá-lo e 43 dias para reparar todos os estragos, como indica no artigo [33]. É necessário entender o ciclo de vida de um ataque para conseguir intervir, momentos antes do mesmo se tornar catastrófico e destruidor. No que diz respeito ao WE-SAND o mesmo pretende intervir nos pontos identificados na Figura 8 (após entrega e instalação de conteúdo malicioso). Deste modo possibilita que as equipas de segurança intervenham antes da intrusão propriamente dita e da exfiltração de dados.

2.3. Limitações dos sistemas tradicionais para lidar com os ataques

Nos últimos anos têm sido desenvolvidas e implementadas diversas técnicas, e as respetivas molduras legais, que permitem controlar o tráfego na Internet, nomeadamente bloquear o acesso a conteúdos ilegais. A solicitação aos prestadores de serviços (*ISP*) para remoção ou bloqueio de

determinados conteúdos é normalmente eficaz. No entanto os utilizadores mal-intencionados trocam de fornecedor. Considera-se que no futuro o bloqueio de conteúdos ilegais será mais complexo e difícil. A imposição de comandos baseados em filtragem é na realidade uma ilusão, pois os utilizadores minimamente informados podem evitar os filtros com recursos a técnicas de comunicação anónima, contornando desse modo a censura dos fornecedores de Internet e das autoridades. Esta prática é recorrente entre os cibercriminosos, inclusive os mesmos desenvolvem as suas próprias ferramentas para comunicarem em total liberdade [34]. Considerando a diversidade de sistemas e a impossibilidade de garantir 100% de segurança nos mesmos faz com que cada ação no sentido de barrar ou dificultar o acesso dos atores maliciosos sejam exploradas novas vias para despoletar as ações maliciosas. É por esta razão que se fala por muitas vezes no jogo do "gato e do rato", isto é, no constante desafio que é colocado aos que defendem os sistemas e aos que o atacam.

A existência de sistemas como *firewalls*, antivírus, sistemas de deteção de intrusão (*IDS*) são sem dúvida importantes para a proteção das organizações. Porém são insuficientes, e tal insuficiência é justificada pelos ataques que vêm a público atingindo organizações de grande dimensão e apetrechadas de tecnologias de ponta. A existência de vulnerabilidades nas aplicações, erros humanos, a instalação de programas adulterados, a exploração por via da engenharia social, os ataques internos, os *e-mails* contendo artefactos maliciosos, a mobilidade e conetividade a partir de qualquer ponto são exemplos de meios explorados pelos atacantes e que contornam/vão além dos sistemas tradicionais de segurança implementados.

2.4.Análise de Artefactos

Um *software* malicioso (*malware*) é um programa desenvolvido especificamente para executar ações danosas e atividades maliciosas num computador. O conceito de código malicioso terá surgido em 1950 [22], embora que para computador apareceu pela primeira vez em 1986. O primeiro *malware* conhecido foi denominado de "Brain. A". Segundo o artigo [35], este foi desenvolvido no Paquistão, por dois irmãos que queriam provar que o computador não era uma plataforma segura. Após criar o *malware* difundiram-no através de disquetes. Ao colocar a disquete no computador o vírus era transmitido para o sistema principal. Quando se colocava uma disquete sem vírus o mesmo era copiado para a disquete. Este vírus não teve impactos negativos, pois a intenção era encontrar problemas e não destruir os sistemas informáticos. Com o decorrer do tempo nasceu o conceito de *malware*, este sim era mais destrutivo. Mais tarde surgiu o vírus Omega. O objetivo deste vírus era infetar o setor de inicialização (*MBR*) do disco, no entanto só provocava estragos se fosse uma sexta-feira 13. Em cada sexta-feira 13, o vírus reescrevia os primeiros 100 setores do disco rígido, desse modo, destruía a tabela de alocação de ficheiros impedindo o computador de inicializar.

No início de 1990 surgiu o vírus Casino [35]. Este copiava os ficheiros da tabela de alocação (*FAT*) para a memória principal (*RAM*) e apagava os ficheiros originais da *FAT*. Após isso surgia um

jogo, onde o utilizador tinha três oportunidades. Caso o utilizador reiniciasse o computador, o mesmo não iria arrancar uma vez que teria perdido a tabela de alocação de ficheiros, no entanto se o utilizador ganhasse o jogo, o vírus copiava os ficheiros novamente para a tabela de alocação e este poderia ser utilizado normalmente.

Em [35] indica que o próximo grande passo na evolução do *malware* foi a introdução do mecanismo de mutação. O mecanismo de mutação foi criado por um *hacker* búlgaro que se chamava Dark Avenger. Este mecanismo adicionava novas funcionalidades ao *malware*, assim seria mais difícil a sua deteção por parte dos antivírus. Basicamente, este foi o primeiro modo de polimorfismo que poderia tornar qualquer vírus mais invisível. Os antivírus utilizavam assinaturas para descobrir ficheiros maliciosos, com o método da mutação as assinaturas eram alteradas e os vírus ficavam camuflados, passando desse modo despercebidos para o sistema, tal como descrito no artigo [35].

Foi a partir de 1999, que surgiram os primeiros rumores sobre ataques massivos, com recurso a *malware*, segundo o artigo [35], alguns especialistas chegaram a afirmar que o novo milénio traria centenas de novos vírus e que estes iriam atacar todos os sistemas existentes e desencadear um apocalipse digital. Esta notícia fez com que os fabricantes de antivírus acelerassem o processo de desenvolvimento para garantir aos utilizadores que não haveria razões para ter medo de um ataque massivo. Com a chegada do novo milénio as previsões do apocalipse caíram por terra, no entanto o *malware* continuou a evoluir de forma constante e daí surgiram novos tipos de *malware*. Na Figura 9, está presente uma linha temporal onde é possível visualizar o *malware* mais marcante em cada ano.

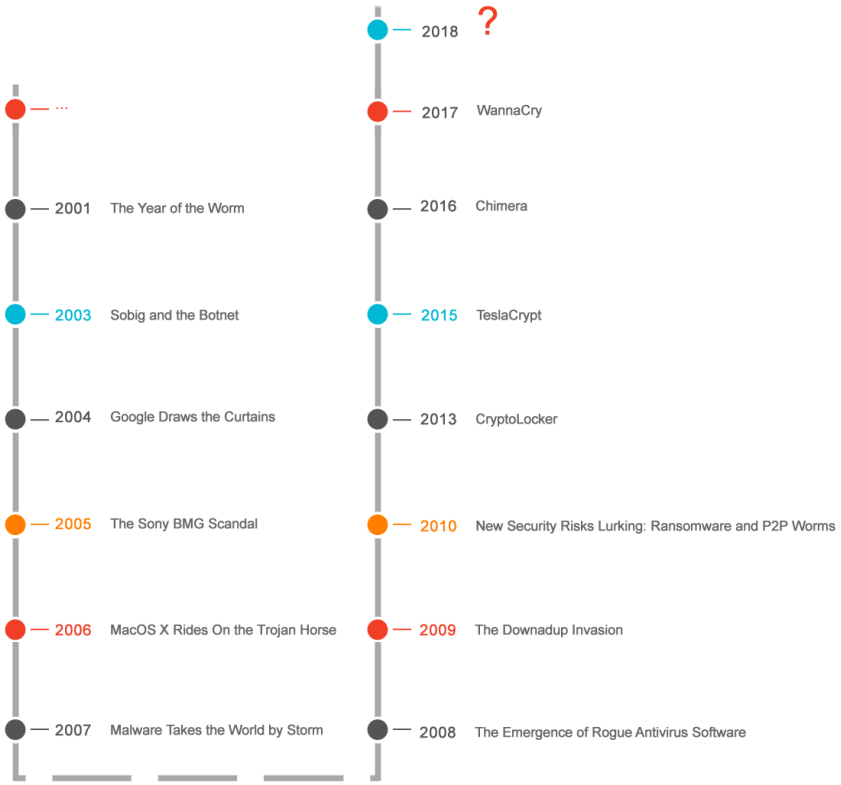


Figura 9 – Linha temporal com os vários períodos de malware [35] [36]

2.4.1. Ataques generalista vs. direcionado

No que diz respeito ao tipo de ataques, o artigo [37] indica que um ataque pode ser generalista quando se destina a múltiplos utilizadores, ou seja, os ataques são realizados em massa com o objetivo de infetar o maior número de computadores ou atingir o maior número de ciberutilizadores. Neste tipo de ataques são utilizadas técnicas como: *phishing* (envio de *e-mails* para um grande número de pessoas a pedir informações sensíveis); *water holing* (criação de um Website falso ou comprometer um legítimo, a fim de roubar informações confidenciais); *ransomware* (cifra dos dados pessoais e pedido de resgaste para desbloqueá-los); *scanning* (escutar as ligações de rede, na tentativa de roubar informações que auxiliam num futuro ataque). Já os ataques direcionados, e de acordo com a explicação do autor no artigo [38] o objetivo é atingir uma determinada organização ou um ciberutilizador em específico. Os cibercriminosos escolhem normalmente organizações que processam e guardam informações sensíveis e possíveis de serem utilizadas para fins maliciosos ou venda em mercados dito negros. Na mira dos atacantes estão empresas de serviços financeiros como bancos, empresas de prestação de serviços e organizações públicas. São utilizadas técnicas tais como *spear-phishing* (envio de *e-mails* que contêm um anexo com *software* malicioso, ou um *link* para descarregar um *software* malicioso dirigido a um setor/pessoa em específico); recurso a *botnets* por exemplo para lançar um ataque do tipo *Distributed Denial of Service*; *subverting the supply chain*, isto é, um ataque na cadeia de fornecimento para por exemplo alterar o *software/hardware* entregue à organização, e desse modo manipular o seu bom funcionamento, na tentativa de controlar os processos da organização [37]. Para cada tipo de ataque existe código malicioso adequado, o mesmo é desenvolvido e ajustado de acordo com a necessidade do atacante.

2.4.2. Ransomware: O mais marcante da atualidade

O *ransomware* é uma ameaça informática que tem afetado milhares de utilizadores em todo o mundo. Este tipo de *malware* invade o sistema e cifra os ficheiros dos utilizadores, tirando partido de algoritmos criptográficos. Neste momento as ferramentas para decifrar a informação são quase nulas, o que leva as vítimas a pagar o resgate pelos seus dados. De acordo com o artigo [39], existem vários tipos de *ransomware* a afetar dispositivos à escala mundial. A maioria das vítimas têm cedido aos atacantes, efetuando os pagamentos solicitados. De acordo com o mesmo artigo, os *ransomwares* mais populares são:

- ***Cryptolocker*** – surgiu pela primeira vez em 2013. Este cifra os ficheiros do utilizador com uma chave de 2048 bits, impossibilitando o acesso aos ficheiros, até que seja pago o resgate exigido [39].
- ***Cryptowall*** – surgiu poucos meses depois do *Cryptolocker* e o seu funcionamento é semelhante ao *Cryptolocker*. O *Cryptowall* versão 4.0, que também é conhecido como *HELP_YOUR_FILES*, chega as vítimas através do *e-mail*. Este *e-mail* pode ser filtrado como lixo eletrónico ou *spam*, mas ainda existe uma possibilidade de o mesmo ser aberto pelo seu recetor [40].

- ***CTB-Locker*** (*Curve-Tor-Bitcoin Locker*) – surgiu pela primeira vez em 2014. Também é conhecido como *Critoni*, e foi desenvolvido para plataformas *Windows*. Este propaga-se via *e-mail*, contendo o ficheiro malicioso em anexo. O anexo normalmente aparece sobre um dos formatos *ZIP*, *SCR*, *CPL*, *CAB*, ou *PIF* [41].
- ***TorLocker*** – surgiu em 2014. Este tipo de *ransomware* infiltra-se no sistema através de mensagem de *e-mail* (spam), vulnerabilidades existentes no sistema operativo ou *software* do utilizador, *bots* e atualizações de *software* fraudulentas. Após infiltrado cifra os ficheiros armazenados no computador e exige um regaste para decifrar os mesmos [42].
- ***Kryptovor*** – surgiu em 2014. Este *malware* rouba os ficheiros dos dispositivos comprometidos, no entanto também possuí um componente *ransomware*.
- ***TeslaCrypt*** – este tipo de *ransomware* propaga-se através de ficheiros associados a jogos e plataformas como *RPG Maker*, *League of Legends*, *Call of Duty*, *Dragon Age*, *StarCraft*, *MineCraft*, *World of Warcraft*, *World of Tanks*, e outros jogos *online*. Ele implementa um padrão criptográfico avançado que leva os jogadores a pagarem pela senha de acesso [43].
- ***Chimera*** – é distribuído via *e-mail*. Este além de cifrar os ficheiros e exigir um resgate, indica se o mesmo não for pago que torna públicos todos os ficheiros e credenciais roubadas [44].
- **WannaCry** – foi o *ransomware* que mais rapidamente de propagou até aos dias de hoje. Em quatro dias, infetou mais de 250 mil dispositivos. Foi inicialmente distribuído via *e-mail*, indicando que seria uma atualização de *software* de impostos. Em seguida, explorou uma vulnerabilidade nos servidores para desse modo cifrar os ficheiros existentes nos mesmos [36].

Para reduzir o impacto dos ataques *ransomware*, o utilizador não deve executar todos os ficheiros que recebe, principalmente se estes forem de origem desconhecida. Para além desta precaução deve manter um *backup* dos dados num dispositivo externo, desligado da rede.

2.4.3.Superfícies de ataque

De acordo com o artigo [45], a superfície de ataque é o conjunto das vulnerabilidades de um dispositivo ou rede informática que pode ser explorada por um atacante. Para visualizar a superfície de ataque de uma organização deve-se identificar e mapear todos os dispositivos e segmentos de rede da organização. Os elementos mapeados devem ser: servidores (Web, aplicacionais, base de dados), pontos de rede (ligações com computadores fixos/portáteis e dispositivos móveis), segmentação da rede (distinguir a rede pública da privada), dispositivos de segurança, sistema de prevenção de intrusão. Após identificar todos os dispositivos, devem ser documentados os indicadores de exposição na rede (*IE*). *IE* são indicadores que indicam se um sistema, dispositivo ou rede está exposto a ataques, este indicador está em contraste com indicadores de compromisso (*IC*), que são os artefactos que indicam se um ataque já foi realizado, isto é, se os sistemas já foram comprometidos. Os *IE* incluem: vulnerabilidades de

software, tais como deficiências em aplicações, navegadores, sistemas operativos, sistemas de base de dados; configurações incorretas e falta de *software* de segurança; violação de políticas de segurança e regras de conformidade, tais como configurações de rede que permitem o acesso a informações confidencias por utilizadores não autorizados. Como exemplos de *IC* temos os *data leaks* tornados públicos ou a existência de comunicações de rede entre as máquinas da nossa organização e servidores sob controlos dos atores maliciosos.

Um dos vetores muito explorado pelos atores maliciosos para disseminar artefactos é o *drive-by-download*. Trata-se de um tipo de ataque realizado através de Websites para os quais os utilizadores são encaminhados e "são convidados" a descarregar conteúdo malicioso. Os atacantes procuram servidores vulneráveis na Web e inserem códigos maliciosos nas páginas Web presentes, quando os mesmos não conseguem faze-lo recriam o Website e divulgam o *URL* via *e-mail*. No que diz respeito ao *e-mail* este contém no corpo da mensagem um *URL* que redireciona a vítima para o Website malicioso e a incentiva a descarregar um *plugin* ou um *software*, que quando instalado permite o acesso ao atacante. Desse modo o *drive-by-download* pode ter dois significados relativos as transferências. Estas transferências podem ser autorizadas pela pessoa ou acopladas a outros ficheiros descarregados, que quando executados descarregam outro *software* sem consentimento do utilizador. No entanto não é necessário clicar em algo para iniciar o *drive-by-download*, basta visitar uma página infetada por um *exploit kit*, deste modo o atacante pode descarregar o *malware* para o computador da vítima. Na Figura 10 são apresentados os passos executados num ataque de *drive-by-download* e o ponto onde o **WE-SAND** se enquadra como potencial solução para mitigar o problema.

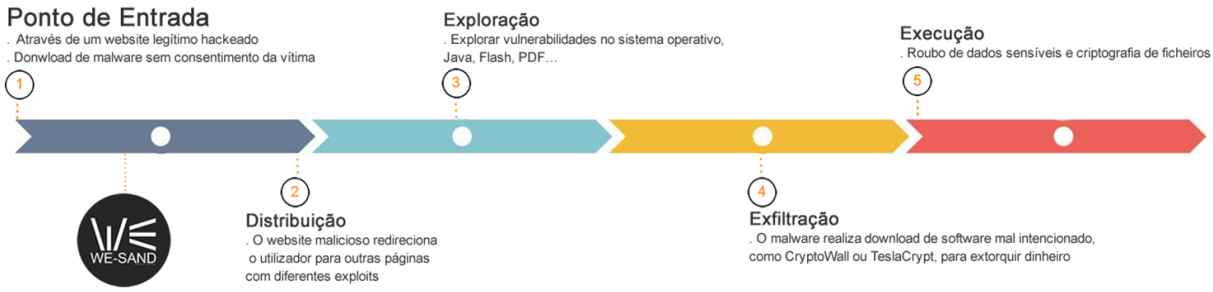


Figura 10 – Anatomia de um ataque de drive-by-download [46]

No artigo [47], o autor indica que o *JavaScript* é uma linguagem de *script* que permite aos programadores executar código do lado do cliente, sendo esta uma das tecnologias mais utilizadas. No entanto, o código *JavaScript* também é utilizado para realizar ataques contra o *browser* do utilizador e suas extensões. Estes ataques resultam geralmente no *download* de *malware* que assume o controlo completo da máquina da vítima o que se denomina por "*drive-by-download*". Dada a natureza dinâmica do *JavaScript* torna-se difícil detetar e bloquear código malicioso, que conduz os utilizadores para Websites maliciosos e instalação de *malware* na própria máquina pode tornar estes membros de uma

botnet. A maioria do conteúdo malicioso é ofuscado, através de várias camadas de ofuscação. Os *scripts* maliciosos usam uma grande variedade de técnicas de ofuscação, como exemplo temos a codificação simples, em formato *base64*. Existem, no entanto, características que podem indicar a atividade maliciosa e ajudar a identificar um *JavaScript* potencialmente perigoso, quando registadas em padrões considerados anormais. Entre os padrões destacam-se:

- Número e destino dos redireccionamentos;
- Diferenças baseadas no histórico para um conjunto de Websites semelhantes;
- Rácio de definição de *strings* e uso das mesmas (uso de cadeias para descodificar código ofuscado em *strings*);
- Duração da execução do código dinâmico;
- Número de *bytes* atribuídos através de operações de cadeia (funções de concatenação e *substring* são monitorizadas para analisar a memória alocada);
- Número de *strings* com *shellcode* incluído;
- Número de componentes instanciados;
- Valores atribuídos e devolvidos pelos métodos;
- Sequência de chamada dos métodos.

Quando um ficheiro *JavaScript* apresenta as características definidas anteriormente, tudo indica, que existe um perigo para o utilizador. O utilizador deve manter o antivírus atualizado, e quando detetar um *pop-up* que não abria anteriormente, informar de imediato a equipa de segurança da empresa. Um simples clique nesse *pop-up* pode desencadear a instalação de um *malware* e comprometer a segurança informática de toda a organização.

2.4.4. Sistemas de Sandbox

Uma *sandbox* é um sistema isolado, que permite a execução de aplicações desconhecidas num ambiente dedicado, sem o risco de afetar os sistemas de produção. Segundo o artigo [48] estes sistemas permitem, por exemplo, a triagem de ficheiros não confiáveis de modo a proteger os utilizadores e sistemas informáticos. Na Figura 11 é ilustrado um diagrama no qual existe um sistema de *sandbox*. De acordo com a figura os pedidos passam pela *sandbox*, e este analisa *URLs* e ficheiros como indica no artigo [49]. Deste modo é possível, por exemplo, verificar se os pedidos realizados ao exterior são confiáveis.

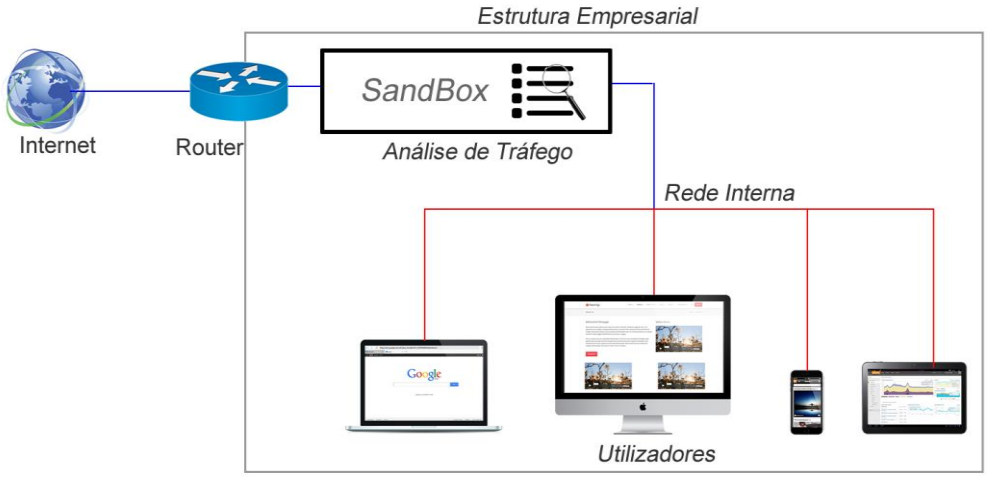


Figura 11 – Exemplo de um Sistema de Sandbox

Atualmente existem vários sistemas de *sandbox*. A listagem que se segue apresenta alguns desses sistemas:

- **Cuckoo** [50] – é um *software* gratuito que permite a análise de *malware*. Por ser gratuito é possível personalizar o sistema de acordo com as necessidades.
- **GFI Sandbox (Threat Analyzer)** [51] – com uma plataforma totalmente personalizável, o Threat Analyze permite recriar os ambientes virtuais e nativos, para seguidamente executar o código malicioso e ver exatamente como o *malware* se comporta.
- **Joe Sandbox Cloud** [52] – executa ficheiros e *URLs* em ambiente controlado. Faz a monitorização do comportamento das aplicações e do sistema operativo, quando deteta atividades suspeitas. Todas as atividades são compiladas num relatório.
- **Capture-HPC** [53] – é um sistema de alta interatividade. As páginas a serem analisadas são processadas dentro de um ambiente virtualizado.
- **Threat Expert** [51] – é um sistema de análise de ameaças *online*. Após submeter um ficheiro, o sistema analisa o comportamento do ficheiro para detetar de forma automática a existência de vírus, *worms*, cavalos de troia, *adware*, *spware*, entre outros.
- **JSDetox** [54] – é um sistema *online* que permite a análise de *JavaScript* de forma estática, possui um motor para executar e desofuscar o código presente em cada ficheiro.
- **Malwr** [55] – é um sistema *online* que permite o envio de ficheiros e obtenção de resultados de forma dinâmica. Este é baseado no Cuckoo Sandbox e VirusTotal.
- **VirusTotal** [56] – é um sistema *online* que analisa ficheiros e *URLs* suspeitos, desse modo deteta facilmente vírus, *worms*, cavalos de troia embebidos em ficheiros maliciosos.
- **VxStream Sandbox** [57] – é um sistema *online* de análise *malware* totalmente automatizado e inovador.
- **JOTTI** [58] – é um sistema *online* gratuito que permite verificar ficheiros, com recurso a vários antivírus. Permite o envio de cinco ficheiros em simultâneo.

- **SCANNI** [59] – é um sistema *online* que utiliza vários mecanismos (antivírus) de análise, para devolver um resultado mais assertivo.

Os sistemas apresentados têm como objetivo capturar e analisar *malware*. Alguns destes sistemas irão ser utilizados no desenvolvimento do protótipo funcional referido neste projeto. A Tabela II apresenta uma comparação entre os sistemas anteriores. As colunas da tabela contêm as *sandbox* acima apresentadas. As linhas contêm um conjunto de parâmetros de comparação. Estes parâmetros foram definidos com base nos objetivos que o projeto a desenvolver se propõe. A ponderação atribuída a cada sistema foi recolhia do Website do seu fornecedor/desenvolvedor.

Tabela II – Análise comparativa dos vários sistemas de sandbox

	Cuckoo	**GFI Sandbox (Threat Analyzer)**	**Joe Sandbox Cloud**	**Capture-HPC**	**Threat Expert**	**JSDetox**	**Malwr**	**Virustotal**	**VxStream Sandbox**	**JOTTI**	**SCANNI**
Última Atualização (Ano)	2016	2016	2016	2008	2016	2010	2016	2016	2017	2018	2018
Funciona Remotamente (R) / Local (L)	L	L	R	L	R	L	R	R	R	R	R
Analisa *URL* (U) / Ficheiros (F)	U/F	F	U/F	U	F	F	U/F	U/F	F	F	F
Gratuito (S/N)	S	**N**	**N**	S	S	S	S	S	**N**	S	S
Documentação Disponível (S/N)	S	S	S	S	S	S	S	**N**	S	S	S
Gera Relatórios (S/N)	S	S	S	N	S	S	S	S	S	S	S
Tipo de Análise Estática (S/N)	**N**	**N**	**N**	**N**	S	S	S	S	**N**	S	S
Tipo de Análise Dinâmica (S/N)	S	S	S	S	**N**	**N**	**N**	**N**	S	N	N

Obter Histórico de Análise (S/N)	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**
Código Aberto (S/N)	S	**N**	**N**	**N**	**N**	S	**N**	**N**	**N**	**N**	**N**

Dos sistemas de *sandbox* apresentados na Tabela II, o Cuckoo é a que apresenta um maior e interessante conjunto de características. O facto de ser gratuita, gerar relatórios, ser de código aberto e ser instalada na rede local são das características que mais se destacam no âmbito do desenvolvimento do **WE-SAND**. No protótipo **WE-SAND** será utilizado o Cuckoo, numa fase inicial, para que seja possível entender a construção do sistema e modo como é processada a análise de artefactos. Numa fase avançada do projeto serão utilizados os seguintes sistemas VxStrem Sandbox, Malwr (baseado em Cuckoo), JOTTI, VirusTotal, SCANNI. A escolha dos sistemas anteriores foi influenciada pela disponibilidade de acesso, forma de integração com o **WE-SAND** e resultados pretendidos após análise.

2.5.Análise de Código Malicioso

A análise de ficheiros executáveis e conteúdos maliciosos desconhecidos não é um problema novo. Já existem soluções nesta área, e podem ser divididas em duas categorias: análise estática e análise dinâmica. Nas próximas subseções são descritas as duas categorias de análise. No seguimento desta descrição é apresentado um descritivo da linguagem *JavaScript*, que pretende demostrar como a análise de código pode auxiliar para a deteção de ameaças.

2.5.1.Análise estática

A análise estática é o processo de analisar um programa sem que este esteja em modo de execução. Neste processo, um binário é normalmente "desassemblado" (*disassembled*) em primeiro lugar. Em seguida, é analisado o código para perceber o fluxo de dados e a rota seguida pelas operações. A análise estática tem a vantagem de cobrir por completo o código do programa. No que diz respeito a análise de *malware*, a situação é um pouco diferente porque o código malicioso é alterado constantemente e o programador faz uso da ofuscação o que torna o processo de conversão e análise do código *assembly* muito complexa. Na realidade o código analisado por um analisador estático pode não ser necessariamente o código que realmente é executado. Este código pode estar preparado para se auto modificar em tempo de execução dificultando o processo de análise [60].

2.5.2.Análise dinâmica

Ao contrário da análise estática, a análise dinâmica consiste na análise durante a execução do código. Esta técnica tem a vantagem de analisar apenas as instruções que se encontram em execução, sendo assim, esta técnica mais resistente à ofuscação e a tentativas de auto modificação. A análise dinâmica poderá ser feita com recurso a ambientes virtualizados ou a ambientes físicos. Os ambientes

virtualizados apresentam vantagens de operação, sendo mais fáceis de montar e impedem que o *malware* seja transmitido para o sistema principal. No entanto existe uma desvantagem pois o *malware* em análise pode detetar que está a ser executado num ambiente virtualizado e alterar o seu comportamento para que o mesmo seja inofensivo e desse modo chegar ao sistema principal. Em alternativa à virtualização existem programas de emulação como por exemplo o *PC emulator*. Trata-se de *software* semelhante a virtualização, no entanto não modifica o sistema operativo, ou seja, este incluí o processador, placa gráfica, disco rígido e outros recursos, com a finalidade de executar um sistema operativo não modificado. Uma vez todas as instruções carregadas no emulador, o sistema é semelhante a um sistema real. As desvantagens associadas é que é mas lento e para um *malware* que funcione com base em sincronização pode detetar que não está a operar num sistema real e por isso mudar o seu comportamento ou não executar de todo [60].

2.5.3. JavaScript

Tal como é referido no artigo [61], o *JavaScript* tornou-se numa das linguagens mais utilizadas da Web. A mesma pode ser integrada com as diferentes linguagens de programação direcionadas para a Web. Os *scripts* são de fácil acesso, quando integrados num Website. Estas características fazem com que seja possível aos atacantes introduzir código ofuscado no ficheiro *JavaScript* descarregado ou replicar esse código para outros Websites desenvolvidos pelos próprios. No código replicado podem introduzir instruções maliciosas de modo a conduzir os utilizadores para Websites de *phishing* ou *download* de *malware*. Com recurso aos exemplos apresentados nos artigos [62] [63] foi extraído um trecho de código que é aqui apresentado na Listagem 1 e Listagem 2. Trata-se de um trecho considerado como malicioso uma vez que dá origem a um ataque do tipo *drive-by-download*.

Na Listagem 1 é apresentado o código desofuscado e na Listagem 2 é apresentado o código ofuscado. As linhas presentes na primeira figura significam: 1 - O comando apresenta uma mensagem, a indicar que o Website se encontra em manutenção; 2 - Descarrega um ficheiro malicioso; 3 - Redireciona para o Website fraudulento.

```
1. (...)
2. <script type="text/javascript">
3.     alert("WebSite em Manutenção.  Será redirecionado para o website secundário. Para melhor
    o funcionamento instale o programa que irá surgir para download.");
4.     window.open('http://www.cgd.com.pt/manutencao.exe', '_blank');
5.     window.location = "http://www.cgd.com.pt";
6. </script>
7. (...)
```

Listagem 1 – Código* JavaScript *desofuscado

```
1. (...)
2. <script type="text/javascript">
3.    document.write(unescape('%3C%73%63%72%69(...)22%3B%0A%3C%2F%73%63%72%69%70%74%3E%0A'))
4. </script>
5. (...)
```

Listagem 2 – Código* JavaScript *ofuscado. Equivalente ao apresentado na Listagem 1

Os utilizadores ao acederem ao Website principal são alertados, que o mesmo está em manutenção. Após confirmação são conduzidos a descarregar e a instalar um *software* malicioso e automaticamente redirecionadas para um Website que pode ser a réplica do original, no entanto o único objetivo é o roubo de dados confidenciais.

O ataque de *drive-by-download*, segundo o descrito no artigo [64] está divido em nove etapas:

1. O utilizador acede de forma involuntária a um Website comprometido;
2. Os ficheiros de *JavaScript* malicioso são descarregados para o sistema do utilizador;
3. Os ficheiros são executados através do *browser*, provocando a infeção por *malware*;
4. Os ficheiros de *JavaScript* infetados redirecionam o tráfego para um servidor de exploração;
5. O *software* instalado no servidor de exploração procura vulnerabilidades no sistema;
6. Quando encontra uma vulnerabilidade, esta é usada para aceder as funções do computador;
7. Após aceder ao computador é verificado se existem privilégios de administrador;
8. O *malware* é transferido para o computador e executado;
9. O *malware* executa as funções pré-determinadas (e.g. recolher informações pessoais e enviar para o servidor controlado pelo atacante, recolher credenciais, cifrar ficheiros e pedir resgate, despoletar ataques contra terceiros).

A Figura 12 exemplifica os passos descritos anteriormente.

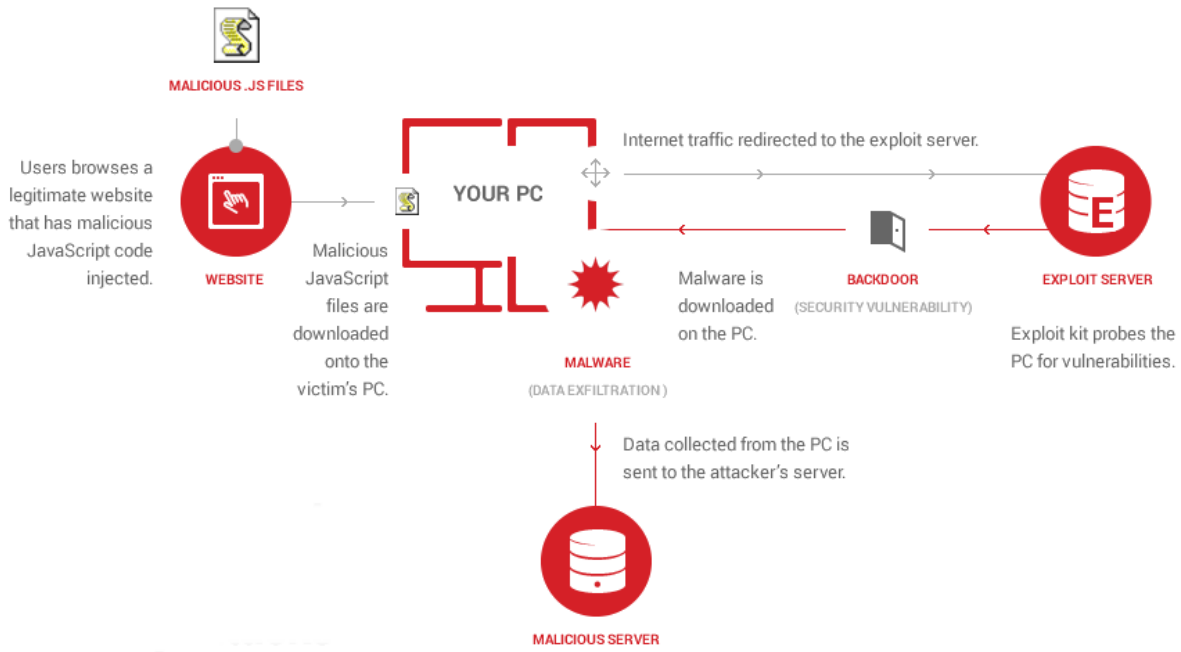


Figura 12 – Exemplo de um ataque de drive-by-download [64]

Considerando a utilização de *JavaScript*, os autores em [65] argumentam que é possível identificar código malicioso a partir de quatro pontos:

1. **Propriedades de *URLs***: identificar se o *URL/IP* é confiável com recurso a uma *blacklist*, e análise de registo de domínio.
2. **Analisar o Output do *JavaScript*:** validar se o resultado gerado pelo *JavaScript* não contém palavras associadas a código malicioso.
3. **Comportamentos do *JavaScript*:** devem ser validadas funções como o *eval()* e *unsecape()*. Estas funções são utilizadas de forma intensiva por atacantes para executar o código ofuscado. Código este, que pode comunicar com entidades externas e roubar informações confidenciais do utilizador.
4. **Conteúdo do código *JavaScript*:** esta característica está relacionada com o tamanho e comportamento do código durante um período temporal. Estas alterações podem ser indicadoras de um ficheiro com código malicioso. No artigo [66], é demonstrado um método de cálculo para encontrar o vetor de semelhança entre dois ficheiros. Estes cálculos podem ser executados com base em vetores de semelhança, comparação de tamanho e assinatura de ficheiros.

Numa era digital onde as ameaças crescem e ganham novas formas, é necessário entender as ameaças mais comuns e os seus impactos no decorrer de um ataque. Ao longo deste capítulo foram apresentados um conjunto de tópicos que se relacionam com o que será o **WE-SAND**. O **WE-SAND**

será desenvolvido com base nos sistemas de *sandbox* apresentados na Tabela II – Análise comparativa dos vários sistemas de *sandbox*. O objetivo é criar um *middleware* que comunica com os vários sistemas de *sandbox*. O **WE-SAND** sempre que possível recolherá os artefactos, estes serão submetidos aos vários sistemas de *sandbox* e os resultados apresentados num *dashboard*. A principal vantagem do **WE-SAND**, é permitir múltiplas análises realizadas por outros sistemas de *sandbox*, sendo assim possivel, comparar resultados e definir o grau de risco com maior precisão. Mediante o grau de risco será possível entender a dimensão do ataque e evitar o seu desencadear, ou seja, travar ataques numa fase inicial.

Capítulo 3 – Especificação do WE-SAND

Neste capítulo são descritas as etapas relativas ao levantamento de requisitos e especificação do **WE-SAND**. Esta fase tem como objetivo, compreender o problema e definir metas para o desenvolvimento da aplicação. Durante o processo de análise são identificados os requisitos de implementação e definidos objetivos para execução do projeto.

3.1. Engenharia de requisitos e suas metodologias

A engenharia de requisitos é o processo que permite descobrir a finalidade para a qual um sistema se destina [67], e desse modo proceder a implementação de forma correta. No caso de projetos com fins lucrativos é realizada uma análise de viabilidade do projeto, custo e retorno do investimento. Os requisitos são categorizados em requisitos funcionais (serviços que o sistema deve oferecer, mediante determinadas situações), não funcionais (definem propriedades e restrições do sistema) e organizacionais (dizem respeito às metas da organização, em consequência de políticas e procedimentos).

No início do desenvolvimento de uma aplicação, é possível que os requisitos indicados pelos *stakeholders* sejam incompletos, ambíguos e inconsistentes, assim sendo, é necessário construir um documento de requisitos e proceder a um conjunto de atividades juntamente com os *stakeholders*. Estas atividades consistem, em analisar os processos de trabalho e a informação a introduzir na aplicação, com o objetivo de produzir resultados finais. Existem inúmeras abordagens para o processo de desenvolvimento, tais como: modelo em cascata, espiral, codificação e correção, prototipagem.

O desenvolvimento do **WE-SAND** teve por base o modelo de prototipagem. A implementação de um protótipo substitui a fase de requisitos e permite que os utilizadores tenham contacto com um produto que devolve os resultados esperados na versão final. Após conclusão do protótipo, o projeto deve seguir a implementação com uma abordagem em cascata/espiral, com o objetivo de cumprir todas as etapas de desenvolvimento de *software*. Enquadrado ainda neste tipo de desenvolvimento, existem as metodologias ágeis. Como indicado em [68] a preocupação centra-se na produção de *software* personalizado a medida do cliente, sendo a maioria da documentação gerada pelas ferramentas usadas na produção. O cliente acompanha o processo de desenvolvimento, tendo um papel importante na definição de novos requisitos, contrariando a prática de tudo ser planeado e definido no início do projeto.

3.2. Diagrama de implementação

Na Figura 13 é ilustrado o diagrama de implementação do **WE-SAND**. O diagrama descreve o processo iniciado na recolha de artefactos até a apresentação dos resultados. A implementação divide-se em dois processos distintos:

- 1 - análise de ficheiros – o utilizador coloca os ficheiros num repositório. Seguidamente estes são recolhidos e submetidos para análise.
- *2 - logs* de acesso Web – o utilizador configura no **WE-SAND** a localização do ficheiro de *log*. Cada linha do ficheiro de *log* corresponde a um *URL* que será recolhido pelo **WE-SAND** e submetido para análise.

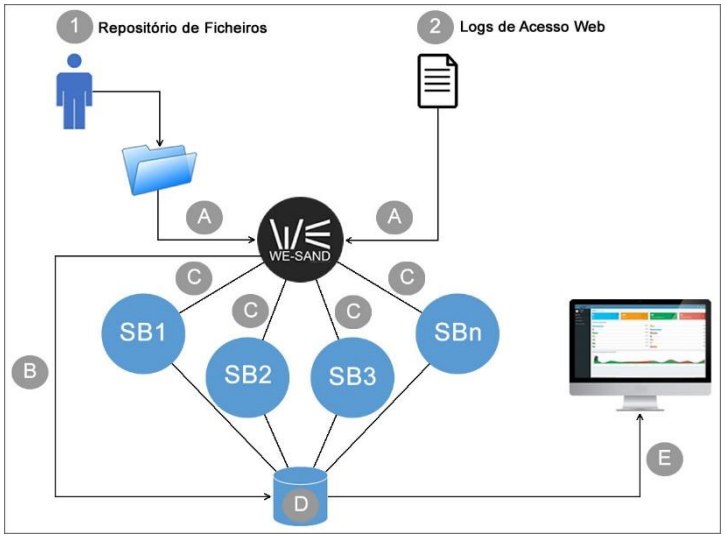


Figura 13 – Diagrama de implementação

3.2.1. Repositório de Ficheiros

No processo número 1 (Repositório de Ficheiros), os ficheiros são colocados num repositório, seguidamente o **WE-SAND** verifica se existe algum sistema de *sandbox* (B), com suporte para envio de ficheiros. Os ficheiros são submetidos para os vários sistemas de *sandbox* (C), e os resultados armazenados na base de dados (D), para posteriormente serem apresentados no *dashboard* e listagem (E). Esta sequência de passos é ilustrada na Figura 14.

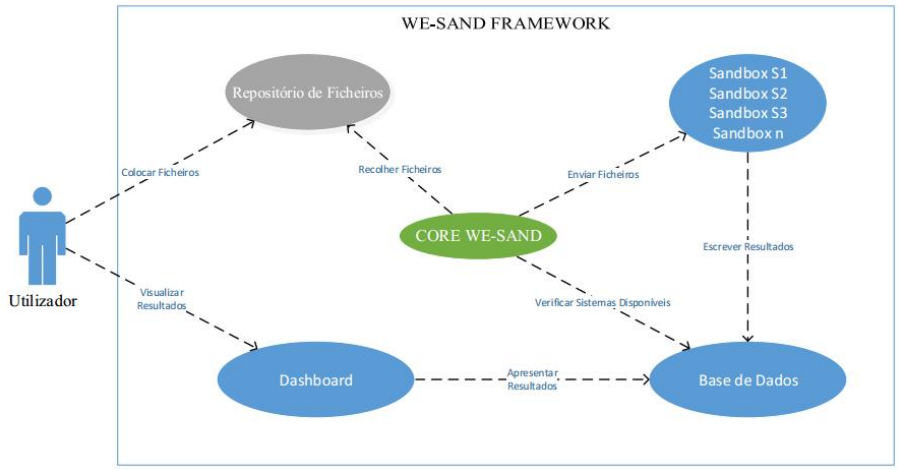


Figura 14 – Repositório de Ficheiros

3.2.2.Logs de Acesso Web

No processo número 2 (*logs* de acesso Web), o **WE-SAND** verifica se existe algum sistema de *sandbox* (B), com suporte para análise de *URLs*. O ficheiro *log* é lido e os vários *URLs* são submetidos aos vários sistemas de *sandbox* (C). Os resultados são armazenados na base de dados (D), para posteriormente serem apresentados no *dashboard* e listagem (E). Esta sequência de passos é ilustrada na Figura 15.

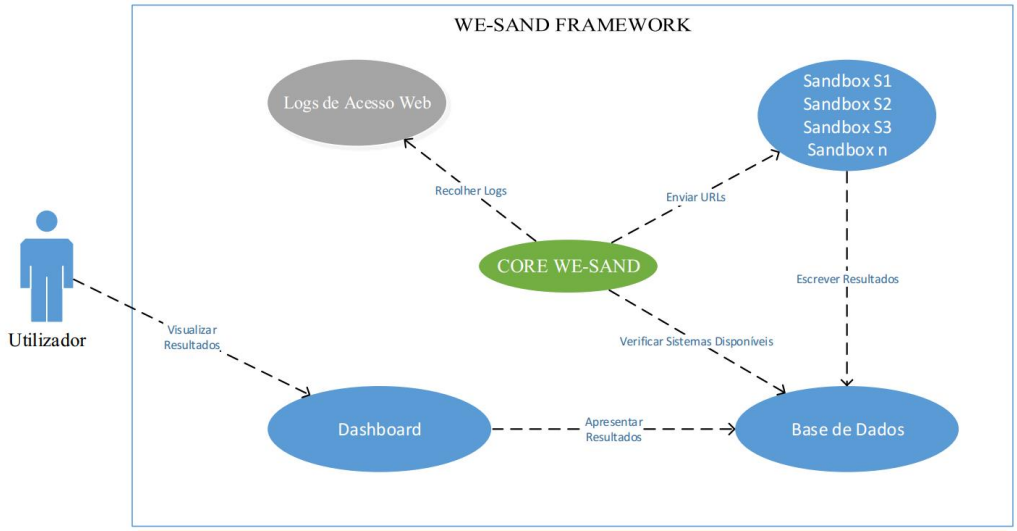


Figura 15 – Logs de Acesso Web

A Figura 14 e Figura 15 representam o fluxo desde da recolha de artefactos até a visualização de resultados. Alguns tipos de sistemas de *sandbox*, não suportam a análise de *URLs*/ficheiros, nesse sentido, durante o processo de inclusão de um sistema de *sandbox*, este deve ser configurando tendo em consideração as suas potencialidades/forma de funcionamento.

3.3.Diagrama de classes

Durante o processo de levantamento de requisitos foi identificada a necessidade de implementar cinco entidades: "*users*", "*access_logs*", "*system_setup*", "*sandbox_system*", "*data_analyzed*".

O diagrama apresentado na Figura 16 é constituído pelas seguintes entidades:

- *access_logs* – registo de acessos realizados ao sistema;
- *users* – utilizadores registados no sistema;
- *system_setup* – pré-configurações do sistema;
- *sandbox_system* – registo de todos os sistemas de sandbox que poderão ser utilizados e respetivas chaves de acesso "*api_key*";
- *data_analyzed* – registo dos resultados da análise.

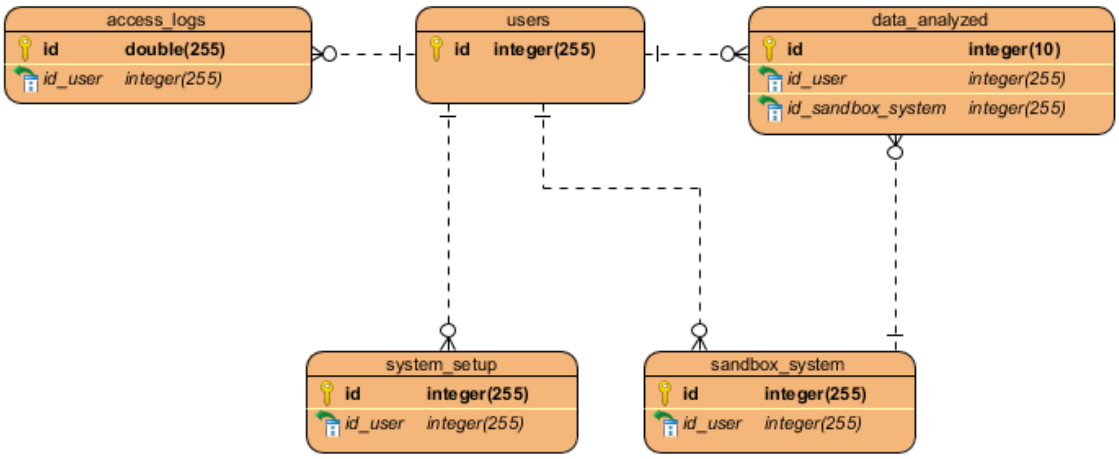


Figura 16 – Diagrama de classes

A estrutura anterior será a base de implementação da base de dados, que dará suporte ao sistema **WE-SAND**.

3.4.Implementação do WE-SAND

A implementação do **WE-SAND** pressupõe a existência de *interfaces* que permitam a submissão dos artefactos para análise. Mediante o artefacto serão selecionados os sistemas de *sandbox* que irão proceder à análise. Os resultados obtidos serão apresentados num *dashboard* e enviados por *e-mail*. Para facilitar a apresentação e acesso a aplicação, a mesma será implementada sob a forma de uma aplicação Web. A aplicação Web será criada com recurso a:

- *HTML* – uma linguagem de marcação utilizada na criação de páginas Web;
- *CSS* – uma linguagem para estilos que define o *layout* do Website;
- *JavaScript* – uma linguagem processada do lado do cliente, ou seja, pelo próprio *browser*. Esta permite criar animações, tornando a interação com a aplicação mais dinâmica;
- *PHP* – uma linguagem processada do lado do servidor, que permite a comunicação entre a base de dados e *HTML*.

Para armazenar os dados relativos aos artefactos e resultados obtidos será utilizado um sistema de base de dados relacional. A este nível não se identificam nesta fase constrangimentos ao nível de requisitos não funcionais pelo que se propõe a utilização do sistema de gestão de base de dados *MySQL*.

Ao longo dos próximos capítulos será focada a implementação e a seleção dos sistemas de *sandbox* que integram, numa primeira fase, o **WE-SAND**.

Capítulo 4 – Estudo comparativo entre sistemas de Sandbox

Este capítulo apresenta um estudo comparativo entre vários sistemas de *sandbox*. Neste estudo foram submetidos a análise 106 artefactos e utilizados cinco sistemas de *sandbox*. O objetivo da análise foi analisar a performance e a eficácia dos sistemas de *sandbox* (VxStream Sandbox, Malwr, Jotti, VirusTotal, Scanni), permitindo através dos resultados obtidos identificar quais os sistemas mais adequados para integrar o protótipo do **WE-SAND**.

4.1. Questões de investigação

Os sistemas de *sandbox* são ferramentas que ajudam os administradores da rede/equipas de segurança informática a identificar ameaças/ataques. Neste projeto propomos a criação de uma *interface* que, de forma transparente para o utilizador, agrega vários sistemas de *sandbox*. A implementação e sucesso de uma solução como o **WE-SAND** esta estritamente ligada à eficácia dos sistemas de *sandbox* e neste âmbito realizamos um estudo comparativo entre sistemas de *sandbox* considerando artefactos maliciosos e não maliciosos. O estudo comparativo pretende dar resposta a duas questões principais:

- Qual é a eficácia de deteção dos sistemas de *sandbox*?
    - Neste âmbito serão considerados vários tipos de artefactos e vários sistemas de *sandbox*, de modo a entender a eficácia de cada um deles. Mais especificamente pretende-se medir a taxa de acerto de cada *sandbox* perante artefactos de vários tipos e deliberadamente maliciosos e não maliciosos.
- Qual é o tempo médio que cada sistema de *sandbox* requer para a análise?
    - Neste âmbito serão recolhidos os tempos de análise, sempre considerando que os tempos podem variar entre sistemas, dependendo, da rapidez do sistema e da ligação a Internet.

4.2. Ambiente de testes

O desenvolvimento das classes de ligação aos vários sistemas de *sandbox* e os primeiros testes que deram origem aos gráficos presentes neste capítulo foram realizados num computador com as seguintes características:

- Portátil ASUS M51VA-AP016C;
- Processador Intel Core 2 Duo;
- Memória *RAM* 4 GB;
- Disco *SSD*;
- Sistema Operativo Linux Mint 17.1 Rebecca;
- Placa *Wi-Fi* com a norma 802.11abgn que dá um máximo de 100 Mbps;
- Ligação à Internet com 4 Mbps de largura de banda *upload* e 31 Mbps de *download*.

Após a primeira fase de testes, o desenvolvimento do protótipo foi realizado num computador com um sistema operativo diferente e características superiores.

- Portátil ASUS N550JV.208;
- Processador Intel Core i7-4700HQ;
- Memória RAM 16 GB;
- Disco mecânico 5400 rotações;
- Sistema Operativo Microsoft Windows 10 Home;
- Placa *Wi-Fi* com a norma 802.11n que dá um máximo de 100 Mbps;
- Ligação à Internet com 4 Mbps de largura de banda *upload* e 31 Mbps de *download*.

Ao alterar o sistema de desenvolvimento foi possível perceber que os tempos de análise eram muito semelhantes.

4.3. Seleção dos sistemas de sandbox

Para a seleção dos sistemas de *sandbox*, foram considerados fatores como a disponibilidade e possibilidade de integração com o **WE-SAND**. Foram também selecionados sistemas menos conhecidos, para ter um espetro mais variado de comparação. Resumindo, para o estudo comparativo, foram selecionados os seguintes sistemas de *sandbox*:

- VxStrem Sandbox – sistema automático de análise, com inúmeros recursos e uma extensa base de conhecimento.
- Malwr – sistema baseado no Cuckoo Sandbox, ferramenta utilizada na base inicial do projeto.
- JOTTI – utiliza vários antivírus (Avast, AVG, ESET…) na análise de ficheiros.
- VirusTotal – sistema em constante evolução e com bibliotecas já desenvolvidas para integração.
- SCANNI – trata-se de um sistema pouco conhecido e que foi incluído para obter outra abordagem comparativa (mais vulgares vs. menos vulgares). Este utiliza vários mecanismos (antivírus) de análise.

4.4. Seleção de artefactos - datasets

Para o estudo comparativo foram selecionados 106 artefactos de forma aleatória. Os artefactos foram recolhidos de vários Websites. Estes estão divididos em duas partes iguais, 53 dizem respeito a artefactos não confiáveis e 53 a artefactos confiáveis. Os artefactos dividem-se em 13 formatos de dados distintos (*JS*, *BAT*, *BIN*, *EXE*, *HTML*, *ZIP*, *PDF*, *PHP*, *PL*, *PNG*, *TXT*, *XLS*, *XLSM*). Todos os ficheiros estavam previamente classificados como confiáveis ou não confiáveis. Na Tabela III é possível

visualizar o número de artefactos divididos por formato de dados, para cada tipo foram recolhidos artefactos em igual número.

Tabela III – Número de artefactos não maliciosos/maliciosos distribuídos por formato de dados

Extensão	***JS***	***BAT***	***BIN***	***EXE***	***HTML***	***ZIP***	***PDF***	***PHP***	***PL***	***PNG***	***TXT***	***XLS***	***XLSM***
Número Artefactos	12	1	6	3	3	3	3	5	2	8	2	4	1

Tal como referido, os artefactos foram recolhidos de forma aleatória, tendo apenas em atenção a classificação de outos Websites de análise de *malware*. De forma a alargar o espetro de confiança, foram incluídos na análise dois ficheiros de texto (*.txt*) criados especificamente para efeitos de teste, um sem conteúdo e outro com apenas um texto. Tal como esperado, os resultados da análise sobre estes ficheiros concluíram que os artefactos não são maliciosos, assim sendo, a conclusão obtida é que os artefactos criados foram analisados corretamente.

4.5. Submissão de Artefactos

Os artefactos identificados no ponto anterior foram colocados em dois diretórios denominados maliciosos e não maliciosos, após esta separação foram submetidos para análise. Por *sandbox* em análise, os resultados foram catalogados em quatro pontos:

- Tempo desde submissão do artefacto até obtenção do resultado final;
- Precisão no número de deteções (Verdadeiro Positivo quando um artefacto é malicioso e é identificado como malicioso pelos sistemas de *sandbox* e Falso Negativo quando um artefacto não é malicioso e não é identificado como malicioso pelos sistemas de *sandbox*. Os casos intermédios (Verdadeiro Negativo e Falso Positivo são também muito relevantes para a análise);
- Cobertura/casos de teste – cada sistema de *sandbox*, agrega vários sistemas antivírus que dão suporte a análise. Nesse sentido num resultado de 15/57 indica que estiveram envolvidos na análise 57 antivírus, mas apenas 15 detetaram o artefacto como malicioso;
- Média de ameaça – é o resultado do número de deteções a dividir pelos casos de teste.

A análise irá permitir ter uma base de comparação quando forem realizados os testes ao **WE-SAND** e desse modo perceber se o funcionamento está de acordo com o pretendido.

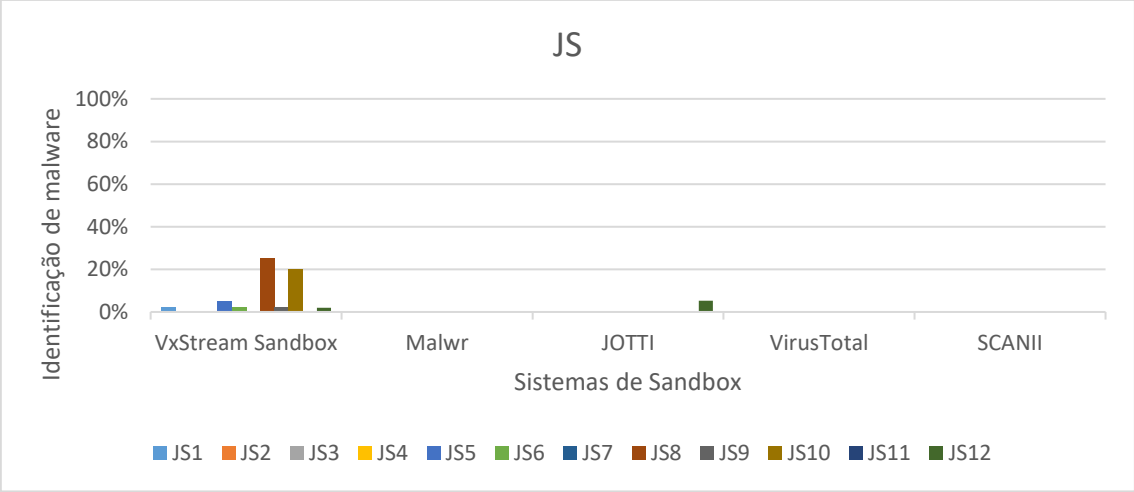
4.6. Resultados Obtidos

Os vários artefactos foram submetidos de forma individual, através do *middleware* desenvolvido em *PHP*. Ressalva-se o caso VxStream Sandbox em que os artefactos foram submetidos através do Website, uma vez que, a versão gratuita não permitia a submissão através do *middleware*. Com base nos resultados foram elaborados os gráficos que serão apresentados nas próximas subsecções.

4.6.1. Artefactos não maliciosos

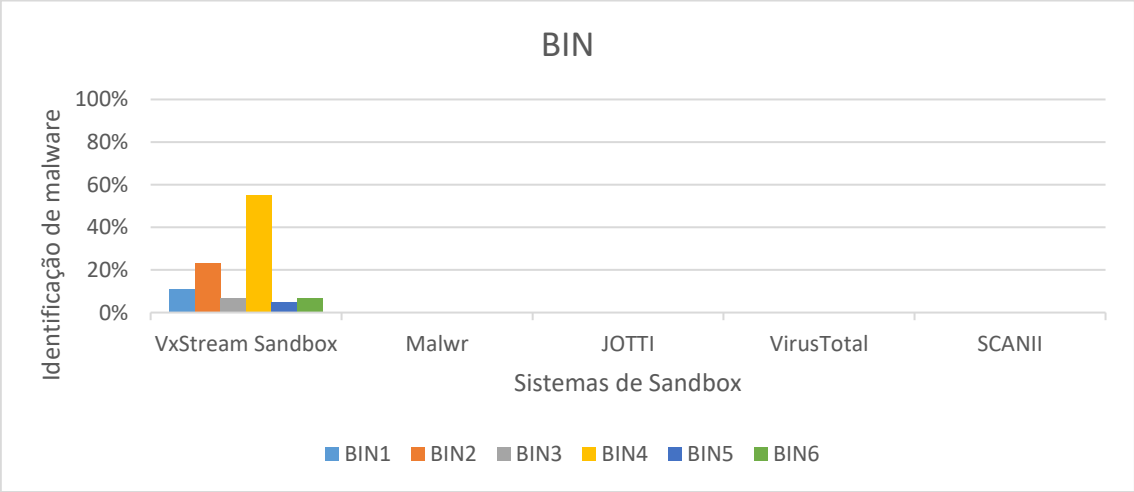
Nesta subsecção são apresentados os gráficos obtidos considerando a análise de artefactos não maliciosos. No Gráfico 1 é apresentado o resultado da análise dos doze ficheiros *JS* (*JavaScript*) não maliciosos. Os ficheiros foram analisados por cinco sistemas de *sandbox*.

Gráfico 1 – Análise ficheiros JS não maliciosos



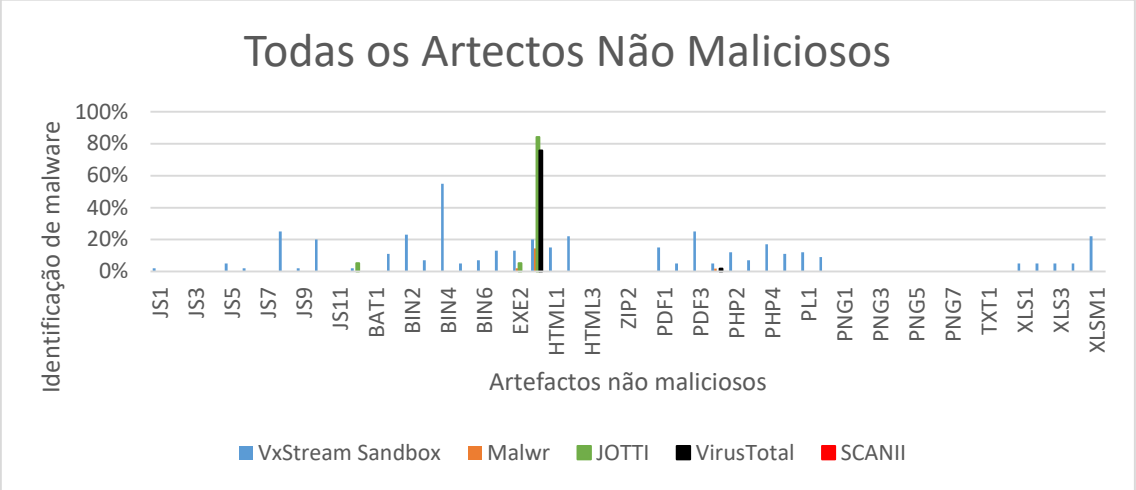
Como se verifica no Gráfico 1, houve dois sistemas de *sandbox* (VxStream e JOTTI) que obtiveram falso positivos (artefacto não malicioso identificado como malicioso), isto é, identificaram os ficheiros como sendo maliciosos quando na realidade não são. No Gráfico 2 são apresentados os resultados obtidos sobre a análise de ficheiros binários (executáveis) não maliciosos.

Gráfico 2 – Análise ficheiros BIN não maliciosos



Como se verifica no Gráfico 2, apenas um sistema de *sandbox* (VxStream), identificou a existência de *malware* em ficheiros binários não maliciosos (falso positivo). Por forma a evitar a apresentação individual por formato de dados, apresentamos no Gráfico 3, o resultado obtido para todos os artefactos não maliciosos.

Gráfico 3 – Comparação de todos os artefactos não maliciosos analisado.

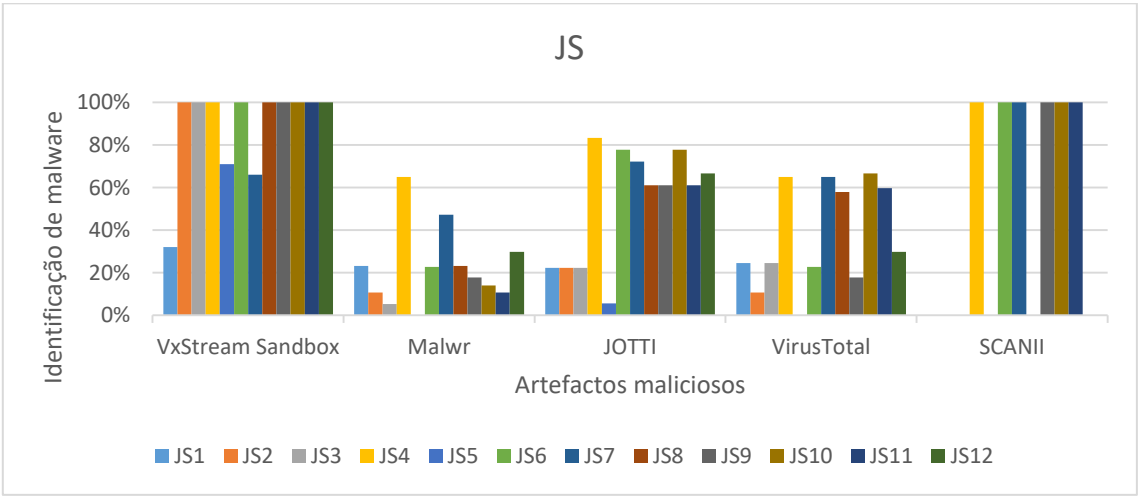


Pela análise do Gráfico 3, pode-se verificar que alguns dos sistemas de *sandbox*, identificaram artefactos como sendo maliciosos quando na realidade não o são. Para cada um dos sistemas de *sandbox*, foi calculada a taxa de falsos positivos, ou seja, artefactos não maliciosos identificados como sendo maliciosos: VxStrem Sandbox – 62%; Malwr – 6%; JOTTI – 6%; VirusTotal – 4%; SCANNI – 0%. Com base nas percentagens anteriores, é possível verificar que o VxStrem Sandbox falha na maioria dos casos.

4.6.2.Artefactos maliciosos

A análise dos verdadeiros positivos foi também verificada. Utilizando o conjunto de artefactos maliciosos, o mesmo foi submetido aos diferentes tipos de *sandbox*. Tal como na subsecção anterior, no Gráfico 4 são apresentados os resultados obtidos, na análise de artefactos maliciosos do tipo *JavaScript*.

Gráfico 4 – Análise ficheiros JS maliciosos

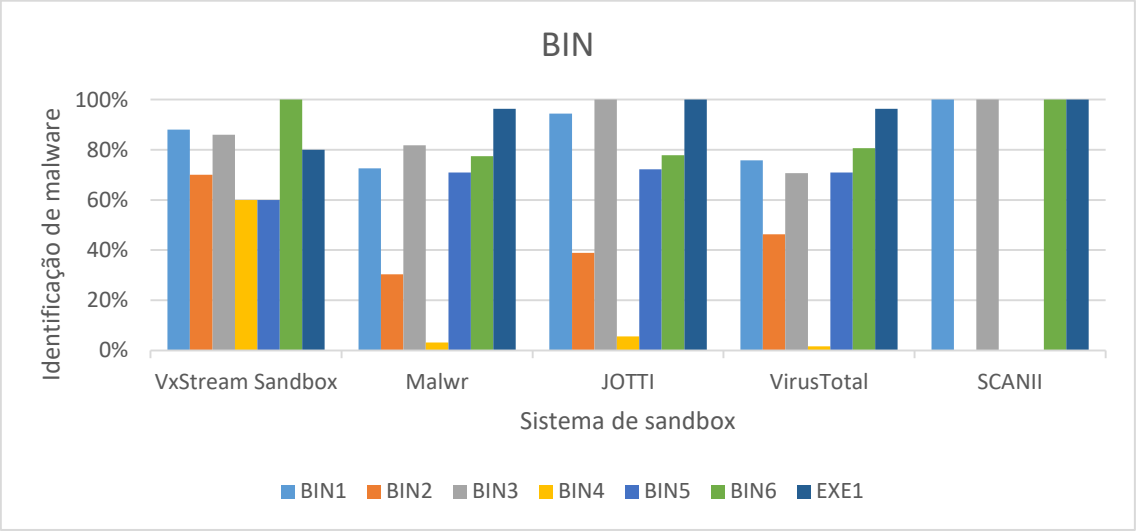


Como se verifica no Gráfico 4, os *JavaScripts* maliciosos foram identificados na sua maioria. Há algumas exceções como a de um ficheiro *JavaScript* (*JS5*) que não foi identificado como malicioso por dois dos sistemas de *sandbox* (Malwr, VirtusTotal). O sistema de *sandbox* SCANII foi incapaz também de detetar dois dos artefactos maliciosos (*JS2* e *JS6*), mas curiosamente todos os que detetou foi com elevado grau de certeza, pois este sistema só devolve dois tipos de resultados, 0 (não malicioso) ou 1 (malicioso), este utiliza vários mecanismos (antivírus) de análise, para devolver um resultado de forma mais assertiva. Os melhores resultados de deteção foram obtidos pelo VxStream e os menos bons pelo Malwr.

No Gráfico 5 são apresentados os resultados para a análise de artefactos maliciosos sob a forma de ficheiros binários, estes ficheiros têm a extensão *.bin* não sendo possível decifrar, sem recurso a ferramentas, o tipo de ficheiro nele contido. Todos são denominados como no exemplo seguinte:

- 590e2bebcf2bd87f76ca80379763abdfa838f98f13231aecfb2642d6c5ecaf4b.bin

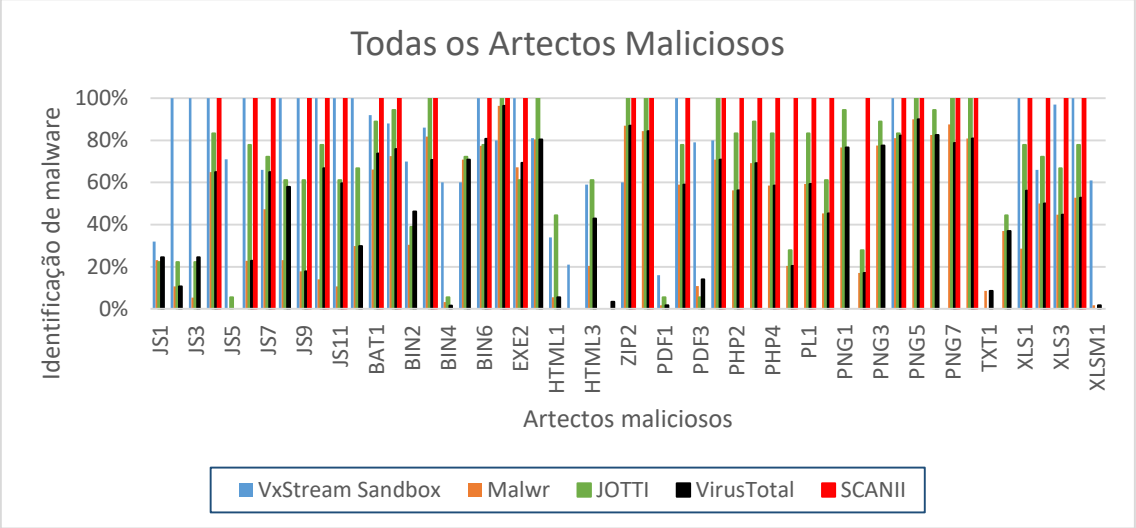
***Gráfico 5 – Análise ficheiros BIN* maliciosos**



Como se verifica no Gráfico 5 a maioria dos sistemas identificou os artefactos como sendo maliciosos. De todos apenas o SCANII falhou a deteção de dois dos sete artefactos. O sistema de *sandbox* com maior grau de confiança foi o VxStream, identificando em todos os casos como sendo maliciosos, o menos assertivo foi o SCANII.

No Gráfico 6 é apresentado o resultado da análise dos vários artefactos.

Gráfico 6 – Comparação de todos os artefactos maliciosos analisados



Como se verifica no Gráfico 6 a maioria dos sistemas identificou os artefactos como sendo maliciosos. É possível verificar que o mais assertivo é o SCANNI e o menos é o VxStream Sandbox. Este é o melhor a detetar ficheiros conhecidos dada a sua base de conhecimento, no entanto, falha na análise de novos ficheiros. Para cada um dos sistemas de *sandbox*, foi identificada uma taxa de verdadeiros negativos, ou seja, artefactos maliciosos identificados como sendo não maliciosos pelos sistemas de *sandbox*: VxStrem Sandbox – 32%; Malwr – 6%; JOTTI – 8%; VirusTotal – 4%; SCANNI – 42%. Com base nas percentagens anteriores é possível concluir que o VxStrem Sandbox e o SCANNI são os sistemas de *sandbox* que mais falharam na identificação de ficheiros maliciosos.

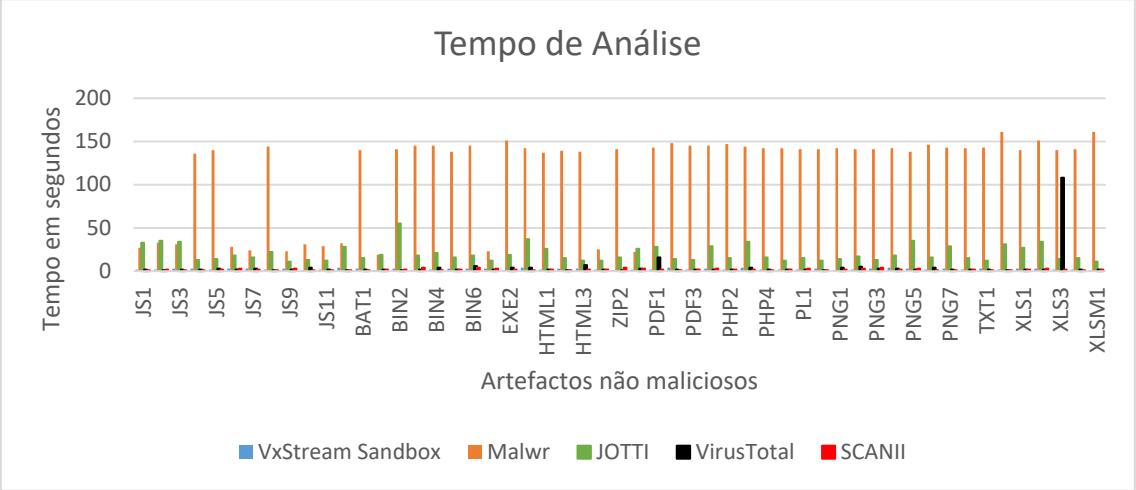
4.7.Análise sobre o tempo de análise

No que diz respeito ao tempo de análise, este varia mediante o tipo de sistema. Nas próximas subsecções é possível visualizar o tempo de análise para cada um dos sistemas.

4.7.1.Artefactos não maliciosos

O Gráfico 7 apresenta o tempo de análise em segundos, para os vários artefactos. O tempo de análise variou entre 1 segundo e 250 segundos.

Gráfico 7 – Comparação de todos os artefactos não maliciosos - tempo de análise



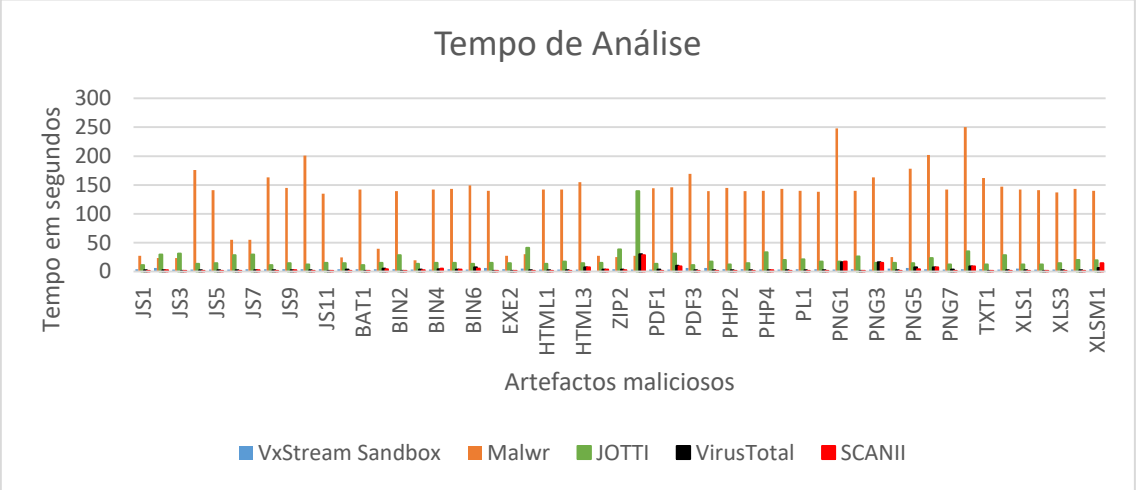
No Gráfico 7 é possível verificar que o sistema Malwr se destacou pela negativa no que diz respeito ao tempo de execução. Para cada um dos sistemas de *sandbox*, é apresentado o somatório total do tempo de análise: VxStrem Sandbox – 129 segundos; Malwr – 6079 segundos; JOTTI – 1075 segundos; VirusTotal – 249 segundos; SCANNI – 104 segundos.

Face aos resultados é possível concluir que o sistema mais rápido é o VxStrem Sandbox, esta rapidez deve-se a base de conhecimento que possuí, o mais demorado a analisar os artefactos é o Malwr.

4.7.2.Artefactos maliciosos

O Gráfico 8 apresenta o tempo de análise em segundos, para os vários artefactos. O tempo de análise variou entre 1 segundo e 142 segundos.

Gráfico 8 – Comparação de todos os artefactos maliciosos - tempo de análise



Para cada um dos sistemas de *sandbox*, é apresentado o somatório total do tempo de análise: VxStrem Sandbox – 199 segundos; Malwr – 6459 segundos; JOTTI – 1100 segundos; VirusTotal – 204 segundos; SCANNI – 175 segundos.

Face aos resultados é possível concluir que o sistema mais rápido é o SCANNI e o mais demorado a analisar os artefactos é o Malwr. Pelos resultados também se verifica que o tempo não vária de forma significativa face ao tipo de artefacto, evidenciado que é o tempo que o próprio sistema necessita para a análise independentemente do tipo de dados.

4.8. Discussão dos Resultados Obtidos

De forma a comprar os falsos positivos/verdadeiros negativos, é apresenta a Tabela IV, que permite visualizar a comparação de todos os sistemas. Esta análise, face aos resultados obtidos, revela o nível de confiança que se poderá ter em cada um dos sistemas de *sandbox.*

As percentagens foram construídas mediante o estado, de detetou ou não detetou como sendo malicioso. A tabela divide-se em duas vertentes distintas:

- Não Maliciosos (Falsos Positivos) – artefactos não maliciosos que foram identificados como maliciosos;
- Maliciosos (Verdadeiros Negativos) – artefactos maliciosos que foram identificados como sendo não maliciosos.

Tabela IV – Comparação entre os falsos positivos/verdadeiros negativos

Sistema de *SandBox*	**Não Maliciosos (Falsos Positivos)**	**Maliciosos (Verdadeiros Negativos)**
VxStrem Sandbox	62%	32%
Malwr	6%	6%
JOTTI	6%	8%
VirusTotal	4%	4%
SCANNI	0%	42%

Com base nas percentagens obtidas é possível concluir que o VxStrem Sandbox é o sistema com maior taxa de erro, e o SCANNI o melhor a analisar ficheiros não maliciosos. Por fim o Malwr, JOTTI e VirusTotal são os mais consistentes na análise, isto é, os que apresentam menor taxa de erro.

Na Tabela V é apresentado o tempo total de análise em segundos, para o conjunto de artefactos.

Tabela V – Comparação entre os falsos positivos/verdadeiros negativos – tempo de análise

Sistema de SandBox	**Não Maliciosos (tempo segundos)**	**Maliciosos (tempo segundos)**
VxStrem Sandbox	129	199
Malwr	6079	6459
JOTTI	1075	1100
VirusTotal	249	204
SCANNI	104	175

Através da Tabela V é possível concluir que o sistema mais demorado na análise é o Malwr e o menos é o SCANNI. No entanto, durante o processo de análise o que se demostrou mais estabilidade em termos de tempo de análise foi o VirusTotal e o SCANII.

Os resultados obtidos através deste estudo comparativo são muito importantes para a criação do **WE-SAND**. Com o **WE-SAND** será possível agregar todos os resultados num só sistema, associando um grau de confiança combinado, que será determinado em função dos resultados obtidos nesta análise, e desse modo obter uma análise mais conclusiva. Perante os resultados obtidos, caberá ao gestor do sistema decidir se pretender entrevir e evitar a evolução de um possível ataque.

Capítulo 5 – Implementação do Sistema WE-SAND

Neste capítulo descrevemos a implementação do protótipo **WE-SAND**. Relembra-se que o principal objetivo do protótipo é integrar um conjunto de sistemas de *sandbox* que facilitem a análise de artefactos, classificando-os como sendo maliciosos ou não maliciosos.

5.1. Estudo de implementação

Para a implementação do protótipo foram utilizadas várias ferramentas, para segmentar o caminho a seguir. Numa fase inicial, foram realizados experimentos baseados nas ferramentas apresentadas na Figura 17. Estes experimentos consistiram na instalação de dois sistemas virtuais que recolhiam os *logs* gerados pelo acesso a Internet dos utilizadores via *browser*.

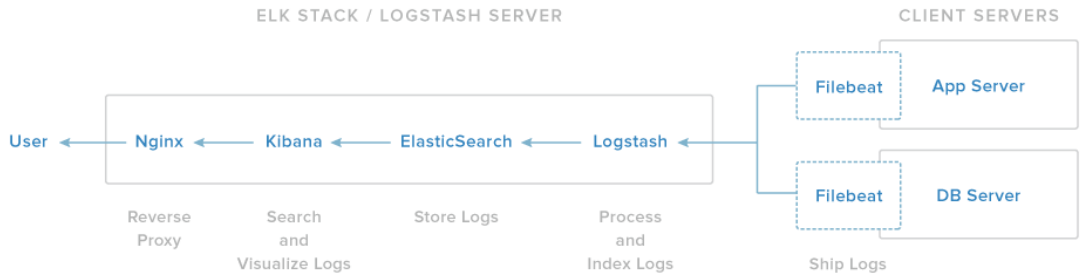


Figura 17 – Ferramentas para recolha e análise de logs [69]

Após análise dos *logs*, foi verificado que existem inúmeros ficheiros acedidos/descarregados através da visita a um Website, assim sendo, prosseguiu-se com a pesquisa de modo a entender os problemas que os ficheiros poderiam trazer. Nos ficheiros descarregados verificou-se a existência de conteúdo malicioso, que poderia ser a porta para um ataque, estes ficheiros foram descarregados de Websites classificados como maliciosos.

Considerando os sistemas de *sandbox* identificados e analisados (conforme poderá ser consultado em capítulos anteriores), deu-se início à implementação do **WE-SAND**.

5.2. WE-SAND Arquitetura do Sistema

A arquitetura geral do **WE-SAND** é ilustrada na Figura 20. O WE-SAND recolhe artefactos e envia-os para análise. Estes artefactos são divididos em duas categorias: ficheiros e *logs* de comunicação Web. Durante a configuração do sistema o utilizador tem que indicar a localização dos artefactos (repositório) e o tipo de categoria (ficheiros/*logs*). Seguidamente o sistema recolhe os artefactos e submete-os aos vários sistemas de *sandbox*. Cada artefacto é identificado por uma chave *MD5* que resulta do seu próprio conteúdo. Esta chave *MD5* é verificada e caso a mesma exista na base de dados,

o sistema devolve o resultado a partir da base de dados, sem comunicar com os sistemas de *sandbox*. Na situação anterior o utilizador é alertado que a comunicação não foi realizada, e caso pretenda, pode configurar o sistema para enviar os artefactos para os vários sistemas de *sandbox*.

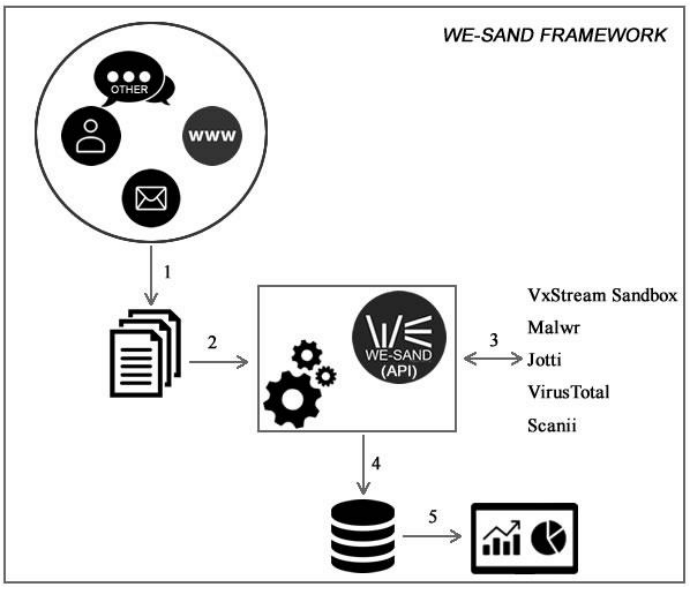


Figura 18 – Arquitetura protótipo WE-SAND

Os números apresentados na Figura 18 representam os passos desde da recolha dos artefactos até ao resultado final.

- 1 / 2 – Recolha de artefactos;
- 3 – Submissão para análise e recolha de resultados;
- 4 – Guardar os resultados na base de dados;
- 5 – *Interface* com apresentação de resultados.

Após conclusão do ciclo anterior o utilizador do sistema consegue identificar se determinado artefacto é ou não confiável e desse modo entrevir numa fase inicial do ataque.

5.3. Desenvolvimento do Sistema

O protótipo **WE-SAND** foi desenvolvido em *PHP*. Esta camada de *software* permite a comunicação entre os vários sistemas de *sandbox*, base de dados, e interface de utilizador. Para cada *sandbox*, foi necessário desenvolver um conetor que permite a integração entre a aplicação desenvolvida e os sistemas de *sandbox*. Na VxStream Sandbox e Malwr foi utilizado o conetor fornecido pelos próprios sistemas (desenvolvido em Python) sendo apenas necessário proceder a ligeiros ajustes. Para os restantes casos o conetor foi desenvolvido de raiz.

5.3.1. Exemplo de conetor de comunicação

Para que seja possível comunicar com os vários sistemas de *sandbox*, foi necessário o desenvolvimento de classes de comunicação. Estas classes são obrigatórias e não foi possível criar uma classe única para comunicar com todos os sistemas, pois, cada um deles tem especificações e formas diferentes de receber dados e devolver resultados.

Na Listagem 3 é apresentada a classe de comunicação com o VirusTotal. Na figura podemos visualizar a declaração e inicialização de variáveis e o método construtor, este apenas recebe a chave de acesso a *sandbox*.

```
1.  (...)
2.  class Connect_VirusTotal {
3.
4.      private $url_api_basic = 'https://www.virustotal.com/vtapi/v2/';
5.      private $apikey = null;
6.      private $url_file_scan = 'file/scan';
7.      private $url_file_report = 'file/report';
8.      private $start_date = null;
9.      private $url_url_scan = 'url/scan';
10.     private $url_url_report = 'url/report';
11.
12.     function __construct($apikey) {
13.         $this->apikey = $apikey;
14.     }
15. (...)
```

Listagem 3 – Declaração de variáveis classe VirusTotal

A função ilustrada na Listagem 4 recolhe um determinado artefacto e envia para o servidor do VirusTotal.

```
1.  (...)
2.  public function set_send_files($file_path, $all_result) {
3.          try {
4.              $this->start_date = date('Y-m-d H:i:s');
5.              $file_name_with_full_path = realpath($file_path);
6.              $api_key = getenv('VT_API_KEY') ? getenv('VT_API_KEY') : $this->apikey;
```

```
7.            $cfile = curl_file_create($file_name_with_full_path);
8.            $post = array('apikey' => $api_key, 'file' => $cfile);
9.            $ch = $this->set_ch($post, True, $this->url_file_scan);
10.           $result = curl_exec($ch);
11.
12.           $status_code = curl_getinfo($ch, CURLINFO_HTTP_CODE);
13.           if ($status_code == 200) { // OK
14.              $js = json_decode($result, true);
15.              if ($all_result == true) {
16.                 return $js;
17.              } else {
18.                 return $js['resource'];
19.              }
20.           } else { // Error occured
21.              return $result;
22.           }
23.           curl_close($ch);
24.        } catch (Exception $exc) {
25.           return $exc->getTraceAsString();
26.        }
27.     }
28. (...)
```

Listagem 4 – Função de envio de ficheiros para o VirusTotal

A função apresentada na Listagem 5 permite recuperar os resultados processados no servidor de análise.

```
1.  (...)
2.  public function get_report($resource, $all_result, $type_report, $scans) {
3.        try {
4.           $url_report = $type_report == 'file' ? $this->url_file_report : $this->url_url_report;
5.           $post = array('apikey' => $this->apikey, 'resource' => $resource);
6.           $ch = $this->set_ch($post, True, $url_report);
7.           $result = curl_exec($ch);
8.           $status_code = curl_getinfo($ch, CURLINFO_HTTP_CODE);
9.           if ($status_code == 200) { // OK
```

```
10.             $js = json_decode($result, true);
11.             if ($all_result == true) {
12.                 return $js;
13.             } else {
14.                 $md5_file = $type_report == 'file' ? $js['md5'] : '';
15.                 $end_date = date('Y-m-d H:i:s');
16.                 //Time between orders in minutes
17.                 $diff_minutes = round((strtotime($end_date) - strtotime($this->start_date)), 2);
18.                 $scans = $scans == true ? $js['scans'] : count($js['scans']);
19.                 return array('error' => 'false', 'md5' => $md5_file, 'resource' => $js['resource'], 's
    tart_date' => $this-
    >start_date, 'end_date' => $end_date, 'time_minutes' => $diff_minutes, 'scan_date' => $js['sca
    n_date'], 'scans' => $scans, 'positives' => $js['positives'], 'total' => $js['total']);
20.             }
21.         } else { // Error occured
22.             return array('error' => 'No results. Order limit exceeded.');
23.         }
24.         curl_close($ch);
25.     } catch (Exception $exc) {
26.         return $exc->getTraceAsString();
27.     }
28. }
29. (...)
```

Listagem 5 – Função de recolha de resultados

O exemplo ilustrado na Listagem 6 dá a conhecer a forma de proceder a um pedido de análise. Neste pedido é enviado um artefacto para análise e recolhido o resultado posteriormente.

```
1. (...)
2. $c = new Connect_VirusTotal('api_key');
3. $resource = $c->set_send_files('../Trojan.BAT', false);
4. $a = $c->get_report($resource, false, 'file', false);
5. print_r($a);
6. (...)
```

Listagem 6 – Exemplo de um pedido de análise de ficheiros

Tal como ilustrado na Figura 19, o resultado é devolvido em formato *JSON*. Esta estrutura é adotada em todas as classes de comunicação.

```
(
        [error] => false
        [md5] => b768f77acb3d8da0edb032f62bb8250a
        [resource] => a8d9ac2b317e69d6eab1f3dee1d312e34439c31f4478bad590abcdab344144a2
        [start_date] => 2017-03-26 15:10:16
        [end_date] => 2017-03-26 15:10:18
        [time_seconds] => 2
        [scan_date] => 2017-03-26 13:38:07
        [scans] => 56
        [positives] => 37
        [total] => 56
)
```

Figura 19 – Exemplo de um resultado de análise de ficheiros

Os sistemas de análise existentes também possibilitam a análise para além de ficheiros, na Listagem 7 é ilustrada uma função que foi criada para submeter *URLs* a análise.

```
1.  (...)
2.  public function set_send_url($scan_url, $all_result) {
3.      try {
4.          $this->start_date = date('Y-m-d H:i:s');
5.          $post = array('apikey' => $this->apikey, 'url' => $scan_url);
6.          $ch = $this->set_ch($post, True, $this->url_url_scan);
7.          $result = curl_exec($ch);
8.          $status_code = curl_getinfo($ch, CURLINFO_HTTP_CODE);
9.          if ($status_code == 200) { // OK
10.             $js = json_decode($result, true);
11.             if ($all_result == true) {
12.                 return $js;
13.             } else {
14.                 return $js['scan_id'];
15.             }
16.         } else { // Error occured
17.             return $result;
18.         }
19.         curl_close($ch);
20.     } catch (Exception $exc) {
21.         return $exc->getTraceAsString();
22.     }
23. }
```

```
24. (...)
```

Listagem 7 – Função de envio de URLs

O excerto de código apresentado na Listagem 8 dá a conhecer a forma de utilização da função que funciona em conjunto com a função ilustrada na Listagem 7.

```
1. (...)
2. $c = new Connect_VirusTotal('api_key');
3. $resource = $c->set_send_url('http://www.we-sand.com', false);
4. $a = $c->get_report($resource, false, 'url', false);
5. print_r($a);
6. (...)
```

Listagem 8 – Exemplo de um pedido de análise de URLs

Independentemente do tipo de análise ou formato do artefacto o resultado obtido é apresentado sempre no mesmo formato. Esta uniformização (Figura 20) facilita a análise e tratamento dos dados.

```
(
        [error] => false
        [md5] => d97a6e259dbb3dd481d54193b3177e36
        [resource] => dd78384dec2526081601787122a44b2a8a5990334042f1a0f1bcf865cce6e4fe-1490536874
        [start_date] => 2017-03-26 15:10:18
        [end_date] => 2017-03-26 15:10:22
        [time_seconds] => 4
        [scan_date] => 2017-03-26 14:01:14
        [scans] => 64
        [positives] => 0
        [total] => 64
)
```

Figura 20 – Exemplo de um resultado de análise de ficheiros

Todas as classes desenvolvidas tem uma estrutura semelhante, as mesmas apenas variam nas funções que as compõem. A Listagem 9 ilustra uma função apenas disponivel na *sandbox* SCANII, esta função tem como objetivo verificar se a *sandbox* está operacional antes de enviar os artefactos para análise.

```
1. (...)
2.  public function get_ping() {
3.      try {
4.          $ch = curl_init();
5.          curl_setopt($ch, CURLOPT_URL, $this->url_api_basic . $this->url_ping);
6.          curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
```

```
7.          curl_setopt($ch, CURLOPT_USERPWD, $this->apikey . ":" . $this->apipass);
8.          $result = curl_exec($ch);
9.          if (curl_errno($ch)) {
10.             return 'Error:' . curl_error($ch);
11.         } else {
12.             return $result;
13.         }
14.         curl_close($ch);
15.     } catch (Exception $exc) {
16.         return $exc->getTraceAsString();
17.     }
18. }
19. (...)
```

Listagem 9 – Classe de teste à disponibilidade do sistema SCANII

5.3.2. Desenho da Base de Dados

A base de dados de suporte ao sistema foi desenvolvida num *SGBD* relacional e mais especificamente em *MySQL*. O esquema de base de dados, apresentado na Figura 21 é constituido por 5 entidades: "*users*", "*access_logs*", "*system_setup*", "*sandbox_system*", "*data_analyzed*".

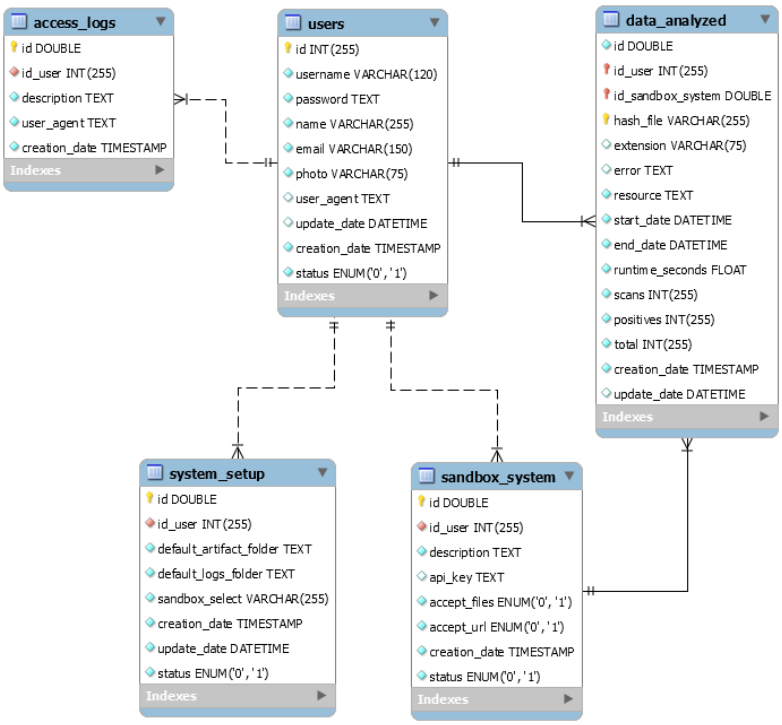


Figura 21 – Base de dados do protótipo WE-SAND.

Cada entidade tem funções específicas no suporte do sistema. Apresentamos de seguida as especificações das entidades:

- *users* – utilizadores registados no sistema, o campo "*user_agent*" permite guardar o último registo de acesso. O campo "*status*" permite definir se um dado utilizador tem ou não acesso ao sistema;
- *access_logs* – permite guardar informação sobre os acessos realizados ao sistema;
- *system_setup* – pré-configurações definidas para um utilizador, tais como "*default_artifact_folder*" e "*default_logs_folder*" pastas com os artefactos/*logs* para análise. Também existe o campo "*sandbox_select*" que permite guardar os últimos sistemas de *sandbox* utilizados por utilizador que opera o sistema;
- *sandbox_system* – nesta entidade é possivel guardar todos os sistemas de *sandbox* que podem ser utilizados e respetivas chaves de acesso "*api_key*". Outra configuração é se o sistema de *sandbox* suporta o carregamento e análise de ficheiros "*accept_files*" e *URLs* "*accept_url*";
- *data_analyzed* – nesta entidade são guardados os resultados da análise. Os resultados são únicos por utilizador, *sandbox* e *hash* do ficheiro, caso, o mesmo ficheiro seja submetido mais que uma vez o resultado da análise é atualizado. O resultado é constituido por:
    - *hash_file* – valor *hash* do artefacto em *MD5*;
    - *extension* – formato de dados do artefacto;
    - *error* – mensagem de erro quando obtida pelo sistema de *sandbox*;
    - *resource* – identificador do artefacto no sistema de *sandbox*;
    - *start_date* / *end_date* – data e hora de início e fim da análise;
    - *runtime_seconds* – tempo em segundos até ao resultado final;
    - *scans* – número de ferramentas de análise utilizadas pelo sistema de *sandbox*;
    - *total* – número de ferramentas de análise disponíveis no sistema de *sandbox*;
    - *positives* – número de incidencias de *malware* identificadas.

Neste protótipo foram apenas criadas as entidades e campos necessários para simular um funcionamento aplicacional de nível iniciado. Os utilizadores registados no sistema têm acesso a todos os sistemas de *sandbox* disponíveis, no entanto, apenas conseguem visualizar as próprias análises. Num nível avançado irão ser acrescentadas outras entidades, para permitir a gestão de chaves de *sandbox* associadas a cada utilizador. Outra *feature* será a partilha de análises entre os vários utilizadores do sistema. Esta partilha irá alargar a base de conhecimento, e desse modo, reduzir as ligações com os sistemas de *sandbox*, reduzindo assim o tempo de espera no resultado de análise.

5.4.Interface do Utilizador

A *interface* do utilizador é uma parte crucial no desenvolvimento de um sistema. A Figura 22 apresenta todas as funcionalidades disponíveis após realização do *login*.

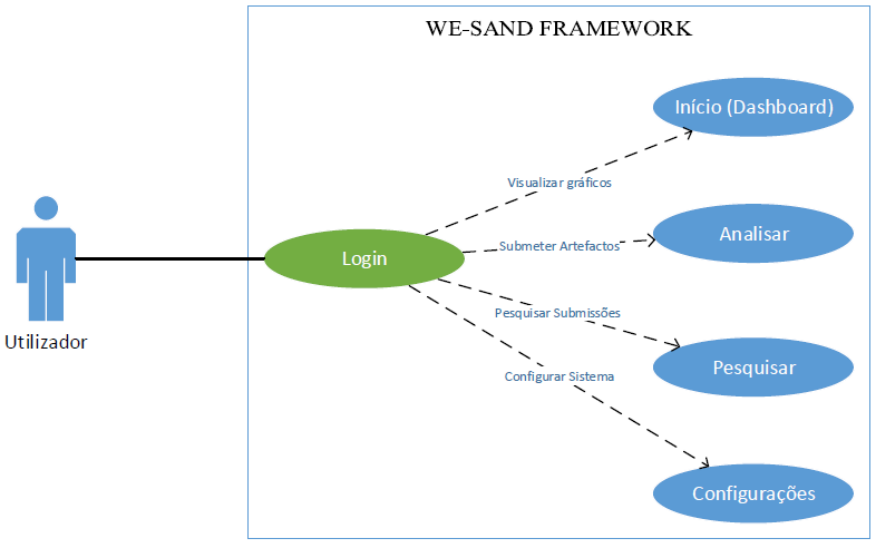


Figura 22 – Funcionalidades associadas ao utilizador com login efetuado

De forma mais detalhada em cada funcionalidade é possível realizar as seguintes operações:

- **Login** – entrada no sistema;
- **Início** – consulta de artefactos submetidos e resumo de análise;
- **Analisar** – seleção do tipo de artefactos e sistemas de *sandbox*;
- **Pesquisar** – pesquisa dos artefactos submetidos para análise;
- **Configurações** – configuração das pastas utilizadas para colocação de artefactos. Seleção dos sistemas de *sandbox* a utilizar na análise.

5.5. Simulação de um acesso ao sistema

Tal como a maioria dos sistemas, o **WE-SAND** possuí um acesso restrito na utilização do sistema, assim sendo, para entrar no sistema o utilizador têm que introduzir as credenciais de acesso. Tal é feito na página principal da aplicação, aqui ilustrada na Figura 23.

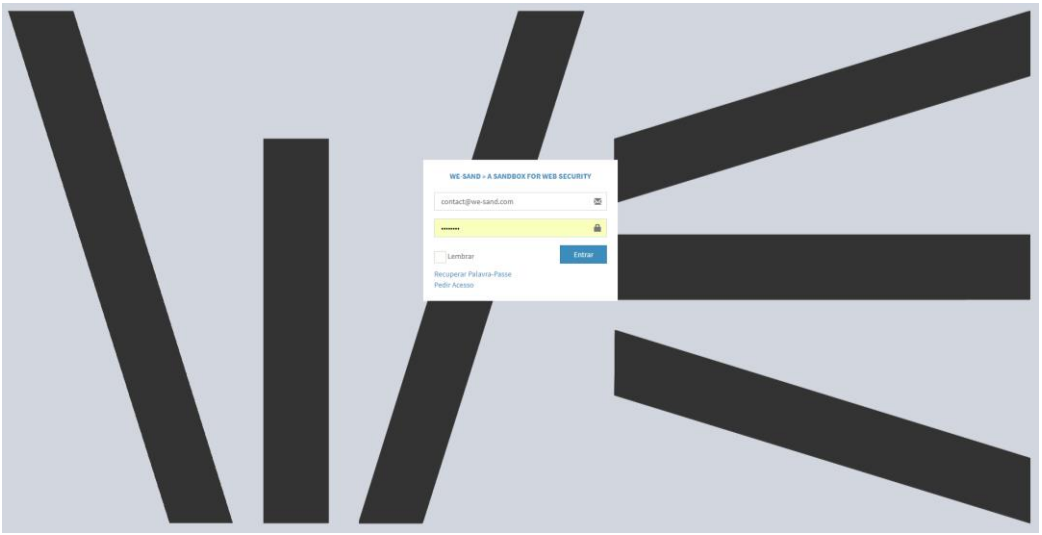


Figura 23 – Acesso ao sistema WE-SAND – página principal para acesso

Após validação das credências o utilizador acede a um *dashboard*, que contem as estatísticas das análises realizadas. As estatísticas estão divididas em categorias:

- **Estatísticas gerais** – análises realizadas a *URLs*/artefactos e o número de conteúdos maliciosos e não malicioso obtido;
- **Artefactos analisados** – número de artefactos submetidos por tipo de ficheiro;
- **Artefactos não maliciosos/maliciosos** – gráfico com o número de artefactos analisados por tipo de ficheiro, onde a cor verde significa conteúdo não malicioso e a cor vermelha conteúdo malicioso.

O *dashboard* é ilustrado na Figura 24.

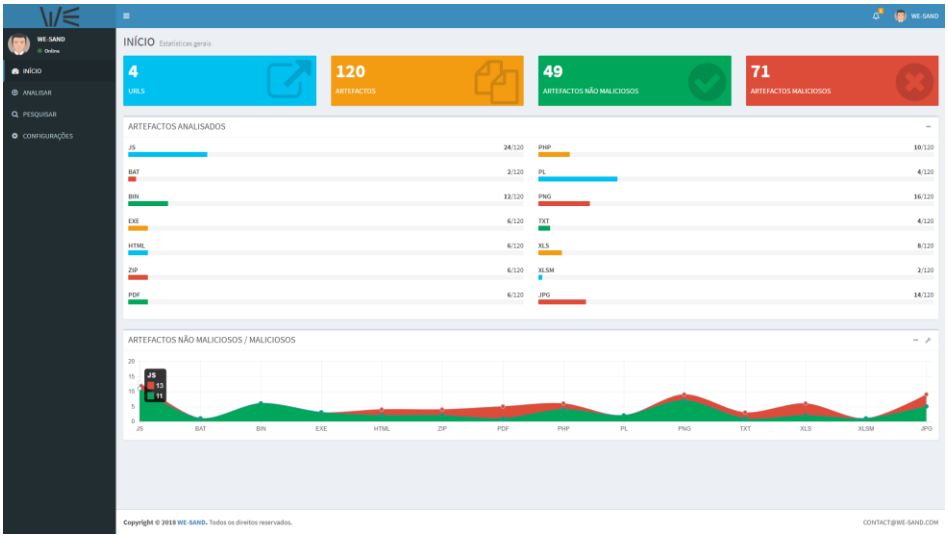


Figura 24 – Dashboard principal do WE-SAND

Tal como referido anteriormente o utilizador tem a possibilidade de pré-configurar o sistema com os caminhos e os sistemas de *sandbox* a serem utilizados na análise. Na Figura 25 é apresentado um exemplo da configuração do sistema.

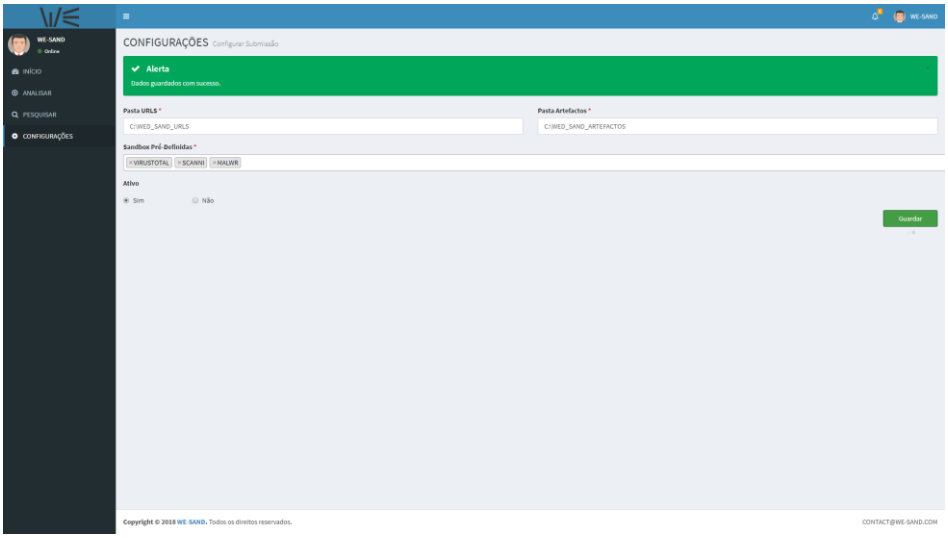


Figura 25 – Configuração do sistema WE-SAND

O utilizador apenas consegue escolher os sistemas de *sandbox* que pretende utilizar e a localização dos ficheiros para análise. Nesta versão do protótipo ainda não é possível ao utilizador, adicionar um novo sistema de *sandbox*. Para adicionar um novo sistema de *sandbox* é necessário criar o conetor que permite a comunicação entre o **WE-SAND** e o sistema de *sandbox* e adicionar um novo registo na entidade "*sandbox_system*".

A Figura 26 ilustra o fluxo necessário para a disponibilização de um novo sistema de *sandbox*. O processo envolve conhecimentos básicos de programação.

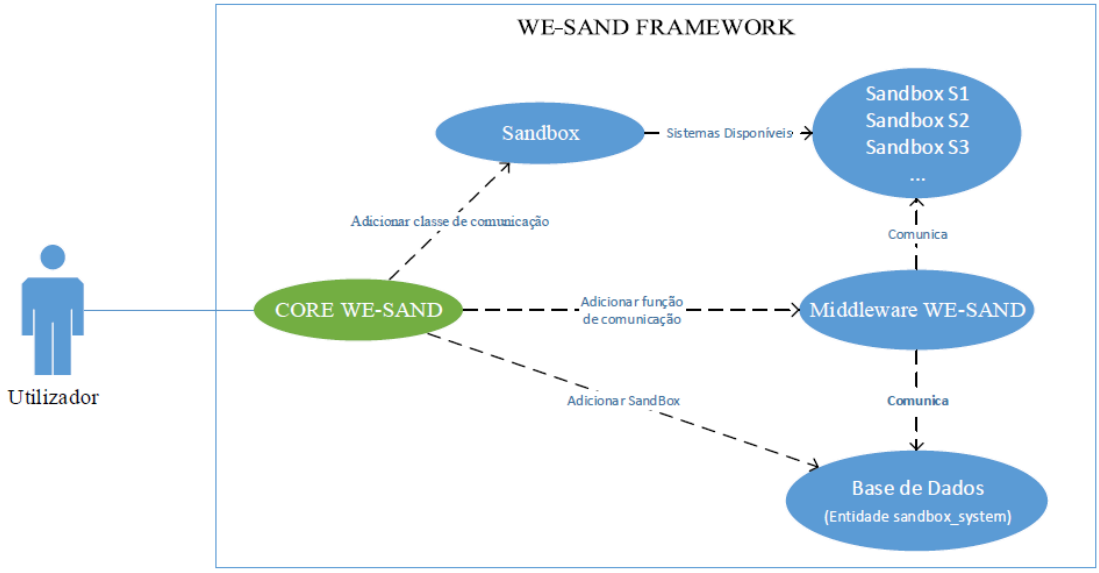


Figura 26 – Inserir novo sistema de sandbox

5.6. Exemplo de submissão

Para iniciar a análise o utilizador tem que realizar a configuração descrita aquando da apresentação da Figura 27, seguidamente seleciona o tipo de artefacto e clica em analisar. Mediante o tipo de artefacto selecionado, são apresentados os sistemas de *sandbox* disponíveis. A Figura 27, Figura 28 e Figura 29 ilustram o processo de análise.

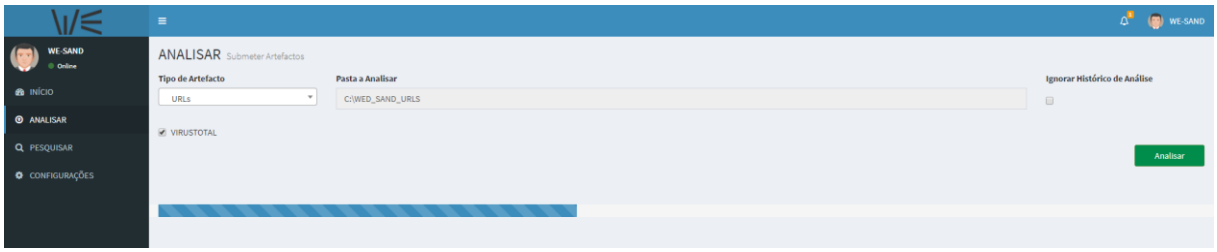


Figura 27 – Análise de URLs

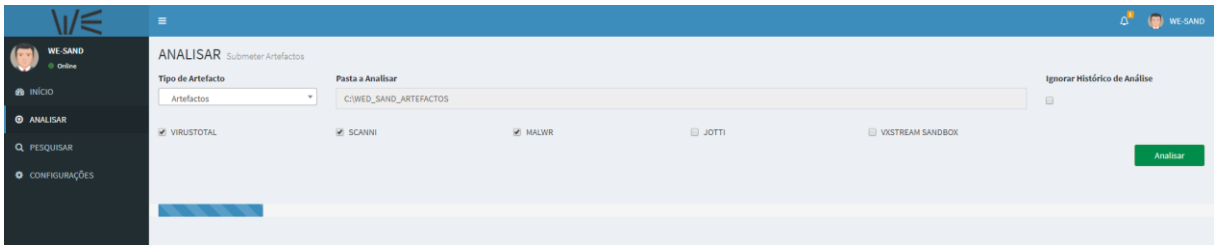


Figura 28 – Analise de artefactos

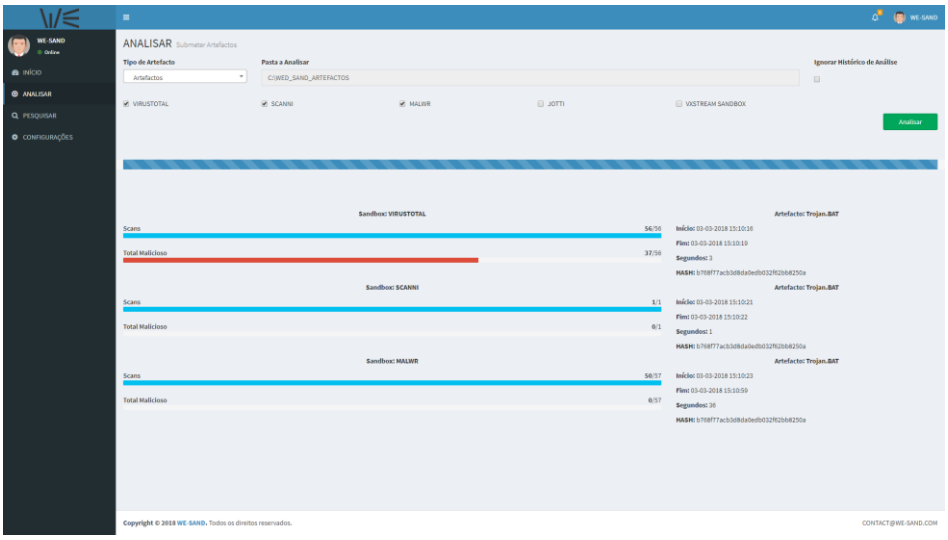


Figura 29 – Resultados obtidos após análise

5.7. Pesquisa de artefactos

Na Figura 30 é apresentado um exemplo de pesquisa na listagem de artefactos submetidos. Quando um artefacto é considerado malicioso o resultado é apresentado a vermelho, caso contrário o resultado aparece a verde.

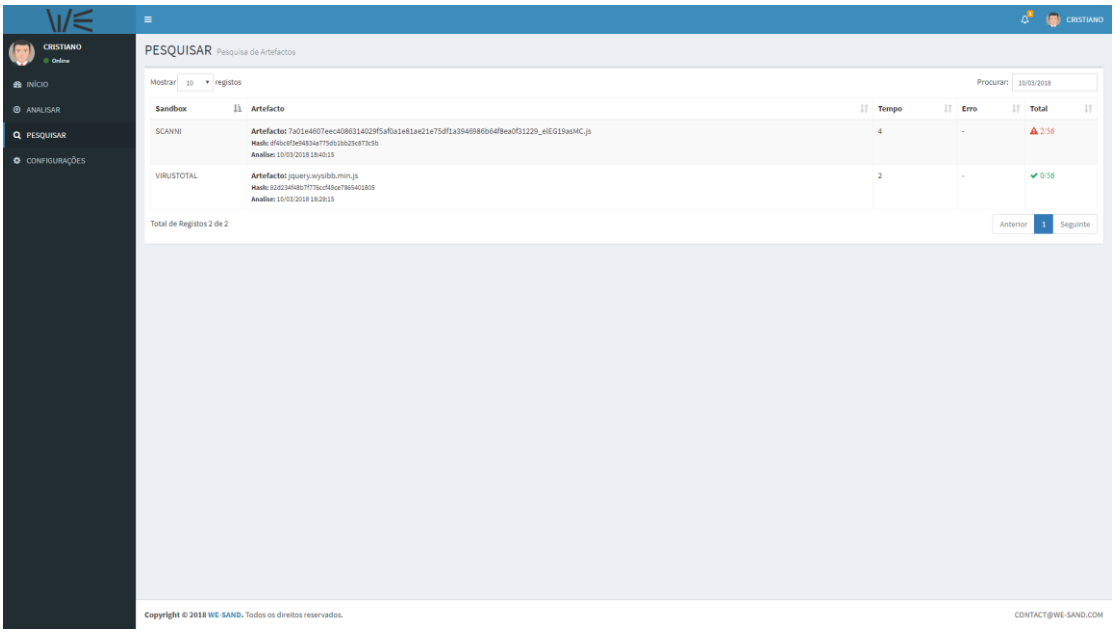


Figura 30 – Pesquisa de artefactos

No sistema apresentado é possível ocultar o descritivo do menu lateral, e dessa forma é aproveitada uma maior área de ecrã para visualização dos conteúdos. Na Figura 31 é visível o espaço extra de visualização quando se oculta o menu.

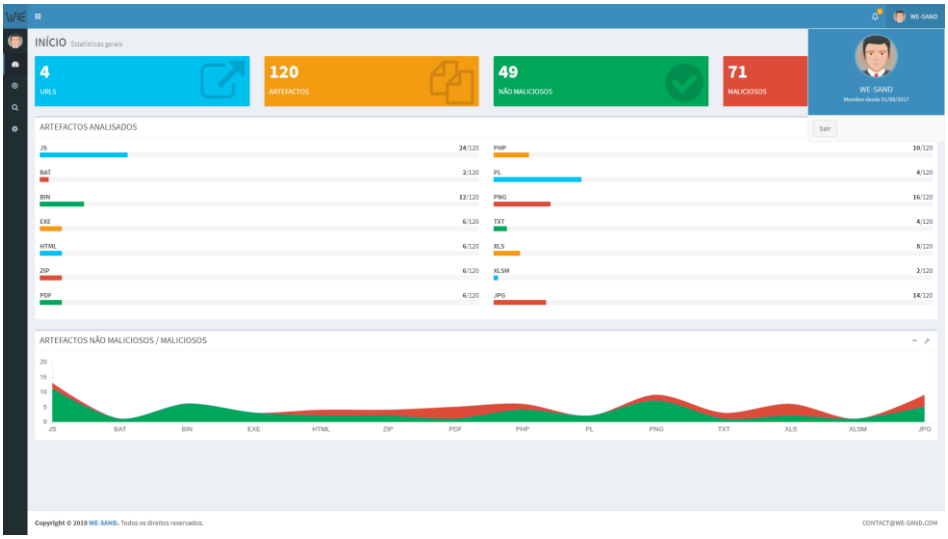


Figura 31 – Ocultar o menu do WE-SAND

5.8. Submissão na sandbox versus WE-SAND

Esta secção pretende apresentar exemplos de submissão de artefactos diretamente nos sistemas de *sandbox* e comparar o resultado obtido com a submissão feita através do **WE-SAND**. Para efeitos de teste, nos exemplos seguintes é possível visualizar uma comparação de artefactos submetidos através do VirusTotal.

Na Figura 32 é ilustrado o resultado obtido com a submissão de um artefacto malicioso no **WE-SAND**. Como se verifica pelo resultado foram obtidos 42 resultados positivos num total de 60 *scans* feitos. O tempo de análise foi de 2 segundos.

a. Exemplo 1 (Trojan.BAT)

```
Array
(
    [error] => false
    [md5] => b768f77acb3d8da0edb032f62bb8250a
    [resource] => a8d9ac2b317e69d6eab1f3dee1d312e34439c31f4478bad590abcdab344144a2
    [start_date] => 2018-05-26 18:36:43
    [end_date] => 2018-05-26 18:36:45
    [time_seconds] => 2
    [scan_date] => 2018-05-26 17:35:37
    [positives] => 42
    [scans] => 60
    [total] => 60
)
```

Figura 32 – Resultado da análise do artefacto submetido através do WE-SAND

Na Figura 33 é ilustrado o resultado obtido com a submissão do mesmo artefacto malicioso, mas desta vez diretamente no VirusTotal. Como se verifica pelo resultado foram obtidos 42 resultados positivos com base em 60 analisadores. O tempo de análise foi de 2 segundos, tal como na análise via **WE-SAND**.

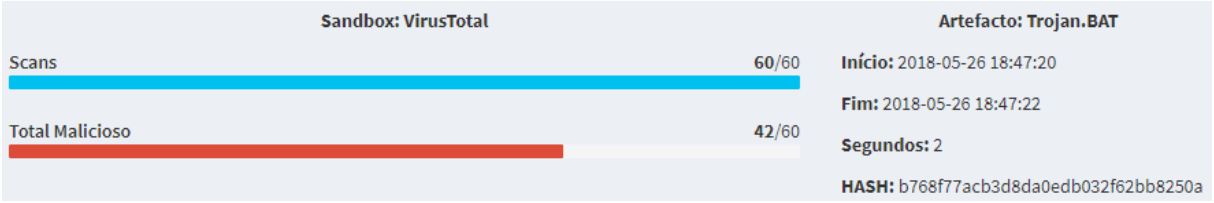


Figura 33 – Resultado da análise do artefacto submetido diretamente na sandbox

Na Figura 34 é ilustrado o resultado obtido com a submissão de um artefacto não malicioso no **WE-SAND**. Como se verifica pelo resultado foram obtidos 2 resultados positivos num total de 59 *scans* feitos. O tempo de análise foi de 3 segundos.

b. Exemplo 2 (1.ZIP)

```
Array
(
    [error] => false
    [md5] => f4b4fec76aa738c47fe1cb5a24796de1
    [resource] => d6797324b7441f0d3719db901ece816b4478ba589c2fcccdd2efaf787ff9d4a1
    [start_date] => 2018-05-26 18:51:56
    [end_date] => 2018-05-26 18:51:59
    [time_seconds] => 3
    [scan_date] => 2017-04-17 19:07:24
    [positives] => 2
    [scans] => 59
    [total] => 59
)
```

Figura 34 – Resultado da análise do artefacto submetido através do WE-SAND

Na Figura 35 é ilustrado o resultado obtido com a submissão do mesmo artefacto não malicioso, mas desta vez diretamente no VirusTotal. Como se verifica pelo resultado foram obtidos 2 resultados positivos num total de 59 *scans*. O tempo de análise foi de 3 segundos, tal como na análise via **WE-SAND**.

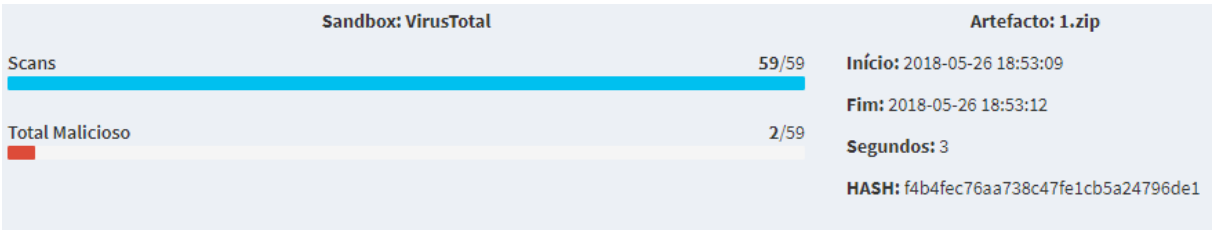


Figura 35 – Resultado da análise do artefacto submetido diretamente na sandbox

Finalmente o caso c), aqui ilustrado na Figura 36, corresponde aos resultados obtidos com a submissão de um artefacto não malicioso no **WE-SAND**. Como se verifica pelo resultado foram obtidos 0 resultados positivos num total de 60 *scans* feitos. O tempo de análise foi de 2 segundos.

c. Exemplo 2 (we_sand.php)

```
Array
(
    [error] => false
    [md5] => 3630966dbdd46b785f8408c5f3ae32b7
    [resource] => 434fe3efa5bd55a1a94bff86e7b9646c3251c64a38d6ae8dd41a4b40b413621d
    [start_date] => 2018-05-26 19:10:09
    [end_date] => 2018-05-26 19:10:11
    [time_seconds] => 2
    [scan_date] => 2018-05-26 18:02:23
    [positives] => 0
    [scans] => 60
    [total] => 60
)
```

Figura 36 – Resultado da análise do artefacto submetido através do WE-SAND

Na Figura 37 é ilustrado o resultado obtido com a submissão do mesmo artefacto não malicioso, mas desta vez diretamente no VirusTotal. Como se verifica pelo resultado foram obtidos 0 resultados positivos num total de 60 *scans*. O tempo de análise foi 1 segundo.

Figura 37 – Resultado da análise do artefacto submetido diretamente na sandbox

As figuras anteriores representam alguns exemplos testados, em todos eles os resultados são muito semelhantes. Por vezes o tempo de análise pode variar, está variação depende de alguns fatores tais como: ligação à Internet e estabilidade de processamento do computador.

Capítulo 6 – Conclusão

Desenvolver um protótipo de análise de artefactos é uma tarefa extramente complexa, pois, existem inúmeros tipos de *malware* a surgir diariamente. O protótipo denominado de **WE-SAND** necessita de uma evolução constante, para aumentar o seu grau de análise e identificação de artefactos maliciosos. Foram estes desafios e a vontade de explorar uma área tão complexa e criativa que deu origem e motivação para o desenvolvimento do protótipo **WE-SAND**.

O número de ciberameaças e de ciberataques, bem como a complexidade destas, eleva os desafios e a necessidade de os detetar e mitigar em tempo útil. A deteção e mitigação das ciberameaças e dos respetivos ataques não se restringe a uma tecnologia, mas sim a solidificação, normalmente por camadas de tecnologias, processos e envolvendo as pessoas. O **WE-SAND** apresenta-se como mais uma camada tecnológica que esta ao dispor das organizações, mais especificamente das equipas de segurança informática, para identificar ameaças e parar as mesmas o mais rápido possível. Acontece que, existe no mercado um vasto número de tecnologias o que dificulta o processo de análise atempado, na medida em que um profissional terá que recorrer a várias tecnologias para obter expressividade e certeza dos resultados obtidos. Neste âmbito o **WE-SAND** ao integrar várias tecnologias de *sandbox* facilita o processo de análise e deteção de ameaças contidas em artefactos que as empresas lidam no dia-a-dia, nomeadamente no que diz respeito ao acesso a Websites por parte da organização que possam resultar em acessos menos seguros e aos ficheiros que entram na empresa por vias como o *e-mail*.

Algo fundamental para a implementação do **WE-SAND**, e que não foi encontrado no estado da arte, foram estudos sobre a eficácia dos vários tipos de sistemas de *sandbox*. Para o efeito conduziu-se um trabalho experimental que consistiu na verificação da eficácia de cada sistema de *sandbox* perante a existência de artefactos maliciosos e não maliciosos e ainda a verificação do tempo necessário a cada análise. Os resultados obtidos evidenciam a existência de verdadeiros negativos, isto é, a não deteção de artefactos maliciosos quando na verdade são maliciosos e ainda a existência de falsos positivos, ou seja, na identificação por parte dos sistemas de *sandbox* de artefactos como sendo maliciosos quando na verdade não o são. Conhecer a eficácia destes sistemas é extremamente relevante e permite o fornecimento de resultados combinados às equipas de segurança com base no grau de confiança de cada sistema de *sandbox*.

Os resultados obtidos sustentam a escolha de um conjunto de sistemas de *sandbox* que compõe o protótipo **WE-SAND**. O protótipo encontra-se desenvolvido e operacional. Dispõe de uma *interface* Web para o envio de ficheiros ou *URLs* para análise e permite visualizar os resultados obtidos através de um *dashboard*. O protótipo poderá desde já ser utilizado em organizações, referindo que durante um período inicial se deverão cruzar os resultados obtidos para obter um maior nível de confiança na solução.

6.1.Agradecimentos no âmbito do trabalho

Durante a implementação e integração com as várias *sandbox*, surgiram algumas dificuldades. A principal dificuldade foi perceber como submeter os artefactos através da *API* e como uniformizar os resultados obtidos após análise. No caso da *sandbox* VxStream Sandbox, foi necessário entrar em contacto com os desenvolvedores, pois a *API* tinha algumas limitações de utilização, infelizmente segundo o senhor Jan Miller a disponibilização de todas as funcionalidades além de ser paga é apenas permitida para empresas. Na sandbox JOTTI foi necessário pedir a chave de acesso para utilização da *API*. Após dificuldades na implementação recebi a ajuda do senhor Jordi Bosveld, representante da JOTTI. Quero desde já agradecer (professores, amigos, familiares) a disponibilidade e ajuda para ultrapassar as dificuldades encontradas.

6.2.Trabalho Futuro

Um dos pontos de trabalho futuro passa pelo evoluir do protótipo ao ponto de ter uma aplicação preparada para integrar nos sistemas de segurança das empresas. Outro aspeto que poderá ser trabalhado é a dimensão e profundidade dos testes a realizar e o modo de funcionamento inline.

Bibliografia

[1] Symantec, “015 Internet Security Threat Report,” *Internet Secur. Threat Rep.*, vol. 20, no. April, p. 119, 2015.

[2] HeartSim, “State of the Phish,” *ThreatSim*, p. 24, 2015.

[3] Sucuri, “Sucuri Website Security | Google Blacklisted | Website Blacklist Removal.” [Online]. Available: https://sucuri.net/website-security/google-blacklisted-my-website. [Accessed: 07-Nov-2015].

[4] C. F. Cybersecurity, “Protecting Websites from Attack with Secure Delivery Networks,” *Cover Featur. CyberSecurity*, p. 9, 2014.

[5] E. El País, “Los dispositivos de Apple sufren el mayor ciberataque de su historia,” 23-Sep-2015. [Online]. Available: http://tecnologia.elpais.com/tecnologia/2015/09/21/actualidad/1442823415_104653.html. [Accessed: 12-Dec-2015].

[6] WhatsApp, “WhatsApp hack attack puts 200,000 at risk.” [Online]. Available: http://www.cnbc.com/2015/09/09/whatsapp-hack-attack-puts-200000-at-risk.html. [Accessed: 10-Dec-2015].

[7] T-Mobile, “15m T-Mobile consumers hacked: SSN and more taken - SlashGear.” [Online]. Available: http://www.slashgear.com/15m-t-mobile-customers-hacked-ssn-and-more-taken-01407526/. [Accessed: 12-Dec-2015].

[8] Observador, “Ataques informáticos. 23% das empresas portuguesas foram alvo - Observador.” [Online]. Available: http://observador.pt/2015/10/29/ataques-informaticos-23-das-empresas-portuguesas-foram-alvo/. [Accessed: 12-Dec-2015].

[9] E. U. A. for L. E. Cooperation, “Mastermind behind EUR 1 billion cyber bank robbery arrested in Spain | Europol.” [Online]. Available: https://www.europol.europa.eu/newsroom/news/mastermind-behind-eur-1-billion-cyber-bank-robbery-arrested-in-spain. [Accessed: 24-May-2018].

[10] T. S. Darya Gudkova, Maria Vergelis, Nadezhda Demidova, “Spam: features of the quarter.” [Online]. Available: https://securelist.com/analysis/quarterly-spam-reports/74682/spam-and-phishing-in-q1-2016/. [Accessed: 09-Jun-2016].

[11] J. Bettencourt, “Kaspersky Lab Spam and Phishing report: FIFA 2018 and Bitcoin among 2017’s most luring topics | Kaspersky Lab US.” [Online]. Available: https://usa.kaspersky.com/about/press-releases/2018_fifa-2018-and-bitcoin-among-2017-most-luring-topics. [Accessed: 24-Feb-2018].

[12] P. Advisor, “15 best security software antivirus of 2015/2016 UK - Test Centre - PC Advisor.” [Online]. Available: http://www.pcadvisor.co.uk/test-centre/software/best-security-software-2015-2016-uk-3265701/. [Accessed: 15-Nov-2015].

[13] Kaspersky, “Kaspersky Security Intelligence Services | Kaspersky Lab PT.” [Online]. Available: http://www.kaspersky.com/pt/enterprise-security/intelligence-services. [Accessed: 29-May-2016].

[14] K. Y, “KILEO ON CYBERSECURITY: CYBER ATTACKS THREAT CENTRAL BANKS AROUND THE WORLD.” [Online]. Available: http://ykileo.blogspot.pt/2016/05/cyber-attacks-threat-central-banks.html. [Accessed: 29-May-2016].

[15] Cyberark, “Lessons Learned from the Bangladesh Bank Heist - CyberArk.” [Online]. Available: http://www.cyberark.com/blog/lessons-learned-bangladesh-bank-heist/. [Accessed: 29-May-2016].

[16] C. Baylon, R. Brunt, and D. Livingstone, “Cyber Security at Civil Nuclear Facilities Understanding the Risks,” *Chatham House*, p. 53, 2015.

[17] Globo, “G1 - Saiba como age o vírus que invadiu usinas nucleares no Irã e na Índia - notícias em Tecnologia e Games.” [Online]. Available: http://g1.globo.com/tecnologia/noticia/2010/10/saiba-como-age-o-virus-que-invadiu-usinas-nucleares-no-ira-e-na-india.html. [Accessed: 25-May-2018].

[18] Hackmageddon, “16-30 April 2016 Cyber Attacks Timeline – HACKMAGEDDON.” [Online]. Available: http://www.hackmageddon.com/2016/05/23/16-31-april-2016-cyber-attacks-timeline/.

[19] Hackmageddon, “1-15 April 2016 Cyber Attacks Timeline – HACKMAGEDDON.” [Online]. Available: http://www.hackmageddon.com/2016/05/09/1-15-april-2016-cyber-attacks-timeline/.

[20] Paolo Passeri, “December 2017 Cyber Attacks Statistics – HACKMAGEDDON.” [Online]. Available: http://www.hackmageddon.com/2018/01/04/december-2017-cyber-attacks-statistics/. [Accessed: 24-Feb-2018].

[21] APAV, “APAV Cibercrime.” [Online]. Available: http://apav.pt/cibercrime/. [Accessed: 26-May-2018].

[22] BitDefender, “Malware History,” *BitDefender*, p. 71, 2010.

[23] P. Vaz, “Sabe o que são Rootkits? Saiba como detectar - Pplware.” [Online]. Available: http://pplware.sapo.pt/internet/sabe-sao-rootkits-saiba-detectar/. [Accessed: 04-Jun-2016].

[24] S. Caixa Geral de Depósitos, “Segurança e Fraude.” [Online]. Available: https://www.cgd.pt/ajuda/Seguranca/Pages/Phishing.aspx. [Accessed: 04-Jun-2016].

[25] P. Security, “Panda Security | PandaLabs Report Q2 2015,” *PandaLabs Rep. Q2 2015*, vol. April - Ju, p. 23, 2015.

[26] Somdev Sangwan, “Malware: Definição, Tipos e Exemplos - Ultimate Hackers.” [Online]. Available: https://teamultimate.in/malware-definition-types-examples/. [Accessed: 24-Feb-2018].

[27] H. Engine, “Brief History of,” *Pixels*, vol. 288, no. 5471, pp. 5–32, 2015.

[28] G. DATA, “A new malware strain was discovered every 4.2 seconds in Q1 2017,” 2018. [Online]. Available: https://www.gdatasoftware.com/news/2017/04/29692-a-new-malware-strain-was-discovered-every-4-2-seconds-in-q1-2017. [Accessed: 24-Feb-2018].

[29] Dyfed Loesche, "• Gráfico: o Ransomware faz parte da crescente ameaça de malware | Statista." [Online]. Available: https://www.statista.com/chart/10045/new-malware-specimen-and-share-of-windows-based-malware/. [Accessed: 24-Feb-2018].

[30] Integrity, "ISO 27001." [Online]. Available: https://www.27001.pt/index.html. [Accessed: 04-Jun-2016].

[31] Sans, "Checklists & Step-by-Step Guides | SCORE | SANS Institute." [Online]. Available: https://www.sans.org/score/checklists/iso-17799-2005. [Accessed: 04-Jun-2016].

[32] Ponemon, "New Ponemon Study on Malware Detection & Prevention Released." [Online]. Available: http://www.ponemon.org/blog/new-ponemon-study-on-malware-detection-prevention-released. [Accessed: 29-May-2016].

[33] Cyberark, "Video: The Cyber Attack Lifecycle - CyberArk." [Online]. Available: http://www.cyberark.com/blog/video-the-cyber-attack-lifecycle/. [Accessed: 29-May-2016].

[34] A. D. Correia, "O Combate ao Cibercrime: Anarquia e Ordem no Ciberespaço." [Online]. Available: https://www.revistamilitar.pt/artigo.php?art_id=854. [Accessed: 04-Jun-2016].

[35] N. Milošević, "History of malware," *arXiv Prepr. arXiv1302.5392*, pp. 1–11, 2013.

[36] Eric Vanderburg, "The evolution of a cybercrime: A timeline of ransomware advances - Security Thinking Cap." [Online]. Available: https://securitythinkingcap.com/evolution-cybercrime-timeline-ransomware-advances/. [Accessed: 24-Feb-2018].

[37] C. Copyright, "Common Cyber Attacks: Reducing The Impact," 2015.

[38] Kaspersky, "Ataques dirigidos de vírus informáticos | Internet Security Threats | Kaspersky Lab." [Online]. Available: http://www.kaspersky.com/pt/internet-security-center/threats/targeted-virus-attacks. [Accessed: 05-Jun-2016].

[39] FireEye, "TeslaCrypt: Following the Money Trail and Learning the Human Costs of Ransomware « Threat Research Blog | FireEye Inc," *2016*. [Online]. Available: https://www.fireeye.com/blog/threat-research/2015/05/teslacrypt_followin.html. [Accessed: 26-May-2016].

[40] SemVirus, "Removendo o CryptoWall 4.0 (Instruções para remoção de vírus)," 2016. [Online]. Available: http://semvirus.pt/cryptowall-4-0/. [Accessed: 26-May-2016].

[41] Pcmedic, "CTB Locker | Novo Ransomware." [Online]. Available: http://www.pcmedic.pt/ctb-locker-novo-ransomware/. [Accessed: 26-May-2016].

[42] Pcrisk, "Vírus TorLocker - como remover?" [Online]. Available: https://www.pcrisk.pt/guias-de-remocao/7857-torlocker-virus. [Accessed: 26-May-2016].

[43] kaspersky, "Jogadores são alvo do novo TeslaCrypt Ransomware | Nós usamos PALAVRAS para salvar o mundo | Blog Oficial da Kaspersky Brasil." [Online]. Available: https://blog.kaspersky.com.br/jogadores-sao-alvo-do-novo-teslacrypt-ransomware/4991/#. [Accessed: 26-May-2016].

[44] Hasherezade, "Inside Chimera Ransomware - o primeiro 'doxingware' em estado selvagem - Malwarebytes Labs | Malwarebytes Labs." [Online]. Available:

https://blog.malwarebytes.com/threat-analysis/2015/12/inside-chimera-ransomware-the-first-doxingware-in-wild/. [Accessed: 24-Feb-2018].

[45] J. Friedman, C. Copyright, S. Security, and S. Security, “Attack Your Attack Surface,” no. March, 2016.

[46] Sophos, “Anatomy of a Drive-by-download,” 2016.

[47] M. Cova, C. Kruegel, and G. Vigna, “Detection and analysis of drive-by-download attacks and malicious JavaScript code,” *Proc. 19th Int. Conf. World wide web - WWW ’10*, p. 281, 2010.

[48] M. Vasilescu, L. Gheorghe, and N. Tapus, “Practical malware analysis based on sandboxing,” *2014 RoEduNet Conf. 13th Ed. Netw. Educ. Res. Jt. Event RENAM 8th Conf.*, pp. 1–6, 2014.

[49] K. Yoshioka, Y. Hosobuchi, T. Orii, and T. Matsumoto, “Your Sandbox is Blinded: Impact of Decoy Injection to Public Malware Analysis Systems,” *J. Inf. Process.*, vol. 19, pp. 153–168, 2011.

[50] C. Foundation, “Automated Malware Analysis - Cuckoo Sandbox.” [Online]. Available: https://cuckoosandbox.org/about.html. [Accessed: 10-Jan-2016].

[51] T. Expert, “ThreatExpert - Automated Threat Analysis.” [Online]. Available: http://www.threatexpert.com/. [Accessed: 10-Jan-2016].

[52] J. Security, “Automated Malware Analysis - Joe Sandbox Cloud.” [Online]. Available: http://www.joesecurity.org/joe-sandbox-cloud. [Accessed: 10-Jan-2016].

[53] Trac, “Capture-HPC.” [Online]. Available: https://projects.honeynet.org/capture-hpc. [Accessed: 09-Jan-2016].

[54] JSDetox, “Javascript malware analysis.” [Online]. Available: http://www.relentless-coding.com/projects/jsdetox/.

[55] Malwr, “Malwr - Malware Analysis by Cuckoo Sandbox.” [Online]. Available: https://malwr.com/about/.

[56] VirusTotal, “VirusTotal - Free Online Virus, Malware and URL Scanner.” [Online]. Available: https://www.virustotal.com/.

[57] Payload-Security, “Automated Malware Analysis - VxStream Sandbox - Payload-Security.com - Home.” [Online]. Available: https://www.payload-security.com/. [Accessed: 14-May-2017].

[58] Jotti, “Jotti’s malware scan.” [Online]. Available: https://virusscan.jotti.org/en-US. [Accessed: 14-May-2017].

[59] “scanii.com.” [Online]. Available: https://support.scanii.com/. [Accessed: 20-Apr-2018].

[60] U. Bayer, A. Moser, C. Kruegel, E. Kirda, U. Bayer, A. Moser, C. Kruegel, ·. E. Kirda, and E. Kirda, “Dynamic analysis of malicious code,” *J Comput Virol*, vol. 2, pp. 67–77, 2006.

[61] B. Sayed, I. Traore, and A. Abdelhalim, “Detection and mitigation of malicious JavaScript using information flow control,” *2014 Twelfth Annu. Int. Conf. Privacy, Secur. Trust*, pp. 264–273, 2014.

[62] K. R. Kishore, M. Mallesh, G. Jyostna, P. R. L. Eswari, and S. S. Sarma, “Browser JS Guard: Detects and defends against Malicious JavaScript injection based drive by download attacks,” *Fifth Int. Conf. Appl. Digit. Inf. Web Technol. (ICADIWT 2014)*, pp. 92–100, 2014.

[63] H.-C. Chen, “Secure multicast key protocol for electronic mail systems with providing perfect forward secrecy,” *Secur. Commun. Networks*, vol. 2, no. May 2012, pp. 100–107, 2013.

[64] ANDRA ZAHARIA, “JavaScript Malware – a Growing Trend Explained for Everyday Users.” [Online]. Available: https://heimdalsecurity.com/blog/javascript-malware-explained/. [Accessed: 24-Feb-2018].

[65] M. Fraiwan, R. Al-Salman, N. Khasawneh, and S. Conrad, “Analysis and Identification of Malicious JavaScript Code,” *Inf. Secur. J. A Glob. Perspect.*, vol. 21, no. 1, pp. 1–11, 2012.

[66] J. Jung, H. Kim, S. Yoon, I. Incidents, R. Architecture, K. Internet, and S. Agency, “M ALICIOUS W EB S CRIPT - BASED C YBER A TTACK P ROTECTION T ECHNOLOGY,” no. Korea, 2014.

[67] I. Simão and P. Varela, “A Engenharia de Requisitos como processo,” *IET Work. Pap. Ser.*, 2009.

[68] M. R. S. Tomás, “Métodos ágeis: características, pontos fortes e fracos e possibilidades de aplicação,” *Rev. Int. Segur. Soc.*, vol. 62, no. 4, pp. 127–129, 2009.

[69] “How To Install Elasticsearch, Logstash, and Kibana (ELK Stack) on Ubuntu 14.04 | DigitalOcean.” [Online]. Available: https://www.digitalocean.com/community/tutorials/how-to-install-elasticsearch-logstash-and-kibana-elk-stack-on-ubuntu-14-04. [Accessed: 22-Jan-2018].

Anexo I

O presente anexo tem como finalidade apresentar os requisitos do protótipo **WE-SAND** e as configurações necessárias para o seu correto funcionamento.

- ***Requisitos do protótipo WE-SAND***
  - Instalação *XAMPP* (Versão 1.8.3 ou superior), ou outro servidor Web;
  - *PHP* (Versão 5.5.15 ou superior);
  - *MySQL* (Versão 5.0.11 ou superior);

- ***Importação da base de dados***
  - Criação da base de dados denominada de "*wesand*";
  - Importação da base de dados presente na pasta "Protótipo **WE-SAND**", denominada de "*bd_wesand*";

- ***Instalação do protótipo WE-SAND***
  - Descompactar o ficheiro "*wesand_api.zip*", presente na pasta "Protótipo **WE-SAND**" e copiar para o servidor Web, no caso do XAMPP pasta "*htdocs*";
  - Configurar a ligação a base de dados - alterar os dados do ficheiro "*db_crud.php*" presente na pasta "*connection*";
  - Deve ficar semelhante a listagem seguinte:

```
1.  (...)
2.   private $dbs = array(
3.            "e_wesand" => array(
4.               "server" => "localhost",
5.               "database" => "wesand",
6.               "username" => "root",
7.               "password" => ""
8.            )
9.         );
10. (...)
```

Listagem 10 – Configuração da ligação a base de dados

- ***Aceder ao Sistema***
  - No *browser* digitar o *link* de acesso (e.g. http://localhost/wesand_api/);

- Aceder com os utilizadores configurados:
    - **Utilizador 1:**
        - *E-mail*: contact@we-sand.com
        - Palavra-Passe: we-sand
    - **Utilizador 2:**
        - *E-mail*: cristiano@we-sand.com
        - Palavra-Passe: we-sand

- ***Outros***
    - Apenas é possível adicionar utilizadores diretamente na base de dados, com uma palavra-passe cifrada em *SHA512*;
    - Para alterar a "*api_key*" de um sistema de *sandbox*, deve aceder a base de dados e realizar a alteração na tabela "*sandbox_system*";