UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

Controlos de Cibersegurança em Ambientes MS Windows de Grandes Empresas: Modelo de Risco para Priorização de Atualizações de Segurança

Gonçalo Lima Tavares Campos dos Reis

Mestrado em Engenharia Informática
Especialização em Arquiteturas, Sistemas e Redes de Computadores

Trabalho de Projeto orientado por:
Prof. Doutor António Casimiro Ferreira da Costa
Eng. José António dos Santos Alegria

2018

Agradecimentos

Em primeiro lugar, gostaria de agradecer ao Professor Doutor António Casimiro, pela sua disponibilidade, orientação e paciência ao longo de todo este projeto. Gostaria de agradecer ao Engenheiro José Alegria, o orientador na Portugal Telecom, pela sua disponibilidaade, apoio e pela confiança que sempre demonstrou ter nas minhas capacidades.

Aos meus orientadores técnicos Ricardo Ramalho e Jorge Silva, por toda a paciência, conhecimentos transmitidos e motivação durante o decorrer deste projeto.

Aos meus colegas de estágio Rodrigo Santos e Fábio Ribeiro pela amizade, aconselhamento e motivação imprescindível ao longo de todo o estágio. Também quero agradecer a toda a equipa do SOC (e DCY em geral) por toda a ajuda prestada e por me terem recebido como se já fizesse parte da equipa desde sempre.

Quero agradecer também aos meus amigos e colegas, em especial ao João Silva e ao Pedro Figueiredo pela disponibilidade e apoio no desenvolvimento deste trabalho.

À minha namorada Mafalda Osório, pela infinita paciência que tem comigo, pela ajuda que me deu durante este projeto, e por estar sempre ao meu lado quando mais preciso.

Por último gostaria de agradecer à minha família, pais e irmão, por me tertem dado as ferramentas e o suporte necessário para o alcance e concretização de todos os meus objetivos.

Resumo

Atualmente, as grandes organizações estão dependentes de amplas redes de computadores para o seu funcionamento. Um dos problemas que estas organizações enfrentam é a facilidade com que as suas redes podem ficar comprometidas, pois basta uma das suas máquinas estar vulnerável para pôr em risco toda a rede. As consequências podem ser consideravelmente graves, como a perda de dados, divulgação de documentos ou perda de confiança por parte dos clientes, entre outras.

De maneira a diminuir a probabilidade da ocorrência de um evento crítico, que possa afetar o normal funcionamento de uma instituição, existem diversas medidas de segurança que devem ser tomadas. Uma delas passa por corrigir vulnerabilidades detetadas o quanto antes e manter atualizadas as assinaturas anti-*malware*, dando especial atenção aos ativos considerados críticos. Esta medida é uma mais-valia para aumentar a segurança do sistema informático de qualquer empresa, dado que permite evitar ataques como o *ransomware WannaCry*, que infetou recentemente milhares de máquinas. As consequências deste ataque consistiram na perda de dados (pois foram encriptados), e no dinheiro gasto em vão, por parte de alguns utilizadores, numa tentativa de reaverem os seus dados.

Deste modo surge a necessidade de existirem modelos para caracterizar o nível de ameaça e risco que determinados ativos representam dentro de uma organização. Com este trabalho pretende-se definir e implementar um destes modelos, com a finalidade de serem produzidos indicadores, visualizações e alarmística sobre os diferentes níveis de criticidade dos servidores do parque informático da Portugal Telecom (PT) e assim melhorar a segurança informática da empresa.

O modelo de risco tem dois objetivos a cumprir: conseguir classificar os servidores da PT que possuem sistema operativo Windows com uma nota que representa o seu nível de risco, e servir como guia para as equipas da PT responsáveis pelo processo de aplicação de atualizações de segurança (aplicação de *patches* e atualização de assinaturas de antivírus). Estas equipas têm uma capacidade limitada, de maneira que apenas podem atualizar um certo número de máquinas de cada vez. O modelo vai ser utilizado pelas equipas para que estas saibam em cada momento quais as máquinas prioritárias que devem ser atualizadas. Para este efeito criámos um indicador que apelidámos de ganho. O ganho é um valor numérico que representa a contribuição de uma máquina na diminuição do risco global do parque informático da PT caso a mesma seja atualizada. Isto significa que as máquinas

com maior ganho (que mais diminuem o risco na sua globalidade) são as consideradas prioritárias na hora de realizar atualizações.

Para realização deste projeto primeiro teve que haver uma investigação inicial sobre os critérios que fazem sentido e que são adequados para classificar os servidores da PT. Para além disso, também temos que realizar uma investigação de maneira a sabermos quais desses critérios se podem utilizar tendo em conta os dados sobre os servidores que a PT consegue disponibilizar. Por último é necessário perceber como conjugar a informação que iremos obter para chegar a uma classificação final de risco.

Palavras-chave: Ciber segurança, Ciberataques, Vulnerabilidades, Malware, Redes de computadores.

Abstract

Today, large organizations rely on wide computer networks for their operation. One of the problems these organizations face it's how easily their networks can be compromised, since it's enough one of their machines being vulnerable to threat the entire network. The consequences can be very serious, such as data loss, disclosure of documents or loss of confidence by customers, among others.

In order to reduce the likelihood of a critical event occurring that could affect the normal functioning of any institution, there are a number of safety measures that must be taken. One of them is to correct detected vulnerabilities as soon as possible and keep anti-malware signatures up to date, paying special attention to critical assets. This measure is an added value to increase the security of any company's computer system, since it allows to avoid attacks like *ransomware WannaCry*, which has recently infected thousands of machines. The consequences of this attack consisted in the loss of data (since it became encrypted), and the money spent in vain, by some users, in an attempt to recover their data.

In this way, there's a need to have models that verify the lvel of threat and risk that certain assets represent within an organization. This work intends to define and implement one of these models, with the purpose of producing indicators, visualizations and alarms about the different levels of criticality of the servers of Portugal Telecom's computer park and thus improve the cyber security of the organization.

The risk model has two objectives: to be able to classify the PT's servers that have Windows operating system with a grade that represents their level of risk, and to serve as a guide for PT's teams that are responsible for the process of applying security updates (patching and updating antivirus signatures). These teams have limited capacity, so they can only update a certain number of machines at a time. The model will be used by these teams so that they know at any moment wich are the machines with higher priority that need to be updated. For this purpose we created an indicator that we called gain. Gain is a numerical value that represents the contribution of a machine in reducing the overall risk of PT's computer park incase it is updated. This means that the machines with the highest gain (which reduce the risk the most) are considered a priority when it comes to making updates.

To carry out this project first there was an initial investigation into the criteria that makes sense and which are appropriate to classify PT's servers. In addition, we also have to conduct an investigation in order to know which of these criteria can be used taking into account the data about the servers that PT can provide. Finally it is necessary to understand how to combine the information that we will obtain to reach a final classification of risk.

Keywords: Cyber security, Cyber attack, Vulnerabilities, Malware, Computer networks.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Motivação

Atualmente, mais do que nunca, as empresas preocupam-se com a cibersegurança dos seus ambientes. Esta preocupação não tem como motivação apenas questões legais, mas também as consequências que um ataque, intrusão ou acesso ilícito com sucesso podem ter na imagem e no negócio da própria empresa. [13]

Uma parte importante de poder evitar certos ataques é corrigir vulnerabilidades detetadas o mais rapidamente possível. Tais correções são efetuadas, por exemplo, através da instalação de atualizações de segurança (*patches*). É essencial garantir que estas sejam instaladas corretamente e atempadamente em todas as máquinas, ou corre-se o risco das mesmas serem infetadas, tal como aconteceu recentemente com milhares de computadores que foram infetados pelo *ransomware WannaCry*. Outra parte importante de poder evitar ataques é conseguir detetar *malware* a partir de assinaturas nos antivírus. Dado que todos os dias surgem milhares de novos *malwares*, é necessário que haja uma constante atualização das mesmas. Qualquer máquina que não atualize as assinaturas arrisca-se a não conseguir detetar certos tipos de *malware*, e como tal, representa uma ameaça para a segurança do ambiente onde se encontra.

Apesar de que qualquer ativo de uma organização que se encontre vulnerável representa um risco para a mesma, cada um tem o seu nível de criticidade face às funções que desempenha. Deste modo, é essencial que existam modelos que relacionem o nível de atualização de cada ativo, no caso deste projeto servidores *Windows*, com o nível de criticidade do mesmo. Dado que o parque da *Portugal Telecom* (PT) é composto por milhares de servidores, a atualização dos mesmos é realizada por partes e nunca todos de uma vez só, até porque muitos são responsáveis por manter serviços críticos da organização. Desta maneira, se existir o relacionamento referido em cima, os técnicos da PT saberão sempre quais os servidores que são mais prioritários.

1.2 Objetivos

Este trabalho tem como objetivo primário conceber e implementar um modelo para caracterizar o nível de risco que determinados ativos representam dentro de uma organização. Como objetivo secundário deve servir de instrumento para as equipas da Portugal Telecom, de modo a tornar mais eficiente o processo de aplicação de *patches* e atualização de definições dos antivírus (definir prioridades entre os ativos). O modelo a definir deverá ser baseado mas não restrito a determinados critérios (conectividades, nível de *patching*, proteção antivírus, serviços expostos) produzindo os indicadores correspondentes. Estes podem depois ser utilizados por outros serviços já existentes de maneira a gerar certo tipo de visualizações e alarmística.

1.3 Contribuições

A contribuição deste projeto está relacionada com a informação que o mesmo vai permitir que seja disponibilizada:

- Nível de risco: O modelo vai produzir informação sobre o nível de risco de cada servidor do parque informático da Portugal Telecom. Com esta informação, os responsáveis pela segurança informática da organização têm sempre uma noção atual do nível de risco dos servidores e podem tomar decisões de acordo com o mesmo.
- Classificação dos servidores em diversos critérios. A nota final de risco dos servidores é baseada num conjunto concreto de critérios. Para além da nota final, o modelo de risco também produz como output a classificação dos servidores em cada um dos critérios usados. Este conjunto de informação que o modelo produz como output vai permitir à organização a elaboração de estratégias específicas a utilizar no processo de *patching*.
- Noção de prioridade. O modelo de risco oferece um indicador que foi denominado como "ganho", e que foi criado para estabelecer a noção de prioridade de intervenção entre os servidores. Este indicador tem como objetivo melhorar a eficiência do processo de *patching* da PT, dado que indica quais os servidores que devem ser atualizados primeiro.

1.4 Estrutura do documento

O documento encontra-se organizado da seguinte forma:

O Capítulo 2 aborda diversos assuntos que estão diretamente relacionados com o projeto e que são essenciais para que o leitor possa compreender o que é apresentado no resto do documento. Neste capítulo também são referidas as ferramentas que direta ou indiretamente estão relacionadas com o projeto. O Capítulo 3 é responsável por explicar

o modelo de risco, desde a definição dos primeiros critérios até à concretização da lógica do algoritmo. O Capítulo 4 explica como é que o modelo foi implementado. Começa por explicar a sua estrutura e de seguida apresenta as diferentes classes desenvolvidas. O Capítulo 5 apresenta as diferentes versões do modelo de risco e explica, para cada uma, o que levou a que a mesma não fosse aceite como versão final. Por último o Capítulo 6 finaliza este documento apresentando algumas ideias que podem ser concretizadas no futuro.

Capítulo 2

Enquadramento teórico

Este projeto tem como objetivo a criação de um modelo de risco que classifica os servidores da PT que possuam o sistema operativo *Windows*, quanto ao risco que estes representam dado um conjunto de critérios. Esta classificação irá permitir que a PT desenhe *heatmaps* e que tenha uma noção permanente do nível de risco do seu vasto parque informático. Para além disso, o output do modelo de risco também irá permitir que a PT desenvolva estratégias de aplicação de *patches* de segurança e atualização de definições de antivírus, baseando-se não só na classificação final de risco de cada servidor como também nas classificações que estes possuem em cada critério utilizado no cálculo da nota final.

Neste capítulo começamos por definir o que é *malware* e apresentamos alguns dos seus principais tipos. É importante que se comece por aqui pois de seguida falamos de um dos maiores ciberataques que já ocorreu e que foi um dos motivos para a elaboração deste projeto, o *ransomware* Wannacry. Depois explicamos o processo de gestão de *patching*, para o qual é necessário classificar os ativos quanto ao risco que representam, e também o processo de gestão de risco. Por último são apresentadas algumas plataformas que foram relevantes, direta ou indiretamente, durante o desenvolvimento deste projeto.

2.1 *Malware*

Malware [3][14] (abreviatura para *software* malicioso) é um tipo de *software* que é criado com o objetivo de se apoderar ou causar danos a um computador ou a uma rede de computadores sem a autorização dos seus utilizadores. Muitas das vezes as vítimas nem se apercebem do que está a acontecer. Existem vários tipos de *malware* que descrevemos de seguida.

2.1.1 *Virus*

Tipo de *malware* que tem a capacidade de conseguir replicar-se sempre que é executado de maneira a infetar outras máquinas ou programas. Alguns vírus utilizam mecanismos

que lhes permitem modificarem-se enquanto se propagam de sistema em sistema com a finalidade de evitarem a deteção. Tipicamente os utilizadores nunca se apercebem que executaram um vírus mas sim outra aplicação. Por exemplo, um agente malicioso pode esconder o código de um vírus no meio dos metadados de uma imagem de maneira a que este seja executado quando a imagem é aberta por um utilizador.

2.1.2 *Trojan horse*

Um *Trojan horse* é um *malware* que tenta passar por *software* legítimo. Normalmente os utilizadores são enganados de alguma forma, por exemplo, através de engenharia social, de maneira a que executem um programa que na verdade é um *Trojan*. Quando ativado este permite que os agentes maliciosos executem ações como espiar as vítimas, roubar dados, criar *backdoors*, etc. Ao contrário dos vírus e *worms* (ver secção 2.1.4) os *Trojans* não se replicam, de maneira a conseguirem passar despercebidos.

2.1.3 *Ransomware*

Este tipo de *malware* ameaça publicar ou bloquear indefinidamente os dados das vítimas a menos que um resgate seja pago. Enquanto que alguns *ransomwares* são simples e bloqueiam o sistema de uma maneira que é possível reverter por alguém com certos conhecimentos, outros são mais avançados e cifram os ficheiros das vítimas, tornando impossível que estes sejam recuperados sem que os agentes maliciosos forneçam a chave para os decifrar. Tipicamente os ataques com *ransomware* são levados a cabo utilizando um *Trojan* que se faz passar por um ficheiro legítimo levando a que os utilizadores façam download ou abram o mesmo quando o recebem num anexo de email.

Wannacry

Wannacry [4][11][16] é o nome dado ao *ranswomware* utilizado num dos maiores ciberataques da história que ocorreu em Maio de 2017 e que teve como alvo os computadores que corriam o sistema operativo *Microsoft Windows*. Os dados de todas as máquinas infetadas foram encriptados e os agentes maliciosos responsáveis pelo ataque exigiram que os utilizadores pagassem uma quantia em *bitcoin* se quisessem receber a chave que supostamente lhes permitiria decifrar os dados.

Este ataque foi possibilitado pela existência de dois *exploits*. O primeiro, *EternalBlue* [4][8], explora uma vulnerabilidade na implementação do protocolo *Server Message Block* (SMB). Este protocolo não consegue lidar corretamente com pacotes especificamente desenhados por parte de agentes maliciosos e permite que os mesmos executem código arbitrário nas máquinas alvo. O segundo, *DoublePulsar* [4][11], é uma *backdoor* que permite a execução e instalação de *malware*. Apesar de ser um *ransomware*, o *Wannacry* também possuía capacidades que normalmente se vêem em *worms*. Depois de

infetar uma máquina iniciava uma procura pela rede de outras máquinas que também se encontrassem vulneráveis.

As vulnerabilidades exploradas para conseguir realizar este ataque já eram conhecidas e inclusive a *Microsoft* já tinha disponibilizado *security patches* de maneira a corrigir tais vulnerabilidades. As máquinas afetadas por este ataque foram aquelas que não tinham aplicados os *security patches* disponibilizados. Dito isto, as proporções que este ataque tomou vieram demonstrar os graves problemas que as organizações a nível mundial têm, quer seja no que toca a utilização de máquinas que correm sistemas operativos obsoletos sem suporte por parte do fabricante, por exemplo *Windows XP*, quer seja pela falta de estratégias de *patching* face aos orçamentos que possuem.

2.1.4 *Worm*

Contrariamente ao que acontece com os vírus os *worms* não necessitam de um hospedeiro (programa infetado) para se conseguirem propagar. Este tipo de *malware* é bastante completo e tem a capacidade de se propagar automaticamente através de redes de computadores explorando as vulnerabilidades de software ou aplicações mal configuradas que existam nas máquinas da rede.

Tradicionalmente os *worms* foram criados apenas para se espalharem. Multiplicavam-se exponencialmente e perturbavam a largura de banda, mas sem alterar funcionalidades dos sistemas. Atualmente muitos *worms* transportam uma *payload* (pedaço de código desenhado para realizar ações maliciosas) de maneira a conseguir por exemplo espalhar um *ransomware* por todas máquinas que conseguirem infetar. [1]

2.2 *Patching*

Os *patches* são um dos instrumentos de segurança informática mais importantes, a par com os antivírus. Ambos têm como principal objetivo manter os sistemas computacionais de uma empresa seguros. Um *patch* é um pequeno pedaço de software que uma empresa disponibiliza de maneira a corrigir vulnerabilidades detetadas nos seus produtos. Traduzindo para português, um *patch* é um remendo, neste caso remove vulnerabilidades impedindo a exploração das mesmas por parte de agentes maliciosos.

Quando os peritos de cibersegurança ou investigadores descobrem uma nova vulnerabilidade, o protocolo que costuma ser seguido é alertar a empresa que desenvolveu o software em questão, de forma a que esta consiga produzir um *patch* o mais rapidamente possível. Normalmente as descobertas não são tornadas públicas. Embora possa parecer contraproducente, dado que o público não terá oportunidade de se proteger contra a exploração da vulnerabilidade, a divulgação da mesma poderia alertar agentes maliciosos sobre a sua existência. Ao avisar a entidade que desenvolveu o software em primeiro lugar, espera-se que esta seja capaz de corrigir a vulnerabilidade antes que mais alguém a

descubra.

2.3 Antivírus

O antivírus é um software que inicialmente foi desenhado para detetar e remover vírus nos computadores. Com o avançar do tempo este software foi melhorado e atualmente consegue prevenir, detetar e remover não só vírus mas também outros tipos de software malicioso(*malware*). De seguida são apresentados os diferentes métodos utilizados pelos antivírus de maneira a lidar com as ameaças de *malware*.

Deteção baseada em *sandbox*

Técnica de deteção baseada no comportamento que consiste em executar programas em ambiente virtual registando as ações que estes realizam. Dependendo das ações realizadas o antivírus classifica os programas segundo sejam maliciosos ou não. Caso não sejam maliciosos os programas são então executados em ambiente real. Ainda que esta técnica seja bastante eficaz é raramente usada, pois é lenta e pesada a nível de consumo de recursos.

Deteção baseada em assinatura

Quando um *malware* chega às mãos de uma empresa que desenvolve antivírus, este é analisado pelos investigadores ou por sistemas de análise dinâmicos. Depois de determinarem que estão perante um *malware*, é extraída uma assinatura do ficheiro do *malware* e é adicionada na base de dados do antivírus que a empresa comercializa. Para escapar à deteção por este método, os autores de *malware* começaram a desenvolver vírus que conseguem cifrar partes deles próprios ou modificarem-se.

Deteção baseada em heurísticas

Depois de ser criado, um vírus pode sofrer mutações ou alterações por parte de agentes maliciosos. Estas alterações fazem com que um vírus possa ter múltiplas variantes.

Denomina-se deteção genérica à deteção e remoção de múltiplas ameaças usando apenas a assinatura de um único vírus. Embora seja vantajoso identificar um vírus específico, pode ser mais rápido detetar uma família de vírus através de uma assinatura genérica ou uma quase exata correspondência com uma assinatura existente. Os investigadores encontram excertos de código em comum que todos os vírus de uma família possuem e assim conseguem criar uma única assinatura genérica. Uma deteção que use este método é apelidada de deteção heurística.

Deteção de *rootkits*

Um *rootkit* é um tipo de *malware* que foi desenhado com o objetivo de ganhar privilégios administrativos num computador sem ser detetado. Os *rootkits* podem alterar o funcionamento do sistema operativo e em alguns casos sabotar o programa de antivírus de maneira a que este seja ineficaz. Os *rootkits* são difíceis de remover, por vezes é necessário a re-instalação do sistema operativo.

Proteção contínua em tempo-real

Os antivírus possuem a capacidade de monitorização em tempo real. Estes monitorizam os computadores à procura de atividades suspeitas enquanto os dados são carregados em memória, por exemplo, quando se insere um CD, ao abrir um email, ao executar um ficheiro, etc.

2.4 Gestão de *patching*

Um *patch* consiste num pedaço de *software* que é utilizado para corrigir diversos problemas como vulnerabilidades de segurança, bugs de *software* ou simplesmente software desatualizado. Vamos falar mais detalhadamente sobre o conceito de *patching* na secção (3.3.1).

Uma grande percentagem de incidentes que são reportados foram causados por uma exploração com sucesso de um número relativamente pequeno de vulnerabilidades em sistemas e aplicações. Por forma a evitar ataques que são levados a cabo a partir de vulnerabilidades conhecidas, as organizações devem assegurar-se que os seus ativos de *IT* têm instalados os últimos *security patches/hot-fixes* disponibilizados. Desta forma surge o processo de gestão de *patching*. Para que este possa ser bem sucedido é necessário que seja um processo robusto e sistemático. O seu ciclo de vida envolve uma série de passos chave: preparação, identificação de vulnerabilidades e aquisição do *patch*, avaliação do risco e priorização, teste do *patch*, implementação e verificação do *patch*.

Preparação

Para este passo é sugerido que em primeiro lugar os administradores do sistema criem e mantenham um registo de inventário, composto por todo o equipamento de *hardware* e pacotes de *software*, em conjunto com as versões dos pacotes mais usados na organização. Devem ser criadas configurações padrão para os maiores grupos de recursos de *IT*, como estações de trabalho e sistemas de ficheiros. Isto vai permitir simplificar o processo de teste e implementação dos *patches*, e como consequência diminuir o tempo do processo global de gestão de *patching*.

Para que o processo de gestão de *patching* seja eficiente é necessária a colaboração e participação dos utilizadores finais da organização. Se estes receberem treino adequado irão poder ocasionalmente implementar eles próprios *patches* que sejam mais simples, o que significa uma redução da carga de trabalho dos administradores do sistema. Para além disso, haver *user awareness* é especialmente importante em organizações que permitem acesso remoto à rede da organização, dado que uma vulnerabilidade explorada através de um computador na casa de um dos trabalhadores representa uma ameaça para a segurança toda a organização.

Identificação de vulnerabilidade e aquisição do *patch*

Existem diversos recursos de informação disponíveis de modo a que os administradores de sistemas possam monitorizar vulnerabilidades e os *patches* que possam aplicar-se, dependendo do *hardware* e *software* que os seus sistemas possuam.

A consulta dos *websites* de vendedores é provavelmente a forma mais direta e fidedigna que os administradores de sistemas têm para conseguir informação relacionada com vulnerabilidades e *patches* sobre produtos específicos. No entanto, por vezes os vendedores não reportam de imediato a existência de novas vulnerabilidades, pois preferem torná-las públicas apenas quando exista um *patch* para as mesmas. Este tipo de informação também pode ser muitas vezes encontrada em *websites* de consultoria de segurança.

De maneira a ajudar os administradores de sistemas a estarem atualizados no que toca à disponibilização de *patches* e *reports* de novas vulnerabilidades, foram criados serviços de alertas que os próprios podem personalizar, para receberem automaticamente notificações com informação sobre novos *patches* e vulnerabilidades descobertas, que afetem especificamente os sistemas pelos quais são responsáveis.

Análise de risco e priorização

Com recursos limitados, os administradores de sistemas muitas vezes têm que priorizar a aplicação de novos *patches*, realizando uma avaliação de risco, para determinar a que sistemas devem ser aplicados primeiro os *patches*. Tradicionalmente a priorização é baseada nos critérios de ameaça, vulnerabilidade e criticidade do sistema. Os administradores devem identificar os riscos associados e ações que têm que ser realizadas quando uma vulnerabilidade de segurança é confirmada, e avaliar o impacto resultante da aplicação do *patch* respetivo.

Teste do *patch*

Este passo é essencial para saber se a aplicação de um *patch* específico vai ou não afetar o funcionamento normal de *software* existente. É importante que os testes decorram num sistema que tenha uma configuração similar ao sistema que se encontra em produção, de

maneira a garantir que a aplicação do *patch* não provoca consequências inesperadas no mesmo.

Para além de identificar problemas inesperados, os testes a *patches* também devem ser utilizados para conseguir ter a certeza que os mesmos corrigiram na totalidade as vulnerabilidades para as quais foram desenvolvidos.

Caso não seja possível aplicar um *patch* devido a resultados negativos dos testes, devem-se implementar controlos de segurança alternativos.

Implementação e verificação do *patch*

Fazer correção de vulnerabilidades num sistema pode ser tão simples como mudar uma definição de configuração, ou pode requerer uma instalação de uma versão completamente nova do *software*.

Antes de aplicar um *patch* os administradores devem fazer um backup do sistema onde vão aplicar o mesmo. Isto possibilita que se possa fazer uma restauração fácil e rápida do sistema, caso o *patch* tenha um impacto inesperado, para o estado em que este se encontrava antes da aplicação do *patch*.

Ferramentas de distribuição e aplicação de *patches*

As organizações podem considerar usar ferramentas automatizadas de gestão de *patching*, de maneira a tornar mais rápido o processo de distribuição e instalação de *patches*. Estas ferramentas podem ser de dois tipos:

- Sistema de gestão de *patching cross-platform* : Este tipo de plataforma consegue lidar com *patches* de diferentes sistemas operativos ou produtos de diferentes vendedores.
- Soluções de gestão de *patching platform specific*: Plataformas que apenas suportam *patches* de vendedores ou sistemas específicos. Um exemplo de uma plataforma deste tipo é o *Windows Server Update Services* (WSUS) que tem como objetivo ajudar os administradores de sistemas a aplicar os últimos *patches* da *Microsoft* apenas para computadores com o sistema operativo *Windows*. Vamos falar mais em detalhe sobre esta plataforma no próximo capítulo.

2.5 Gestão de risco

Risco em segurança de informação é o impacto negativo resultante da exploração de uma vulnerabilidade, tendo em conta a probabilidade e o impacto da ocorrência.

O processo de gestão de risco em segurança de informação [15] deve ser contínuo. Deve estabelecer o contexto externo e interno, avaliar os riscos, e tratar dos mesmos usando um plano de tratamento de risco que implemente as recomendações e decisões.

Este processo pode ser aplicado à organização como um todo, a uma parte da organização (como por exemplo um departamento) ou a qualquer sistema de informação.

O processo de gestão de risco consiste num estabelecimento do contexto, *risk assessment*, tratamento do risco, aceitação do risco, comunicação e consulta do risco, e monitorização e revisão do risco.

Estabelecimento do contexto

Estabelecer o contexto interno e externo da gestão de risco da segurança de informação requer a configuração de critérios básicos necessários, definição do âmbito (ativos a ter em conta) e limites, e estabelecer uma organização apropriada para operar o processo de gestão de risco.

Risk assessment

A tarefa de *risk assessment* quantifica ou descreve qualitativamente o risco e permite que os *managers* priorizem os riscos de acordo com a sua perceção de seriedade ou outros critérios estabelecidos. Esta tarefa consiste nas seguintes atividades:

- Identificação do risco: Determinar o que pode causar uma potencial perda, e obter informações sobre sobre como, onde e porquê que tal perda possa ocorrer.
- Análise do risco: Pode ser usada uma metodologia qualitativa, quantitativa, ou uma mistura de ambas. A análise qualitativa é frequentemente usada primeiro para obter uma indicação geral do nível de risco e revelar os maiores riscos.
- Avaliação do risco: É usado o conhecimento obtido através da análise do risco para tomar decisões sobre ações futuras.

Uma abordagem iterativa da tarefa de *risk assessment* fornece um bom balanço entre a minimização do tempo e do esforço gastos na identificação de controlos, enquanto se continua a assegurar que os maiores riscos são avaliados corretamente.

Tratamento do risco

As opções no tratamento do risco devem ser escolhidas baseadas no resultado da tarefa de *risk assessment*, no custo expetável da sua implementação e nos benefícios expetáveis das mesmas.

Quando existem opções que garantem grandes reduções no risco e que têm baixos custos, estas devem ser implementadas. As restantes opções devem ser analisadas de maneira a perceber se são justificáveis.

Existem quatro tipo de opções não exclusivas para o tratamento do risco:

- Modificar o risco
- Reter o risco
- Evitar o risco

• Partilhar o risco

Depois de um plano de tratamento do risco ser definido é necessário determinar o risco residual. Isto envolve uma atualização ou uma re-iteração da tarefa de *risk assessment*, tendo em conta os efeitos expetáveis do tratamento do risco proposto.

Aceitação do risco

Os planos de tratamento do risco devem descrever como é que os riscos que foram analisados na tarefa de *risk assessment* devem ser tratados de maneira a irem de encontro com os critérios de aceitação do risco. Estes critérios podem ser mais complexos que simplesmente determinarem se um risco residual está acima ou abaixo de um *treshold* específico.

Comunicação e consulta do risco

A comunicação do risco é uma atividade para conseguir um acordo sobre como gerir o risco através da troca e/ou partilha da informação sobre o risco entre os *decision-makers* e outros *stakeholders*.

Esta atividade assegura que aqueles que são responsáveis por implementar a gestão do risco e os investidores percebem a base na qual as decisões são tomadas e o porquê de certas ações serem necessárias.

Monitorização e revisão do risco

Os riscos não são estáticos. Ameaças, vulnerabilidades, verossimilhança ou consequências podem mudar abruptamente sem qualquer indicação. É necessário que exista uma monitorização constanre de maneira a detetar tais mudanças. Esta monitorização pode ser suportada por serviços externos que fornecem informação sobre novas ameaças e vulnerabilidades.

Fatores que afetam a verossimilhança e consequências da ocorrência de ameaças podem mudar, tal como os fatores que afetam a sustentabilidade ou o custo das várias opções de tratamento do risco. Assim sendo, as atividades de monitorização do risco devem ser repetidas regularmente e as opções escolhidas para tratar o risco devem ser revistas periodicamente.

2.6 Ferramentas

Nesta secção vamos começar por falar da plataforma HIDRA e dos seus componentes. Esta plataforma agrega uma grande parte dos dados da PT e, como tal, alimenta o modelo de risco que foi desenvolvido no âmbito deste projeto. Em seguida apresentamos duas plataformas da *Microsoft* que estão ligadas ao processo de *patching*, e que foram

utilizadas também para se conseguir classificar as máquinas da PT quanto ao seu nível de atualização tanto de *patching* como também das definições de antivírus.

2.6.1 Plataforma HIDRA

High performance Infrastructure for Data Research and Analysis (HIDRA) é uma plataforma ágil usada para processar fluxos de eventos, desenvolvida na Portugal Telecom pela Direção de *Cyber Security and Privacy* (DCY)/*Cyber Security Development and Integration* (CSD). É baseada em *Elasticsearch*, *RabbitMQ*, Kibana e programas em *Ruby*. O objetivo principal desta plataforma é possibilitar o processamento de eventos complexos de forma rápida e ágil.

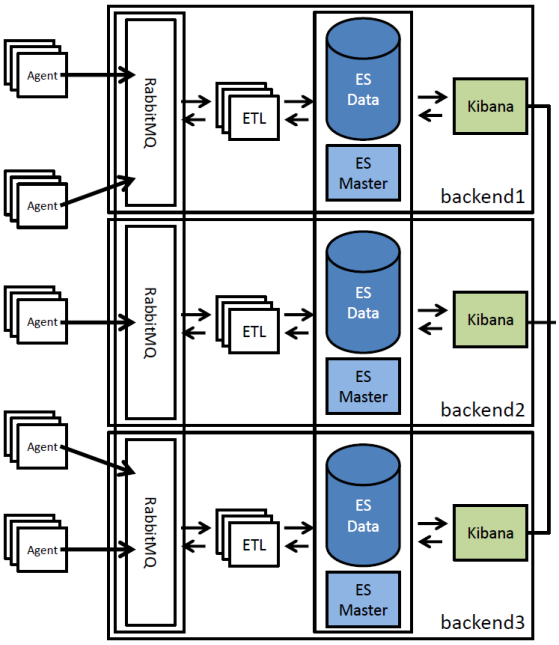

Figura 2.1: Arquitetura da plataforma HIDRA.

A figura 2.1 mostra a arquitetura da plataforma HIDRA, onde se pode observar que tudo começa a partir de agentes que passam a informação (*raw*) para um *broker* (*RabbitMQ*). Depois de sair do *broker*, a informação passa por um processo de ETL (*Extract, Transform, Load*) antes de chegar e ser indexada no *Elasticsearch*. Por último, a informação pode ser visualizada através da interface Kibana. A existência do broker é necessária de maneira a que os agentes e os processos de ETL possam funcionar de maneira assíncrona.

Elasticsearch

O *Elasticsearch* é um motor de pesquisa e análise distribuído criado em 2010 que permite o acesso a um grande volume de dados de forma rápida e eficiente. Esta ferramenta armazena os dados de uma forma centralizada.

Através do *Elasticsearch* podem ser feitas pesquisas estruturadas ou desestruturadas, bem como pesquisas geo ou metric. Por ser distribuído é altamente escalável e permite que sejam obtidas respostas quase instantaneamente. Isto acontece porque os dados se encontram indexados, o que permite aumentar o seu desempenho enquanto motor de pesquisa.

Esta ferramenta usa também uma *Application Programming Interface (API) Representational State Transfer (REST)* e tem suporte para vários tipos de linguagens de programação, o que a torna bastante popular por ser fácil de usar.

Através da agregação de dados é possível explorar padrões existentes nos mesmos.

Kibana

O Kibana é um *plugin* de visualização de dados *open-source* para o *Elasticsearch*. Fornece recursos de visualização sobre o conteúdo indexado num *cluster Elasticsearch*. Os utilizadores podem criar gráficos de barra, linha e dispersão, gráficos e mapas tipo *pie-chart* para visualização de grandes volumes de dados, ou simplesmente visualizar os documentos indexados.

2.6.2 WSUS

O WSUS é uma plataforma gratuita da Microsoft que permite gerir a implementação de atualizações de segurança (tanto *patches* como assinaturas *anti-malware*) e *hotfixes* em ambiente empresarial. Para tal, é necessário que exista pelo menos um servidor WSUS com ligação à *Internet*, de maneira a conseguir fazer downloads do site de atualizações da Microsoft. Depois de realizados, estes são distribuídos pelos computadores e/ou por outros WSUS conforme a configuração de cada rede.

Os administradores têm poder para aprovar ou rejeitar as atualizações disponíveis, forçar a instalação de atualizações até uma certa data, e consultar diversos relatórios, como por exemplo, quais as atualizações que cada máquina precisa. Existe a possibilidade de configurar o WSUS de maneira a que este aprove automaticamente certo tipo de atualizações. Podem ser utilizadas políticas de grupo para configurar o agente de atualizações automáticas, e assim assegurar que os diversos utilizadores não conseguem desativar ou contornar as políticas de atualizações da empresa. [12]

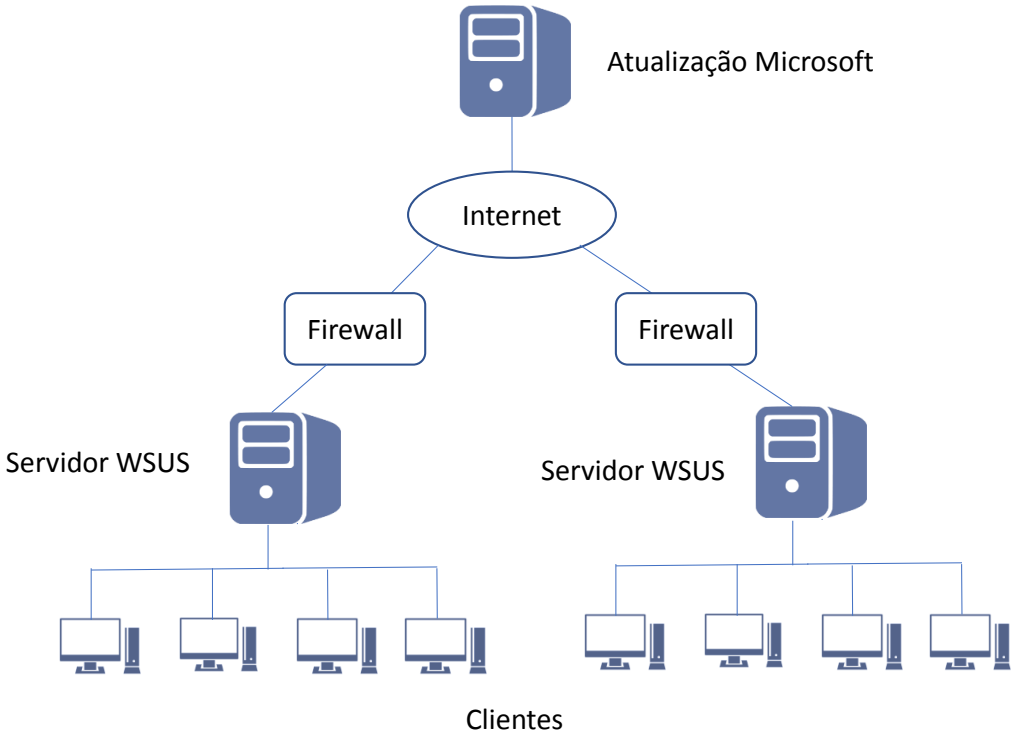

Figura 2.2: Arquitetura WSUS.

2.6.3 SCCM

O *System Center Configuration Manager* (SCCM) é uma Plataforma da Microsoft que encontra qualquer dispositivo (servidores, PC's cliente, e smartphones) ligado à rede através de *Active directory* e instala um *software client* em cada nó. Constrói uma base de dados do inventário com registos de cada ativo (*software* instalado e especificações de *hardware*), e permite reforçar as políticas da instituição através de definições de gestão, como por exemplo, forçar a instalação de novas atualizações de certas aplicações, caso estas não tenham sido instaladas passado um certo tempo após a sua disponibilização. O SCCM integra o WSUS que é responsável pela parte de *patch management*.

2.7 Conclusão

As organizações têm bastantes falhas no que toca a terem um bom processo de gestão de *patching*. Estas falhas são visíveis no ciberataque do *ransomware Wannacry*, que conseguiu explorar vulnerabilidades para as quais já haviam sido disponibilizados *patches* anteriormente. A dimensão deste ataque prova que o problema das organizações com o processo de gestão de *patching* é global e portanto preocupante.

Este projeto abarca diversos assuntos, cujo domínio é relevante para o seu desenvolvimento. Neste capítulo apresentaram-se esses assuntos e conceitos, fornecendo assim o contexto essencial para a compreensão do que é apresentado nos capítulos seguintes.

Capítulo 3

Modelo de Risco

De maneira a conseguir melhorar o processo de gestão de risco da PT, mais concretamente a fase de aplicação de *patches* e atualização de definições de antivírus, criámos um modelo de risco que classifica os servidores da PT.

Neste capítulo começamos por explicar a necessidade da existência do nosso modelo de risco em grandes organizações como a PT. De seguida apresentamos como é que o mesmo foi elaborado e o conjunto de critérios que utiliza para classificar os servidores da PT com um uma nota que representa o seu risco. Por último descrevemos em detalhe o algoritmo por detrás do modelo de risco.

3.1 Contexto

Nos dias de hoje, grandes empresas como a PT precisam de aplicar as atualizações de segurança que vão surgindo da maneira mais responsável e eficiente possível. Este trabalho é complicado devido à dimensão dos parques informáticos deste tipo de empresas que geralmente são compostos por milhares de máquinas. Saber por que máquinas devem começar com o processo de *patching* é essencial, pois a janela de tempo entre a descoberta de vulnerabilidades e a sua exploração por parte de agentes maliciosos é cada vez menor. Portanto, conseguir aplicar as atualizações com a maior brevidade possível e começar a aplicá-las nos elementos que representam o maior risco é de extrema importância.

O processo de *patching* em organizações com parques informáticos de dimensão igual ou superior ao da PT é sempre muito complexo, pois sempre que é disponibilizado um novo *patch* é preciso aplicá-lo a milhares de máquinas. Embora algumas destas tenham funções menores que podem ser interrompidas, outras são responsáveis por manter certos serviços disponíveis sem interrupções. Estas máquinas não podem simplesmente parar para uma atualização sem que previamente haja uma preparação, muitas vezes complicada, de maneira a que máquinas substitutas fiquem responsáveis por tais serviços enquanto se aplicam os *patches*.

O objetivo do modelo de risco é conseguir classificar os servidores do parque infor-

mático da PT com uma nota que representa o seu nível de risco, utilizando para isso diversos critérios que foram escolhidos no decurso do projeto e que são apresentados na secção 3.3. É esperado que depois de os servidores estarem classificados os técnicos da PT responsáveis pela aplicação de *patches* consigam formular, com base nos resultados, estratégias de aplicação não só de *patches* que vão sendo disponibilizadas pela Microsoft mas também de novas definições de antivírus, o que vai contribuir para uma melhoria do nível de segurança informática da organização.

3.2 Elaboração do Modelo

A elaboração do modelo começou com a definição de um conjunto de critérios que iriam ser usados para classificar os servidores da PT de maneira a conseguirmos chegar a uma nota final de risco. O primeiro esboço do modelo contava com quatro critérios: *patching*, importância, exposição e vizinhança. No segundo esboço foram adicionados outros dois critérios: antivírus e alarmes. Estes seis critérios foram escolhidos não só por acharmos que são os mais relevantes como também porque existe uma limitação no que toca aos dados que a PT tem disponíveis. Após a escolha dos critérios atribuímos a cada um deles um peso que representa a sua relevância no cálculo do risco. Mais tarde decidimos que o critério da importância não seria tido em conta da mesma maneira que os restantes. Vamos falar individualmente sobre cada critério na Secção 3.3.

Cada servidor é classificado em cada critério com uma letra de A a F, sendo A a melhor classificação e F a pior. Por exemplo, no critério de alarmes, caso um servidor não tenha gerado nenhum alarme (não foi atacado), recebe a classificação A. Como consequência de outros projetos, os ativos da PT já se encontravam com uma classificação de A a F nos critérios de *patching* e antivírus, de maneira que para o nosso projeto foi necessário definir as regras de classificação dos restantes quatro critérios com base nos dados disponíveis.

Depois dos servidores estarem classificados com uma letra em cada critério (exceto a vizinhança), para efeitos do cálculo da nota final é necessário ter uma representação numérica de cada letra. No caso da vizinhança, a classificação deste critério resulta de uma média de notas numéricas, portanto não é necessário que haja traduções. Após o cálculo da nota final é necessário que exista uma tradução da nota final e da nota da vizinhança para uma letra. De seguida apresentamos as tabelas genéricas que são utilizadas para realizar as traduções das letras que representam as classificações dos servidores em cada critério em números, e das notas finais e da vizinhança em letras (Figura 3.1). Estas tabelas já eram utilizadas na PT.

A tabela de tradução de letras em *ratings* da figura 3.1 foi um ponto de partida e é utilizada no modelo para tradução das notas dos critérios de exposição à *Internet* e alarmes. Tal como foi dito anteriormente, o critério de importância é diferentes dos restantes. Ao contrário de participar no cálculo da nota final com um peso associado, o objetivo deste

Nota	Rating
A	900
B	820
C	740
D	690
E	445
F	250

Rating	Nota
900-820	A
820-740	B
740-690	C
690-640	D
640-445	E
445-250	F

Figura 3.1: Tabelas de tradução entre letras e *ratings*.

critério no modelo de risco é o de influenciar de forma positiva ou negativa (conforme a importância do servidor) os *ratings* dos critérios de *patching* e antivírus. Durante o projeto modificámos a tabela genérica já existente na PT e criámos três novas tabelas de tradução de letras em *ratings* para os critérios de *patching* e antivírus que têm em conta a importância do servidor. Estas tabelas são apresentadas na Secção 3.3.4 que é dedicada ao critério da importância.

3.3 Critérios

3.3.1 Patching

De maneira a saber-se o risco que os ativos representam no que toca a vulnerabilidades que possuem e que podem ser exploradas, cada ativo da PT está classificado com uma nota de A a F que foi dada consoante o seu nível de *patching*, sendo A a melhor nota e F a pior. Esta nota é baseada em critérios como a atualização mais crítica que o ativo tem em falta, o tempo que passou desde que essa atualização foi disponibilizada e o departamento que gere o servidor (Direção de *IT* ou Direção de *Engineering Network Operations*). Exemplo de uma parametrização possível:

Departamento DIT

- Se a máquina requer uma atualização de segurança de nível crítico:
  - Se a atualização já está disponível há mais de 14 dias => F
  - Caso contrário => D
- Se a máquina requer uma atualização de segurança de nível alto/moderado:
  - Se a atualização já está disponível há mais de 14 dias => E
  - Caso contrário => C
- Se a máquina requer uma atualização de segurança de nível baixo ou sem nível:
  - Se a atualização já está disponível há mais de 14 dias => D
  - Caso contrário => B
- Caso a máquina tenha todas as atualizações instaladas => A

Departamento DEO

- Se a máquina requer uma atualização de segurança de nível crítico:
  - Se a atualização já está disponível há mais de 15 dias => F
  - Caso contrário => D
- Se a máquina requer uma atualização de segurança de nível alto/moderado:
  - Se a atualização já está disponível há mais de 30 dias => E
  - Caso contrário => C
- Se a máquina requer uma atualização de segurança de nível baixo ou sem nível:
  - Se a atualização já está disponível há mais de 30 dias => D
  - Caso contrário => B
- Caso a máquina tenha todas as atualizações instaladas => A

3.3.2 Antivírus

Tal como no patching, os ativos da PT também se encontram classificados de A a F quanto ao nível de atualização do antivírus que possuem. Neste caso, a nota vai variar consoante o tempo que passou desde que foram disponibilizadas novas definições. Exemplo de uma parametrização possível:

DIT McAfee Update Status - Servers

- Se as últimas definições do antivírus foram disponibilizadas há 5 dias ou mais => F
- Se as últimas definições do antivírus foram disponibilizadas há 4 dias ou mais => E
- Se as últimas definições do antivírus foram disponibilizadas há 3 dias ou mais => D
- Se as últimas definições do antivírus foram disponibilizadas há 2 dias ou mais => C
- Se as últimas definições do antivírus foram disponibilizadas há 1 dias ou mais => B
- Caso a máquina tenha as últimas definições instaladas => A

3.3.3 Vizinhança

Nem todo o *malware* é igual ou se propaga da mesma maneira. Um *worm* por exemplo é um tipo de *malware* que se espalha maioritariamente através da rede. Depois de infetar um *host*, o *worm* auto replica-se de maneira a propagar a infeção para os restantes *hosts* presentes na rede (propagação lateral).

Tendo em conta o que foi dito no parágrafo anterior, é essencial saber para cada máquina, que outras máquinas se encontram ao seu redor (na mesma rede) e o nível de risco que a sua rede representa. Esta informação é uma mais valia para a empresa pois caso uma dada máquina seja infetada, permite-nos saber que máquinas podem potencialmente ser contaminadas, e consequentemente a criticidade das mesmas. Para cada máquina, a nota do critério de vizinhança reflete o risco da rede onde esta se encontra. A forma de calcular a nota da vizinhança encontra-se explicada na secção 3.4.

3.3.4 Importância

Nem todas as máquinas têm a mesma relevância dentro de uma organização, pois certas máquinas contêm informação muito importante, confidencial, ou têm mesmo uma função essencial na empresa, de modo que são consideradas prioritárias no que toca a assegurar a sua integridade. O critério de importância pretende influenciar a nota de risco de máquinas críticas de maneira a que estas tenham um nível global de risco mais elevado que outras que possuam notas semelhantes nos outros critérios mas que não sejam tão importantes.

De seguida são apresentadas as três tabelas (Figura 3.2, Figura 3.3, Figura 3.4) que criámos a partir da tabela genérica da Figura 3.1 e que representam a tradução de letras em ratings dos critérios de *patching* e antivírus tendo em conta a importância de cada servidor. Para conseguir influenciar o nível de risco dos servidores a partir da importância dos mesmos decidimos utilizar o intervalo que existe entre *ratings* de uma mesma letra que pode ser observado na Figura 3.1. Desta maneira, dentro de uma mesma letra temos valores de *rating* diferentes para servidores com importâncias diferentes.

Criámos estas tabelas e definimos que o *rating* dos servidores com importância alta estaria no extremo inferior do intervalo de *ratings* para cada letra (pior nota possível dentro do intervalo), enquanto que o *rating* daqueles que são pouco importantes encontra-se no extremo superior do intervalo menos 5 pontos. Os servidores com importância média recebem um *rating* que se encontra a meio do intervalo. Apercebemo-nos da necessidade de haver diferenças de *ratings* durante o processo de refinamento do modelo, e portanto a explicação para esta necessidade encontra-se no Capítulo 5

Nota	Rating
A	825
B	815
C	735
D	685
E	635
F	440

Figura 3.2: Tabela de tradução entre letras e ratings. Máquinas com importância baixa.

Nota	Rating
A	860
B	780
C	715
D	665
E	540
F	340

Figura 3.3: Tabela de tradução entre letras e ratings. Máquinas com importância média.

Nota	Rating
A	900
B	820
C	740
D	690
E	445
F	250

Figura 3.4: Tabela de tradução entre letras e ratings. Máquinas com importância alta.

3.3.5 Exposição à Internet

Num mundo cada vez mais digital onde os casos de ataques informáticos aumentam de dia para dia, é essencial que as empresas se preocupem seriamente em proteger os seus ativos expostos à *Internet* por forma a não só manter os seus dados e os dos seus clientes seguros mas também para assegurar a disponibilidade dos seus serviços. Este tipo de ativos têm que ser melhor protegidos pois dado que se encontram expostos à *Internet* estão constantemente em risco de serem sondados por agentes maliciosos, que procuram vulnerabilidades que possam ser exploradas para a realização de um ataque com sucesso.

Tendo em conta o que já foi dito, o critério de exposição à *Internet* tem como finalidade aumentar o nível de risco dos servidores da PT que estejam diretamente expostos à *Internet*, tornando-os prioritários face a outros semelhantes com menor nível de exposição.

Classificação

A classificação foi definida durante o projeto e aprovada pela PT.

- Se o servidor estiver exposto diretamente à *Internet* => F
- Se o servidor não estiver exposto diretamente à *Internet* => C

3.3.6 Alarmes

O critério de alarmes permite aumentar o nível risco das máquinas que sofreram ciberataques recentemente. É importante ter em conta os ataques pois as máquinas que sofreram ataques recentemente estão mais suscetíveis a serem atacadas num futuro próximo. Por exemplo, se um agente malicioso conseguir explorar uma vulnerabilidade com sucesso numa máquina, nada lhe garante que a vulnerabilidade irá ser *patched* de imediato. Como tal, continua a existir a possibilidade de exploração da mesma durante um certo período. Caso existam vários alarmes para uma dada máquina o algoritmo apenas tem em conta o mais crítico para o cálculo da nota do critério. O nível dos alarmes varia entre 1 e 10, sendo 1 um alarme com pouca relevância e 10 um alarme crítico. O intervalo é de 1 a 10 pois os alarmes são gerados pelo *AlienVault* (Plataforma de Gestão e Correlação de Eventos de Segurança utilizada pela PT) que os classifica com um valor nesse intervalo.

Classificação

A classificação foi definida durante o projeto e aprovada pela PT.

- Caso exista pelo menos um alarme e o nível de risco seja 9 ou 10 => F
- Caso exista pelo menos um alarme e o nível de risco seja 7 ou 8 => E
- Caso exista pelo menos um alarme e o nível de risco seja 5 ou 6 => D
- Caso exista pelo menos um alarme e o nível de risco seja 3 ou 4 => C
- Caso exista pelo menos um alarme e o nível de risco seja 1 ou 2 => B
- Caso não existam alarmes => A

3.4 Algoritmo

No âmbito deste projeto criámos um algoritmo que tem como objetivo gerar a classificação final de risco de cada servidor e as classificações que cada um obteve nos diferentes critérios. Para que possa gerar esse output, o algoritmo recebe inicialmente um conjunto de dados da plataforma HIDRA como input. No caso do *patching* e do antivírus recebe diretamente a classificação (letra) dos servidores nesses critérios (tal como foi dito nas secções anteriores, a PT já possuía uma classificação destes dois critérios). No caso da exposição à *Internet* e alarmes, o algoritmo recebe informação sobre se o servidor está exposto diretamente ou não à *Internet*, e caso existam alarmes o nível do alarme mais crítico. No caso da *importância* recebe um valor gerado artificialmente de 1 a 3 sendo 1 o menor nível de importância e 3 o maior. Para a vizinhança o algoritmo recebe os *hostnames* de todos os servidores que se encontram nas mesmas redes que o servidor que o algoritmo está a classificar no momento.

O algoritmo do modelo de risco encontra-se dividido em duas fases. A necessidade desta divisão vem da maneira como a nota do critério da vizinhança é calculada. A nota do

critério da vizinhança é calculada a partir das notas que os servidores vizinhos obtiveram nos restantes critérios, e portanto é necessário que primeiro se calculem as notas dos servidores em todos os critérios menos na vizinhança (fase 1), e que depois exista uma fase 2 onde se calculam as notas do critério da vizinhança e a nota final de risco para cada servidor.

Fase 1

O algoritmo começa por criar as estruturas de dados necessárias onde são guardadas as informações sobre os servidores. De seguida são realizadas *queries* à HIDRA de maneira a obter os *hostnames* dos servidores a analisar e a informação de input necessária mencionada no início desta secção. Como a vizinhança é apenas calculada na fase 2, o input também é obtido apenas na fase 2.

Depois de obtida a informação de input o algoritmo começa por classificar com uma letra cada servidor nos critérios de alarmes e exposição à *Internet* segundo as regras apresentadas nas Subsecções 3.3.6 e 3.3.5 (relembrar que o input relativo a *patching* e antivírus já é a letra em si). De seguida traduz as letras em números para mais tarde efetuar cálculos. No caso dos critérios de *patching* e antivírus utiliza as tabelas segundo a importância apresentadas na Secção 3.3.4 e para os critérios de alarmes e exposição à *Internet* utiliza a tabela da Figura 3.1. Por último calcula a média das notas numéricas dos quatro critérios mencionados no ponto anterior para cada servidor. Esta média vai ser utilizada no cálculo da nota da vizinhança.

$$Média = (Cp + Ca + Ce + Ca)/4$$

- Cp (*Patching*), Ca (Antivírus), Ce (Exposição à *Internet*), Ca (Alarmes).

Fase 2

O algoritmo realiza uma série de *queries* à HIDRA de maneira a descobrir em que redes se encontra cada servidor e consequentemente que servidores cada um tem como "vizinhos" (na mesma rede). A nota do critério de vizinhança de cada servidor é calculada utilizando a média das notas dos critérios calculada na fase 1 de cada servidor que seja seu "vizinho", fazendo uma média de notas geral da rede onde cada um se encontra. Por exemplo, se o algoritmo estiver a classificar o servidor S1 quanto ao critério de vizinhança, e este tem como vizinhos os servidores S2 e S3, então a nota de vizinhança é:

$$NVS1 = (MC2 + MC3)/2$$

- NVS1 (Nota do critério de vizinhança do servidor S1), MC2 e MC3 (Média das notas dos critérios do servidores S2 e S3 que foi calculada na fase 1).

Depois de calculada a nota de vizinhança, é então calculada a nota final que vai representar o nível de risco de cada servidor. Multiplica-se a nota de cada critério pelo peso correspondente e soma-se os resultados. No Capítulo 5 vamos apresentar as diferentes fórmulas e valores dos pesos dos critérios que o algoritmo utilizou para calcular a nota final de risco dos servidores durante o projeto, até chegarmos a uma versão final.

3.5 Conclusão

O processo de *patching* é complexo em organizações com parques informáticos de grande dimensão, e dado que os recursos para o realizar são limitados é uma mais valia que existam instrumentos que tenham como finalidade reduzir a sua complexidade. O nosso projeto produziu um modelo de risco que é capaz de classificar os servidores da PT com uma nota de risco a partir de certos critérios, que foram definidos por nós depois de uma investigação inicial sobre não só o que seria mais relevante ter em conta para calcular o nível de risco, como também sobre os dados que a PT poderia disponibilizar. A partir não só desta classificação final como também das notas que cada servidor recebeu nos diferentes critérios, a PT poderá elaborar estratégias de aplicação de *patches* no seu parque informático e assim ter um processo de *patching* mais eficiente.

Capítulo 4

Implementação

Neste capítulo descrevemos a parte de programação que foi realizada durante o desenvolvimento deste projeto.

A linguagem de programação utilizada foi Ruby, pois é a mais utilizada na área de cibersegurança da PT. Foram criadas duas classes: a primeira (Algorithm.rb) é responsável pela parte lógica do algoritmo, recebe os dados e processa-os de maneira a chegar a uma classificação do risco para cada servidor. A segunda classe (HidraConnect.rb) trata das ligações à plataforma HIDRA. Tem como missão ir buscar os dados necessários de modo a que o algoritmo possa correr.

4.1 Estrutura do modelo de risco

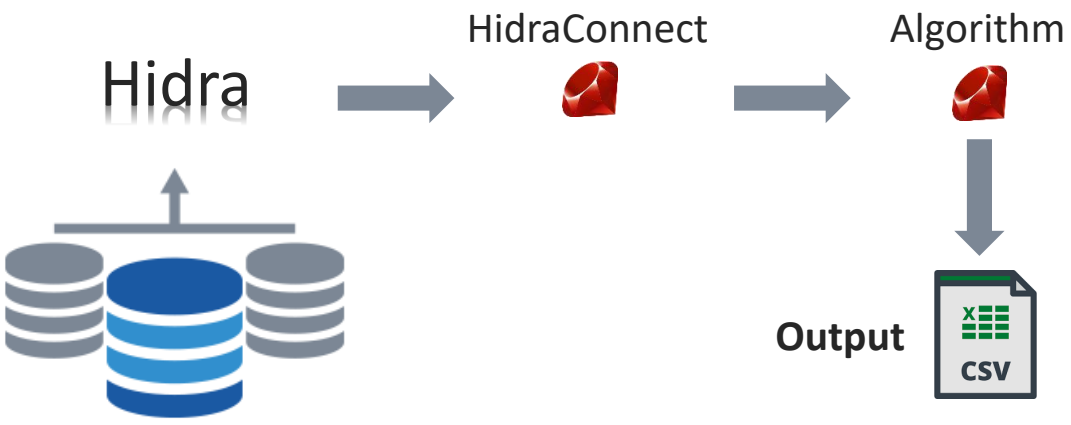

Figura 4.1: Estrutura do modelo de risco.

Como podemos observar na Figura 4.1 em primeiro lugar temos a fonte de dados do modelo de risco, a plataforma HIDRA. Os dados da plataforma HIDRA encontram-se armazenados em índices, e cada índice possui diferentes campos. Por exemplo, enquanto que a partir de um índice ao qual realizamos uma *query* podemos retirar informação sobre *hostnames* e níveis de atualização de *patching* e antivírus de cada servidor, noutros apenas

conseguimos retirar informação sobre alarmes. Por esta razão é necessário realizar várias *queries* para obter todos os dados que o algoritmo requer. A classe *Algorithm* chama os métodos presentes na classe *HidraConnect* que fazem as ligações necessárias à HIDRA de maneira a obter os dados que o algoritmo precisa. O output do algoritmo é depois enviado para um ficheiro .csv.

4.2 Classe Algorithm

A classe *Algorithm* é responsável pela parte lógica do algoritmo. Começa por invocar os métodos da classe *HidraConnect* de maneira a receber os *hostnames* dos servidores a analisar e os dados necessários para os classificar em cada critério (exceto vizinhança). Depois de receber os mesmos, armazena-os numa estrutura de dados chamada *Hash*, que funciona como um dicionário onde existem chaves únicas, cada uma com valor(es) associado(s). Neste projeto, as chaves da *Hash* correspondem aos *hostnames* dos servidores. Cada uma das chaves vai ter associada um *array* que contém os dados recebidos. Depois de armazenar os dados é então criada outra *Hash*. Esta, tal como a anterior, vai usar os *hostnames* como chaves. Cada chave vai ter como valor um array preenchido com os valores numéricos correspondentes às notas os servidores obtiveram em cada critério.

Tal como foi dito na explicação do algoritmo na Subsecção 3.4 este divide-se em duas fases. Durante cada uma delas, a *Hash* que armazena os dados de input é percorrida iterativamente.

Ainda que na próxima secção, que fala sobre a classe *HidraConnect*, apresentamos os diversos métodos da mesma, não considerámos relevante fazer o mesmo para a classe *Algorithm*.

4.3 Classe HidraConnect

Esta classe é responsável pelas diversas ligações à HIDRA que têm como objetivo a obtenção de dados para alimentar o algoritmo do modelo de risco. As ligações são realizadas utilizando uma linguagem específica do domínio tipo JSON própria do Elasticsearch através do protocolo HTTPS. De seguida são apresentadas as funções que compõem esta classe:

```
def machines_list ()
```

Função que é responsável por aceder ao índice da HIDRA que possui os *hostnames* de todos os servidores a analisar pelo algoritmo. Nesse índice também se encontra informação relativa ao nível de atualização de *patching* e de antivírus de cada um. Resumindo, a função utilizando apenas uma *query* devolve não só os *hostnames* dos servidores, como também a nota de *patching* e antivírus dos mesmos.

```
def importance_grade (hostname)
```

Função responsável por devolver o nível de importância do servidor que possui o *hostname* recebido como parâmetro. Neste momento esta informação não se encontra disponível e portanto é emulada. Primeiro é feito um *Hash* do *hostname* e posteriormente converte-se o mesmo para um inteiro. Em seguida aplica-se a operação de módulo de maneira a obtermos um valor de 1 a 3 que representa a importância da máquina (1 - Baixa, 2 - Média, 3 - Alta). Este processo permite que os valores de importância obtidos para cada máquina sejam pseudo-aleatórios mas constantes nas diversas execuções.

```
def exposure (hostname)
```

Esta função executa uma ligação à HIDRA de maneira a descobrir se o servidor que possui o *hostname* recebido como parâmetro se encontra exposto diretamente ou não à *Internet*. Devolve *true* caso o servidor se encontre diretamente exposto e *false* caso contrário.

```
def alerts(hostname)
```

Função que recebe informação da HIDRA sobre os alarmes que surgiram num período até 24h antes para o servidor cujo nome é recebido como parâmetro, e os respetivos níveis de risco associados aos mesmos.

```
def neighborhood (hostname)

def interfaces_ip(hostname)

def ip_to_range(ip)

def hosts_in_range(range_gte, range_lte, hostname)
```

A função *neighborhood* é a responsável por devolver os *hostnames* de todos os servidores que são vizinhos do servidor que possui o *hostname* recebido como parâmetro. Dado que para chegar ao output pretendido é necessário realizar diversas *queries*, foram criadas três funções auxiliares.

A função *interfaces_ip* é responsável por pegar no *hostname* recebido como parâmetro e obter os endereços *IP* das diferentes interfaces de rede que o servidor em questão tem. As interfaces de rede que são relevantes para o modelo de risco são as que cujo endereço de *IP* se encontre nas redes de *backend, frontend* ou gestão.

A função *ip_to_range* é chamada para cada um dos endereços *IP* devolvidos pela função *interfaces_ip* e identifica o segmento de rede onde cada um se encontra.

Por último, a função *hosts_in_range* pega nos diferentes segmentos de rede identificados e obtém os *hostnames* das máquinas presentes em cada um deles.

Depois de eliminados os duplicados a função neighborhood devolve um array com todos os *hostnames* que correspondem a servidores vizinhos.

4.4 Configuração

De maneira a facilitar a realização de alterações ou atualizações do código, foi criado um ficheiro de configuração, escrito na linguagem *yaml*. Ao existir este ficheiro, certas alterações ao modelo podem ser realizadas sem que seja necessário mexer no códgio. Constam neste ficheiro entre outros:

- Diferentes formatos base das *queries* utilizadas na classe *HidraConnect*

```
q1:
  size: 10000
  query:
    bool:
      must:
        query_string:
          query:
          analyze_wildcard: "true"


q:
  size: 10000
  query:
    bool:
      must:
        -
          query_string:
            query:
            analyze_wildcard: "true"
        -
          range:
            ts:
              gte:
              lte:
```

Podemos observar dois exemplos de *queries* base que foram utilizadas neste projeto. A diferença entre elas é que com a segunda podemos especificar um intervalo em termos de data. Por exemplo, com as palavras chave "*range*"e "*ts*"indicamos que se trata de um intervalo relativo ao campo *timestamp*. Depois atribuímos duas datas, uma ao campo "*gte*"(*greater than or equal*) e outra ao campo "*lte*"(*lower than or equal*). Desta maneira apenas são apresentados resultados cujo campo *timestamp* se encontre entre as duas datas especificadas.

- Os pesos dos diversos critérios utilizados no modelo de risco

```
:patching_weight: 0.35
:antivirus_weight: 0.20
:internet_exposure_weight: 0.10
:alerts_weight: 0.10
:neighborhood_weight: 0.25
```

- Os *ratings* que cada letra pode ter consoante a importância de cada servidor

```
:value_grade_A_high: 900
:value_grade_B_high: 820
:value_grade_C_high: 740
:value_grade_D_high: 690
:value_grade_E_high: 445
:value_grade_F_high: 250

:value_grade_A_med: 860
:value_grade_B_med: 780
:value_grade_C_med: 715
:value_grade_D_med: 665
:value_grade_E_med: 540
:value_grade_F_med: 340

:value_grade_A_low: 825
:value_grade_B_low: 825
:value_grade_C_low: 735
:value_grade_D_low: 685
:value_grade_E_low: 635
:value_grade_F_low: 440
```

- Os tipos de interfaces de rede relevantes

```
:network_types: ["ip_management", "ip_backend", "ip_frontend"]
```

- Nota que é atribuída por omissão a cada critério caso a informação necessária para o cálculo da mesma não esteja disponível.

```
:unknown_grade_patching: "F"
:unknown_grade_antivirus: "F"
:unknown_grade_importance: "3"
:unknown_grade_exposure: true
:unknown_grade_neighborhood: []
```

4.5 Conclusão

Tal como referido no início deste capítulo, a implementação do modelo de risco foi realizada utilizando a linguagem de programação *Ruby*, pois é a linguagem utilizada na área de cibersegurança da PT. Foram programadas duas classes em *Ruby*. A primeira (*HidraConnect*) é responsável pelas ligações à plataforma HIDRA, que é a fonte de dados que alimenta o modelo. A segunda (*Algorithm*) trata da lógica do modelo. Chama as funções da classe *HidraConnect* de modo a obter os dados que precisa para classificar os servidores e produz um output que é enviado para um ficheiro .csv. Por último também foi criado um ficheiro de configuração em YAML. Este permite que se possam realizar alterações no futuro, como por exemplo mudanças nos pesos dos critérios, com maior facilidade sem que se tenha que mexer no código.

Capítulo 5

Avaliação e ajuste do modelo de risco

O modelo de risco tem dois objetivos a cumprir: conseguir classificar os servidores da PT que possuem sistema operativo Windows com uma nota que representa o seu nível de risco, e servir como guia para as equipas da PT responsáveis pelo processo de aplicação de atualizações de segurança (aplicação de *patches* e atualização de assinaturas de antivírus). Estas equipas têm uma capacidade limitada, de maneira que apenas podem atualizar um certo número de máquinas de cada vez. O modelo vai ser utilizado pelas equipas para que estas saibam em cada momento quais as máquinas prioritárias que devem ser atualizadas. Para este efeito criámos um indicador que apelidámos de ganho. O ganho é um valor numérico que representa a contribuição de uma máquina na diminuição do risco global do parque informático da PT caso a mesma seja atualizada. Isto significa que as máquinas com maior ganho (que mais diminuem o risco na sua globalidade) são as consideradas prioritárias na hora de realizar atualizações. O ganho é a diferença entre o nível risco de uma máquina depois de ser atualizada e o nível risco da mesma antes de ser atualizada, mais o valor com que esta contribui para a diminuição da nota do critério da vizinhança dos vizinhos ao ser atualizada. Relembramos que a nota do critério da vizinhança é calculada utilizando a média de notas dos critérios dos vizinhos. Ao atualizarmos uma máquina, estamos a melhorar as notas que esta possui nos critérios de *patching* e antivírus. Como tal, a média das notas dos seus critérios vai melhorar, influenciando assim positivamente as notas do critério da vizinhança dos seus vizinhos.

Para a definição dos pesos dos critérios do modelo decidimos que o que se devia valorizar mais era a proteção dos servidores, neste caso, os critérios de *patching* e de antivírus. Aceitámos como verdade que tanto direta como indiretamente expostos à *Internet* existe sempre a possibilidade dos servidores sofrerem ataques, e como tal, a melhor forma de os neutralizar é ter todas as atualizações em dia. Ao ter na memória o ataque ocorrido no ano passado utilizando o *ransonware Wannacry* que se propaga como um *worm* pela rede, também decidimos dar um peso considerável ao critério da vizinhança.

Nesta secção vamos descrever a evolução do modelo de risco ao longo do desenvolvimento deste projeto, de modo a que este conseguisse cumprir os dois objetivos descritos

no início do parágrafo anterior. No total foram concretizadas quatro versões do modelo de risco. A necessidade de fazer alterações no modelo deveu-se ao facto de que este não conseguia definir uma ordem aceitável de priorização entre servidores. De versão para versão as alterações realizadas são ao nível da fórmula de cálculo do nível de risco. Como consequência das mudanças na fórmula, alguns pesos também foram sendo alterados, ainda que mantendo sempre como base as premissas utilizadas originalmente para repartir os pesos entre os critérios que foram descritas no parágrafo anterior.

A metodologia de avaliação consistiu em *expert knowledge*. Após a concretização de uma nova versão geravam-se resultados que foram avaliados em conjunto com os peritos responsáveis por aplicar as atualizações nos servidores da PT. Em cada versão são apresentados os resultados de alguns casos concretos. Apesar de terem sido utilizados mais casos para efeitos de avaliação, estes são aqueles que consideramos serem os melhores para explicarmos o porquê da necessidade de realizar uma nova versão. As tabelas com os dados de cada caso apresentam: as notas dos servidores em cada critério, a classificação de risco obtida após a aplicação da fórmula da versão em questão (Risco), a classificação de risco com que ficam caso sejam atualizados (Risco final) e o ganho.

5.1 Versão 1

Esta versão foi o primeiro teste ao peso dos critérios que definimos e funcionou como base para as restantes.

Pesos dos critérios

- *Patching* (P): 25%
- Antivírus (AV): 20%
- Exposição (E): 10%
- Importância (I): 15%
- Alarmes (A): 10%
- Vizinhança (V): 20%

Fórmula

$$Risco = P * 0.25 + AV * 0.20 + E * 0.10 + A * 0.10 + I * 0.15 + V * 0.20$$

Avaliação

Com os casos que apresentamos de seguida procurámos perceber as diversas interações existentes no modelo. Nas tabelas podemos observar os diferentes níveis de risco e valores do ganho entre máquinas com notas semelhantes nos diversos critérios. No primeiro caso

com servidores com níveis de risco baixos e no segundo caso com servidores que possuem níveis de risco altos.

Resultados

Caso 1: Existência de uma máquina importante com nível de risco baixo (M1) e na sua vizinhança a presença de máquinas pouco importantes com nível de risco também baixo (M2, M3, M4 e M5).

Host	P	AV	I	E	A	V	Risco	Risco Final	Ganho
M1	820	820	250	900	900	802,3	747,0	783,0	44,0
M2	900	900	900	740	690	780,3	839,1	839,1	0,0
M3	820	820	900	900	900	769,8	838,0	874,0	44,0
M4	820	820	900	740	445	800,5	782,6	818,6	44,0
M5	900	900	900	900	250	794,3	813,9	813,9	0,0

Tabela 5.1: Resultados caso 1 Versão 1

Caso 2: Existência de uma máquina importante com nível de risco alto (M1) e na sua vizinhança a presença de máquinas pouco importantes com nível de risco também alto (M2, M3, M4 e M5).

Host	P	AV	I	E	A	V	Risco	Risco Final	Ganho
M1	445	445	250	250	900	507,5	454,3	659,0	250,3
M2	250	250	900	250	900	494,5	461,4	753,9	357,5
M3	445	445	900	250	740	483,0	530,9	735,6	250,3
M4	250	250	900	250	690	505,0	442,5	735,0	357,5
M5	445	445	900	250	440	498,0	503,9	708,6	250,3

Tabela 5.2: Resultados caso 2 Versão 1

Discussão e conclusões

Ao observarmos ambas as tabelas conseguimos destacar um padrão. Na coluna onde se registam os ganhos, caso uma máquina seja atualizada, podemos verificar que os ganhos são iguais quando as notas de *patching* e antivírus são iguais, ainda que as notas dos restantes critérios sejam diferentes. Como exemplo deste problema temos as máquinas M1 e M3, que apesar de terem importâncias diferentes o ganho de as atualizarmos é 44 para ambas. Isto quer dizer que os ganhos dependem exclusivamente das notas de *patching* e antivírus. Isto faz sentido pois estes dois critérios são os únicos que podem ser melhorados. Os restantes critérios são considerados como características permanentes dos servidores. A dependência entre o ganho e os critérios de *patching* e antivírus significa que uma máquina importante vai ter o mesmo ganho que uma máquina pouco importante

caso possuam as mesmas notas de *patching* e antivírus. Esta versão do modelo não cumpre o segundo objetivo, pois não consegue indicar corretamente quais as máquinas que realmente são as mais prioritárias.

5.2 Versão 2

Depois de concluir que o problema na versão 1 estava relacionado com a importância dos ativos decidimos aplicar esse critério de uma maneira diferente. Para diferenciar máquinas mais importantes de máquinas menos importantes implementámos um mecanismo que prejudica as notas dos critérios dos servidores consoante a sua importância.

Começámos por definir três níveis de importância (baixa, média e alta). De seguida criámos uma tabela onde normalizámos os valores que os critérios podem tomar (250 a 900) para valores entre 0 e 1. Depois da criação da tabela definimos três valores (1.15, 1.55 e 2.4). Estes valores estão associados a cada uma das importâncias, baixa, média e alta respetivamente. A ideia é normalizar as notas que os servidores receberam em cada critério e depois elevá-las a um dos três fatores consoante a importância do servidor, de maneira a piorar as notas. Ao ter valores diferentes para cada importância estamos a assegurar que as notas dos servidores mais importantes são mais penalizadas do que as dos servidores menos importantes.

Nesta versão a média das notas dos critérios que é feita no final da fase 1 do algoritmo (ver Secção 3.4 do Capítulo 3) é realizada depois da penalização descrita no parágrafo ser aplicada às notas. Deste modo, dado que as mesmas vão ser utilizadas para o cálculo da nota da vizinhança, o mecanismo do parágrafo anterior não se aplica à nota do critério da vizinhança.

Como nesta versão o critério da importância afeta o cálculo do risco de uma forma diferente (influenciando negativamente as notas dos critérios), dividimos o peso que este tinha na equação pelo critério de *patching* e pelo da vizinhança.

Pesos dos critérios

- *Patching* (P): 35%
- Antivírus (AV): 20%
- Exposição (E): 10%
- Alarmes (A): 10%
- Vizinhança (V): 25%

Fórmula

Relembramos o mecanismo que foi descrito no início desta secção. As notas não se elevam diretamente, mas são primeiro normalizadas para valores de 0 a 1. Depois de normalizadas são elevadas aos expoentes apresentados em baixo consoante a importância

do servidor. Por último sofrem o processo inverso à normalização que sofreram. Quanto maior a importância, maior o expoente. Isto significa que as notas dos servidores importantes serão as mais penalizadas.

$$Risco = (P)^x * 0.35 + (AV)^x * 0.20 + (E)^x * 0.10 + (A)^x * 0.10 + V * 0.25$$

- Se a importância do servidor é alta, $x = 2.4$
- Se a importância do servidor é média, $x = 1.55$
- Se a importância do servidor é baixa, $x = 1.15$

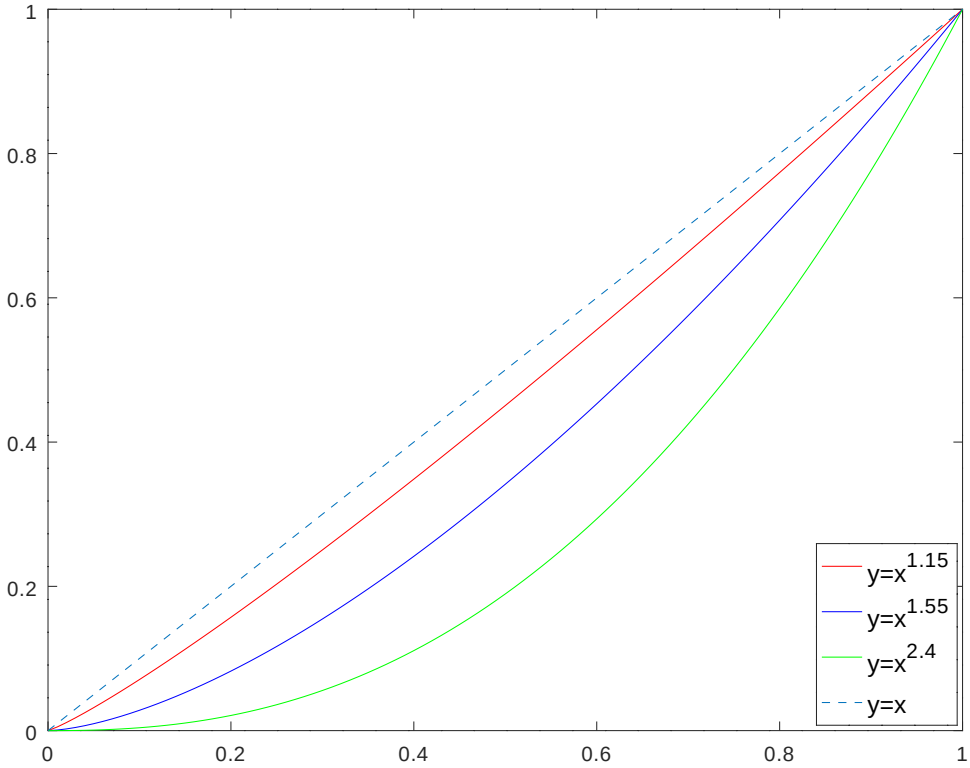

Figura 5.1: Funções referentes ao mecanismo de penalização

Na Figura 5.1 podemos observar o que acontece aos valores normalizados de 0 a 1 (e consequente à classificação de risco de cada servidor) quando estes são elevado aos fatores de penalização descritos em cima. O que podemos verificar no gráfico é o quanto os valores são afetados tendo em conta a importância dos servidores. Por exemplo, enquanto que quando a função que corresponde aos valores dos servidores com importância baixa (cor vermelha) quase que não é afetada, a função que corresponde aos servidores com importância alta (cor verde) possui uma concavidade considerável. Esta concavidade significa que as notas começam a piorar rapidamente à medida que se vão afastando do *rating* 900 (que é representado pelo valor 1).

Avaliação

Nesta versão estávamos maioritariamente preocupados com os resultados que poderiam existir no que toca a diferenciar os servidores quanto à sua importância, quando as notas dos servidores estivessem nos extremos. Por exemplo, no caso dos servidores terem nota F em todos ou quase todos os critérios. Isto porque se a nota de um servidor já se encontra em F, o mecanismo de penalização descrito na apresentação desta versão pode não ter o impacto desejado.

Resultados

Para cada caso vamos primeiro apresentar a tabela com as notas originais dos critérios de cada máquina. Na segunda tabela já foram aplicadas as funções exponenciais às notas que são apresentadas.

Caso 1: Existência de uma máquina importante com nível de risco alto (M1) e na sua vizinhança a presença uma máquina pouco importante com nível de risco alto (M2).

Host	P	AV	I	E	A
M1	250	250	Alta	250	900
M2	250	250	Baixa	250	900

Tabela 5.3: Notas originais caso 1 Versão 2

Host	P	AV	I	E	A	V	Risco	Risco Final	Ganho
M1	250,0	250,0	Alta	250,0	900,0	412,5	355,6	713,1	438,8
M2	250,0	250,0	Baixa	250,0	900,0	412,5	355,6	713,1	438,8

Tabela 5.4: Resultados caso 1 Versão 2

Caso 2: Existência de duas máquinas importantes com nível de risco alto (M1 e M2) e na sua vizinhança a presença de uma máquina pouco importante com nível de risco alto (M3).

Host	P	AV	I	E	A
M1	445	250	Alta	250	250
M2	250	445	Alta	250	250
M3	250	250	Baixa	250	250

Tabela 5.5: Notas originais caso 2 Versão 2

Host	P	AV	I	E	A	V	Risco	Risco Final	Ganho
M1	286,1	250,0	Alta	250,0	250,0	252,3	263,2	608,1	423,8
M2	250,0	286,1	Alta	250,0	250,0	254,5	257,8	608,1	429,3
M3	250,0	250,0	Baixa	250,0	250,0	252,3	251,1	608,6	438,8

Tabela 5.6: Resultados caso 2 Versão 2

Discussão e conclusões

Nesta versão esperávamos que o mecanismo de penalização de notas dos critérios segundo a importância fizesse com que ocorressem mudanças significativas no indicador ganho face aos resultados da versão 1. No entanto, na Tabela 5.4 podemos observar que quando as notas originais dos servidores são o mínimo, o mecanismo de penalização de notas não tem qualquer efeito, dado que não se podem piorar mais as notas. Na Tabela 5.6 o problema é semelhante. As máquinas importantes apenas têm uma das notas (*patching* ou antivírus) um pouco mais elevadas que a máquina pouco importante, e com isso os ganhos já são menores. Esta versão não consegue fazer com que os ganhos das máquinas importantes sejam sempre maiores que os das menos importantes com notas semelhantes, logo não cumpre com os objetivos.

5.3 Versão 3

De maneira a tentar resolver os problemas da versão 2, nesta versão decidimos aplicar o mecanismo de penalização de notas descrito na Secção anterior ao resultado de calcular o risco utilizando todos os critérios menos a vizinhança, em vez de individualmente a cada critério. Tal como na versão anterior, devido ao modo como a vizinhança é calculada, esta não é submetida ao mecanismo de penalização de notas.

Como a vizinhança fica de fora e tem um peso de 25% , isso quer dizer que o valor resultante de calcular o risco usando os outros critérios e consequente aplicação do mecanismo de penalização vale 75% . Como esse valor vai então ser multiplicado por 0.75, houve uma necessidade de aumentar os pesos dos critérios na fórmula, de modo a que estes no final continuem a ter o mesmo peso original.

Na subsecção onde a fórmula é apresentada podemos verificar que o fator associado ao mecanismo de penalização de notas, mais concretamente quando a importância é alta, foi reduzido de 2.4 para 1.75. Esta redução deveu-se ao facto de que o mesmo prejudicava em demasia as notas das máquinas com importância alta.

Pesos dos critérios

- *Patching* (P): 35%
- Antivírus (AV): 20%

- Exposição (E): 10%
- Alarmes (A): 10%
- Vizinhança (V): 25%

Fórmula

$$Risco = ((P * 0.4 + AV * 0.30 + E * 0.15 + A * 0.15)^x) * 0.75 + V * 0.25$$

- Se a importância do servidor é alta, $x = 1.75$
- Se a importância do servidor é média, $x = 1.55$
- Se a importância do servidor é baixa, $x = 1.15$

Avaliação

De maneira a verificar se o problema das versões anteriores (ganhos de máquinas importantes inferiores ou iguais aos de máquinas pouco importantes) utilizámos um caso semelhante aos utilizados nas mesmas.

Resultados

Caso 1: Existência de duas máquinas importantes com nível de risco alto (M1 e M2) e na sua vizinhança a presença de uma máquina pouco importante com nível de risco alto (M3).

Host	P	AV	I	E	A	V	Risco	Risco Final	Ganho
M1	250	250	Alta	250	250	262,2	253,0	514,2	342,4
M2	250	445	Alta	250	250	250,0	257,2	511,2	323,0
M3	250	250	Baixa	250	250	262,2	253,0	576,5	404,7

Tabela 5.7: Resultados caso 1 Versão 3

Discussão e conclusões

Como se pode observar na tabela 5.7 o problema da importância verificado nas versões anteriores não só não foi resolvido, como piorou. Nesta versão, os ganhos das máquinas importantes no caso apresentado são muito menores que os da máquina pouco importante. Isto deve-se ao facto de estarmos a aplicar as funções exponenciais a um resultado quase final. As notas das máquinas importantes nunca conseguem ser muito elevadas mesmo depois de serem atualizadas, o que diminui os ganhos.

5.4 Versão Final

Na figura 3.1 podemos observar que existe um intervalo de *ratings* para cada letra. Nesta versão do modelo de risco decidimos usar esses intervalos para conseguir distinguir as máquinas importantes das pouco importantes no que toca aos ganhos que se obtêm ao atualizá-las. Consoante a importância da máquina (alta, média ou baixa), as notas dos seus critérios podem estar no extremo inferior, a meio ou no extremo superior do intervalo do *rating* da nota correspondente.

As tabelas de tradução de letras em rating consoante a importância já foram apresentadas anteiormente neste no Capítulo 3 na Secção 3.3.4.

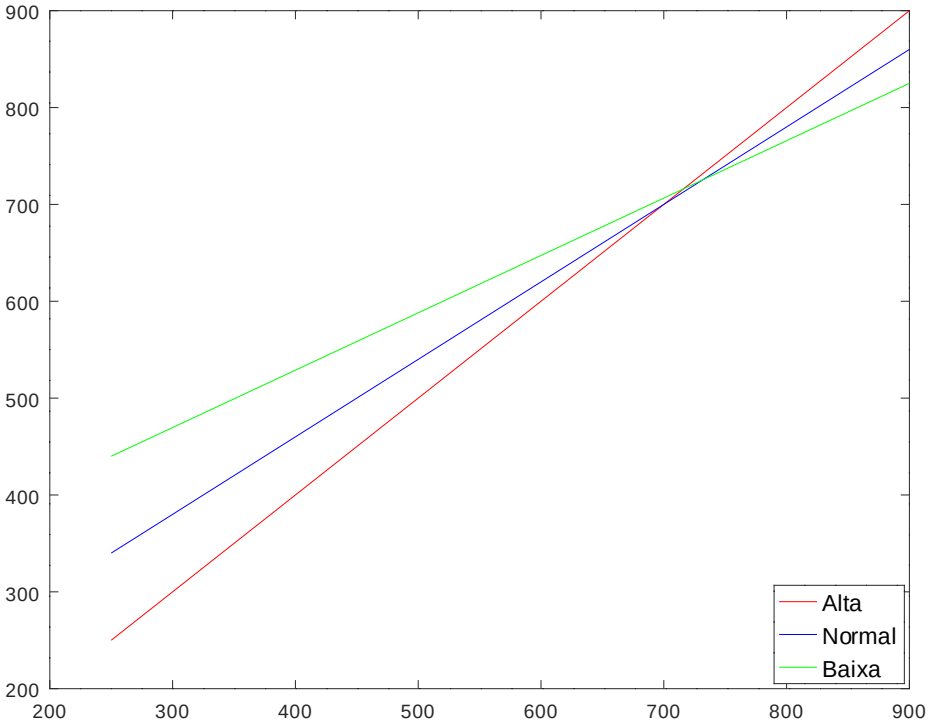

Figura 5.2: Classificações de risco em função da importância.

Na Figura 5.2 podemos observar três retas que representam as três tabelas de *ratings* segundo a importância que foram apresentadas na Secção 3.3.4. Como dados de entrada temos o intervalo completo de *ratings*. Como dados de saída temos os *ratings* correspondentes (recordamos que apenas as notas dos critérios de *patching* e antivírus é que são afetadas) tendo em conta a importância dos servidores.

Pesos dos critérios

- *Patching* (P): 35%
- Antivírus (AV): 20%
- Exposição (E): 10%

- Alarmes (A): 10%
- Vizinhança (V): 25%

Fórmula

$$Risco = P * 0.35 + AV * 0.20 + E * 0.10 + A * 0.10 + V * 0.25$$

Avaliação

À semelhança das versões anteriores, nesta versão também nos focámos nos casos onde as notas dos servidores se encontram nos extremos (notas dos critérios com A e B para o extremo positivo, E e F para o negativo).

Resultados

Caso 1: Existência de uma máquina importante com nível de risco alto (M1) e na sua vizinhança a presença de uma máquina pouco importante com nível de risco alto (M2). Ambas as máquinas com notas de *patching* e antivírus a F.

Host	P	AV	I	E	A	V	Risco	Risco Final	Ganho
M1	250	250	Alta	250	900	392,5	350,6	708,1	438,8
M2	440	440	Baixa	250	440	412,5	414,1	667,1	310,5

Tabela 5.8: Resultados caso 1 Versão final

Caso 2: Existência de máquinas importantes com nível de risco alto com as notas de *patching* e antivírus a variar entre E e F (M1, M2, M3 e M4), e na sua vizinhança a presença de uma máquina pouco importante com nível de risco alto e com ambas as notas de *patching* e antivírus a F (M5).

Host	P	AV	I	E	A	V	Risco	Risco Final	Ganho
M1	250	250	Alta	250	900	496,9	376,7	734,2	438,8
M2	250	445	Alta	250	900	484,7	412,7	731,2	387,6
M3	445	250	Alta	250	900	484,7	441,9	731,2	358,3
M4	445	445	Alta	250	900	472,5	477,9	728,1	307,1
M5	440	440	Baixa	440	900	461,3	491,3	744,3	310,5

Tabela 5.9: Resultados caso 2 Versão final

Caso 3: Existência de máquinas importantes com nível de risco baixo com as notas de *patching* e antivírus a variar entre A e B (M1, M2, M3 e M4) e na sua vizinhança a presença de uma máquina pouco importante com nível de risco baixo e com ambas as notas de *patching* e antivírus a A (M5).

Host	P	AV	I	E	A	V	Risco	Risco Final	Ganho
M1	740	740	Alta	250	900	849,4	734,3	822,3	108,0
M2	740	820	Alta	900	900	500,0	728,0	800,0	87,0
M3	820	740	Alta	900	900	803,8	815,9	875,9	75,0
M4	820	820	Alta	900	900	798,8	830,7	874,7	54,0
M5	815	815	Baixa	900	900	799,4	828,1	874,8	57,4

Tabela 5.10: Resultados caso 3 Versão final

Discussão e conclusões

Como se pode verificar na tabela 5.8, que apresenta os resultados do caso 1, o problema de distinção de ganhos entre máquinas importantes e pouco importantes que ocorria nas versões anteriores foi resolvido. Ainda que todas as máquinas possuam nota F nos critérios de *patching* e antivírus, os maiores ganhos são obtidos ao atualizar a máquina importante, tal como pretendido.

Os resultados das tabelas 5.9 e 5.10 são idênticos entre si embora para extremos opostos de notas. Enquanto que na primeira tabela as notas variam entre E e F, na segunda estas variam entre A e B. Os resultados mostram que máquinas pouco importantes apenas têm ganhos superiores quando ambas as notas de *patching* e de antivírus são inferiores (máquinas M4 e M5 em ambas as tabelas). Com esta versão do modelo conseguimos fazer com que exista uma distinção clara nos ganhos tendo em conta a importância das máquinas, tal como pretendido.

5.5 Conclusão

Para a criação do modelo de risco começámos por definir critérios para podermos classificar os servidores da PT com uma nota que representasse o seu nível de risco. Depois de definidos os critérios foi-lhes atribuído um peso consoante a sua relevância e criada uma versão inicial do modelo.

Depois de realizar alguns testes da primeira versão do modelo reparámos que havia um problema. Não havia distinção nos ganhos entre atualizar uma máquina importante ou uma máquina pouco importante. Este acontecimento era um problema pois um dos objetivos do modelo é servir de guia para as equipas responsáveis pelas atualizações das máquinas saberem que máquinas são prioritárias.

Durante algumas iterações fomos alterando a maneira de calcular o risco, modificando certas partes da fórmula que considerámos relevantes, por forma a conseguirmos resolver o problema de distinção de máquinas com importâncias diferentes.

Conseguimos resolver o problema observado durante as diferentes iterações na última versão apresentada. Esta versão cumpre ambos os objetivos delineados no início deste projeto: conseguir classificar os servidores da PT que possuem sistema operativo

Windows com uma nota que representa o seu nível de risco, e servir como guia para as equipas da PT responsáveis pelo processo de aplicação de atualizações de segurança, ao disponibilizar um indicador (ganho) que introduz uma noção de prioridade entre servidores.

Capítulo 6

Conclusão

Depois de enfrentarmos vários desafios durante o decorrer deste projeto, como por exemplo a necessidade de definir critérios que não só fizessem sentido mas que também pudessem ser alimentados com os dados disponíveis na PT, conseguimos desenvolver um modelo de risco que cumpre os objetivos. A partir dos critérios definidos que consideramos que são os mais relevantes, conseguimos classificar cada servidor da Portugal Telecom quanto ao seu nível de risco (objetivo primário). Depois de conseguido o objetivo primário, utilizando a classificação do nível de risco desenvolvemos um indicador que denominámos de "ganho". Este representa, para cada servidor, a diminuição no risco global do parque informático que ocorre caso o próprio seja atualizado. Assim sendo permite que as equipas da PT responsáveis pela atualização dos servidores saibam a ordem pela qual estes devem ser atualizados (objetivo secundário). Ainda que este indicador exista, através do *output* do modelo de risco podem-se criar diferentes estratégias de atualização. Por exemplo, uma estratégia poderia consistir em atualizar os servidores com importância alta, nota de risco média (C ou D) e cuja vizinhança seja péssima (E ou F).

No que diz respeito à evolução do projeto, foi um processo de constante aprendizagem. Fomos modificando o modelo de risco ao longo de diversas versões tentando sempre perceber o que falhou e porquê, e o que possivelmente teria que ser mudado na versão seguinte para alcançar os objetivos.

6.1 Trabalho futuro

Maioritariamente, o tempo que o modelo de risco demora a classificar os servidores é usado nas *queries* à HIDRA. Tal como foi explicado neste relatório, a informação na HIDRA encontra-se armazenada em índices. Ao fazer uma *query* é necessário especificar o índice, de maneira que em cada *query* apenas se pode retirar informação desse índice. De maneira a diminuir o tempo de execução do algoritmo do modelo de risco, o ideal seria conseguir concentrar toda a informação que este precisa num mesmo índice, o que permitiria que o algoritmo apenas tivesse de realizar uma *query* para obter toda a informação,

melhorando assim o desempenho.

Outra melhoria a fazer diz respeito ao critério da exposição. Atualmente este critério apenas classifica os servidores com estado expostos à *Internet* ou não. Embora de momento não exista informação disponível por parte da PT que possibilite esta melhoria, deveria haver uma divisão entre aqueles servidores que se encontram expostos à *Internet* baseada nos diferentes "obstáculos"que se encontram entre eles e a saída para a *Internet*. Um exemplo de obstáculo seria uma *firewall*.

Por último, dado que os dados de input do critério da importância são emulados, é necessário que essa informação comece a ser usada neste no modelo.

Glossário

CSD *Cyber Security Development and Integration*. 16

DCY Direção de *Cyber Security and Privacy*. 16

HIDRA *High performance Infrastructure for Data Research and Analysis*. 16

PT *Portugal Telecom*. 3

SCCM *System Center Configuration Manager*. 18

WSUS *Windows Server Update Services*. 13, 17, 18

Bibliografia

[1] Computer worm. https://searchsecurity.techtarget.com/definition/worm/. [Online; acedido 09-Maio-2018].

[2] Lovisa johansson: Part 1: Rabbitmq for beginners - what is rabbitmq? https://www.cloudamqp.com/blog/2015-05-18-part1-rabbitmq-for-beginners-what-is-rabbitmq.html/. [Online; acedido 30-Outubro-2017].

[3] Malware: O que é malware? https://www.microsoft.com/pt-br/security/resources/malware-whatis.aspx. [Online; acedido 16-Maio-2018].

[4] Malwarebytes: How did the wannacry ransomworm spread? https://blog.malwarebytes.com/cybercrime/2017/05/how-did-wannacry-ransomworm-spread/. [Online; acedido 16-Maio-2018].

[5] Manageengine : Desktop central. https://www.manageengine.com/products/desktop-central/windows-patch-management.html. [Online; acedido 30-Outubro-2017].

[6] Roxana elliott: The elk stack - elasticsearch, logstash, and kibana. https://www.section.io/blog/what-is-the-elk-stack-elasticsearch-kibana/. [Online; acedido 30-Outubro-2017].

[7] Solarwinds: Top 13 reasons to maximize your wsus investment. http://www.solarwinds.com/patch-manager/maximize-your-wsus. [Online; acedido 30-Outubro-2017].

[8] Wikipedia: Eternalblue. https://en.wikipedia.org/wiki/EternalBlue. [Online; acedido 16-Maio-2018].

[9] Wikipedia: It risk management. https://en.wikipedia.org/wiki/IT_risk_management. [Online; acedido 21-Maio-2018].

[10] Wikipedia: Syslog. `https://pt.wikipedia.org/wiki/Syslog/`. [Online; acedido 30-Outubro-2017].

[11] Wikipedia: Wannacry. `https://en.wikipedia.org/wiki/WannaCry_ransomware_attack`. [Online; acedido 02-Outubro-2017].

[12] Wikipedia: Wsus. `https://en.wikipedia.org/wiki/Windows_Server_Update_Services`. [Online; acedido 30-Outubro-2017].

[13] Tiago Filipe Eleutério Duarte. *An automated system to search, track, classify and report sensitive information exposed on an intranet*. PhD thesis, 2015.

[14] Allen Harper, Shon Harris, Jonathan Ness, Chris Eagle, Gideon Lenkey, and Terron Williams. *Gray hat hacking the ethical hackers handbook*. McGraw-Hill Osborne Media, 2011.

[15] I ISO and I Std. Iso 27005: 2011. *Information technology–Security techniques–Information security risk management. ISO*, 2011.

[16] Savita Mohurle and Manisha Patil. A brief study of wannacry threat: Ransomware attack 2017. *International Journal*, 8(5), 2017.

[17] Tiago Manuel Simões Sequeira. Investigação e desenvolvimento de um sistema automático de detecção, monitorização e análise da propagação de worms em redes empresariais. Master's thesis, 2011.