INSTITUTO POLITÉCNICO DE BEJA

Escola Superior de Tecnologia e Gestão

Mestrado em Engenharia de Segurança Informática

iCatch - Sistema de *Intelligence* para Caracterização de Ciberataques Multi-Etapa

João Pedro Páscoa Mendes Orvalho

2019

INSTITUTO POLITÉCNICO DE BEJA

Escola Superior de Tecnologia e Gestão

Mestrado em Engenharia de Segurança Informática

iCatch - Sistema de *Intelligence* para Caracterização de Ciberataques Multi-Etapa

Elaborado por:

João Pedro Páscoa Mendes Orvalho

Orientado por:

Professor Doutor Rui Miguel Soares Silva

2019

Resumo

iCatch - Sistema de Intelligence para Caracterização de Ciberataques Multi-Etapa

Ataques informáticos têm ameaçado utilizadores e organizações, e têm ficado cada vez mais complexos e sofisticados.

Os NIDSs (Network Intrusion Detection Systems) são das ferramentas mais importantes utilizadas para monitorizar redes de computadores. Estes sistemas fornecem informações importantes, na forma de alertas, quando atividades potencialmente indesejadas são identificadas. Contudo, cada alerta isolado é baseado na observação de uma atividade de ataque individual, levando à ausência de um modelo adequado para detetar ataques multi-etapa.

A presente dissertação, com o intuito de melhorar a segurança em redes informáticas, aborda aspetos importantes relacionados com a correlação de alertas de intrusão, propõe um sistema automático capaz de identificar e reconhecer ataques de rede multi-etapa com base em alertas provenientes de NIDSs, e descreve a implementação realizada de uma prova de conceito com a finalidade de demonstrar a potencialidade do sistema proposto.

Os resultados dos testes realizados ao sistema revelaram-se bastante promissores, mostrando através do uso do CPM (Clique Percolation Method) a possibilidade de separar ataques potencialmente não relacionados, mostrando que os algoritmos de *machine learning* não supervisionados são uma mais valia na identificação de passos de ataque, e que o reconhecimento de cenários de ataque com base na semelhança de passos de ataque individuais permite oferecer uma abordagem flexível no reconhecimento de ataques multi-etapa.

Palavras-chave: *Correlação de Alertas, Ataque Multi-Etapa, Segurança em Redes, Reconhecimento de Ataques, Identificação de Ataques.*

Abstract

iCatch - Intelligence System to Characterize Multi-Stage Cyber Attacks

Cyberattacks have threatened users and organizations and are becoming more complex and sophisticated.

NIDSs (Network Intrusion Detection Systems) are one of the most important tools used to monitor computer networks. These systems provide important information, in the form of alerts, when potentially unwanted activities are identified. However, each isolated alert is based on the observation of an individual attack activity, leading to the absence of a suitable model to detect multi-stage attacks.

This master's thesis, in order to improve security in computer networks, addresses important aspects related to the correlation of intrusion alerts, proposes an automatic system capable of detecting and recognizing multi-stage network attacks based on alerts from NIDSs, and describes the implementation of a proof of concept to demonstrate the potential of the proposed system.

The results of the system tests were quite promising, showing through the use of CPM (Click Percolation Method) the possibility to split potentially unrelated attacks, showing that unsupervised machine learning algorithms provide added value to identify attack steps, and recognizing attack scenarios based on the similarity of individual attack steps provides a flexible approach to multi-stage attack recognition.

Keywords: *Alert Correlation, Multi-Stage Attack, Network Security, Attack Recognition, Intrusion Detection.*

Agradecimentos

O presente trabalho não poderia chegar a bom porto sem o precioso apoio de diversas pessoas. Correndo o risco de injustamente não mencionar alguém, quero deixar expresso os meus profundos agradecimentos, sem qualquer ordem de importância definida:

- Ao Professor Doutor Rui Silva, por ter aceite ser o orientador, pela orientação prestada, pelo seu incentivo, paciência, disponibilidade e apoio que sempre demonstrou;
- A todos os docentes do Mestrado em Engenharia de Segurança Informática por tudo o que me ensinaram;
- A todos os colegas de Mestrado que permitiram a partilha de valores e conhecimentos, ajudando na conclusão desta etapa;
- Por último, quero agradecer à minha família pelo apoio incondicional que deram, pela compreensão e incessante incentivo que sempre demonstraram.

Enfim, quero demonstrar o meu agradecimento, a todos aqueles que, de um modo ou de outro, tornaram possível a realização desta dissertação.

Índice

Índice de Figuras

Abreviaturas e Siglas

AA	Autoassociador
CPM	Clique Percolation Method
CSV	Comma Separated Values
CVE	Common Vulnerabilities and Exposures
DDoS	Distributed Denial-of-Service
DMZ	Demilitarized Zone
EM	Expectation-Maximization
GAC	Graph-based Alert Correlation
HAC	Hierarchical Agglomerative Clustering
IDMEF	Intrusion Detection Message Exchange Format
IDPS	Intrusion Detection and Prevention System
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
JSON	JavaScript Object Notation
NIDS	Network Intrusion Detection System
NIPS	Network Intrusion Prevention System
OSI	Open System Interconnection
SO	Sistema Operativo
SOM	Self-Organizing Map
TCP	Transmission Control Protocol
TLS	Transport Layer Security

Capítulo 1

Introdução

Ataques informáticos têm ameaçado utilizadores e organizações desde o início da Internet, sendo que estes ataques são caracterizados por serem cada vez mais complexos e sofisticados, causando consideráveis prejuízos financeiros [1]. Os atacantes são movidos pelas mais variadas razões, desde motivações económicas, políticas, táticas ou competitivas, pela destruição ou dano, e fama ou vingança [2].

Os NIDSs são frequentemente implementados para monitorizar tráfego em redes de computadores. Todavia, tais sistemas são conhecidos pelos seus alertas de baixa qualidade, levando a que a revelação da estratégica dum ataque diretamente (sem processamento posterior) a partir de alertas seja impraticável [3].

Extrair informação útil de alertas isolados não é uma tarefa simples, visto que pode ser emitido um elevado número de alertas seja, por exemplo, devido a problemas de configuração em equipamentos na rede, porque se pretende obter melhor cobertura de monitorização de ataques informáticos levando a incluir diversos sensores, ou até devido à natureza do tipo de ataque. Podem também ser misturados alertas falsos positivos (resultantes, a título de exemplo, de regras genéricas) com verdadeiros positivos.

Os mecanismos de segurança podem reportar de forma redundante aspetos relativos ao mesmo evento de segurança, e as relações causais entre alertas podem não ser identificadas [4].

Além disso, os mecanismos de segurança geram frequentemente alertas de baixo nível (ex.: apenas informações como endereço IP de origem e destino, natureza da anomalia e *timestamp*, não descrevendo um problema por completo) ou alertas com informação incompleta (levando a que ataques individuais possam ser representados por vários alertas), e pode não ser possível detectar todas as tentativas de ataque (e consequentemente não serem emitidos alertas) [4, 5].

Tipicamente, os atacantes necessitam de realizar diversas etapas de intrusão sucessivamente para alcançar um objetivo final. O conjunto dessas etapas é conhecido por ataque multi-etapa, ou até cenário de ataque [1]. Num ataque multi-etapa, ações prévias alteram e/ou fornecem conhecimento a um atacante com vista a realizar ações posteriores [6, 7]. Na prática, um cenário de ataque pode consistir em diversas etapas de ataque, onde cada etapa consiste em um ou mais passos de ataque [8].

Uma característica dos IDSs (Intrusion Detection Systems) é que cada alerta (isolado) é baseado na observação de uma atividade associada a um passo de ataque individual [9]. De forma geral, este tipo de sistemas de segurança auxiliam na deteção de passos de intrusão individuais, mas não na deteção de ataques com múltiplas etapas. Isto deve-se à ausência de um modelo adequado para detetar ataques multi-etapa e à falta de um método para ligar as diversas etapas de um ataque ao comportamento de um atacante.

De modo a que seja obtida uma visão panorâmica (visão ampla de um ataque completo), fundamental para identificar as características de um ataque informático e consequentemente entender o mesmo, é necessário correlacionar a informação associada ao ataque, com vista a que seja possível posteriormente reconstruir e caracterizar cenários de ataques informáticos [10, 11].

O processo de ligação das etapas de um ataque e a identificação de cenários de ataque multi-etapa é usualmente feito de forma manual [9]. Contudo, ataques que pertençam ao mesmo cenário de ataque podem estar espacialmente e temporalmente dispersos, podendo levar a que uma análise manual seja morosa e pouco confiável, sendo propensa a erros [9, 11]. Além disso, podem existir alertas que não são relevantes e que podem ser ignorados [12].

Uma abordagem promissora para analisar automaticamente alertas de intrusão é chamada de correlação [13]. As técnicas de correlação de alertas de intrusão permitem que essa tarefa seja facilitada, pois a partir delas são descobertas as relações entre alertas ao serem extraídas características em comum que os eventos reportados possuam. Estas técnicas facilitam a interpretação dos resultados oferecidos pelos IDSs, pois a partir delas é extraída informação de alto nível, permitindo que novos significados sejam atribuídos aos alertas de intrusão [14].

Tem existido um grande esforço no campo da correlação de alertas de segurança, sendo que existem diversas técnicas propostas ao longo dos anos, cada uma com a sua própria arquitetura, técnica, capacidade, filosofia, e o seu próprio critério de desempenho [7].

Múltiplos procedimentos de correlação de alertas têm sido adaptados, entre eles:

agregação de alertas; redução de alertas; priorização de alertas; remoção de falsos positivos; descoberta de alertas em falta; previsão de sequências e tipos de ataque; e identificação das causas e relações entre alertas. No entanto, muitas destas abordagens confiam em regras pré-definidas de correlação de alertas, e/ou requerem um trabalho humano considerável para criar e/ou manter um sistema ao dependerem da definição de regras complexas de correlação e conhecimento prévio (ex.: características de ataques) para a identificação dos ataques, o que leva a dificuldades de implementação, a uma manutenção periódica notável, e a capacidades limitadas em detetar novas estratégicas de ataque [15, 16]. Muitas abordagens são ainda pouco flexíveis no reconhecimento de ataques, e não têm a eficiência que é desejável na análise de alertas, apresentando algumas limitações.

O objetivo desejado de qualquer sistema de segurança é ser o mais autónomo possível na identificação de ataques [1]. Tendo em conta o referido, o presente trabalho surge no contexto da segurança em redes de computadores com a necessidade do desenvolvimento de um sistema automático que permita, de forma prática e eficaz, a identificação e reconhecimento (sendo o reconhecimento realizado com base em cenários conhecidos) de cenários de ataque em rede multi-etapa, a partir de alertas provenientes de NIDSs com o intuito de melhorar a segurança nas redes informáticas.

Neste sentido, com base essencialmente em artigos técnicos e científicos, a presente dissertação aborda aspetos importantes relacionados com a correlação de alertas de intrusão e propõe um sistema de identificação e reconhecimento de ataques de rede multi-etapa. É ainda descrita a implementação realizada de uma prova de conceito com a finalidade de demonstrar a potencialidade do sistema proposto, contribuído assim com uma abordagem inovadora.

Este trabalho encontra-se incluído na desafiante área de investigação dos sistemas de detecção de intrusões, mais especificamente no âmbito da correlação de alertas que pode ser considerada como uma forma de aprimorar a deteção de intrusões realizada pelos IDSs. Tal como mencionado em [1], pesquisas nesta área de investigação são atualmente ainda bastante ativas, sendo que ainda se está longe de um modelo automático de deteção de ataques multi-etapa confiável.

É importante referir que o sistema desenvolvido não é um IDS, mas sim uma ferramenta de pós-processamento de alertas de intrusão provenientes de NIDSs.

A abordagem adotada teve em conta as vantagens das diversas abordagens analisadas com o intuito de dar a melhor resposta aos requisitos do presente trabalho.

Neste sentido, para identificar ataques multi-etapa é aplicado o CPM para reunir alertas prováveis de pertencer ao mesmo ataque multi-etapa, evitando desta

forma misturar alertas provenientes de ataques potencialmente não relacionados para análise, e é utilizado o algoritmo de *machine learning* não supervisionado HAC (Hierarchical Agglomerative Clustering) com o intuito de agrupar alertas relativos a um ataque multi-etapa em passos de ataque. Desta forma, não é necessário conhecimento prévio pelo sistema proposto sobre ataques de forma a identificar ataques multi-etapa.

No que diz respeito ao reconhecimento de ataques multi-etapa, são utilizados métodos matemáticos relativos ao cálculo de distâncias com vista a calcular o grau de semelhança dos ataques identificados com os cenários de ataque conhecidos (presentes numa base de conhecimento) com base em passos de ataque.

O sistema proposto foi testado com sucesso num sistema controlado com recurso a virtualização, com base em duas experiências de cenários de ataque distintos, com o IDS Snort 3, onde foram obtidos resultados bastante interessantes, que expressam a viabilidade do sistema proposto, incentivando a continuação do seu desenvolvimento no futuro.

Nestas experiências foi avaliado o desempenho do sistema a identificar *clusters* de alertas e a reconhecer cenários de ataque conhecidos, sendo que na segunda experiência foi ainda avaliada a função de divisão de ataques do sistema.

Partindo do princípio que muitos atacantes tendem a ser metódicos nos ataques informáticos que realizam (ao efetuarem procedimentos e ataques semelhantes), o sistema permitirá auxiliar na identificação de um atacante ou grupo de atacantes (ex.: equipas de *crackers*) que tenham um *modus operandi*/"assinatura" de ataque conhecido ao serem reconhecidos passos de ataque semelhantes que possam ser característicos dos atacantes.

O presente trabalho, além deste capítulo que serve de introdução essencial, encontra-se dividido em seis capítulos. No capítulo 2 é descrita a fundamentação teórica dos conceitos envolvidos no trabalho, no capítulo 3 é feito o estudo do estado da arte, no capítulo 4 é descrito o funcionamento do sistema de correlação de alertas proposto, e no capítulo 5 são apresentados os resultados experimentais obtidos com base nos testes realizados. Por fim, no capítulo 6 são mencionadas algumas considerações e trabalhos a realizar com vista à continuação de futuras evoluções do sistema, e no capítulo 7 são referidas as conclusões do trabalho realizado.

Capítulo 2

Fundamentação Teórica

Este capítulo tem como objetivo descrever a fundamentação teórica dos conceitos envolvidos no presente trabalho. Inicialmente, na secção 2.1, são referidas definições relativas aos diversos termos relacionados com o âmbito do trabalho, na secção 2.2 são referidas algumas tecnologias de deteção e prevenção de intrusões, na secção 2.3 são abordadas técnicas de análise de alertas, e a secção 2.4 finaliza com as considerações finais do capítulo.

2.1 Termos e Expressões de Segurança

Existem múltiplos termos, relacionados com os sistemas de correlação de alertas, que serão abordados ao longo deste trabalho e que podem ser interpretados de diferentes formas. Neste sentido, de modo a evitar que tais termos sejam mal interpretados, é importante clarificar os seus significados adotados neste trabalho para que os mesmos fiquem familiares.

2.1.1 Segurança de Computadores

Proteção oferecida a sistema de informação com vista a preservar a integridade (as informações são modificadas apenas de uma maneira especificada e autorizada), a disponibilidade (acesso quando requisitado, redundância e resistência a falhas) e a confidencialidade (acesso restrito a entidades devidamente autorizadas) dos recursos dum sistema de informação (incluindo *hardware*, *software*, *firmware*, informações/dados e telecomunicações) [17, 18].

2.1.2 Atacante

Indivíduo ou sistema singular autónomo que tenta realizar um ou mais ataques para atingir um objetivo [19].

2.1.3 Vulnerabilidade

Fraqueza num sistema, permitido ações não autorizadas [19].

2.1.4 Vetor de Ataque

Meio pelo qual um atacante pode obter acesso não autorizado a um dispositivo ou rede para propósitos nefastos (ex.: dispositivo removível, anexo de email, página de Internet) [20].

2.1.5 Ataque Informático

Qualquer tipo de atividade maliciosa que tenta coletar, perturbar, negar, degradar ou destruir recursos dum sistema de informação ou a informação em si [21].

De acordo com a ISO/IEC 27000, um ataque no contexto das redes de computadores é definido como uma tentativa para destruir, expor, alterar, desativar, roubar ou ganhar acesso não autorizado ou fazer uso não autorizado de um ativo [1, 22].

2.1.6 Intrusão em Sistema Informático

Associado a um ataque bem sucedido, após o qual o atacante obtém acesso ao sistema alvo. No caso da intrusão explorar uma vulnerabilidade ainda não corrigida ou tornada pública, a mesma designa-se por *zero-day* [23].

2.1.7 Incidente de Segurança

Ações tomadas através da utilização de uma rede de computadores que resultam num efeito adverso sobre um sistema de informação e/ou a informação aí armazenada [21].

Um incidente de segurança corresponde a um ou mais alertas indesejados ou inesperados que tenham grande probabilidade de comprometer as operações de um negócio e ameaçar a segurança da informação [24].

2.1.8 Evento

Um evento é qualquer ocorrência observável num sistema ou rede [20]. Resulta de ações direcionadas a alvos específicos. Um único evento pode não gerar qualquer alerta, mas pode também gerar múltiplos alertas [19, 14].

2.1.9 Ameaça

Possibilidade de violação da segurança que existe quando há uma circunstância, capacidade, ação ou evento que pode causar a quebra da segurança e causar danos [18].

2.1.10 Risco

Potencial prejuízo para uma organização em caso de ataque [17].

2.1.11 Alerta

Também conhecido como alarme, é uma mensagem de aviso que pode indicar um ataque ou um evento suspeito [14]. Um único alerta pode descrever um conjunto ou sequência de eventos. Todo o alerta é suspeito, mas um evento não é necessariamente suspeito [25].

Característica/atributo de alerta

Campo específico contido dentro de um alerta. Alguns exemplos são: data, nome do alerta, origem, e destino [14].

Tipos de alertas

Uma característica sempre presente num sistema de classificação como um IDS é a impossibilidade de obter resultados absolutamente corretos. Desta forma, um alerta pode ser entendido como:

- Verdadeiro positivo: Acontece quando é gerado um alerta e ocorreu uma tentativa de ataque;
- Verdadeiro negativo: Quando não é gerado qualquer alerta nem ocorreu qualquer tentativa de ataque;

- Falso negativo: Quando não é gerado um alerta e ocorreu uma tentativa de ataque;
- Falso positivo: Acontece quando é gerado um alerta mas não ocorreu uma tentativa de ataque, tratando-se de ações legítimas.

Apesar de ser impossível eliminar completamente os falsos positivos e os falsos negativos, um IDS deve ser configurado de forma a reduzir os resultados erróneos através de um processo de afinação. Os limiares de rejeição/aceitação devem ser escolhidos adequadamente de acordo com as necessidades.

Correlação de alertas

Processo de múltiplas etapas que recebe alertas como entrada e atua como uma plataforma para gerir e entender os alertas [3].

Cluster de alertas

Coleção de alertas que são semelhantes entre si e são dissemelhantes em relação aos alertas pertencentes a outros *clusters* [26].

Hiper Alerta

Quando dois ou mais alertas são fundidos como parte do processo de correlação de alertas, o resultado é chamado de meta-alerta ou hiper alerta, ou seja, alertas com informações generalizadas e que representam os vários alertas que deram origem ao hiper alerta, levando a um maior nível de abstração e consequentemente permitindo facilitar a análise humana [27].

2.1.12 Estratégia de Ataque

Uma estratégia de ataque é definida como sendo um ataque completo lançado pelo atacante, que consiste em passos e etapas de ataque, sendo que cada etapa pode incluir um ou mais passos de ataque.

Ao contrário de alguns trabalhos nesta área, na presente dissertação é realizada a distinção do conceito de etapa e passo de ataque.

Na Figura 2.1 é possível verificar uma ilustração, criada com base num esquema de [3], que facilita a compreensão dos distintos termos em questão.

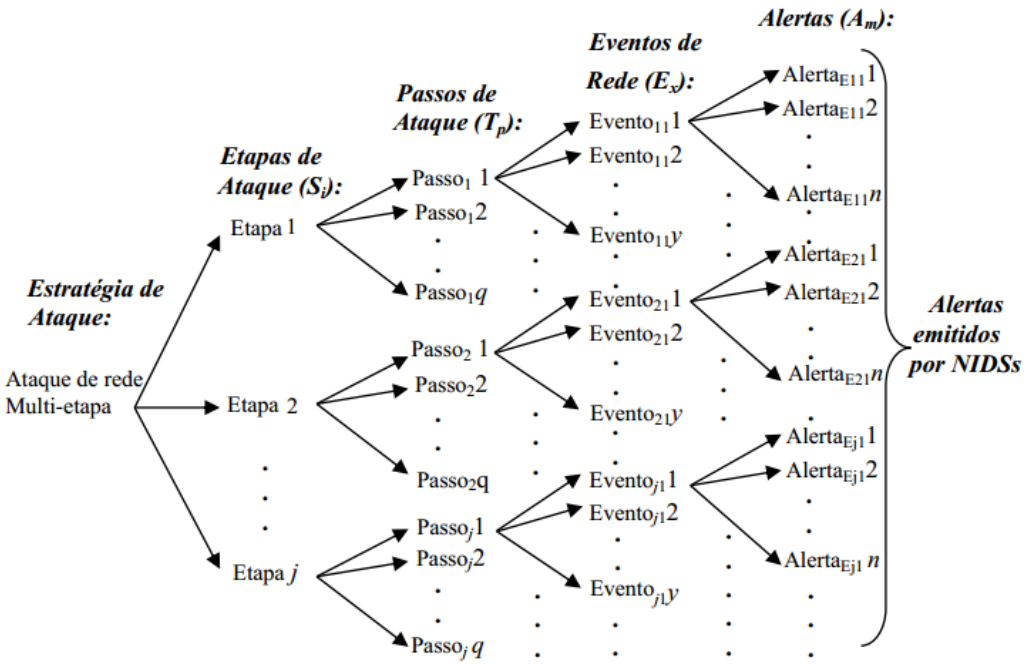

Figura 2.1: Estrutura de cenário de ataque em rede

Existe um conjunto de j etapas de ataque indicado por $S_i = \{S_1, S_2, ..., S_j\}$ num ataque de rede multi-etapa, sendo que S_i contém q passos de ataque (tendo em conta o objetivo final do atacante).

Cada passo de ataque é indicado por T_p, onde $p = 1, 2, ..., q$ e $T_p \subseteq S_i$, sendo que T_p contribui com y eventos que são avaliados por um NIDS.

Um evento de rede identificado como malicioso é indicado como E_x, onde $x = 1, 2, ..., y$ e $E_x \subseteq T_p \subseteq S_i$.

Para todo o E_x ocorrido na rede, um NIDS emitirá n alertas, sendo que um alerta é indicado por A_m, onde $m = 1, 2, ..., n$ e $A_m \subseteq E_x \subseteq T_p \subseteq S_i$ [3].

De forma geral, um ataque lançado por um atacante pode ser descrito em etapas relacionadas entre si, sendo que alguns ataques podem servir de preparação para outros ataques [10].

Um famoso exemplo de um ataque multi-etapa é o cenário simulado especificado em [28], que contém cinco etapas, nomeadamente: descoberta de *hosts* ativos (*host scanning*) em sub-redes, com os *hosts* descobertos são localizadas as máquinas Solaris com a ferramenta de administração remota sadmind em execução, invasão através de exploração de vulnerabilidade de Solaris sadmind, instalação de *daemon* DDoS (Distributed Denial-of-Service) responsável por receber e executar os comandos do atacante, e realização de ataque DDoS a partir de *hosts* comprometidos.

Numa etapa de descoberta de *hosts* ativos podem ser realizados passos de ataque

distintos tais como: TCP SYN ping, NetBIOS *scan*, ARP *scan*, envio de pacotes ICMP Echo *request*, ou até de uma forma mais despercebida através da análise (passiva) de DHCP *requests* cada vez que um *host* se liga à rede.

Num ataque podem ser adotadas estratégias de ataque distintas para atingir os objetivos de ataque. As estratégias podem incluir [17]:

- Ataques diretos - O atacante ataca o alvo sem qualquer intermediário (sem contar com o encaminhamento normal de tráfego). Exemplo: ataques de *cache poisoning*;
- Ataques progressivos - O atacante usa *hosts* intermediários até chegar ao alvo, dificultando o seu bloqueio visto que esta estratégia de ataque sugere que existam várias opções até chegar ao alvo. De certa forma, esta estratégia pode ser entendida como uma série de ataques diretos aplicados de forma sucessiva;
- Ataques em massa - O atacante compromete um grupo de *hosts* e utiliza-os em conjunto contra o alvo. Em alguns casos, o grupo de *hosts* atacantes (que são também vítimas) são comprometidos num ataque para serem mantidos e usados em ataques posteriores. É utilizado particularmente em ataques de DDoS, ex.: através de *botnets*;
- Ataques de desorientação - O atacante gera tráfego para confundir ou distrair os defensores da rede de lidarem com os "verdadeiros" ataques (ex.: ataque direto) através de um ataque mascarado (ex.: com recurso a IP *spoofing*) e/ou através de "manobras de diversão" (onde são simulados ataques contra recursos que os atacantes sabem que os defensores devem proteger), aumentando assim as hipóteses de um ataque de sucesso.

A Figura 2.2, criada com base no trabalho [17], retrata os diversos tipos de estratégias de ataque mencionadas.

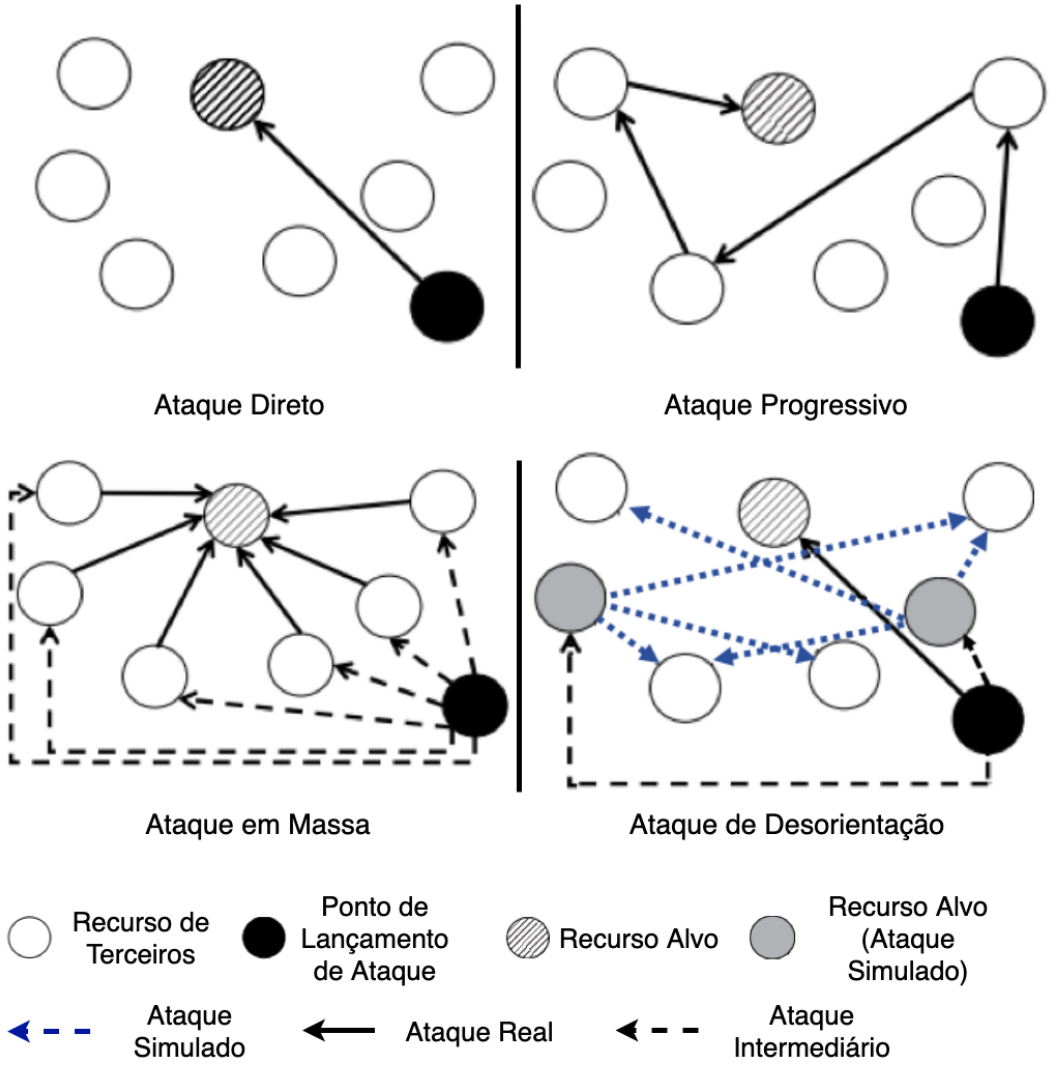

Figura 2.2: Tipos de estratégias de ataque

2.1.13 *Exploit*

Meio pelo qual um atacante tira partido de uma vulnerabilidade, podendo ser desde poucas linhas de código a blocos de código com maior dimensão e complexidade. [29].

Payload

Código que o atacante quer que a vítima execute (ex.: *reverse shell*).

Shellcode

Tradicionalmente é o código binário que executa uma *shell* (ex.: Meterpreter *shell*). Contudo, pode ser definido como o código que é executado assim que um *exploit* tem sucesso. É uma série de instruções usadas como um *payload* aquando da exploração de uma vulnerabilidade.

2.2 Deteção e Prevenção de Intrusões

Alertas de segurança podem ser provenientes de diversas tecnologias de uma infra-estrutura de sistemas de informação. Nesta secção são referidas algumas tecnologias de deteção e prevenção de intrusões.

2.2.1 Firewall

Firewall é um componente ou conjunto de componentes, de *software* e/ou *hardware*, utilizados para dividir, controlar, e restringir o acesso entre partes protegidas de uma rede e a Internet, ou entre diferentes grupos de redes [24, 30].

Estes dispositivos são dedicados para análise e bloqueio de tráfego não desejado, tradicionalmente com base no *header* dos pacotes, sendo adequados para impedir apenas certos tipos de ataques pois, de forma geral (com exceção, a título de exemplo, de *firewalls* de nível 7 do modelo OSI (Open System Interconnection)), não têm a capacidade de procurar por anomalias ou padrões específicos no conteúdo dos pacotes, tal como os IDSs têm. Por estas razões, as *firewalls* devem trabalhar em conjunto com os IDSs na proteção das redes [31, 32].

As *firewalls* podem ser consideradas com sendo de natureza *stateless* ou *statefull*. O tipo *stateless* consome menos recursos pois não armazena registos das ligações, lidando com regras explicitamente definidas por um administrador. Por outro lado, o tipo *statefull* necessita de mais recursos pois armazena os registos das ligações, sendo possível escrever regras mais flexíveis, tal como: "Permitir ligações originadas de uma máquina local e não aceitar ligações provenientes da Internet, exceto as que sejam respostas às ligações iniciadas pela rede local"[33].

2.2.2 IDS

Visto que é irrealista construir um sistema perfeitamente seguro, os IDSs têm um papel importante a desempenhar na identificação de ameaças e ações mal-intencionadas.

A comunidade de segurança de computadores desenvolveu diversos IDSs, cada um com o seu foco e com as suas funcionalidades [24]. Estes dispositivos, de *hardware* e/ou *software*, têm como função a monitorização de eventos em um ambiente que seja um potencial alvo de um ataque informático [34].

A análise dos eventos observados pode revelar a existência de ameaças relacionadas com a exploração de vulnerabilidades tecnológicas dos sistemas monitorizados ou de violações das políticas de segurança de uma organização.

Os dois tipos mais comuns de IDSs são os de deteção por assinaturas (também chamados de deteção por abuso) e de deteção por anomalia, sendo que ambos têm em comum um resultado: alertas [10, 35]. Os IDSs podem funcionar em ambos os modos, podendo operar, de forma complementar, a nível de rede (NIDS) ou de *hosts* (HIDS) [30].

Existem diversos IDSs atualmente, entre eles: Snort, Suricata, Bro e OSSEC. O Snort [36] é um dos IDSs mais populares, sendo atualmente desenvolvido pela Cisco. Este IDS, de código aberto, é baseado em libpcap, e pode atuar como *sniffer* e/ou *logger* de pacotes e também como NIDS/NIPS (Network Intrusion Prevention System).

Os IDSs do tipo HIDS baseiam o seu funcionamento na monitorização de diferentes tipos de entidades dentro de um sistema computacional, tais como: sequência de chamadas (*system calls*) ao SO (Sistema Operativo), alterações ao sistema de arquivos, arquivos de *log*, configuração do SO, entre outros.

No que se refere aos IDSs do tipo NIDS, tipicamente operam sobre a análise de pacotes que trafegam num segmento de rede. A análise pode envolver tanto o *header* quanto o *payload* dos pacotes monitorizados.

Os sistemas de deteção de anomalias estabelecem um padrão de uso normal (ou perfil), sendo que esse comportamento é descrito através de heurísticas e estatísticas, e é comparado com o comportamento corrente com base em diversos indicadores, tais como o volume de tráfego, tipos de acesso, uso do processador e disco, entre outros [10]. Os perfis gerados podem ser estáticos (alterados durante o funcionamento do sistema e devem ser reconstruidos periodicamente) ou dinâmicos (alterados em tempo real, em função dos eventos observados).

Mudanças percebidas em relação ao padrão são traduzidas em alertas de intrusão. Estes sistemas procuram determinar aquilo que foge do comum, ou seja, do comportamento considerado normal num sistema. Numa etapa inicial devem ser recolhidos alguns dados capazes de definir certos tipos de comportamentos do ambiente em análise. Esses dados são então submetidos a uma "fase de treino". Nesta fase são estabelecidos os limites do comportamento normal e definidos os limiares

para a deteção dos eventos [34].

Relativamente aos sistemas baseados em assinaturas, os mesmos codificam sequências de ações como assinaturas de ataques que representam o comportamento de ataques já conhecidos. Essas assinaturas são então confrontadas com as atividades correntes do sistema: caso a intrusão previamente modelada ocorra e combine com a assinatura gerada a deteção será realizada [24, 34]. Esta é a técnica mais utilizada na prática [37].

Embora os sistemas baseados em assinaturas sejam conhecidos pela sua boa precisão de deteção devido ao número reduzido de falsos positivos que são detetados, a grande dificuldade dos mesmos é a deteção de novos ataques ou implementações distintas de um ataque (onde se incluem as ameaças polimórficas e *zero-day* [38]), visto que estes sistemas utilizam os comportamentos de ataques previamente conhecidos para realizar a deteção. Por esta razão, este tipo de sistemas devem ser atualizados regularmente.

No que se refere aos IDSs baseados em anomalia, estes têm como principal vantagem a capacidade de detetar novos ataques, no entanto, geralmente não apresentam informações claras sobre os eventos maliciosos, sofrem com a deteção de altas taxas de falsos positivos e a definição do comportamento normal para um sistema pode tornar-se difícil e extensiva [30, 34].

É importante referir ainda que existem essencialmente dois métodos de análise, que podem ser utilizados em conjunto, adotados pelos NIDSs para a análise dos dados que recebem, nomeadamente [31]:

- Análise baseada em pacotes - Também conhecido por NIDSs tradicional, processa cada pacote que recebe. Além dos cabeçalhos, também analisa o *payload*. Apesar de produzir menos falsos alertas, exige mais tempo de processamento;

- Análise baseada em fluxos de pacotes - Em vez de analisar todos os pacotes, analisa informação resumida de pacotes relacionados na forma de fluxo, sendo que a informação a ser analisada é reduzida e consequentemente é exigido menos tempo de processamento. É utilizado normalmente em redes de alta velocidade.

Um fator que afeta adversamente a eficácia de IDS consiste na utilização, por parte dos atacantes, de técnicas evasivas que alteram o formato ou a temporização das suas mensagens para que estas aparentem ser inofensivas. Alguns IDS possuem métodos que detetam técnicas evasivas, realizando um processamento das mensagens de forma semelhante ao sistema alvo [39].

Os IDSs, por si só, não são confiáveis o suficiente, levando a que os alertas necessitem de serem correlacionados com vista a revelar as relações entre alertas para que as estratégias de ataque sejam identificadas [40].

Os dados obtidos de um IDS podem ser utilizados para mitigar um ataque no momento em que o mesmo ocorre, ou para uma análise posterior da segurança da rede, e até como evidência de um ataque.

Frequentemente as informações relativas a atividades maliciosas são recolhidas de forma centralizada usando um SIEM (Security Information and Event Management), utilizado para uma melhor organização, auxilio na deteção, análise, e mitigação de incidentes de segurança, permitindo ter uma visão dos eventos de segurança de toda uma organização.

2.2.3 IDPS

Um IDPS (Intrusion Detection and Prevention System), também conhecido como IDS ativo, é uma das formas eficazes de proteger uma rede, oferecendo uma plataforma unificada que permite monitorizar o estado da mesma (rede), impedindo que ataques causem danos através de medidas de resposta apropriadas.

Estes tipos de sistemas consistem em três domínios: deteção de intrusões (relativo aos IDSs), correlação de alertas (processa e analisa alertas com vista a descobrir as relações entre eles), e prevenção de intrusões (relativo aos IPSs (Intrusion Prevention Systems), sugere um plano de resposta apropriado para a intrusão detetada, de modo a impedir que a rede seja danificada) [3].

2.2.4 Localização

A localização dos sistemas de segurança referidos numa rede de computadores varia consoante a categoria a que pertencem e de acordo com os requisitos de cada sistema. De forma geral, um NIDS é instalado de forma passiva, capturando as comunicações num segmento de rede, sem interagir diretamente com o tráfego de rede. No que se refere ao IPS, o mesmo deve ser instalado em linha antes do segmento de rede que se pretende monitorizar, de forma a poder bloquear ativamente a comunicação em caso de deteção de intrusões.

Na figura 2.3 encontra-se ilustrado a arquitetura típica de uma rede.

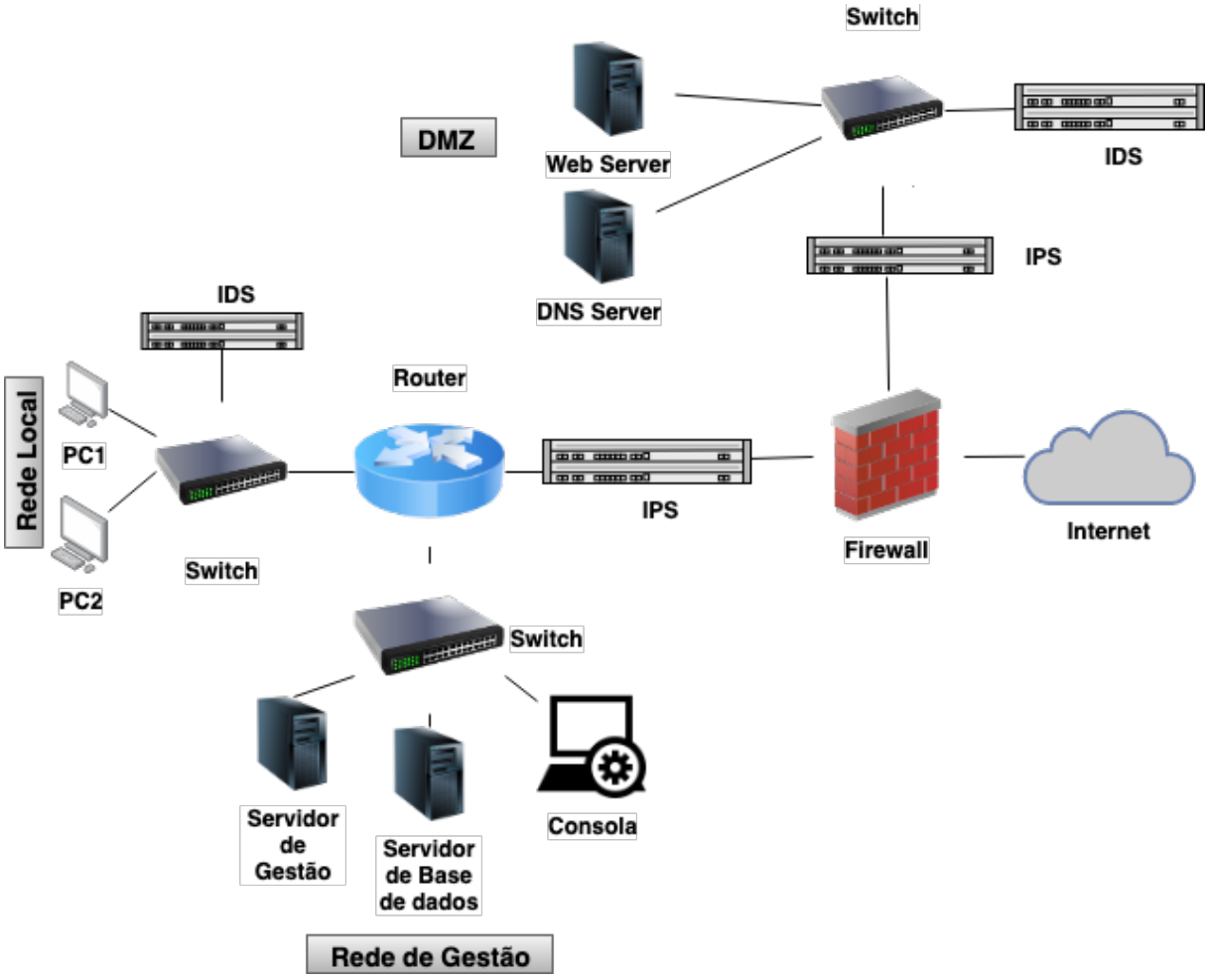

Figura 2.3: Arquitetura típica de rede

É também comum a utilização de um dispositivo "Network TAP" para dividir o tráfego de rede com o intuito de criar uma cópia do tráfego para inspeção.

Os IDSs e IPSs podem enviar alertas para serem geridos na "Rede de Gestão" com vista a que seja realizada a correlação de eventos de forma a que sejam reconhecidos padrões de ataque de modo mais abrangente ao invés de serem examinados ataques detetados individualmente por cada um dos IDS/IPS.

Embora a deteção de intrusões esteja dispersa por diversos pontos da rede, o controlo e monitorização do seu funcionamento está concentrado num ponto central.

Enquanto que a consola é utilizada para a monitorização e/ou configuração de sensores, bem como os vários servidores do sistema, o servidor de gestão realiza uma análise mais detalhada da informação recolhida pelos sensores e efetua a correlação de eventos registados pelos vários sensores. O servidor de base de dados pode, a

título de exemplo, armazenar a informação gerada pelos sensores.

É importante referir que atualmente existem soluções que incluem num único dispositivo várias soluções de segurança. Por exemplo, um UTM (Unified Threat Management) inclui funções como: *antimalware*, *firewall* de rede, deteção e prevenção de invasões, entre outras.

DMZ

Uma DMZ (Demilitarized Zone), conhecida também como rede de perímetro, é uma sub-rede lógica ou física que contém e expõe serviços na fronteira externa de uma organização com uma rede maior e não confiável (geralmente a Internet). Os dispositivos presentes nesta região, ou seja entre a rede confiável (geralmente a rede local) e a rede não confiável estão na zona desmilitarizada.

A DMZ tem como função manter todos os serviços que possuem acesso externo (tais como servidores HTTP, FTP, E-mail, DNS, entre outros) juntos numa sub-rede, limitando assim, o potencial dano em caso de comprometimento de algum desses serviços por um atacante. De forma a atingir esse objetivo, os computadores presentes numa DMZ não devem ter forma de acesso à rede privada local (rede confiável).

Esta configuração é realizada através do uso de *firewalls*, que realizam o controlo de acesso entre a rede local, a Internet, e a DMZ [41].

2.2.5 Formato IDMEF

Para cada tipo de ferramenta de segurança uma saída é utilizada visando o propósito de cada ferramenta. Devido às inúmeras formas e focos de deteção de intrusões, cada IDS realiza uma análise e emite um alerta num formato particular, focado na sua metodologia de deteção.

Com vista a alcançar o principal objetivo da correlação de alertas, é recomendado que exista uma padronização desses alertas. A obtenção de uma saída padronizada é importante para que exista uma independência no uso dos variados sistemas de defesa de intrusões [42].

Com o objetivo de viabilizar a interação e integração de distintas tecnologias de segurança, o Internet Engineering Task Force publicou um padrão orientado a objetos voltado para alertas de segurança denominado IDMEF (Intrusion Detection Message Exchange Format), que se encontra descrito no RFC 4765 [43]. O padrão define os formatos dos dados e procedimentos para trocas de informações entre sistemas de deteção de intrusão e sistemas de gestão de segurança [24].

Este padrão é implementado em XML (Extensible Markup Language) e tem sido utilizado em diversos trabalhos de correlação de alertas como forma de representar os alertas [8, 24].

Este é um padrão bastante dinâmico que pode permitir desde a padronização de um simples alerta, com poucas informações, até um alerta mais robusto que contemple informações extra [34]. Desta forma, o IDMEF pode exibir informações bastante simples ou até mesmo um alerta mais detalhado. Essa flexibilidade deriva da necessidade de atender os mais diversos tipos de alertas que podem ser gerados pelos múltiplos sistemas de segurança [42].

O IDMEF está principalmente preocupado com as regras de escrita. No entanto, é importante referir que é comum os IDSs escolherem diferentes nomes para classificar os alertas, fornecerem informação incompleta para determinados atributos, ou decidirem incluir campos adicionais para guardar dados relevantes [27]. Neste sentido, uma classificação comum de ataques pode ser necessária para que exista interoperabilidade entre entidades distintas [44].

2.3 Análise de Alertas

Os problemas associados aos IDSs levaram ao surgimento de diversos métodos de processamento de alertas.

O tratamento de alertas de intrusão pode ser dividido em duas categorias: classificação de alertas e correlação de alertas [10, 25]. No entanto, é importante referir que é utilizado frequentemente o termo de correlação de alertas como um nome genérico que abrange diversas técnicas distintas de processamento de alertas, entre as quais: normalização, correlação, agregação e fusão [7]. Neste sentido, a análise de alertas é popularmente conhecida como correlação de alertas [40].

A categoria de classificação de alertas corresponde aos métodos que manipulam alertas com base nos seus atributos, geralmente realizando agrupamentos que representam categorias de alertas com vista a oferecer resultados mais relevantes. Este processo tem como objetivo classificar os alertas pelos seus tipos com vista a, por exemplo, remover falsos positivos, reduzir alertas redundantes, identificar causas raiz (a razão para tal alerta ocorrer), diferenciar entre ataques realizados com sucesso e sem sucesso ou priorizar alertas (de modo a avaliar a importância dos alertas gerados pelos sensores, com base em fatores como a criticidade do alvo/origem, a criticidade do ataque e a certeza do alerta) [10]. Estes métodos são utilizados para pré-processamento de alertas e podem levar à perda de informação [25].

Em relação à categoria de correlação de alertas, a mesma procura entender as relações de causalidade entre os alertas, buscando ligar alertas numa sequência lógica, tentando, a título de exemplo, reconstruir um cenário de ataque multi-etapa ou prever próximos tipos de alertas.

Ambas as categorias podem recorrer a técnicas estatísticas, *machine learning*, regras para a criação de relações entre alertas, entre outros. Os métodos de classificação e os métodos de correlação de alertas são, geralmente, utilizados em conjunto [10].

A diferença entre a correlação de alertas (na sua essência) e outras técnicas de análise de alertas, é que a mesma tenta encontrar relações entre alertas, mostrando o progresso de ataques [35]. Com o intuito de simplificar, é utilizado no presente trabalho o termo de correlação de alertas como uma expressão genérica.

2.3.1 Correlação de Alertas

Os alertas de segurança são gerados de forma independente, contudo estes alertas podem ter ligações entre si. A correlação de alertas pode ser definida como um processo composto por diversos componentes com o propósito de analisar alertas e fornecer uma visão de alto nível sobre o estado de segurança da rede em observação [45].

A correlação de alertas define o processo automático que encontra a relação entre alertas e os seus atributos, sendo que tais relações são cruciais para revelar o comportamento de um atacante (em termos de estratégia de ataque) [3].

Um sistema de correlação de alertas pode, a título de exemplo, receber alertas de sensores distintos e formatar e padronizar alertas de intrusão, reduzir falsos alertas, eliminar alertas de intrusão redundantes, detetar padrões de ataque, priorizar alertas de intrusão, prever estados de ataques futuros, identificar os objetivos do atacante, descobrir cenários de ataque e reconhecer estratégias de ataque, detetar a raiz do problema em ataques, e reconstruir cenários de ataque a partir dos alertas obtidos, permitindo oferecer uma melhor compreensão dos incidentes ocorridos. É essencial referir que é possível variar na abordagem da correlação, dependendo dos objetivos de cada sistema [46, 14, 42, 45, 3].

Diversas abordagens têm sido propostas para dar resposta aos desafios da correlação de alertas. Os algoritmos de correlação de alertas, segundo o artigo [46], podem ser divididos essencialmente em três categorias, com base nas suas características. Estes algortimos podem ser baseados em: semelhanças, conhecimento, e estatísticas.

Baseados em Semelhanças

Alertas que pertencem ao mesmo ataque individual têm frequentemente atributos semelhantes [47]. Além disso, alertas semelhantes tendem a ter causas semelhantes ou efeitos semelhantes sobre os recursos de uma rede [45].

Esta abordagem correlaciona alertas com base nas semelhanças das características dos mesmos, tais como o endereço de origem, endereço de destino, e número de porta, sendo que alertas com um elevado grau de semelhança são correlacionados [48].

Apesar deste tipo de técnicas terem a vantagem de conseguirem agrupar alertas sem a definição precisa dos tipos de ataque, não conseguem encontrar a relação casual entre alertas, visto que estes métodos agrupam alertas com base nas semelhanças dos seus atributos [46, 44].

Os trabalhos [49, 27] são das referências mais citadas na literatura.

Baseados em Conhecimento

Os algoritmos existentes nesta categoria são divididos em duas principais subcategorias, nomeadamente pré-requisitos e consequências, e cenário de ataque.

Na subcategoria pré-requisitos e consequências cada incidente é encadeado a outro incidente através de uma rede de conjunções e disjunções predefinidas gerando uma rede de possíveis ataques.

Visto que os ataques e as suas variantes geram normalmente um elevado número de possíveis cenários, a utilização de regras (com pré-condições e pós-condições de ataques) tem sido empregada para colmatar este problema, reduzindo o número de possíveis cenários de ataque armazenados [45].

Embora estes algoritmos não necessitem de uma definição precisa de cada cenário de ataque como os algoritmos baseados em cenários, é necessário um conhecimento prévio para a definição dos pré-requisitos que podem gerar os ataques e dos possíveis resultados desses ataques. Desta forma, não é necessário inserir uma grande quantidade de regras de correlação, bastando declarar as condições necessárias para que um ataque seja realizado com sucesso e as possíveis consequências do ataque em questão. Contudo, diversas abordagens deste tipo, descrevem as pré e pós condições de forma manual, levando a que seja um processo intensivo e difícil [25, 50].

Durante o processo de construção dos pré-requisitos e consequências, os mesmos são construídos um por um sem definir a sequência dum ataque, visto que a mesma é construída automaticamente através da ligação dos pré-requisitos e consequências [1].

Com o conhecimento dos pré-requisitos e consequências, é possível correlacionar alertas relacionados ao identificar as relações causais entre eles (ex.: ao realizar a correspondência entre as consequências dos alertas anteriores com os pré-requisitos dos alertas posteriores) [25]. A relação causal pode ser utilizada para tentar prever os alertas consequentes possíveis [10].

É importante referir que nesta abordagem um novo ataque pode não conseguir ser "emparelhado" com outros ataques caso os seus pré-requisitos e consequências não estejam definidos [51]. Além disso, informação sobre o ambiente e o estado do sistema a ser monitorizado pode ser especialmente crucial neste método [47].

Esta abordagem pode ser adotada, por exemplo, através das linguagens de modelação de ataques (ex.: LAMBDA [52] e JIGSAW [53]) utilizadas para descrever as pré e pós condições [47].

Os trabalhos [54, 55] são dos mais citados na literatura.

Na subcategoria cenário de ataque os algoritmos são baseados na ideia que muitas intrusões incluem diversos passos que devem ser executados um por um, para que um ataque seja realizado com sucesso. Portanto, os alertas emitidos por sistemas de segurança podem ser comparados com os passos de intrusão pré-definidos e correlacionada a sequência de alertas relativas a cada ataque [46].

Neste método a relação de causalidade é especificada em termos de cenários, onde os alertas correlacionados podem ser combinados para construir um cenário de ataque [40].

Tipicamente, os trabalhos nesta área têm sido focados em dois métodos: o uso de modelos formais definidos por especialistas humanos para especificar os cenários de ataque ou utilizando *machine learning* para criar tais cenários [45].

O principal desafio nestes algoritmos é definir cenários de ataque mesmo que existam métodos automáticos de aprendizagem de cenários de ataque (ex.: com *machine learning*) [46]. Além disso, geralmente este tipo de algoritmos falham na correlação de alertas de ataques desconhecidos, estando limitados a situações conhecidas. Destes forma, é necessário uma atualização frequente da base de conhecimento de ataques [56, 14].

Diversas linguagens de correlação formal para descrever cenários de ataque têm sido desenvolvidas (ex.: LAMBDA, STATL [57] e ADeLe [58]) [47, 44].

O trabalho [57] é uma referência na literatura.

Baseados em Estatística

Estes algoritmos seguem o conceito de que ataques relevantes vão ter dados estatísticos semelhantes e como tal podem ser associados.

Os algoritmos estatísticos puros não têm conhecimento prévio de cenários de ataque, portanto novos cenários de ataque podem ser identificados [14].

A motivação para a realização da análise estatística é que todos os ataques multi-etapa têm semelhanças estatísticas nos seus atributos, e que os passos de ataque têm relações causais. Exemplo: se o passo X é a causa do passo Y, então X tem que preceder a Y e muito provavelmente os passos de ataque decorrem juntos num curto espaço de tempo. Esta informação pode ser útil, a título de exemplo, para oferecer níveis de confiança a alertas (com base em repetições de alertas equivalentes), para prever a ocorrência de futuros alertas, e para perceber que tipos de alertas podem causar um alerta do tipo A e como a probabilidade condicional de A está relacionada com as suas causas, podendo até ser possível capturar a relação temporal entre os alertas [44].

Um exemplo prático desta abordagem é executar um sistema *offline* com um conjunto de dados para treino, e analisar e guardar as correlações para serem utilizadas posteriormente *online* para a correspondência de padrões [47, 25, 46].

Apesar desta abordagem não necessitar de conhecimento sobre ataques para operar, não funciona em vários domínios e apresenta uma taxa de erro maior [46, 25].

É comum este tipo de métodos utilizarem redes bayesianas, cadeias de Markov e testes de causalidade de Granger.

Os trabalhos [5, 59] são uma referência na literatura.

Considerações

A atribuição de um algoritmo a uma categoria é baseado no facto de o algoritmo ter mais semelhanças com a categoria em questão. É importante referir que a categorização de um algoritmo pode não ser completamente precisa, já que pode existir a partilha de características de categorias distintas. As técnicas híbridas utilizam características das diferentes técnicas mencionadas para a correlação de alertas [46, 14].

Dado que a segurança informática se encontra incluída num contexto bastante dinâmico, a abordagem com recurso a regras pré-definidas leva a que exista uma necessidade de melhoria constante das regras de correlação existentes, ao ajustar os parâmetros que devem seguir a evolução do sistema. Ao não existir uma rápida criação de novos padrões de correlação, até os eventos críticos que sejam emitidos por uma aplicação de segurança com as últimas atualizações podem ser descartados por não fazerem parte de nenhum padrão de correlação [15].

Os sistemas baseados em conhecimento podem levar a que seja exigido uma atualização periódica dos padrões de ataque visto que os mesmos podem mudar drasticamente [32].

Não existe uma única solução que seja considerada a "melhor" em termos de precisão e/ou complexidade, para resolver um problema genérico de correlação de alertas. Contudo, existem estudos que indicam uma tendência na adoção de diferentes abordagens para dar uma melhor resposta aos problemas em redes complexas [14].

2.3.2 Origem do Conhecimento sobre Ataques

Uma forma de diferenciar os trabalhos nesta área de investigação é através da origem da informação no processo de identificação de ataques multi-etapa. De acordo com [1], os métodos podem ser classificados em: manuais, supervisionados, ou de extração automática.

Nos métodos manuais o conhecimento sobre um ataque é manualmente codificado por um especialista tendo em conta ataques conhecidos ou variações dos mesmos (ex.: métodos baseados em pré-requisitos e consequências).

Os métodos supervisionados contam com uma fase preliminar de aprendizagem automática a partir de um conjunto de dados de treino, extraindo conhecimento através de algoritmos de *machine learning* supervisionados.

No que se refere aos métodos automáticos, os mesmos não recorrem a conhecimento prévio sobre os ataques, sendo que aprendem sobre o mesmo conjunto de dados onde as deteções são realizadas (em tempo real).

A clara vantagem dos métodos automáticos sobre outros é que os mesmos conseguem encontrar ataques desconhecidos. Contudo, podem não ser tão confiáveis e reportar diversos falsos positivos [1].

É importante referir no entanto que os métodos supervisionados dependem da disponibilidade de um conjunto de dados sólido e confiável para a fase de treino. Todavia, os conjuntos de dados disponíveis estão longe de refletir as situações complexas que são encontradas no mundo real, devido a problemas de privacidade e segurança que impedem as organizações de tornarem os seus dados de rede públicos [1].

Métodos de *Machine Learning*

No que se refere aos métodos de utilização de *machine learning*, os mesmos podem ser caracterizados essencialmente em três tipos, nomeadamente [60]:

- Supervisionado - Aprendizagem através exemplos. Necessita de dados previamente rotulados (representando a resposta certa) para serem utilizados na

sua execução. É utilizado quando existe um número expressivo de exemplos rotulados;

- Não-supervisionado - Aprendizagem por observação e descoberta, sendo capaz de encontrar automaticamente padrões a partir de um conjunto de dados. Não necessita de exemplos rotulados para a sua execução, aprendendo por observação em vez de aprender por exemplos;
- Semi-supervisionado - Utilizado em aplicações semelhantes às da aprendizagem supervisionada, todavia o perfil dos dados é diferente. De forma geral, é formado por uma pequena quantidade de dados rotulados.

É importante mencionar que os algoritmos de agrupamento, também conhecidos como *clustering*, fazem parte do método de *machine learning* não supervisionado.

Algumas abordagens de correlação de alertas recorrem a algoritmos de *machine learning* supervisionados, todavia, tal como referido em [13], o trabalho moroso de preparação e manutenção dos sistemas leva a que os sistemas de correlação baseados em aprendizagem supervisionada sejam menos práticos.

2.4 Considerações Finais

Este capítulo apresentou os principais conceitos envolvidos neste trabalho, tendo sido também importante como forma de contextualização ao presente trabalho.

De forma geral, os componentes envolvidos num processo de correlação de alertas não são sempre os mesmos, visto que depende da abordagem de correlação, sendo que a correlação de alertas é demasiado complexa para ser abordada numa única fase. Algumas *frameworks* têm sido sugeridas para correlacionar alertas [8, 44].

O sistema proposto teve em conta as diferentes técnicas de análise de alertas que podem ser adotadas nos sistemas de correlação de alertas.

Capítulo 3

Estado da Arte e Hipótese de Investigação

No presente capítulo é efetuada a descrição de trabalhos relacionados, com vista a encontrar novas e inspiradoras ideias para o trabalho e para a formulação da hipótese de investigação.

Na secção 3.1 encontra-se um estudo do estado da arte e na secção 3.2 é mencionado o contexto do problema e hipótese de investigação. A secção 3.3 finaliza com as considerações finais do capítulo.

3.1 Trabalhos Relacionados

Em [42] é apresentado uma abordagem para correlacionar alertas de segurança, onde a metodologia adotada tem como princípio a utilização de *data mining* para a obtenção de informações constituintes nos alertas provenientes de um NIDS, com vista a classificar, agrupar e correlacionar os alertas visando obter cenários de ataque. Desta forma, é possível listar de forma organizada ataques que foram realizados etapa a etapa numa rede ou num determinado *host*.

Inicialmente os alertas, provenientes de um NIDS, são classificados, ou seja, agrupados por tipo de ataque (ex.: *Brute Force*, *Scan*, *Dos*) com recurso ao algoritmo de aprendizagem não supervisionada AutoClass com base em informações fornecidas pelo padrão IPFIX (utilizado pelo NIDS baseado em anomalia empregue no trabalho em questão), que realizava as suas deteções através da análise de fluxos bidirecionais e que não fornecia a classe dos alertas emitidos.

Posteriormente, sem ter em conta os resultados obtidos com o Autoclass, e utilizando o algoritmo k-means (com o intuito de agrupar os alertas que fazem parte de

um mesmo cenário de ataque), são criados *clusters* de alertas de forma a agrupar alertas com características semelhantes entre si, com base no endereço de origem, endereço de destino e porta de destino.

Assim que os alertas estão classificados e agrupados é estabelecida uma relação entre os dois resultados. A correlação é realizada procurando pela ocorrência de alertas com o par <endereço de origem, endereço de destino> igual, que pertençam a classes distintas dentro de um determinado *cluster*. Os alertas são depois organizados de acordo com a data de deteção e os cenários de ataque são encontrados.

É identificado pelo autor que alguns cenários de ataque não são detetados pela metodologia apresentada, nomeadamente ataques de DDoS.

No artigo [49], muito referenciado na literatura, foi utilizada uma abordagem probabilística para correlacionar alertas. Cada alerta é correlacionado com o hiper alerta mais semelhante, assumindo que a semelhança é maior que um limite de semelhança mínimo. No caso dos alertas serem dissimilares, é criado um novo hiper alerta que pode ser composto por alertas de sensores distintos.

Medir a semelhança das características de alertas é a principal preocupação nesta abordagem. No modelo proposto são consideradas quatro métricas distintas para medir a semelhança entre dois alertas, sendo elas:

- Sobreposição de características - Cada alerta tem algumas características, e um novo alerta e alertas existentes partilham algumas características, tais como endereço de origem, endereço de destino, tempo, e tipo de ataque;

- Semelhança de características - Valor da semelhança baseado em características comuns dos alertas depende do tipo de informação. Por exemplo, a semelhança entre dois endereços de destino pode ser obtida com base nos bits mais altos de endereços IP (que podem identificar uma sub-rede), enquanto que a semelhança entre classes de ataque é calculada com base numa matriz de similaridade de classes de incidentes criada pelos autores (que indica a probabilidade de um alerta do tipo X ser seguido de um alerta do tipo Y);

- Expectativa de semelhança - A expectativa de semelhança depende de situação para situação, sendo que esta medida é utilizada para normalizar a semelhança geral. Por exemplo, num *port scan*, um atacante, tenta ligar-se a diferentes *hosts*, e portanto, a semelhança entre endereços de destino pode ser baixa. Além disso, em ataques que sejam possíveis realizar com recurso a IP *Spoofing*, o peso da semelhança entre IPs de origem é diferente em comparação com ataques onde o IP *Spoofing* não seja possível;

- Semelhança mínima - Grau de semelhança que tem que ser cumprido por certas características de cada alerta. No caso do grau de semelhança ser menor que a semelhança mínima, o alerta não será correlacionado (independentemente do valor de semelhança geral).

A semelhança geral de dois alertas encontra-se entre 0 e 1, e é calculada com a média ponderada dos atributos semelhantes, usando a expectativa pré-definida de semelhança (para cada atributo) como valor de peso.

Ao utilizar valores de semelhança esperada (através da matriz de correlação) e valores mínimos de semelhança diferentes, é possível obter diferentes visões dos incidentes. A título de exemplo, ao diminuir-se o valor esperado de semelhança da identificação do sensor e exigir uma elevada probabilidade para a classe de ataque, é esperado obter uma melhor visão dos incidentes de segurança individuais reportados por sensores distintos. Por outro lado, ao diminuir a expectativa de semelhança da classe de ataque, é esperado reconhecer diversas etapas de um ataque multi-etapa.

Segundo [47], uma escolha cuidadosa dos critérios de semelhança pode levar a sistemas com um bom funcionamento que capturem a essência de diversos tipos de ataques previamente conhecidos. No entanto, é importante mencionar que decidir o nível de semelhança de diversos atributos não é de forma alguma uma tarefa trivial, sendo que, neste contexto, considerar uma correspondência perfeita entre atributos é, em muitos casos, uma abordagem demasiado simplificada.

Diversos trabalhos têm dado seguimento ao trabalho de [49]. A utilização de matrizes de correlação de alertas para armazenar a relação casual entre alertas é comum na literatura. Um exemplo disso é o trabalho [48], que utiliza uma abordagem semelhante, sendo que a diferença essencial entre as duas abordagens é no calculo da probabilidade de correlação entre dois tipos de alertas, onde o grau de correlação é calculado adaptativamente em [48]. Este trabalho recorre a dois algoritmos de *machine learning* supervisionados relativos a redes neurais (Multilayer Perceptron e Support Vector Machine) para determinar a probabilidade de correlação entre alertas.

A abordagem adotada por [49] também está relacionada com o trabalho de [32], porém o trabalho de [49] é baseado em métodos probabilísticos, e portanto depende da distribuição comportamental dos atributos, subjacente aos desvios relativos ao que é esperado.

Com a abordagem adotada no trabalho [32], através de *machine learning*, foi possível evitar as restrições impostas por tais modelos paramétricos, que recorrem a assunções através dos pesos associados às características dos alertas, podendo

prejudicar os resultados no caso de não estarem corretas.

No trabalho em questão foi proposto um sistema, de baixa manutenção, para correlacionar alertas de NIDS, utilizando *machine learning* com aprendizagem não supervisionada. Este sistema é baseado na ideia de que os ataques informáticos podem ser decompostos em passos de ataque, onde cada passo corresponde a uma ação do "plano" do atacante.

O sistema é implementado em duas fases de correlação e é compatível com NIDSs distintos. Na primeira fase, utilizando a técnica AA (Autoassociador), baseada numa rede neural artificial, os alertas são agrupados com base apenas nas semelhanças de características dos pacotes IP associados aos alertas (ex.: icmpType, ipTos, tcpFlagSyn, udpLen, entre outros), de modo a que cada grupo forme um passo de ataque. Na segunda fase, com recurso ao algoritmo EM (Expectation-Maximization), os grupos criados na primeira fase são combinados, com base em informações de cada *cluster* criado na primeira fase e em atributos extraídos dos alertas (ex.: nº de alertas, dimensão média do *payload*, tamanho médio de cabeçalho IP, partes comuns nos endereços IP, entre outros), de forma a que cada combinação de grupos (chamado de *super-cluster*) contenha os alertas de um ataque completo.

Também foi testado a rede neural SOM (Self-Organizing Map), tendo apresentado uma taxa de erro superior ao AA e EM. É também importante referir que os autores testaram, numa fase inicial do sistema, o algoritmo k-means, todavia o mesmo apresentou um baixo nível de desempenho neste sistema.

Apesar do sistema proposto necessitar de algumas melhorias em termos das características dos alertas a ter em conta e dos algoritmos a utilizar (uma vez que diversos alertas do mesmo tipo foram agrupados em grupos distintos, e alguns alertas, ainda que poucos, estavam em *clusters* com alertas não relacionados), os resultados experimentais foram interessantes.

No artigo [13], semelhante ao trabalho de [32], é proposto um modelo de *clustering* com base no método Improved Unit Range de modo a melhorar os resultados de *clustering*, Principal Component Analysis para reduzir a dimensionalidade dos dados e o algoritmo EM com vista a agregar alertas semelhantes e reduzir o número de alertas.

A abordagem sugerida foi testada com quatro algoritmos de aprendizagem não supervisionada, nomeadamente SOM, k-means, Fuzzy c-means e EM, tendo o algoritmo EM apresentado melhores resultados de *clustering* e a rede neural SOM os piores nos testes realizados.

Os autores do artigo defendem que agrupar alertas com base nas semelhanças das suas características pode revelar os passos de ataque efetuados por atacantes,

tendo na abordagem adotada agrupado alertas semelhantes em *clusters* com vista a que cada *cluster* represente um passo de ataque.

O artigo [61] teve como objetivo resolver problemas relativos à descoberta de novos padrões de ataque e a dificuldades na definição e manutenção de regras complexas de associação de ataques.

Este método é composto por dois componentes. Inicialmente, utilizando janelas de tempo extensivas (que se adaptam), são classificados alertas de intrusão em três tipos de hiper alertas (com base na semelhança entre o tipo de ataque, IP de origem e IP de destino) e ligados os mesmos (hiper alertas) a partir de ligações temporárias de acordo com associações de endereços IP, criando assim sequências de ataque candidatas. Posteriormente, é calculado o nível de correlação, com recurso a grafos de correlação, das ligações temporárias entre hiper alertas para filtrar as ligações que não representem um relacionamento real. Segundos os autores, o produto final do método proposto pode ser utilizado para construir regras de associação, que podem ajudar na deteção de possíveis ataques futuros, em tempo real, num sistema.

No seguimento de [61], é proposto em [16] uma *framework* para reconhecer cenários de ataque multi-etapa a partir de alertas gerados por IDSs. Neste trabalho é referido que a abordagem adotada em [61] leva ao aumento de sequências candidatas com redundância e não garante que alertas dentro de uma sequência de ataque candidata pertençam ao mesmo cenário de ataque, podendo prejudicar a precisão do resultado final.

A abordagem dos autores tem por objetivo aumentar a eficiência no reconhecimento de ataques multi-etapa e prever próximos ataques em tempo-real. Esta *framework* consiste em dois componentes: *online*, onde são recebidos alertas gerados pelos sensores IDS e são reconhecidos e previstos ataques multi-etapa em tempo real, e *offline* onde são construídos modelos de estratégias de ataque para serem posteriormente utilizados para a extração de padrões sequenciais através do algoritmo Generalized Sequential Pattern, cujo o resultado pode ser transformado automaticamente em regras ao serem construídas árvores de cenários de ataque, com vista a serem utilizadas no componente *online*.

A abordagem adotada no componente *offline* segue métodos semelhantes aos do artigo [61], contudo é calculado o nível de correlação entre hiper alertas aquando da criação de sequências de ataque candidatas (representando um cenário de ataque completo), permitindo confirmar assim que todos os hiper alertas nas sequências candidatas se encontram correlacionados, enquanto que no trabalho de [61] são criadas sequências de ataque candidatas com base apenas em endereços IP e só posteriormente são corrigidas algumas ligações que foram indevidamente realizadas entre os

hiper alertas.

O trabalho [62], desenvolvido com base em [63], teve como objetivo a especificação e implementação de um sistema inteligente, que realiza a classificação de comportamentos suspeitos na rede através do correlacionamento de alertas de segurança com base em estatística, procurando encontrar padrões e relações entre os alertas.

Este sistema pode ser entendido como um filtro, de modo a evitar alertas falsos positivos provenientes de múltiplos IDSs. Nesta abordagem são aplicados cálculos das redes bayesianas para entender o correlacionamento de eventos de vários IDSs, utilizando abordagens que conseguem encontrar comportamentos suspeitos a partir da análise da variação destes correlacionamentos, e utiliza o raciocínio baseado em casos para comparar esses comportamentos suspeitos com outros registados anteriormente (e validados por um operador de segurança), com vista a classificar se os comportamentos suspeitos são efetivamente ataques ou não, permitindo filtrar automaticamente possíveis ataques que são muito semelhantes aos já registados (com base na experiência de casos já classificados pelos operadores de segurança).

Com a abordagem deste trabalho foi possível: obter modelos estatísticos que permitem modelar o estado normal de uma rede com base na correlação de eventos de segurança de diversas fontes, entender a probabilidade de ocorrer um tipo de alerta depois de outro, detetar comportamentos suspeitos, saber quais os tipos de alertas que mais podem influenciar a ocorrência de outros, entre outras utilidades.

Alguns trabalhos na literatura, tal como o [64], utilizam os testes de causalidade de Granger, realizando uma análise estatística de causalidade, baseada em séries temporais, para correlacionar alertas e gerar cenários de ataque sem qualquer conhecimento pré-definido. Contudo, apesar deste tipo de técnicas permitir detetar ataques desconhecidos, têm algumas dificuldades em lidar com situações onde os atacantes tenham demoras entre passos de ataque [35].

No artigo [7] foi proposto um sistema de correlação de alertas que unifica alertas provenientes de IDSs distintos, remove falsos alertas que não são prejudiciais para o sistema devido às regras de segurança em vigor, agrega alertas, e correlaciona-os para criar cenários de ataque multi-etapa.

Neste sistema é adotada uma abordagem de correlação de alertas baseada em grafos e cadeias de Markov absorventes que funciona em tempo real, com foco numa arquitetura de IDSs distribuídos, com vista a permitir o reconhecimento de ataques e a extração de propriedades dos mesmos, e a análise e previsão de ataques.

As cadeias de Markov absorventes são utilizadas para definir os estados iniciais e os objetivos de cada ataque, e a probabilidade de transição de uma ação de ataque

para outra, de acordo com um dado cenário, até alcançar um dos objetivos de ataque.

Nesta abordagem, os grafos são obtidos aquando do processo de correlação de alertas, oferecendo uma análise simples, intuitiva e em profundidade de ataques, de modo a descrever cenários de ataque.

O sistema proposto permite que seja realizada a correlação e previsão de ataques multi-etapa com base em probabilidades de transição semelhantes, sem necessidade de conhecimento prévio nem de um conjunto de dados para treino.

A deteção de ataques conhecidos é realizada com base em grafos de ataque existentes, sendo considerado um novo cenário de ataque caso o cenário em questão não seja semelhante a nenhum dos grafos de ataque existentes. Através destes grafos conhecidos é ainda possível a deteção de alertas que não foram reportados pelos IDSs erroneamente (falsos negativos).

Neste artigo são também referidos alguns métodos (alternativos) que são baseados em grafos de ataque de rede (diferentes de grafos de ataques de alertas), que podem ser gerados a partir dos detalhes de configuração da rede e de vulnerabilidades conhecidas dentro da rede, representando as possíveis formas que um atacante pode violar a política de segurança de uma organização tendo em conta a arquitetura da mesma. Contudo, segundo o que é referido no artigo, os grafos de ataque podem ser bastante largos e densos, e as ferramentas que geram grafos de ataque de rede devem ser continuamente atualizadas e não apresentam bons resultados. Estes métodos não são o foco do presente trabalho.

No artigo [12] foi sugerida uma abordagem de correlação de alertas com base em grafos intitulada de GAC (Graph-based Alert Correlation) que pode ser utilizada para detetar ataques distribuídos, tais como DDoS, *port scans* e disseminação de *Worms*. Para tal, são realizados três passos, nomeadamente:

1. Agrupamento de alertas com base em endereços IP e portas (origem e destino), e com o CPM obter *clusters* com vista a que cada *cluster* inclua todos os alertas que pertençam a uma etapa de ataque individual;

2. São identificados os padrões de comunicação entre *hosts* dentro de cada *cluster* para que seja oferecido um contexto, sendo eles 1:1 (uma origem e um destino), 1:N (uma origem com vários destinos), N:1 (várias origens e um destino) ou N:N (várias origens e vários destinos). Desta forma, é facilitada a análise humana e a identificação das relações entre *clusters* distintos;

3. Os *clusters* são interligados, com base nos padrões de comunicação nos *clusters* e nos endereços IP de origem e destino, com vista a obter um cenário de ataque

completo. Como resultado é obtido um grafo com todos os *clusters* que sejam considerados semelhantes.

O GAC transforma alertas em representações de grafos distintos em cada passo, que são sucessivamente processados durante o processo de correlação.

De modo a ser tolerante a erros, o modelo dos autores considera a incerteza aquando da identificação dos padrões de comunicação (passo 2), calculando o valor da mesma (incerteza) de forma a refletir a confiança no tipo de padrão identificado.

A abordagem sugerida não requer conhecimento prévio e permite revelar a relação entre alertas.

O trabalho [6] apresenta uma abordagem que agrega e processa alertas de IDSs com vista a formar sequências de etapas de ataque com base na severidade dos alertas e nos modelos probabilísticos correspondentes, permitindo a comparação de sequências de ataque, com vista a identificar comportamentos de ataque únicos e semelhantes.

O trabalho utiliza uma abordagem baseada em semelhança para agregar alertas, uma abordagem baseada em conhecimento para desenvolver as sequências de etapas de ataque e uma abordagem baseada em estatística de modo a comparar as sequências de ataque.

Com base numa série temporal, foram definidas as etapas de ataque através das tendências de crescimento e decrescimento. No modelo proposto foi assumido que uma mudança nessas tendências é um provável indicativo de um atacante alterar o tipo de ataque, a intensidade de um tipo de ataque aumenta assim que um atacante faz uma etapa de ataque especifica, e quando a intensidade atinge um pico (assumido como o objetivo da etapa de ataque) e começa a decrescer significa que um atacante encerra uma etapa de ataque.

Os autores do modelo testaram a abordagem sugerida, com vista a descobrir semelhanças e estratégias únicas, com dez equipas num ambiente controlado de competição de testes de penetração, cujo objetivo era descobrir, explorar e documentar vulnerabilidades na rede. O modelo proposto permitiu identificar padrões de comportamento interessantes, com base na comparação entre equipas, tais como as sequências de ataque: mais longas, mais semelhantes, e mais prováveis de serem únicas (podendo indicar assinaturas de cada equipa) para cada equipa.

Apesar de não ser o foco do presente trabalho, é importante referir que existem abordagens, tal como o trabalho [27], que verificam os alertas com vista a identificar se existiu ou não sucesso numa intrusão (com o objetivo de reduzir a influencia dos falsos alertas num processo de decisão) através dos rastos deixados pelo atacante (ex.:

ligação de rede para o atacante, portas abertas deixadas pelo atacante relativas a um *backdoor*), e da análise de requisitos para que o ataque tenha efeito (ex.: verificação se os serviços estavam vulneráveis, se os atacantes alcançaram o alvo dada a tipologia de rede e configurações de *firewall*, entre outros).

3.2 Contexto do Problema e Hipótese de Investigação

Nesta secção é abordado o contexto do problema com vista a entender os desafios do trabalho e a hipótese de investigação do presente trabalho.

3.2.1 Contexto do Problema

Tal como mencionado em [3], existem principalmente dois pontos que devem ser considerados aquando de uma investigação na área da correlação de alertas, nomeadamente:

1. Os alertas são de baixa qualidade em termos de elevada redundância, número, e falsos alertas, podendo afetar a eficácia de um sistema de correlação. Estes problemas podem ser causados, a título de exemplo, por:

    - Baixo desempenho dos NIDSs - Emissão de diversos falsos positivos, visto que atividades consideradas normais são erroneamente entendidas como intrusões;
    - Ataques intensivos em simultâneo contra múltiplos *hosts* na rede - Tal cenário pode confundir os NIDSs e produzir falsos positivos e aumentar a redundância de alertas;
    - Organizações terem a tendência de implementar diversos NIDSs (homogéneos ou heterogéneos) - Pode levar a uma grande quantidade de alertas.

2. As estratégias de ataque não podem ser reconhecidas diretamente a partir de alertas (em formato bruto), pois:

    - Os alertas gerados por múltiplos NIDS podem encontrar-se num formato diversificado e são representados por informação de baixo nível, levando a que revelação das estratégias de ataque diretamente a partir de alertas seja impraticável;

- O desenvolvimento contínuo de novos ataques de rede, dado que os métodos de ataque estão cada vez mais sofisticados, leva a novas estratégias de ataque e novos padrões de alertas.

Além dos pontos referidos, no artigo [46] é também mencionado que existem diversos problemas associados aos sistemas que emitem alertas como os IDSs, levando a que sistemas de processamento de alertas sejam necessários. Entre os problemas estão:

- Eventos heterogéneos - Receção de uma ampla variedade de alertas de diferentes sensores, o que leva a que existam alertas em diferentes formatos. Com vista ao processamento de alertas, é requerido a normalização dos mesmos;
- Incidentes não identificados - Em todos os tipos de sistemas de deteção de intrusões, falhas na deteção são cometidas devido ao facto de existir falta de informação que descreva os incidentes e imprecisão dos padrões de ataque;
- Incapacidade de ligar alertas - A maioria dos ataques são atividades sequenciais, em que o invasor realiza diversas fases para atingir os seus objetivos. Detetar tais ligações entre fases de ataques é, por vezes, deveras difícil, uma vez que o padrão da primeira fase de ataque não é necessariamente sempre único e não é determinável com plena certeza. Um atacante pode realizar uma etapa de ataque em substituição de outra etapa que realizou sem sucesso, e pode também não realizar partes de um ataque por ter acesso direto à informação;
- Não fornecer um nível de confiabilidade e prioridade dos alertas - Muitos IDSs não fornecem atributos de confiabilidade dos alertas gerados, e em caso de oferecerem tal informação os resultados não são comparáveis com os resultados de outros recursos devido à ausência de um padrão comum entre os diferentes recursos. Por outro lado, a importância de um alerta depende da importância do alvo, que não está relacionada a sensores como IDSs.

No artigo [1] é referido que o grande número de alertas emitido por IDSs deve-se essencialmente devido a: inclusão de múltiplos IDSs, definição imprecisa de incidentes, incompatibilidade de dispositivos na rede, número de intrusões reais em simultâneo e intensivamente, ou comportamento ilegítimo que tenta disfarçar um ataque principal.

É, desta forma, fundamental ter estes desafios em conta, de modo a obter conhecimento útil dos alertas.

3.2.2 Hipótese de Investigação

Apesar dos esforços que têm sido realizados nesta área de investigação, há ainda algumas limitações nos trabalhos existentes que devem ser ultrapassadas.

Muitas das técnicas de correlação de alertas não fornecem informação detalhada sobre as táticas ou estratégias das intrusões a partir de alertas de intrusão, criando apenas grupos de alertas semelhantes. Além disso, diversos sistemas apenas implementam poucos aspetos de correlação, são pouco flexíveis no reconhecimento de ataques, e não têm a eficiência que é desejável na análise de alertas, apresentando algumas limitações.

Com vista a fazer face aos problemas que têm vindo a ser mencionados, e aos desafios associados ao reconhecimento de estratégias de ataque, assim como superar as limitações dos trabalhos existentes é necessário um sistema de correlação de alertas capaz de identificar e reconhecer estratégias de ataque de forma mais prática e eficaz. Portanto, a principal questão de investigação é:

Como identificar ataques de rede multi-etapa e reconhecer cenários de ataque conhecidos, de forma prática e eficaz, a partir de alertas provenientes de NIDSs com o intuito de melhorar a segurança nas redes informáticas?

A hipótese de investigação surge partindo da questão anterior e no seguimento dos estudos existentes, apresentando como objetivo melhorar a qualidade dos alertas, identificar ataques multi-etapa, e reconhecer cenários de ataque de rede conhecidos com base em alertas gerados por NIDSs.

Os alertas gerados por NIDSs têm pouco significado, a menos que sejam analisados através de correlações. O conhecimento extraído a partir da correlação de alertas permitirá inclusivamente facilitar investigações forenses, e dar suporte no desenvolvimento de mecanismos de resposta apropriados [8, 3].

Analisar alertas de intrusão é desafiante particularmente devido ao elevado número de alertas produzidos por NIDSs, sendo que a automatização da correlação de alertas irá reduzir os esforços de um analista de segurança.

3.3 Considerações Finais

O presente capítulo ajudou a aumentar o conhecimento do domínio, e a entender diversas abordagens utilizadas nesta área de investigação, bem como estas poderiam ser utilizadas no presente trabalho. Existiu um ênfase maior em abordagens de

correlação de alertas que não necessitassem de conhecimento prévio para operarem e que fossem de baixa manutenção.

Através da análise realizada foi possível perceber que existem abordagens distintas cada uma com as suas vantagens e desvantagens, e com os seus próprios critérios de desempenho e objetivos. Tal como referido, muitas pesquisas têm sido realizadas ao longo dos anos nesta área.

Na literatura existem trabalhos que tiram partido de eventos, de apenas alertas, ou de ambos (híbridos entre eventos e alertas), sendo que geralmente são utilizados apenas alertas. Os alertas são dos "vestígios" mais utilizados na deteção de ataques multi-etapa. Contudo, os eventos que não representem alertas de segurança podem fornecer informação contextual que pode ser fundamental em um cenário de ataque.

Existem diversas razões importantes que levam à preferência dos alertas de IDS, nomeadamente devido à composição de alguns conjuntos de dados públicos, e porque os ataques multi-etapa são mais fáceis de detetar se forem compostos por elementos que pressuponham ameaças por si só, se tiverem uma estrutura pré-definida, e um número limitado de tipos de eventos (ao contrário dos eventos gerais que não têm) [1].

Nos ataques multi-etapa pode ser necessário não só inferir a natureza de cada ataque, bem como as ligações entre os mesmos. A ligação entre ações de ataques elementares pode ser realizada de diversas formas. Em [1] são apresentados alguns exemplos, sendo eles com base em: comparações entre IPs de origem e/ou IPs de destino; ações que são encontradas usualmente em conjunto; intervalos de tempo entre ações ou ações que pertençam a uma janela de tempo específica; ações que preparam para as condições necessárias de outras ações; ou no mesmo tipo de ação. Conclui-se, desta forma, que a comunidade de investigação ainda não sabe como fornecer uma definição consistente de uma ligação entre ações de ataques [1].

No que se refere aos métodos baseados em semelhanças, ao serem ligados pesos de ataque de forma simples, apenas tendo em conta poucas características, pode resultar em diversos falsos positivos. Não obstante, a utilização de um processo de ligações complexas com base na aplicação de métricas de correlação e usando diferentes pesos para cada característica pode levar a que se seja demasiado específico para capturar as características de todo um ataque multi-etapa. É também importante destacar que as ligações dependem da semelhança inerente entre as características de cada ação ou conjunto de ações [1].

Existem diversos desafios associados aos ataques multi-etapa que devem ser tidos em conta, entre eles [1]:

- Etapas individuais podem parecer inofensivas;

- Um atacante pode ter múltiplos planos de ataque e pode parar um plano porque perdeu o interesse ou porque não é capaz de tirar partido das vulnerabilidades;
- Os ataques podem não ser detetados devido a limitações de dispositivos de rede ou devido a forma como estão configurados;
- Um atacante não necessita de seguir uma ordem precisa para executar um ataque multi-etapa, levando a que as possíveis sequências de ações sejam bastante complexas;
- O intervalo entre etapas de ataque consecutivas pode ser bastante alto.

Capítulo 4

Sistema Proposto

Este capítulo menciona os requisitos do sistema, e descreve o funcionamento do sistema de correlação de alertas proposto bem como seus componentes e o processo de desenvolvimento dos mesmos. São também referidos os desafios enfrentados e como foram ultrapassados.

Foi adotada uma arquitetura modular com vista a facilitar a manutenção, diminuir a repetição de código e melhorar a legibilidade do código desenvolvido.

Na secção 4.1 são abordados os requisitos do sistema e na secção 4.2 a arquitetura do sistema.

4.1 Requisitos do Sistema

De forma a assegurar que o sistema cumpre o seu propósito são identificados nesta secção os requisitos funcionais.

- Receber alertas - O sistema deve ser capaz de receber alertas de NIDSs através da rede;
- Analisar alertas - Quando são recebidos alertas de intrusão, os mesmos devem ser analisados e interpretados;
- Identificar ataques multi-etapa - Identificar alertas que pertençam ao mesmo ataque multi-etapa;
- Reconhecer cenários de ataque semelhantes - Com base nos alertas recebidos, calcular o grau de semelhança dos ataques identificados com os cenários de ataque já conhecidos;

- Apresentar os resultados da correlação - Após o processamento necessário dos alertas, apresentar o resultado para que seja possível a sua análise.

É também importante que o sistema apresente resultados com uma boa precisão e seja de fácil integração.

4.2 Arquitetura do Sistema

Nesta secção é apresentada a descrição da arquitetura do sistema.

Para cada módulo foi definida uma responsabilidade, sendo que todos os módulos dependem dos anteriores para poderem ser executados.

O sistema proposto é composto por sete módulos principais responsáveis pela análise de alertas, nomeadamente:

- Receção de alertas - Recebe, em tempo real, os alertas emitidos pelos NIDSs;
- Normalização - Padroniza os alertas no formato IDMEF;
- Divisor de ataques - Separa os alertas, de modo a que cada grupo de alertas represente um ataque multi-etapa;
- Pré-processamento - Converte atributos de alertas num formato que seja compatível com o algoritmo utilizado para criar *clusters* de alertas;
- *Clustering* de alertas - Agrupa alertas em grupos, de modo a que cada grupo de alertas represente um passo de ataque;
- Redução de alertas - Remove alertas considerados redundantes e prepara os alertas para serem processados no módulo de reconhecimento de cenários de ataque;
- Reconhecimento de cenários de ataque - Encontra cenários de ataques conhecidos que sejam semelhantes aos ataques em análise.

Na Figura 4.1 é possível observar, de forma sucinta, a arquitetura de *software* desenhada para o sistema.

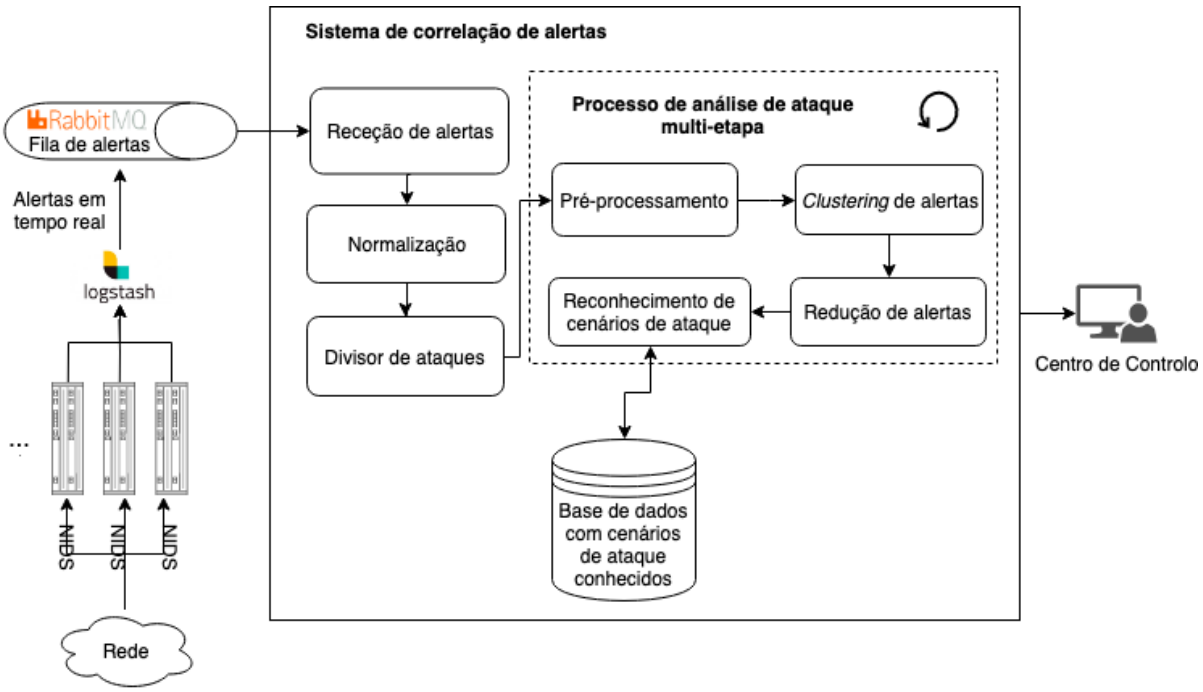

Figura 4.1: Arquitetura de *software* do sistema

O centro de controlo serve para gerir o sistema de correlação de alertas e visualizar os resultados oferecidos pelo mesmo.

No que diz respeito à arquitetura de hardware do sistema, a mesma encontra-se ilustrada na Figura 4.2.

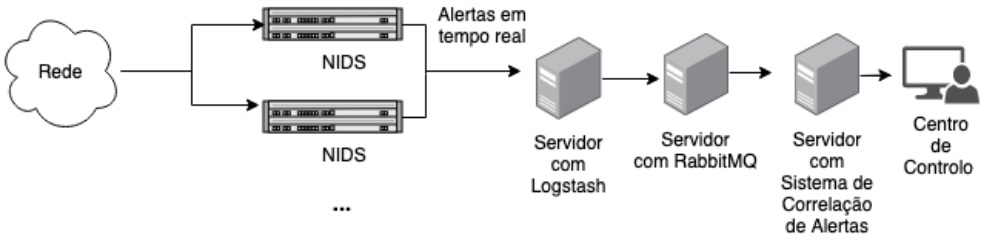

Figura 4.2: Arquitetura de *hardware* do sistema

É importante referir que é possível incluir diversos serviços num só servidor, exemplo: NIDS, Logstash, RabbitMQ e sistema de correlação de alertas num só servidor.

Apesar do Logstash e RabbitMQ serem sugeridos para cooperarem com o sistema de correlação, os mesmos podem ser substituídos por aplicações equivalentes.

Na restante parte da presente secção é realizada a descrição, de forma mais detalhada, dos módulos que compõem o sistema, onde são mencionadas as tecnologias

utilizadas, a responsabilidade de cada módulo e a suas ligações com os restantes módulos.

4.2.1 Receção de Alertas

Este módulo é responsável por receber, em tempo real, os alertas emitidos pelos NIDSs para o sistema.

Para coletar os alertas dos NIDSs pode ser utilizada uma ferramenta como o Logstash [65], que permite receber dados em diversos formatos a partir de uma variedade de fontes simultaneamente, possibilitando filtrar, transformar e enviar os dados para um ou mais destinos à escolha (ex.: base de dados).

Esta ferramenta *pipeline*, pertencente à família de ferramentas ELK Stack (Elasticsearch, Logstash e Kibana) da Elastic [66], e é utilizada frequentemente em contextos de *Big Data* pois consegue receber dados de uma grande quantidade de fontes simultaneamente.

O Logstash realiza três etapas de processamento, nomeadamente *input*, *filter* e *output*. Desta forma, além de ser utilizado para coletar os alertas de diversos NIDSs, é utilizado para filtrar e transformar os dados com o intuito de garantir que apenas são enviados dados ao sistema relacionados aos alertas e que os dados recebidos pelo sistema, através de uma aplicação como o RabbitMQ, são úteis e têm uma estrutura semelhante.

O RabbitMQ é um *message broker* capaz de entregar mensagens, assincronamente, de forma confiável entre aplicações com recurso a uma ligação TCP (Transmission Control Protocol), que foi implementado para suportar o protocolo de mensagens denominado Advanced Message Queuing Protocol.

A ideia do RabbitMQ é disponibilizar uma estrutura que facilite os fluxos de mensagens, especialmente em grandes aplicações, permitindo a comunicação entre sistemas completamente distintos de uma maneira confiável e de uma forma desacoplada.

Ao ser colocado um sistema como o RabbitMQ entre o Logstash e o sistema de correlação de alertas vai permitir que exista menos acoplamento entre as duas partes, visto que ambas têm os parâmetros de configuração do gestor de mensagens (RabbitMQ).

Na arquitetura deste *message broker* (RabbitMQ) existem aplicações cliente, chamadas *producers*, também conhecidas como *publishers*, (que enviam mensagens ao *broker*), e existem os *workers* também chamados de *consumers* (que se ligam às

queues e subscrevem as mensagens a serem processadas). As mensagens colocadas numa *queue* são armazenadas até que um *consumer* as recolha.

É importante referir que o *consumer* e o *publisher* podem estar em diferentes localizações.

Na Figura 4.3, criada com base num esquema de [67], é possível observar, de forma resumida, a arquitetura de um *message broker*.

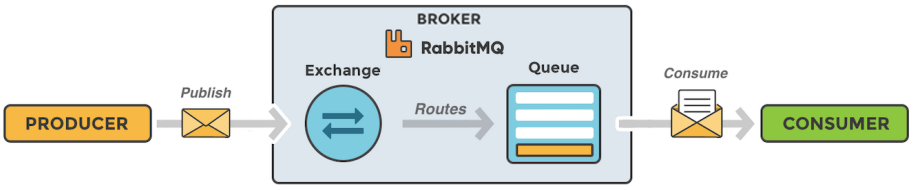

Figura 4.3: Arquitetura de *message broker*

Aquando da obtenção de alertas dos NIDSs, são realizados os seguintes passos:

1. O Logstash (*producer*) recolhe os alertas dos NIDSs (ex.: ao ler os ficheiros de *log* que contêm informações relativas aos alertas de intrusão), e após a recolha filtra-os, transforma-os e envia-os aos RabbitMQ no formato JSON (JavaScript Object Notation);

2. Um *exchange* aceita as mensagens do *producer* e encaminha-as para uma ou mais *queues*. Nos testes realizados foi apenas utilizada uma *queue*, visto que a mesma era o suficiente para responder às necessidades do sistema;

3. O sistema de correlação de alertas (*consumer*) recebe as mensagens enviadas pelo *producer* através da *queue*.

Tal como é possível perceber, as mensagens (dados relativos aos alertas) não são diretamente enviadas para uma *queue*, pois estas são enviadas inicialmente para uma *exchange* que é responsável por encaminhar as mensagens às *queues* corretas (tal como referido, pode ser mais que uma).

É importante referir que no caso de um NIDS não oferecer um formato de saída que facilite a leitura pelo Logstash, é possível utilizar ferramentas como o ids-tools [68] com vista a converter, por exemplo, a informação do formato unified2 (formato binário) em JSON (que é compatível com o Logstash).

A escolha deste *broker* deve-se à fácil integração com outros sistemas, à estabilidade e segurança (ao ser tolerante a falhas e poder ser configurado para utilizar TLS (Transport Layer Security)), e por responder às necessidades do sistema. Os principais objetivos do protocolo TLS são o fornecimento de autenticidade entre duas

entidades, a confidencialidade nas comunicações e a integridade dos dados transmitidos.

Uma característica fundamental num sistema de segurança da informação é a capacidade de garantir a segurança das suas próprias comunicações internas. Desta forma, as mensagens podem ser cifradas de forma a garantir a segurança nas comunicações.

Ao utilizar criptografia a nível de transporte para as *queues* é possível cifrar o tráfego (protegendo contra adulteração e interceção de mensagens), e verificar a autenticidade das entidades envolvidas na comunicação (assegurando a autenticidade, através dos certificados), evitando desta forma que ataques como Man-in-the-Middle (classe de ataques onde um atacante tenta representar o papel de uma entidade legítima) tenham sucesso.

4.2.2 Normalização de Alertas

Nesta fase os alertas recebidos são padronizados para serem preparados para análise. Muitas organizações implementam NIDSs cooperativos com vista a oferecer uma melhor deteção e visão global das atividades de intrusão, contribuindo para a diversidade dos formatos dos alertas.

A normalização de alertas é o processo de converter diferentes formatos de dados dos alertas num padrão comum para ser apropriado e aceite por outros componentes do processo de correlação [8, 27]. Desta forma, os alertas recolhidos são convertidos no formato IDMEF.

É importante salientar que todos os NIDSs devem estar com os relógios sincronizados para garantir a correta correlação dos alertas. Uma forma prática de garantir a sincronização é através do protocolo Network Time Protocol.

Tal como referido, diferentes IDSs podem escolher nomes distintos para classificar os alertas (onde é descrito o tipo/classe de ataque). Desta forma, caso sejam utilizados IDSs que utilizem diferentes nomes para classificar os alertas é necessário utilizar uma classificação comum.

Os atributos tidos em conta neste módulo são: data do alerta, endereço de origem, endereço de destino, porta de origem, porta de destino, prioridade, protocolo, classe do alerta, número da regra que originou o alerta, entidade que gerou o alerta (o NIDS), e dimensão do pacote associado ao alerta.

4.2.3 Divisor de Ataques

O presente módulo tem como objetivo separar alertas, de modo a evitar que alertas não relacionados sejam analisados em conjunto.

Filtrar alertas apenas por endereço IP de um atacante permite analisar as etapas de um atacante ao longo da rede, todavia não é o suficiente caso se deseje ter uma visão mais panorâmica dos ataques na rede. Ter em conta a identificação do agente (sensor NIDS) é importante no caso de se querer ter foco em ataques realizados em determinados segmentos de rede (onde o agente se encontra responsável), e até caso se pretenda determinar que segmentos de rede têm mais interesse para os atacantes. Enfim, são diversas as hipóteses que podem ser adotadas de acordo com a visão sobre a rede que se pretende ter.

É importante ter em mente que em diversos ataques os endereços IP podem ser falsificados (IP *spoofing*). Além disso, um atacante pode utilizar diferentes endereços IP para realizar ataques distintos contra um sistema alvo, portanto o IP de origem, por si só, não pode ser sempre utilizado para identificar um atacante [48].

É também fundamental considerar que podem ser realizados vários ataques ao mesmo tempo. Desta forma, é provável que ataques individuais tenham como alvo o mesmo *host* na rede sem existir uma ligação entre eles [12]. Num ataque pode existir um *scan* por SQL Injection num servidor web, e noutro ataque, em simultâneo mas sem estar relacionado, estar a existir um *port scan* na mesma sub-rede. Dado que em ambos os ataques podem ser emitidos alertas em simultâneo com informações em comum (ex.: endereço IP de destino e porta de destino), pode levar a querer que se tratam de alertas relacionados apesar de não serem.

A natureza dos alertas também oferece alguns desafios, pois num ataque podem ser emitidos alguns alertas que tenham como origem o IP da vítima e como destino o IP do atacante como consequência de uma ligação da vítima ao atacante ser considerada maliciosa (ex.: devido a *reverse shell*) em virtude de ataques realizados anteriormente (pelo atacante). Existem diversos trabalhos nesta área que não têm em consideração esta particularidade, contudo no contexto do presente trabalho é importante ter tal particularidade em consideração.

Tendo em atenção os diversos desafios mencionados, teve-se em conta o artigo [12] com a finalidade de dar resposta aos requisitos deste módulo. Optou-se por ter foco em ataques diretos/grupos de ataques diretos relacionados, ou seja, ataques que tenham um ou mais alvos/origens em comum.

Inicialmente são criados grafos não direcionados onde cada alerta é representado num nó (vértice). A ligação entre nós (criação de arestas) é realizada caso os alertas

associados a cada nó se encontrem num determinado intervalo de tempo (tempo máximo assumido para um ataque completo multi-etapa) e caso se verifique uma das condições mencionadas na Listagem 4.1.

```
1 (Alert1_Src_IP=Alert2_Src_IP) OR (Alert1_Dst_IP=Alert2_Dst_IP) OR
      (Alert1_Src_IP=Alert2_Dst_IP AND Alert2_Src_IP=Alert1_Dst_IP)
```

Listagem 4.1: Condições para criação de aresta

Desta forma, são tidos em conta ataques que, a título de exemplo, recorram a IP *spoofing* para mascarar o IP de origem, ou que levem a alertas onde no endereço de origem se encontre o IP da vítima e no endereço de destino o IP do atacante.

Com o intuito de definir comunidades coesas de alertas, é aplicado o CPM para cortar ligações entre alertas com acopolações fracas. Assim, evita-se que alertas potencialmente não relacionados fiquem na mesma comunidade, o que levaria a que os alertas não relacionados fossem futuramente analisados em conjunto (caso os mesmos fossem atribuídos à mesma comunidade).

Um clique é definido como sendo um sub-grafo completo no qual todos os pares de nós estão ligados por uma aresta. Desta forma, um clique de tamanho k é um sub-grafo com k nós onde o grau de todos os nós no sub-grafo induzido é k-1.

No CPM assume-se que as comunidades são formadas por um conjunto de cliques e arestas que conectam esses cliques. Dois k-cliques são adjacentes se partilharem k-1 nós, sendo que a união de k-cliques adjacentes é chamada de *k-clique chain*, e dois k-cliques estão ligados se fizerem parte do mesmo *k-clique chain*.

Uma comunidade é definida como sendo composta por vários cliques que partilham entre si muitos dos seus nós. Cada comunidade k-clique é composta por todos os cliques de tamanho k que podem ser alcançados uns aos outros através de uma série de k-cliques adjacentes. Este conceito de comunidade satisfaz o objetivo de representar grupos de nós fortemente coesos.

Este método é baseado no conceito de que as arestas internas de uma comunidade são prováveis de formar cliques devido à sua alta densidade e que é improvável que arestas que ligam comunidades formem cliques.

Na Figura 4.4 é apresentada uma ilustração de comunidades k-clique com k=3.

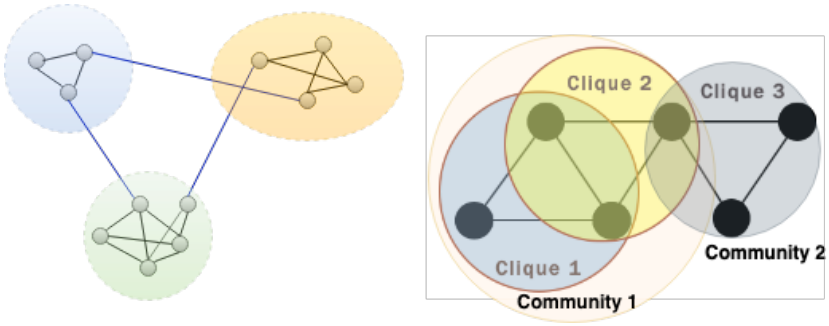

Figura 4.4: Ilustração de comunidades k-clique com k=3

O parâmetro k do método CPM é ajustável. Um valor de k=2 significa que é considerado a percolação de sub-grafos completos de dois nós, sendo que grupos coesos são os componentes desconectados do grafo.

Ao aumentar o valor para k=3 os cliques, neste caso, são constituídos por triângulos onde cada nó encontra-se ligado a outros dois nós. Com k=4, é ainda mais restritivo, já que os cliques precisam ser densos o suficiente para permitir a percolação de sub-grafos completos de quatro. Ao aumentar o valor de k, o tamanho das comunidades diminui, no entanto, as comunidades encontradas são mais coesas.

Um dos desafios dos autores do trabalho [12] foi encontrar o melhor valor de k. No contexto deste trabalho, este valor deve ter em consideração o ambiente onde o sistema de correlação de alertas se vai enquadrar.

Na Listagem 4.2 é possível verificar os passos realizados pelo CPM.

```
1 Input: value of k and original graph
2  Cliques_k= find out all cliques of size k in original graph
3  Create clique graph G(V,E), where |V| = |Cliques_k|
4  E = {e_ij | clique i and clique j share k-1 nodes}
5  Return all connected components of G
```

Listagem 4.2: Clique Percolation Method

Cada k-clique no grafo original é representado por um nó no novo *clique graph* e as arestas neste novo grafo são usadas para representar a sobreposição de cliques no grafo original.

Todos os componentes conectados no *clique graph* representam uma comunidade, sendo que os membros de uma comunidade são conseguidos ao obter todos os nós dos cliques individuais que formam o componente conectado.

O CPM padrão, para encontrar comunidades procura pelos "clique maximal" (conjunto de vértices adjacentes entre si que não estão estritamente contidos em outros cliques), sendo este problema caracterizado por ser do tipo NP-hard [69].

Apesar deste método ter a desvantagem de permitir que existam alertas que pertençam a mais do que uma comunidade (no local onde é feito o corte), no contexto deste trabalho é possível colmatar esta particularidade ao verificar onde existe maior atividade dos endereços IP com o intuito de saber a que comunidade existe uma maior probabilidade do alerta pertencer.

Existe também a limitação de poderem existir nós que não são incluídos em nenhuma das comunidades identificadas (devido às fracas ligações). Tais nós são considerados, neste contexto, como não fundamentais para análise pelo sistema de correlação.

Cada uma das comunidades finais resultantes neste módulo é analisada individualmente pelos módulos seguintes do sistema de correlação.

4.2.4 Pré-processamento de Alertas

A necessidade deste módulo encontra-se diretamente relacionada com os requisitos do módulo de *clustering* de alertas, visto que é necessário preparar os atributos dos alertas num formato que seja compatível com o algoritmo utilizado pelo módulo referido.

Neste módulo, são utilizados apenas os atributos dos alertas que são tidos em conta no módulo de *clustering* de alertas. Com base na própria experiência e conhecimento, nos trabalhos analisados em 3.1, e tendo em conta o artigo [70] (com um estudo sobre os melhores atributos a usar no agrupamento de alertas) são utilizados os seguintes atributos dos alertas:

- Porta de origem - Associado ao porto utilizado para originar uma comunicação;
- Porta de destino - De forma geral, está associado a um serviço em execução na máquina de destino;
- Prioridade - Identifica a severidade associada ao alerta;
- Protocolo - Protocolo associado ao alerta;
- Classe do alerta - Classificação do alerta (ex.: web-application-attack, shellcode-detect), sendo definido com base no tipo de atividade detetada pelo NIDS;
- Dimensão do pacote - Tamanho do pacote IP associado ao alerta.

É importante referir que o IP de origem e IP de destino não são tidos em conta neste módulo pois o mesmo recebe como entrada os dados processados provenientes do módulo de divisão de ataques que já teve em consideração esses atributos.

As TCP Flags não são consideras neste módulo pois podem existir alertas com protocolos que não tenham essa informação (ex.: associados a protocolos como UDP e ICMP), podendo levar a que os resultados no módulo de *clustering* de alertas pudessem ser afetados ao ser atribuído um valor (ex.: 0) que represente a não existência de TCP flags.

Os atributos extraídos dos alertas encontram-se no formato numérico e não numérico, todavia o algoritmo utilizado no módulo de *clustering* de alertas requer apenas dados no formato numérico. Desta forma, os atributos extraídos neste módulo necessitam de ser processados.

Tendo em conta o resultado pretendido na fase de *clustering* de alertas, todos os atributos são convertidos no formato nominal (considerando-os como categorias) com recurso ao método "One Hot Encoding". Este método é utilizado de forma comum em *machine learning*, com o intuito de representar no formato binário cada categoria.

De modo a ser tolerante a variações nos tamanhos dos pacotes, é atribuída uma categoria ao tamanho dos pacotes consoante o seu intervalo de valores. De forma geral, em redes TCP/IP a gama de valores [40-79] bytes é associada a pacotes de controlo TCP, a gama [80-159] bytes é associada a ferramentas de *networking* (ex.: ping, traceroute), e a gama [1280-2559] bytes relacionada aos pacotes de dados totalmente cheios.

Todos os valores dos atributos são representados em valores numéricos e definidos numa escala de [0,1]. Neste módulo foram assumidos todos os atributos como nominais, todavia no caso de se utilizarem valores no formato numérico (sem ser binário) poderá ser necessário recorrer a fórmulas de normalização ou padronização.

Desta forma, neste contexto, é garantido que são concedidos a cada atributo pesos iguais, pois uma diferença nas escalas em atributos distintos resultaria em pesos também diferentes, sendo que os atributos com maior escala ficariam com um peso maior [26].

Este é um módulo deveras importante no sistema de correlação que tem implicação direta nos resultados do módulo de *clustering* de alertas. É importante destacar que os atributos dos alertas extraídos devem ser processados consonante o resultado pretendido no módulo de *clustering* de alertas. Por exemplo, o protocolo ICMP, que opera na camada 3 do modelo OSI, não tem portas lógicas de origem e destino atribuídas. Como alternativa, podem ser utilizadas informações como o "ICMP type" e "ICMP code" respetivamente.

4.2.5 *Clustering* de Alertas

Este módulo recebe os dados processados do módulo de pré-processamento, e tem como objetivo agregar alertas em grupos, de modo a que cada grupo de alertas semelhantes (*cluster*) represente um passo de ataque.

Neste módulo encontram-se associados alguns desafios, entre eles: os alertas relativos ao mesmo passo de ataque podem estar dispersos no tempo, definir critérios de semelhança entre alertas não é uma tarefa trivial, e um passo de ataque pode ter mais que um atacante e/ou vítima.

Existem algumas questões que são levantadas quando se utilizam métodos de correlação de alertas baseados em semelhança, nomeadamente que atributos comparar, que algoritmo utilizar, como decidir se alertas são semelhantes e o peso que cada atributo deve ter na comparação [47].

Tal como mencionado na secção 3.1 alguns trabalhos (ex.: [49]) utilizam funções baseadas em probabilidade de semelhança pré-definida para medir a semelhança entre dois alertas, contudo estas abordagens podem ser demasiado limitadoras devido às restrições impostas por tais modelos paramétricos, podendo levar a que novos alertas fiquem isolados e não possam ser correlacionados.

Segundo o que é referido em [13, 8], agrupar novos alertas (alertas não conhecidos) pode ser alcançado recorrendo a algoritmos de *machine learning* de aprendizagem não supervisionada, com vista a agrupar alertas com base nas semelhanças dos seus atributos. Esta abordagem tem mostrado não só que agrupar alertas com atributos semelhantes pode revelar passos de ataque, mas também permite reduzir o número de alertas.

Alertas semelhantes tendem a ter causas raiz semelhantes ou efeitos semelhantes nos recursos de rede [44].

Tendo em conta a literatura analisada, essencialmente [70, 32, 8, 13], e os objetivos do sistema, adotou-se para esta fase do processo de correlação uma abordagem baseada em semelhança com recurso ao algoritmo de *machine learning* de aprendizagem não supervisionada *hierarchical clustering* (agrupamento hierárquico), mais especificamente do tipo aglomerativo.

Neste módulo de *clustering* foi adotada uma abordagem empírica com vista a dar resposta aos requisitos do presente módulo.

Hierarchical Clustering

Este algoritmo de *machine learning* procura construir uma hierarquia de *clusters* ao construir uma estrutura em árvore com base na hierarquia criada.

Existem dois tipos principais de estratégias utilizadas por este método, nomeadamente:

- Agglomerative Clustering - Conhecido por uma abordagem ascendente, esta abordagem leva a que no início seja criado um *cluster* para cada instância e sucessivamente sejam aglomerados pares de *clusters* (de acordo com a semelhança) até que todos os *clusters* tenham sido combinados num único *cluster* que contenha todos os dados;

- Divisive Clustering - Conhecido por uma abordagem descendente, requer um método que divida um *cluster* que contenha todos os dados e prossiga com a divisão de *clusters* até que os dados individuais tenham sido divididos em *clusters* individuais (para cada instância).

A escolha da adoção do HAC neste módulo deve-se ao facto do algoritmo começar com cada atributo (de um alerta) numa classe separada e em seguida, em cada passo do algoritmo, fundir dois *clusters* que sejam mais semelhantes. Esta particularidade é bastante vantajosa no agrupamento de alertas para identificar com precisão passos de ataque [70].

Este algoritmo determinístico (ao contrario do algoritmo k-means que cada vez que é iniciado pode produzir diferentes *clusters*) é por norma considerado do tipo *hard clustering*, pois cada objeto é atribuído a apenas um *cluster*.

Nesta abordagem, com vista a que seja decidido que *clusters* devem ser combinados, é necessária uma medida de distância entre os conjuntos de dados. Desta forma, é utilizada uma métrica para medir a distância entre pares de dados (ex.: distância de Manhattan, Euclidiana), e um critério de ligação que especifica a distância de conjuntos.

Na Listagem 4.3 é possível verificar os passos realizados por este algoritmo.

```
1  Input: dataset (d1, d2, d3, ...dN) of size N
2  # compute the distance matrix between the input data points
3   for i=1 to N:
4      for j=1 to i:
5         dis_mat[i][j] = distance[di, dj]
6   let each data point be a cluster
7   repeat
8      Merge the two closest clusters
9      Update the distance matrix
10  untill only a single cluster remains
```

Listagem 4.3: Algoritmo Hierarchical Agglomerative Clustering

Diferentes definições de distância entre *clusters* pode levar a diferentes resultados, sendo que existem diversos critérios para avaliar a distância entre *clusters*, entre eles [71]:

- Single linkage - Distância entre os pontos mais próximos;
- Complete linkage - Distância entre os pontos mais distantes;
- Centroid linkage - Distância entre os centróides (pontos centrais dos grupos) de dois *clusters*;
- Average linkage - Distância média entre todos os pares de pontos dos diferentes *clusters*;
- Median linkage - Mediana entre todos os pares de pontos de diferentes *clusters*;
- Ward linkage - Une *clusters* que minimizem a soma dos erros quadráticos (SSE). Requer a distância Euclidiana.

A Figura 4.5 ilustra alguns dos métodos referidos, nomeadamente: single, complete e centroid respetivamente.

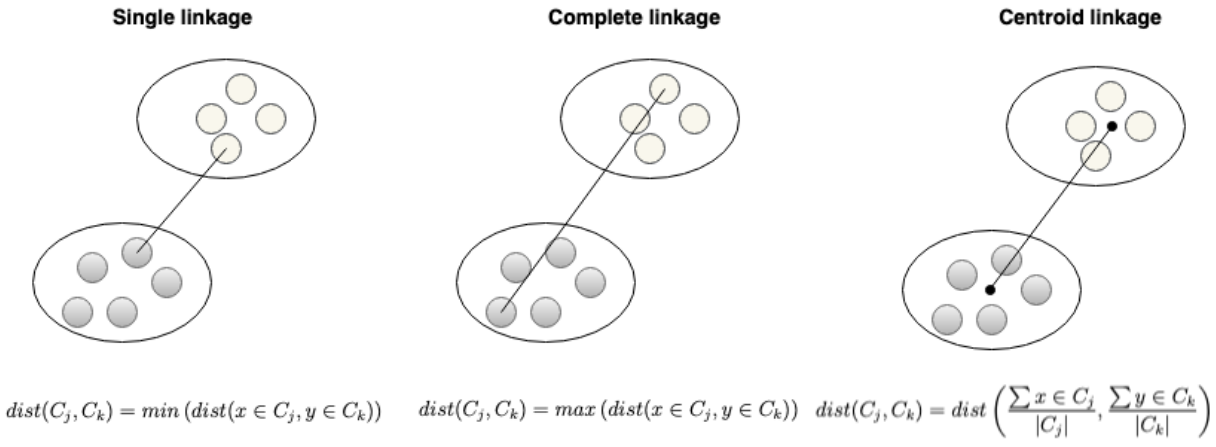

Figura 4.5: Métodos single, complete e centroid

Em termos de complexidade de tempo, esta abordagem é, de forma geral, $O(n^3)$, contudo pode ser melhorada com restrições nos critérios de avaliação da distância entre *clusters* [71].

Escolher o melhor número de *clusters* No algoritmo HAC, com vista a que sejam agrupados os alertas em *clusters* é necessário indicar o número de *clusters* previamente, visto que este algoritmo descobre *clusters*, que não são conhecidos *a priori*. Todavia, neste caso, não existe essa informação.

Um problema popular caracterizado por ser difícil, e que é frequentemente ignorado em problemas de *clustering* é "Quantos *clusters* estão no meu conjunto de dados?"[72].

Infelizmente, não existe uma resposta definitiva a esta questão que tem gerado bastante discussão na comunidade de investigação, sendo que o número "ótimo" de *clusters* é, de certa forma, subjetivo dependendo do método utilizado para medir a semelhança e dos parâmetros utilizados para o particionamento.

O resultado de ambas as abordagens de agrupamento hierárquico podem ser representadas num dendrograma (formato em árvore, que não necessita previamente do número de *clusters*) que reflete como os dados estão organizados. Na Figura 4.6 é possível observar um exemplo dum dendrograma.

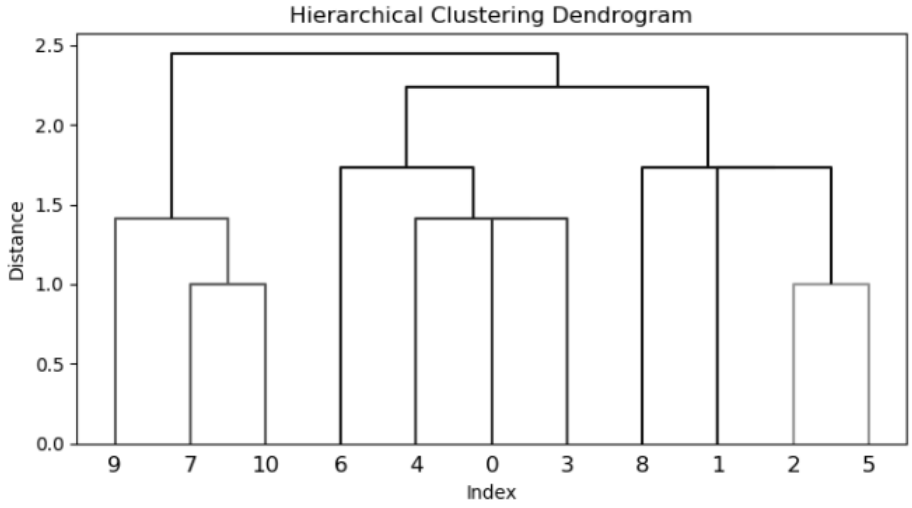

Figura 4.6: Exemplo de dendrograma

Neste exemplo o conjunto de dados é composto por onze instâncias (folhas), sendo que os mesmas são associadas a *clusters* individuais. Os *clusters* são então sucessivamente aglomerados (em nós) até que todos os *clusters* tenham sido combinados num único *cluster* no topo.

Quanto mais abaixo for feita a aglomeração de *clusters*, mais semelhantes as observação são. Da mesma forma que quanto mais acima a aglomeração acontecer, menos semelhantes as observações são.

A observação dum dendrograma pode sugerir a escolha do "melhor"número de *clusters*. Esta escolha é dada pelo número de linhas verticais no dendrograma que intersectem uma linha horizontal que atravesse a distância máxima vertical sem intersectar um *cluster*.

No presente exemplo o "melhor" número de *clusters* aparenta ser três. Ainda assim, é uma solução que está longe de ser "ótima", pois nem sempre é fácil reconhecer o "melhor" número de *clusters* a partir de um dendrograma e é exigida uma análise manual.

Em alguns trabalhos, tal como [13], é variado o número de *clusters* até ser encontrada a "melhor" opção. Contudo, é importante que o número de *clusters* "ótimo" seja obtido de forma automática.

Uma abordagem utilizada para contornar este problema consiste em executar o algoritmo de agrupamento diversas vezes com diferentes parâmetros e escolher o melhor resultado de acordo com um critério. É possível utilizar diversas medidas como regra de paragem que possibilitam a identificação do número "óptimo" de grupos a serem definidos para o conjunto de dados.

A título de exemplo, Glenn Milligan e Martha Cooper [73] testaram no total de 30 medidas de validação de agrupamento com o objetivo de determinar o melhor número de *clusters* no intervalo de [2,5] para cada uma das medidas utilizadas num processo de agrupamento hierárquico. Os resultados de validação externa com os índices "Calinski and Harabasz" e "Duda and Hart" mostraram ser bons indicadores para o número de *clusters* no problema em questão.

Na literatura têm sido propostas uma variedade de medidas com o intuito de validar os resultados da análise de agrupamento. O pacote NbClust [74] da linguagem R oferece os 30 índices (entre eles gap e silhouette) do artigo [73] com vista a auxiliar na escolha do número de *clusters* num conjunto de dados. Este pacote pode ser utilizado por dois algoritmos de *clustering*, nomeadamente K-means e HAC, ao variar o número de *clusters*, as medidas de distância, e os métodos (critérios) de *clustering* [75].

Com base numa abordagem empírica, com recurso a pacote NbClust, o índice de validação interna Point-Biserial com o critério "complete linkage" revelou resultados interessantes no problema que se pretende resolver.

Identificação dos Padrões de Comunicação

Após obtenção dos *clusters*, são identificados os padrões de comunicação entre atacantes e vítimas dentro de cada *cluster*. A identificação dos padrões de comunicações,

além de auxiliar o reconhecimento de passos de ataques, possibilita facilitar a análise dos alertas recebidos e consequentemente o estudo dos ataques identificados.

Com vista a dar resposta a este requisito foram tidas em conta as fórmulas sugeridas no artigo [12], sendo distinguidos quatro tipos de comunicações (1:1, 1:N, N:1, N:N).

Nas fórmulas utilizadas para identificar os padrões de comunicação, cada métrica consiste em três somatórios com pesos iguais que descrevem três proporções, nomeadamente:

- $\mid A \mid$ - Nº de atacantes;
- $\mid T \mid$ - Nº de alvos;
- $\mid\mid A \mid - \mid T \mid\mid$ - Diferença entre nº de atacantes e alvos.

As fórmulas são construídas de forma a que exista uma relação linear entre uma correspondência perfeita do respetivo padrão de comunicação ($\delta = 1$), ou apenas parcialmente ($0 < \delta < 1$).

Desta forma, as quatro métricas $\delta 0t0$ 4.1, $\delta 0tN$ 4.2, $\delta Nt0$ 4.3, e δNtN 4.4 indicam a certeza entre [0,1] de uma correspondência com os padrões 1:1, 1:N, N:1, e N:N.

Um *cluster* que tenha um *match* perfeito com o padrão 1:1 contém: 2 endereços IP distintos $\mid V \mid = 2$, 1 alvo $\mid T \mid = 1$, e 1 atacante $\mid A \mid = 1$.

$$\delta 0t0 = \frac{1}{3} \cdot \left(\frac{\mid V \mid - \mid A \mid}{\mid V \mid -1} + \frac{\mid V \mid - \mid T \mid}{\mid V \mid -1} + \frac{\mid V \mid - \mid\mid A \mid - \mid T \mid\mid}{\mid V \mid} \right) \tag{4.1}$$

Neste caso, todos os três somatórios para $\delta 0t0$ ficam a 1. Quantos mais alvos ou atacantes estiverem envolvidos, mais o valor da métrica reduz, convergindo para 0.

No padrão 1:N é esperado: 1 atacante $\mid A \mid = 1$ e $\mid V \mid -1$ alvos, o que resulta numa diferença de $\mid V \mid -2$.

$$\delta 0tN = \frac{1}{3} \cdot \left(\frac{\mid V \mid - \mid A \mid}{\mid V \mid -1} + \frac{\mid T \mid}{\mid V \mid -1} + \frac{\mid\mid A \mid - \mid T \mid\mid}{\mid V \mid -2} \right) \tag{4.2}$$

Portanto, o primeiro somatório da métrica $\delta 0tN$ fica 1, se $\mid A \mid = 1$ e o segundo somatório é 1 se $\mid T \mid = \mid V \mid -1$, resultando na diferença entre atacantes e alvos de $\mid V \mid -2$.

No que diz respeito ao padrão N:1 (métrica $\delta Nt0$), é esperado: $\mid V \mid -1$ atacantes e $\mid T \mid = 1$ alvos, resultando na diferença $\mid V \mid -2$.

$$\delta Nt0 = \frac{1}{3} \cdot \left(\frac{\mid A \mid}{\mid V \mid -1} + \frac{\mid V \mid - \mid T \mid}{\mid V \mid -1} + \frac{\mid\mid A \mid - \mid T \mid\mid}{\mid V \mid -2} \right) \tag{4.3}$$

Uma correspondência perfeita no padrão N:N (métrica δNtN) necessita de ter: $\mid V \mid$ atacantes e $\mid V \mid$ alvos para que a diferença entre eles seja 0.

$$\delta NtN = \frac{1}{3} \cdot \left(\frac{\mid A \mid}{\mid V \mid} + \frac{\mid T \mid}{\mid V \mid} + \frac{\mid V \mid - \mid\mid A \mid - \mid T \mid\mid}{\mid V \mid} \right) \tag{4.4}$$

Tal como referido na secção 3.1, os autores de [12], construíram as fórmulas de modo a serem tolerantes a erros, sendo portanto considerada a incerteza, calculando o valor da certeza que reflete a confiança no tipo de padrão de comunicação identificado.

Neste sentido, a certeza da identificação do tipo de padrão de comunicação é dada por: $max\{\delta 0t0, \delta 0tN, \delta Nt0, \delta NtN\}$. O padrão de comunicação que tenha o valor de certeza maior é associado a um *cluster*.

Considerações

Num *cluster* podem surgir alertas que apesar de serem semelhantes encontram-se bastante afastados no tempo. Neste caso, podem ser criados *clusters* distintos para esses alertas afastados no tempo, sendo desta forma considerados passos de ataque distintos.

Este módulo devolve um cenário de ataque composto pelos passos de ataque identificados.

4.2.6 Redução de Alertas

Este módulo recebe como entrada um cenário de ataque e remove em cada passo de ataque os alertas redundantes. Desta forma, alertas repetidos em cada *cluster* são representados num só alerta, levando a que o conjunto dos alertas seja significativamente reduzido.

Inicialmente, os alertas em cada passo de ataque que apresentem todos os atributos iguais e que estejam próximos a nível temporal (ex.: 2s) são considerados o mesmo alerta. Para tal, é considerado o tamanho exato da dimensão dos pacotes (ao invés de uma gama de valores como no módulo de pré-processamento).

Este procedimento inicial é útil com vista a incluir tais alertas (sem repetições) no relatório final de correlação de alertas.

Posteriormente, estes alertas são novamente processados com vista a que os mesmos sejam utilizados pelo módulo de reconhecimento de cenários. Neste sentido, os alertas são novamente reduzidos, contudo agora com base no protocolo e no número

da regra associada ao alerta, sendo que alertas com tais atributos iguais são considerados o mesmo alerta.

O número da regra associada ao alerta serve como identificador do alerta, contendo informação associada ao mesmo como a classe do alerta e prioridade. No Snort, o número da regra é um identificador único de cada alerta composto por: ID do gerador (que corresponde ao componente do Snort que gerou o alerta), ID da regra (identificador único da regra), e número de revisão (versão da regra), sendo que cada valor se encontra separado por virgulas.

Atividades suspeitas podem criar múltiplos pacotes semelhantes, levando a que possam ser emitido múltiplos alertas de forma repetida [76]. Além disso, tal como referido, alertas de segurança podem também ser produzidos em elevado número como resultado de falsos positivos, sondagens a alvos de forma repetida, e até tentativas de dissimular um ataque real feito paralelamente [9]. Desta forma, é importante realizar estes procedimentos com vista a obter resultados mais precisos e de forma mais eficiente.

Este modulo além de facilitar a análise humana de alertas, prepara o cenário de ataque recebido através do módulo anterior com vista a ser analisado pelo módulo seguinte.

4.2.7 Reconhecimento de Cenários de Ataque

Este módulo recebe como entrada um cenário de ataque identificado e processado pelo sistema de correlação, de modo a ser analisado com o intuito de calcular o grau de semelhança do cenário em análise com os cenários de ataque já conhecidos, com base nos passos de ataque identificados.

É bastante difícil reconhecer o comportamento de atacantes, visto que o mesmo pode mudar com o tempo e ambiente onde é executado [14]. Reconhecer um cenário de ataque automaticamente é bastante desejável, todavia é importante mencionar que existem diversos desafios. Um cenário de ataque pode ser executado de formas distintas, mas que sejam equivalentes para chegar ao mesmo objetivo do atacante. A título de exemplo, a ordem temporal de um ataque pode mudar ou um ataque pode ser substituído por outro que tenha um funcionamento equivalente [9].

Na deteção de cenários de ataque multi-etapa podem ser utilizadas assinaturas de ataques que especificam os ataques constituintes e a ordem dos mesmos. Esta abordagem é utilizada nos sistemas de deteção de intrusões baseados em assinaturas, para a deteção de ataques que envolvam múltiplos eventos. Porém, esta abordagem apresenta fraquezas no reconhecimento de cenários de ataque complexos, nomeada-

mente quando, tal como mencionado, ataques são substituídos por outros equivalentes ou quando a ordem dos ataques é alterada sem afetar o resultado, podendo existir diversas variantes de um cenário de ataque [9].

Nesta subsecção são descritos os passos realizados até chegar ao método criado, responsável por oferecer o grau de semelhança entre cenários de ataque.

No presente módulo considerou-se que se o cenário em análise possuir novos dados a distância entre os cenários não deve aumentar, podendo o cenário conhecido ser um "subconjunto" do que está em análise, todavia a falta de dados em comum no cenário em análise com o cenário conhecido é sim considerada uma distância entre os cenários.

Com vista a dar resposta aos desafios e requisitos deste módulo, foi utilizada a combinação das seguintes técnicas:

Cálculo da distância de hamming 4.5: permite medir a distância mínima de substituições a uma string binária para transformar noutra.

$$hamming\_dist(p, q) = \sum_{i=1}^{n} dist(p_i, q_i) \tag{4.5}$$

Esta medida é utilizada neste módulo para realizar a comparação utilizando o valor binário resultante da existência ou não de alertas e padrões de comunicação equivalentes nos passos de ataque em comparação, sendo que neste módulo são tidos em conta apenas dois atributos, nomeadamente: protocolo e número da regra associada ao alerta.

Como a distância é relativa a dois pontos, pode-se realizar esta comparação utilizando o valor binário resultante da existência ou não de algo equivalente, sendo que vai ser representada a existência com o valor 0 e a não existência com o valor 1 4.6.

$$dist(p_i, q_i) = \begin{cases} 0, & \text{se } p_i = q_i \\ 1, & \text{se } p_i \neq q_i \end{cases} \tag{4.6}$$

Desta forma, o número de alertas em comum num passo de ataque pode ser dado pela fórmula 4.7.

$$count\_equivalent\_alerts(p, q) = \sum_{i=1}^{m} \sum_{j=1}^{n} 1 - dist(p_i, q_j) \tag{4.7}$$

m representa o nº de alertas associados ao passo de ataque do cenário conhecido, e n representa o nº de alertas associados ao passo de ataque do cenário em análise.

Cálculo da distância euclidiana 4.8: possibilita o cálculo da distância entre dois pontos que tenham o mesmo número de dimensões entre si.

$$euclidean\_dist(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{4.8}$$

Esta distância é utilizada no presente módulo para calcular a distância entre o número de alertas equivalentes nos passos de ataque em comparação.

Visto que o cálculo é utilizado em apenas uma dimensão, este pode ser simplificado para a fórmula 4.9.

$$euclidean\_dist(x_i, y_i) = |x_i - y_i| \tag{4.9}$$

Com vista em obter um valor da distância euclidiana no intervalo de [0,1] (pois a escala entre este valor e o valor da distância de hamming é díspar) é aplicada a normalização 4.10.

$$min\_max(X) = \frac{X - Xmin}{Xmax - Xmin} \tag{4.10}$$

O resultado da normalização, é atribuído a $alerts\_dist(p, q)$.

É também importante ter em consideração o padrão de comunicação (1:1, 1:N, N:1 ou N:N) associado a um passo de ataque no caso de existirem alertas equivalentes entre os passos de ataque em comparação. Neste sentido, a distância entre os padrões de comunicação dos passos de ataque é dada pela fórmula 4.11.

$$comm\_pattern\_dist(p, q) = \begin{cases} dist(p.pattern, q.pattern), & \text{se } alerts\_dist(p, q) < 1 \\ 1, & \text{se } alerts\_dist(p, q) = 1 \end{cases} \tag{4.11}$$

Após estas considerações, chegou-se à fórmula 4.12 para calcular a distância entre passos de ataque.

$$attack\_step\_dist(p, q) = alerts\_dist(p, q) + comm\_pattern\_dist(p, q) \tag{4.12}$$

Com isto é obtida uma generalização do cálculo da distância entre passos de ataque. Com o intuito de obter a distância mínima entre pares de passos de ataque, é utilizada a fórmula 4.13.

$$min\_attack\_step\_dist(\alpha, new\_\alpha) = \min_{\substack{1 \le i \le m \\ 1 \le j \le n}} attack\_step\_dist(\alpha_i, new\_\alpha_j) \tag{4.13}$$

m representa o número de passos de ataque do cenário conhecido, n representa o número de passos de ataque do cenário em análise, new_α o cenário em análise e α o cenário conhecido.

Tendo em conta as fórmulas mencionadas, surgiu o algoritmo apresentado na Listagem 4.4 para calcular a distância entre cenários.

```
1 Input: Known scenario N and scenario under examination S
2  foreach n ∈ N do
3       Get distance score of the most similar attack step of S to
            the attack step n
4       Append distance score of the most similar pair of attack
            steps
5       Remove the most similar pair of attack steps from distance
            comparison
6  end foreach
7  Return Sum of distance score / (length(N) * 2)
```

Listagem 4.4: Algoritmo para o cálculo da distância entre cenários

É importante referir que o método para calcular o grau de semelhança entre cenários de ataque é caracterizado por ser genérico, podendo ser ajustado (ex.: em termos de pesos) de acordo com as necessidades.

No caso de não serem encontrados cenários semelhantes, sugere-se que o cenário em análise seja armazenado para ser investigado com vista a que se torne num cenário conhecido e que se tome as devidas medidas defensivas.

Este módulo abre portas a diversas funcionalidades deveras importantes que podem ser tidas em conta, nomeadamente a previsão de passos de ataque, levando a que seja possível evitar que um ataque atinja o seu objetivo final (ao ser reconhecido o ataque completo enquanto ele se encontra a ser realizado).

É fundamental referir que no reconhecimento de cenários de ataque não são tidos em conta atributos como endereços IP, portas, TCP flags, tamanho dos pacotes e data associada a cada alerta pois são atributos que facilmente variam num ataque.

Ao contrário de muitos outros sistemas de correlação de alertas, o presente trabalho ao reconhecer cenários de ataque com base em passos de ataque individuais permite oferecer uma abordagem flexível ao não necessitar que o atacante realize a mesma ordem de ataque e ao ser tolerante a passos de ataque que são substituídos por outros ou não são realizados.

Este módulo é especialmente eficaz em ataques multi-etapa onde seja realizado um *modus operandi* semelhante, podendo ter a capacidade de reconhecer ataques de um atacante ou grupos de atacantes que tenham comportamentos comuns. No projecto ATT&CK do MITRE [77] são descritas formas de ataque conhecidas por

serem utilizadas por grupos de atacantes, sendo também mencionados os *software* e as técnicas usadas com recurso aos mesmos por tais grupos de atacantes.

Ainda que cada ataque multi-etapa realizado por um atacante possa não ser exatamente igual, visto que os sistemas numa rede podem também ser diferentes, é possível existirem passos de ataque em comum, e o padrão de procedimentos a seguir pode ser semelhante num atacante ou grupo de atacantes.

Caso se conheça a "assinatura" de um grupo de atacantes, ou seja, tipos de ataques e procedimentos associados a grupos de atacantes é possível contribuir no reconhecimentos destes indivíduos a partir dos passos de ataque realizados por estes, sendo útil, a título de exemplo, numa investigação forense. Ao ter em conta os passos de ataque é possível ter um grau de especificidade suficiente para o reconhecimento de cenários de ataque de forma eficaz.

É importante referir que, para uma maior eficácia, os cenários conhecidos na base de conhecimento devem ser criados a partir do mesmo conjunto de regras (dos NIDSs) que foram utilizadas para identificar os ataques.

Para armazenar um cenário de ataque conhecido na base de conhecimento não é necessário que todos os módulos do sistema de correlação sejam utilizados. Na Figura 4.7 apresenta-se uma ilustração da arquitetura de sistema adotada para o armazenamento de cenários conhecidos.

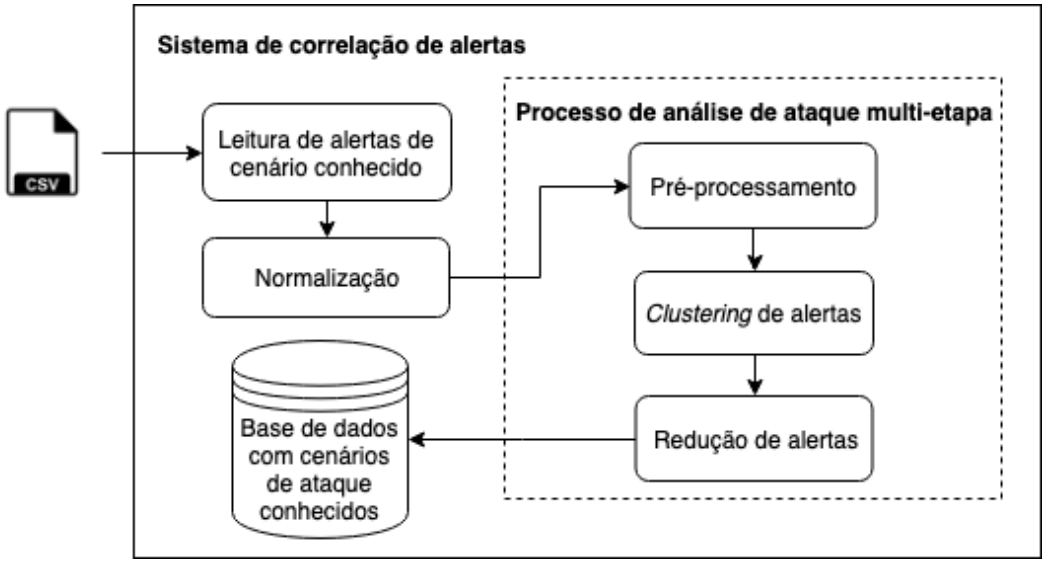

Figura 4.7: Arquitetura de sistema para o armazenamento de cenários conhecidos

Os cenários de ataque conhecidos são importados no sistema através de ficheiros no formato CSV (Comma Separated Values), onde cada linha de um ficheiro repre-

senta um alerta associado ao cenário, de modo a facilitar a leitura dos cenários pelo sistema.

4.2.8 Protótipo de Prova de Conceito

Com o intuito de exclusivamente testar e demonstrar a potencialidade do sistema proposto foi desenvolvido um protótipo de prova de conceito tendo como linguagem de programação principal o Python dado a sua compatibilidade com diferentes sistemas, e devido às bibliotecas existentes que são uma mais valia para dar resposta às necessidades do sistema.

Nos testes realizados ao sistema proposto, descritos no capítulo 5, foi utilizada uma base de dados SQLite [78] para o armazenamento de dados estruturados relativos aos cenários de ataque conhecidos. A escolha desta base de dados justifica-se pela simplicidade, portabilidade (podendo ser livremente distribuída) e finalidade que seria utilizada. É importante referir que as transações no SQLite são atómicas, consistentes, isoladas e duráveis (conceito ACID).

Na Figura 4.8 encontra-se o diagrama de classes utilizado para a base de dados responsável por armazenar os cenários de ataque conhecidos.

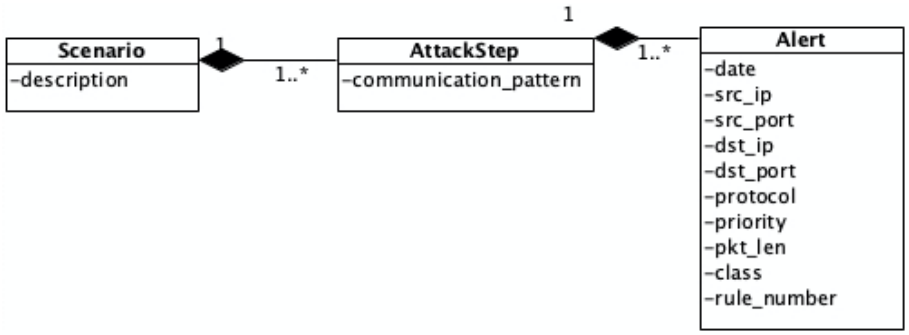

Figura 4.8: Diagrama de classes utilizado para a base de dados

Foi também desenvolvida uma pequena aplicação web, com o intuito meramente demostrativo, com recurso à *framework* Flask [79] para apresentar alguns dos resultados do sistema de correlação, com vista a facilitar o entendimento dos mesmos.

Na Figura I.1 do apêndice I é possível observar a página principal da aplicação web. Nesta página é possível: obter os relatórios de correlação de alertas gerados pelo sistema, gerir a captura de alertas em tempo real, e iniciar uma nova análise de correlação de alertas, sendo consequentemente gerado um relatório no formato PDF com o resultado final da correlação, onde são incluídos os alertas não redundantes de cada cenário analisado, o grau de semelhança dos cenários identificados com cada

um dos cenários conhecidos, e o número de cenários identificados (representando o número de ataques multi-etapa analisados).

Na Figura I.2, presente no apêndice I, é possível visualizar um excerto de um exemplo de um relatório gerado.

Na aplicação web é utilizado o formato JSON com o intuito de transferir dados de forma estruturada entre o sistema de correlação e a aplicação web.

Durante o desenvolvido deste protótipo foi utilizado um controlo de versões Git, sendo utilizado como serviço de alojamento o GitLab. Foram também realizados testes unitários e de integração com vista a economizar o tempo de implementação e de testes.

O projeto encontra-se disponível para consulta no endereço:

https://gitlab.com/joaoorvalho/projecto_dissertacao

Além do código desenvolvido, no projeto encontram-se incluídos exemplos dos ficheiros de configurações utilizados para o Logstash, comandos úteis para as ferramentas utilizadas e para o sistema proposto, e são incluídos os conjuntos de dados criados para fins de teste.

Foram também desenvolvidas funções para importar cenários de ataque conhecidos no formato CSV e funções para converter ficheiros com conjuntos de alertas no formato JSON para CSV.

Todos os módulos mencionados no presente capítulo foram implementados com recurso a diversas bibliotecas disponíveis no Python. As principais bibliotecas utilizadas foram:

- Pika [80] - Cliente utilizado no módulo de receção de alertas para interagir com o RabbitMQ broker;
- NetworkX [81] - Utilizado no módulo de divisão de ataques para criar os grafos e para aplicar o CPM;
- LibPrelude [82] - Biblioteca *open-source* utilizada no módulo de normalização que realiza a implementação do IDMEF;
- Pandas [83] - Utilizado no módulo de pré-processamento para aplicar o método "One Hot Encoding";
- rpy2 [84] - Aplicado no módulo de *clustering* de alertas para utilizar o pacote NbClust;
- Reportlab [85] - Biblioteca para gerar relatórios PDF com o resultado da correlação;

- Sqlite3 - Utilizado para gerir a base de conhecimento de cenários conhecidos;
- Flask - Utilizado para suportar a aplicação web.

O sistema foi desenvolvido com recurso ao *software* PyCharm [86].

4.2.9 Considerações Finais

No presente trabalho, um ataque de rede multi-etapa é representado por um cenário de ataque composto por diversos *clusters*, onde cada *cluster* representa um passo de ataque.

Durante a investigação realizada neste trabalho foram efetuados diversos testes utilizando, numa fase inicial, o *software* Weka [87] que é composto por uma coleção de algoritmos *open source* para *data mining*. Estes algoritmos possuem aplicações para pré-processamento de dados, regras de associação, classificação, correlação e visualização. Esta ferramenta foi uma mais valia para o presente trabalho.

Um atacante pode realizar diversas ações para alcançar o seu objetivo. O processo de correlação é bastante importante pois facilita a identificação da intenção do atacante.

Ferramentas como o Inundator [88], que geram tráfego malicioso falso (levando a diversos falsos positivos) com o intuito de despistar o verdadeiro tráfego malicioso, originam diversos desafios neste tipo de sistemas.

Capítulo 5

Avaliação Experimental

Com o intuito de avaliar o sistema desenvolvido, são apresentados neste capítulo os resultados obtidos com base em testes realizados com duas experiências num ambiente virtualizado, com vista a validar a eficácia do sistema proposto e provar que o sistema desenvolvido dá resposta aos requisitos do sistema enunciados em 4.1.

Os testes permitem identificar elementos que podem ser alvo de melhoria e, se necessário, correção.

Na literatura é possível verificar diversos conjunto de dados, nomeadamente: DARPA 2000, DEF CON (com base no concurso "capture the flag"), NSA, entre outros. Contudo, tal como mencionado, é importante realçar a falta de conjuntos de dados relevantes com dados reais que possam refletir situações complexas encontradas na realidade, devido a problemas de privacidade que impedem as organizações de tornarem tais dados públicos [1].

O conjunto de dados público mais importante e mais utilizado é o DARPA 2000, patrocinado pela Agência de Projetos de Pesquisa Avançada de Defesa nos Estados Unidos e gerado pelo MIT Lincoln Laboratory, com vista a ser utilizado por métodos de análise de ataques de alto nível e reconhecimento dos objetivos de ataque. Este conjunto de dados contém tráfego de duas redes e dados de auditoria provenientes de máquinas Solaris [1]. Não obstante, por ser um conjunto de dados partilhado no ano 2000, as suas assinaturas de alertas não abrangem novas técnicas de ataque, levando a que atualmente não seja utilizado este conjunto de dados em diversos trabalhos.

O desenvolvimento de aplicações informáticas em ambiente virtualizado apresenta vantagens na rapidez de deteção e resolução de problemas. Neste sentido, tendo em conta o que foi referido no presente capítulo, optou-se por realizar duas experiências de ataque distintas com recurso ao *software* de virtualização VirtualBox [89].

Tendo em contas as limitações em termos de *hardware* para o ambiente de virtualização e com vista a facilitar o processo de realização de testes, foi utilizado o *sniffer* Wireshark [90] para a captura das comunicações em rede, que posteriormente foram analisadas (a partir do formato .pcap) pelo IDS Snort 3.0.

Foram utilizadas as regras do Snort para utilizadores registados e foi adicionado uma regra personalizada para a deteção do ataque de SYN *flood* realizado na experiência 2 com vista a que fossem deixados rastos (na forma de alertas) dos ataques realizados.

Com vista a avaliar os resultados das duas experiências, foi validado o desempenho do sistema a identificar *clusters* e a reconhecer cenários de ataque conhecidos. Na experiência 2 foi ainda avaliada a função de divisão de ataques do sistema.

A precisão na identificação de *clusters* é dada utilizando duas medidas principais, nomeadamente:

- Taxa de erro - Percentagem de alertas que ficaram em *clusters* que não deviam, dado por (Erro de *clustering* / Total de alertas) x 100, sendo que o erro de *clustering* representa o número de alertas que ficaram em *clusters* que não deviam;

- Precisão - Percentagem de alertas que ficaram nos *clusters* corretos, dado por 100 - Taxa de erro.

A avaliação dos testes foi realizada com base na experiência e conhecimento possuído, sendo que os alertas utilizados para a validação e teste de *clustering* foram rotulados com a identificação prevista de cada *cluster*.

5.1 Experiência 1

Esta experiência foi composta por um atacante com o SO Kali Linux instalado, definido com o endereço IP 192.168.0.3/24, e duas vítimas com os endereços IP 192.168.0.1/24 (vítima 1) e 192.168.0.2/24 (vítima 2), sendo que as vítimas têm o SO Windows XP SP2 e Windows 7 SP1 instalado respetivamente.

Para esta experiência foram realizadas as seguintes etapas de ataque pelo atacante:

1. Reconhecimento de SOs e descoberta de serviços e versões em execução nas vítimas com Nmap em ambas as vítimas;

2. Invasão de vítima 1 com *exploit* "exploit/windows/smb/ms08_067_netapi" do Metasploit com recurso a uma *reverse shell* (com o *payload* "windows/meterpreter/reverse_tcp");

3. Invasão de vítima 2 com *exploit* "caseiro", escrito em Python, para explorar falha de Remote Buffer Overflow de aplicação SLmail 5.5 em execução na vítima 2, sendo também utilizada uma *reverse shell*.

Um ataque de *buffer overflow* é um caso especial de violação de segurança de memória (anomalia), onde um *software* ao escrever dados num *buffer*, ultrapassa os limites do mesmo (*buffer*), sobrescrevendo a memória adjacente, podendo levar a que sejam sobrescritos registos críticos do processador como o Extended Instruction Pointer, que armazena o endereço em memória da próxima instrução a ser executada [29].

Os ataques por *buffer overflow* foram uma das vulnerabilidades de segurança mais comuns há alguns anos, sendo que atualmente é notável a preocupação com o desenvolvimento de *software* cada vez mais elaborado, e com maior controlo de qualidade. Ainda assim, muitas técnicas que surgiram com o intuito de dificultar a exploração de falhas de *buffer overflow* falharam.

As vulnerabilidades a ataques por *buffer overflow* começaram a ser estudadas em reposta aos mecanismos de proteção existentes. Apesar da existência de alguns obstáculos, tais como o Data Execution Prevention que impede certas áreas da memória de serem executadas (ex.: a stack), o Structured Exception Handler que é um mecanismo nativo do Windows para manipular exceções, e o Address Space Layout Randomisation que permitem definir aleatoriamente endereços de memória, não levou a que as explorações de falhas de *buffer overflow* se tornem impraticáveis.

Vários *exploits* dependem do conhecimento prévio dos endereços de memória. Portanto, a aleatoriedade é interessante como uma forma de proteção genérica. Contudo, basta que algum endereço seja fixo para que esta proteção não tenha efeito algum.

Foram utilizadas *reverse shells* para fazer ligações remotas das máquinas das vítimas à maquina do atacante. Algumas *firewalls* estão destinadas a bloquear o tráfego de entrada por padrão, permitindo o tráfego de saída. Isto porque é comum confiar nos dispositivos da rede interna e bloquear os dispositivos exteriores que inicializam comunicações com a rede interna.

No caso de as comunicações serem feitas para a porta 443 da máquina externa (do atacante), as comunicações são assumidas como HTTPS o que facilita ainda mais a permissão de ligação ao exterior.

5.1.1 Resultados da Identificação de *Clusters*

Foram emitidos no total 33 alertas, sendo identificados 9 *clusters* pelo algoritmo HAC. Os resultados da identificação de *clusters* apresentaram uma taxa de erro de $\simeq 12\%$ e consequentemente uma precisão de $\simeq 88\%$, ao existir um erro de 4 alertas.

Nesta experiência, devido à natureza dos ataques, foram emitidos alertas bastante semelhantes entre si, apesar de pertencerem a *clusters* distintos.

No que se refere aos erros, os 4 alertas foram atributos a dois *clusters* individuais erroneamente, onde cada *cluster* tem dois alertas, pois foram classificados pelo Snort com um tipo de ataque distinto relativamente a outros alertas semelhantes como consequência de regras complementares do Snort. Ainda assim, com base no tipo de ataque, os alertas atribuídos aos dois *clusters* encontram-se relacionados.

Os alertas emitidos associados a indicadores de *shellcode* relativos aos ataques ao SLmail e à exploração da vulnerabilidade MS08-067 foram atribuídos pelo HAC ao mesmo *cluster*. Considerou-se sensato, neste caso, aceitar tal solução dada a natureza dos ataques em questão. No entanto, ainda assim, o módulo de *clustering* de alertas pode atribuir tais alertas em *clusters* separados, dado que os mesmos se encontram afastados no tempo, levando a considerar, acertadamente, dois passos de ataque distintos.

5.1.2 Resultados do Reconhecimento de Cenários de Ataque

Com vista a testar o reconhecimento do cenário de ataque identificado, foi adicionado à base de conhecimento um ataque completo com as seguintes etapas de ataque:

1. Reconhecimento de SOs e descoberta de serviços e versões em execução nas vítimas com Nmap em ambas as vítimas;

2. Detetar se máquina da vítima 1 é vulnerável a execução de código remoto através da vulnerabilidade com o CVE (Common Vulnerabilities and Exposures) "CVE-2008-4250" (conhecida como MS08-067) com o *script* do Nmap "smb-vuln-ms08-067.nse";

3. Invasão de vítima 1 com *exploit* "exploit/windows/smb/ms08_067_netapi" do Metasploit com recurso a uma *reverse shell* (com o *payload* "windows/meterpreter/reverse_tcp").

Apesar do ataque completo do atacante não ser exatamente igual ao cenário conhecido, existem ataques em comum. O cenário conhecido encontra-se incluído

nos ataques do atacante, com exceção da etapa de verificação se a vulnerabilidade "CVE-2008-4250" é explorável na máquina da vítima 1.

O sistema conseguiu reconhecer os passos de ataque equivalentes, apresentando um grau de semelhança de 88.89%.

5.2 Experiência 2

Esta experiência foi realizada tendo duas máquinas atacantes com o SO Kali Linux instalado, definidos com os endereços IP 192.168.0.2/24 (atacante 1) e 192.168.0.3/24 (atacante 2), e uma vítima com o endereço IP 192.168.0.1/24 e com o SO Lubuntu composto por um servidor WEB Apache e um servidor SSH.

Os ataques realizados na presente experiência foram:

1. Atacante 1 identifica máquinas ativas na rede com recurso a "ICMP echo probes" através do *software* fping;
2. Atacante 1 realiza análise TCP com o *software* Nmap com vista a descobrir mais informações sobre os serviços e versões em execução na máquina da vítima;
3. Atacante 2 realiza um ataque, através de um *script* de python "caseiro"(através da biblioteca de Python scapy [91]), de SYN *flood* com recurso a IP *Spoofing* com alteração de IP de origem e porta de origem frequente, tendo como alvo a porta 80 do servidor Web;
4. Simultaneamente ao ataque de SYN *flood* é realizado um ataque de força bruta SSH do tipo dicionário pelo atacante 1 com recurso ao *software* THC Hydra.

Neste caso, o ataque de SYN *flood* do atacante 2 serviu para tentar retirar o foco de atenção do ataque de força bruta realizado em paralelo pelo atacante 1.

Um ataque de SYN *flood* funciona de forma semelhante ao HTTP *flood*. O atacante inunda a vítima com pacotes TCP com a flag SYN ativa, sendo que o endereço IP de origem é geralmente mascarado (pois o tipo de ataque assim o permite). Cada pacote enviado pelo atacante tem como intenção realizar uma ligação, o que leva a vítima a reservar uma determinada quantidade de memória para cada ligação e devolver um pacote TCP SYN-ACK para o qual espera uma resposta TCP ACK do cliente (neste caso o atacante), que irá permitir estabelecer uma nova ligação.

Como os pacotes TCP ACK esperados nunca são enviados pelo atacante, quando a memória da vítima é completamente reservada os pedidos legítimos de ligações são

impedidos de serem atendidos até que o "time to live" do pacote TCP expire ou o ataque acabe.

É difícil saber se um pacote SYN é legitimo, levando a que um ataque deste tipo leve a que a vítima seja inundada através de muitas tentativas de ligação.

5.2.1 Resultados da Identificação de *Clusters*

Foram emitidos no total 440 alertas, sendo identificados 6 *clusters* pelo algoritmo HAC. Os resultados da identificação de *clusters* apresentaram uma taxa de erro de 0.2% e consequentemente uma precisão de 99.8%, ao existir um erro de 1 alerta.

No que se refere aos erros, o alerta foi atribuído a um *cluster* individual erroneamente visto que, tal como aconteceu na experiência 1, foi atribuído ao alerta em questão uma classificação de ataque distinta em comparação a outros alertas semelhantes como consequência de regras do Snort complementares.

Ainda assim, a maioria dos alertas relativos a *clusters* distintos que tinham a mesma classificação de ataque foram consideradas corretamente pelo HAC como *clusters* diferentes.

Ainda que no ataque de Syn *Flood* realizado pelo atacante 2 as portas de origem e os endereços IP de origem variassem constantemente, o HAC considerou acertadamente que os alertas pertenciam ao mesmo *cluster*. Além disso, alguns alertas relativos ao ataque de força bruta encontravam-se misturados com os alertas de Syn *Flood* (nos *logs* do Snort), mesmo assim, o desempenho do sistema de correlação não foi afetado.

Dado o elevado número de alertas, devido à natureza dos ataques e ao tipo de regras do Snort existentes, é uma mais valia o sistema ter a capacidade de remover alertas redundantes, permitindo facilitar o entendimento dos ataques ocorridos.

Considerou-se que o ataque de força bruta deu origem a mais do que um passo de ataque (ao criar mais que um *cluster*), pois além das excessivas tentativas de autenticação, também foram emitidos alertas aquando da troca de chaves criptográficas devido a um padrão, considerado pelo Snort como suspeito, presente no código binário.

É importante destacar que os alertas emitidos, como consequência dos ataques realizados nesta experiência, caracterizam-se por não serem muito semelhantes entre passos de ataque, justificando desta forma o excelente grau de precisão obtido na identificação de passos de ataque.

5.2.2 Resultados do Reconhecimento de Cenários de Ataque

Para testar o reconhecimento do cenário de ataque identificado, foi adicionado à base de conhecimento um ataque completo com todos os ataques mencionados, com exceção da etapa inicial (de descoberta de máquinas ativas).

O sistema conseguiu reconhecer os passos de ataque equivalentes, apresentando um grau de semelhança de 100.0%, pois neste caso todos os ataques do cenário conhecido se encontravam incluídos nos ataques realizados pelo atacante.

5.2.3 Resultados da Divisão de Ataques

Um ambiente de rede não controlado oferece diversos desafios aos sistemas de correlação de alertas, entre eles: existirem milhares de eventos, ataques não relacionados em simultâneo e a existência de vários falsos positivos.

O módulo de divisão de ataques permite separar alertas de modo a evitar que alertas potencialmente não relacionados sejam analisados em conjunto, sendo desta forma evitado misturar "lixo" com alertas úteis, o que poderia afetar o desempenho do sistema.

Com vista a avaliar esta função do sistema, além dos ataques (relativos ao atacante) realizados na experiência 2 foram adicionados os seguintes ataques realizados em simultâneo:

1. Descoberta de serviços e versões em execução nas vítimas com Nmap a vítimas com os IPs 192.168.0.1/24 e 192.168.0.4/24;

2. Com o Metasploit invadir vítima com IP 192.168.0.4/24 com recurso ao *exploit* "exploit/windows/smb/ms08_067_netapi" e a uma *reverse shell* (com o *payload* "windows/meterpreter/reverse_tcp").

A vítima com o IP 192.168.0.1 é a mesma que na experiência 2 original, e a vítima com o IP 192.168.0.4 é uma nova vítima com SO Windows XP SP2. O atacante, com o SO Kali Linux instalado, associado aos ataques adicionais foi definido com o IP 192.168.0.5/24.

Nestes ataques adicionados, apesar de na primeira etapa de ataque existir uma vítima em comum com a experiência 2 original, o ataque multi-etapa concentrou-se essencialmente noutro foco (neste caso na vítima com o IP 192.168.0.4/24).

O sistema conseguiu separar com sucesso os alertas relativos à experiência 2 original dos alertas associados aos ataques acrescentados, levando a que tivessem

sido identificados dois ataques multi-etapa, ao ser definido no módulo de divisão de ataques um k=4. Consequentemente, os dois ataques identificados foram analisados em separado, levando a que o cenário de ataque adicionado, realizado em simultâneo com outros ataques, não tivesse qualquer interferência no desempenho do sistema de correlação.

5.3 Considerações Finais

Os resultados de ambas as experiências foram bastante animadores.

É importante referir que erros de separação (alertas que estão noutro *cluster* em relação ao que deveriam estar, apesar dos elementos desse *cluster* estarem relacionados) é muito menos grave que erros onde são criados *clusters* compostos por alertas não relacionados.

Os ataques relativos aos cenários de ataque conhecidos foram realizados à parte (sem ter em conta os alertas provenientes dos ataques simulados do atacante) com vista a simular um ataque real.

Diversas abordagens de correlação analisadas enfatizam aspetos distintos de correlação, dificultando a comparação de resultados de cada solução. Além disso, a utilização de um conjunto de dados distintos com expectativas de resultados díspares, dificulta a comparação do desempenho entre sistemas de correlação de alertas semelhantes.

O número de alertas em análise poderia ser maior no caso de se ter em conta mais vítimas e/ou atacantes.

Capítulo 6

Considerações e Trabalho Futuro

Neste capítulo são mencionadas algumas considerações e trabalhos futuros a realizar com vista a dar continuidade ao sistema, permitindo assim futuras evoluções do mesmo.

Como é sabido, diferentes IDSs podem ter as suas próprias normas, portanto quando diferentes IDSs detetam os mesmo ataques, eles podem gerar alertas com terminologias diferentes. Desta forma, é conveniente uniformizar o formato dos alertas, particularmente na classificação (categoria de ameaça) dos mesmos com vista a que o maior número de IDSs distintos sejam compatíveis para trabalhar cooperativamente.

Apesar do módulo de divisão de ataques permitir evitar que alertas não relacionados sejam analisados em conjunto, é benéfico o estudo de outras abordagens que possam ser aplicadas neste trabalho cujo o principal intuito seja a redução de falsos positivos provenientes dos NIDSs, pois os mesmos (falsos positivos) podem afetar os resultados do sistema de correlação.

Tecnicamente, os alertas falsos positivos podem ser causados, por exemplo, por [8]:

- Limitações de *runtime* - por vezes o contexto em que uma atividade ocorre determina se a mesma é ou não intrusiva, contudo nem sempre é possível analisar o contexto de todas as atividades;

- Especificação na deteção de assinaturas - escrever assinaturas que descrevam padrões de intrusão é uma tarefa bastante difícil pois, em alguns casos, o balanço entre uma assinatura demasiado especifica que pode não capturar todos os ataques e as suas variações, e uma assinatura demasiado geral que classifica ações legitimas como intrusões, pode ser difícil de determinar;

- Dependência do ambiente - ações que são normais em certos ambientes podem ser consideradas maliciosas em outros.

Ao ter a capacidade de reconhecer cenários de ataque multi-etapa numa fase inicial enquanto se encontram em progresso, é possível ter a capacidade de prever os possíveis passos de ataque futuros. Desta forma, pode ser possível tomar uma ação para parar um ataque multi-etapa antes que seja alcançado o objetivo final do mesmo. Neste sentido, esta abordagem pode ser bastante benéfica ao permitir reduzir significativamente os danos causados por ataques numa rede.

Com vista a prever ataques, é necessário ter especial atenção ao tempo de processamento dos alertas. Desta forma, é necessário adaptar o sistema para que o mesmo consiga dar resposta em tempo útil.

Ainda que os alertas sejam dos "vestígios" mais utilizados na deteção de ataques multi-etapa, poderá ser benéfico considerar diversos tipos de eventos com vista a melhorar a aplicabilidade deste tipo de sistemas em redes reais [1].

Ter também em conta alertas de HIDS permite ter uma visão ainda mais abrangente dos ataques realizados, pois assim é também possível saber o que acontece dentro de cada *host*. Nas futuras evoluções do sistema poderá ser vantajoso ter também em consideração alertas de HIDS.

No futuro será realizado um estudo aprofundado sobre a dimensão dos pacotes de diversas aplicações pois o mesmo poderá beneficiar consideravelmente a qualidade dos resultados do sistema. A título ilustrativo, as ferramentas Nmap e Advanced Port Scanner [92] recorrem durante as suas análises a pacotes TCP SYN diferentes, sendo que o tamanho do pacote IP no caso do NMAP é de 44 bytes e do Advanced Port Scanner é 52 bytes. A razão encontra-se na forma como os pacotes TCP SYN são criados, isto porque alguns *scanners*, tal como o Nmap, são criado por uma interface "raw socket" para utilizarem cabeçalhos personalizados, em vez dos mesmos serem criados por padrão pelo SO.

No que diz respeito ao reconhecimento de cenários de ataques, além de se ter em conta os passos de ataque, ter também em consideração as etapas de ataque pode melhorar a capacidade de reconhecimento de ataques pelo sistema. Neste sentido, é necessário um componente responsável por revelar individualmente cada etapa de ataque.

A continuação da realização de testes ao sistema de correlação em diversos ambientes, em especial em ambiente reais, é benéfico com vista a serem identificadas possíveis melhorias que possam ser realizadas.

A continuação do estudo dos atributos e dos algoritmos a serem utilizados durante a correlação de alertas poderá também ser vantajoso, com vista a melhorar

ainda mais a precisão deste tipo de sistemas.

Para aplicações em tempo real é necessário considerar uma janela de tempo para os alertas recebidos, de forma a que seja permitido iniciar a correlação de alertas automaticamente, por exemplo, quando são detetados alertas com uma criticidade elevada. É importante, no entanto, ter em consideração que podem surgir diversos desafios. A título de exemplo, um atacante pode dificultar a correlação de alertas ao aumentar o intervalo de tempo entre etapas de ataque ou preenchendo a janela de tempo com alertas inúteis com o intuito de enganar [4]. A realização de um estudo sobre a melhor forma de como estes sistemas podem iniciar, de forma totalmente automática, uma análise de correlação de alertas (além da recolha de alertas) é importante.

Capítulo 7

Conclusões

Como consequência do número crescente de ataques informáticos tornou-se crucial adotar um conjunto de estratégias que visassem proteger a informação dos sistemas de informação. A realização de ataques multi-etapa está provavelmente maior do que nunca [1].

A segurança da informação é um problema cada vez mais relevante para as organizações. A ocorrência de intrusões informáticas pode causar consideráveis prejuízos.

Tem existido cada vez mais sofisticação em termos dos ataques levados a cabo por cibercriminosos, originando um elevado número de ataques sem precedentes realizados a qualquer momento e a partir de qualquer parte do mundo [93].

O aumento da segurança de um sistema não é garantia total de que este será inviolável. A segurança dos sistemas informáticos atuais exige mais do que a instalação e manutenção de *firewalls*, o controlo das atualizações de *software* ou a sensibilização dos utilizadores para a segurança.

A segurança de sistemas de informação é um problema complexo pois trata-se de uma luta desigual. Ao contrário de um atacante que tem que explorar apenas um vetor de ataque, o defensor tem de garantir a eliminação de todas as fraquezas [39]. É impossível eliminar todas as vulnerabilidade de um sistema considerando a complexidade na construção de tais sistemas e a interatividade entre os seus componentes. Os desafios da segurança colocam uma grande pressão sobre as organizações.

Encontrar uma forma eficaz de proteger sistemas de informação dentro de uma infraestrutura de informação critica é desafiante mesmo com a tecnologia mais avançada e profissionais treinados [94].

A cibersegurança está cada vez mais na ordem do dia, prova disso é a primeira Estratégia Nacional de Segurança do Ciberespaço, visando aprofundar a segurança das redes e dos sistemas de informação e potenciar uma utilização livre, segura e

eficiente do ciberespaço, por parte de todos os cidadãos e das entidades públicas e privadas [95].

A correlação de alertas permite ajudar a encontrar as relações entre os alertas que indiquem os motivos e métodos de uma tentativa de ataque [32].

Modelar cenários de ataque multi-etapa é uma tarefa bastante difícil que enfrenta múltiplos desafios. Ainda assim, a abordagem inovadora adotada permite além de identificar ataques multi-etapa, calcular o grau de semelhança dos ataques identificados com os cenários de ataque conhecidos, presentes numa base de conhecimento, através da comparação de passos de ataque.

A correlação de alertas depende dos sistemas que emitem os alertas, levando a que em diversos casos os IDSs sejam utilizados em conjunto com os analistas de segurança. Alguns ataques que façam parte de um cenário de ataque podem não ser mencionados nos alertas, dada a falta de cobertura de sensores de segurança, ou porque o ataque é indistinguível de uma atividade considerada normal [9]. Desta forma, recuperar as sequências de ataques exatas é deveras difícil devido à natureza imperfeita dos sensores [6].

Estudar os comportamentos dos atacantes é um trabalho desafiante visto que estes tendem a mudar o seu comportamento com o objetivo de não serem identificados. Além disso, à medida que novas vulnerabilidades são continuamente descobertas, os atacantes podem utilizar novas estratégias de ataque [48].

É fundamental existir um sistema que analise alertas de baixo nível produzidos por NIDSs, isto porque pode não ser fácil localizar a origem e alvo das intrusões ou a falha em questão ao analisar estes alertas, e porque sensores de baixo nível consideram alertas de forma isolada, sem considerar ligações lógicas entre os alertas [44].

Apesar dos diversos desafios associados às técnicas de correlação de alertas, as técnicas existentes oferecem a possibilidade de reconstruir cenários de ataque [14].

A análise de alertas a partir de sistemas de correlação de alertas permite apresentar um guia importante para planear e desenvolver mecanismos de resposta e de prevenção, sendo que a automatização destes sistemas é crucial não apenas com vista a obter planos de resposta precisos e confiáveis, mas também para revelar as estratégias de ataque em constante mudança [13].

Neste trabalho, além de ter sido descrito o sistema de correlação de alertas proposto, foi implementado um protótipo de prova de conceito com o intuito de demonstrar a potencialidade do sistema.

Com base nos testes realizados, a solução alcançada demonstrou ser eficaz no que se propunha realizar devido às técnicas que implementa, constituindo um mecanismo

de defesa acessível e preparado para os futuros desafios da segurança da informação e dos sistemas de informação. No entanto, é fundamental a continuação do seu desenvolvimento e a realização de testes em diferentes ambientes, de forma a permitir futuras evoluções do sistema.

O CPM revelou ser útil em separar ataques potencialmente não relacionados.

A adoção de *machine learning* de aprendizagem não supervisionada mostrou ser uma mais valia na revelação de passos de ataque. Ainda assim, não se descarta a hipótese de no futuro também se recorrer a outro tipo de algoritmos com vista a obter resultados ainda mais precisos.

O presente trabalho, ao contrário de múltiplos trabalhos nesta área de investigação, ao reconhecer cenários de ataque com base em passos de ataque, permite oferecer uma abordagem flexível, pois apesar de cada ataque multi-etapa poder ser sempre diferente, podem existir passos de ataque em comum, e o padrão de procedimentos a seguir pode ser semelhante num atacante ou grupos de atacantes.

Este sistema pode, a título de exemplo, ser utilizado num *honeypot* de pesquisa (recurso de rede cuja função é de ser atacado e comprometido) com o intuito de serem reunidos dados sobre os ataques e investigados os mesmos, de forma a descobrir novas ferramentas, motivações e novas táticas de atacantes para futuro aperfeiçoamento dos sistemas de deteção de intrusão e assim controlar melhor as ações intrusivas.

Modelar o comportamento de atacantes pode ser importante para melhor perceber como desenvolver regras de deteção, treinar profissionais de segurança, assim como desenvolver mecanismos de resposta apropriados e precisos [1, 3].

O sistema proposto pode também ser uma mais valia em análises forenses. Dado que os alertas não são particularmente importantes se estiverem isolados, encontrar as relações entre alertas com vista a revelar a estratégia de ataque é um importante componente nos sistemas de gestão de segurança [40]. Os sistemas de correlação de alertas permitem oferecer uma grande ajuda, ao encontrarem automaticamente relações entre alertas de intrusão que representem o mesmo ataque e ao indicarem os métodos de ataque numa rede de computadores [32].

No que diz respeito às maiores dificuldades que surgiram no decorrer do desenvolvimento deste trabalho, destaca-se a utilização de técnicas de *machine learning* para dar resposta ao problema essencialmente associado ao módulo de *clustering*, pois não tinha lidado com nenhum desafio semelhante até então. Tecnologias de *machine learning* têm a capacidade otimizar produtos de segurança, levando a que cada vez sejam mais utilizadas nos mesmos.

Para uma defesa efetiva de atacantes, é fundamental conhecer os mesmos, ou seja, conhecer os seus métodos de ataque, as suas ferramentas, táticas e os seus

possíveis objetivos. Neste sentido, os conhecimentos obtidos durante o mestrado foram uma mais valia no desenvolvimento do presente trabalho.

Esta dissertação, diferente de todos os trabalhos apresentados no capítulo 3, foi extremamente desafiante, exigindo bastante trabalho e dedicação.

Bibliografia

[1] J. Navarro, A. Deruyver, e P. Parrend, "A systematic survey on multi-step attack detection," *Computers & Security*, vol. 76, pp. 214–249, Julho 2018. [*Online*]. Disponível: https://doi.org/10.1016/j.cose.2018.03.001 (citado nas págs. 1, 2, 3, 6, 20, 23, 34, 36, 65, 74, 77 e 79)

[2] E. Domingues, "Os Ciberataques como um Novo Desafio para a Segurança: o Hacktivismo," Dissertação de mestrado, Instituto Superior de Ciências Policiais e Segurança Interna, 2015. [*Online*]. Disponível: https://comum.rcaap.pt/bitstream/10400.26/15403/1/Disserta%C3%A7%C3%A3o%20de%20mestrado%20Final%20Elisabete%20Domingues.pdf (citado na pág. 1)

[3] M. Siraj, "Hybrid of structural-causal and statistical model for intrusion alert correlation," Tese de doutoramento, Universiti Teknologi Malaysia, 2013. [*Online*]. Disponível: http://eprints.utm.my/id/eprint/33791/ (citado nas págs. 1, 8, 9, 15, 19, 33, 35 e 79)

[4] Z. Zali, M. R. Hashemi, e H. Saidi, "Real-time attack scenario detection via intrusion detection alert correlation," in *2012 9th International ISC Conference on Information Security and Cryptology*. IEEE, sep 2012, pp. 95–102. [*Online*]. Disponível: http://ieeexplore.ieee.org/document/6408197/ (citado nas págs. 1 e 75)

[5] X. Qin e W. Lee, "Statistical causality analysis of infosec alert data," in *Recent Advances in Intrusion Detection*, G. Vigna, C. Kruegel, e E. Jonsson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 73–93. [*Online*]. Disponível: https://doi.org/10.1007/978-3-540-45248-5_5 (citado nas págs. 1 e 22)

[6] S. Moskal, S. J. Yang, e M. E. Kuhl, "Extracting and Evaluating Similar and Unique Cyber Attack Strategies from Intrusion Alerts,"

in *2018 IEEE International Conference on Intelligence and Security Informatics (ISI)*. IEEE, nov 2018, pp. 49–54. [*Online*]. Disponível: https://ieeexplore.ieee.org/document/8587402/ (citado nas págs. 2, 32 e 78)

[7] O. B. Fredj, "A realistic graph-based alert correlation system," *Sec. and Commun. Netw.*, vol. 8, n. 15, pp. 2477–2493, Outubro 2015. [*Online*]. Disponível: http://dx.doi.org/10.1002/sec.1190 (citado nas págs. 2, 18 e 30)

[8] M. Siraj, H. Albasheer, e M. Din, "Towards predictive real-time multi-sensors intrusion alert correlation framework," *Indian Journal of Science and Technology*, vol. 8, n. 12, 2015. [*Online*]. Disponível: http://www.indjst.org/index.php/indjst/article/view/70658 (citado nas págs. 2, 18, 24, 35, 44, 50 e 73)

[9] S. Cheung, U. Lindqvist, e M. W. Fong, "Modeling multistep cyber attacks for scenario recognition," in *Proceedings DARPA Information Survivability Conference and Exposition*, vol. 1, April 2003, pp. 284–292 vol.1. [*Online*]. Disponível: https://doi.org/10.1109/discex.2003.1194892 (citado nas págs. 2, 57, 58 e 78)

[10] C. Kemmer, "Correlação de alertas de intrusão utilizando pré-requisitos e consequências," 2016. [*Online*]. Disponível: http://www.uel.br/cce/dc/wp-content/uploads/TCC-CRISTIANO_CROTI_KEMMER-BCC-UEL-2015.pdf (citado nas págs. 2, 9, 13, 18, 19 e 21)

[11] K. Pei, Z. Gu, B. Saltaformaggio, S. Ma, F. Wang, Z. Zhang, L. Si, X. Zhang, e D. Xu, "Hercule: Attack story reconstruction via community discovery on correlated log graph," in *Proceedings of the 32Nd Annual Conference on Computer Security Applications*, ser. ACSAC '16. New York, NY, USA: ACM, 2016, pp. 583–595. [*Online*]. Disponível: http://doi.acm.org/10.1145/2991079.2991122 (citado na pág. 2)

[12] S. Haas e M. Fischer, "Gac: Graph-based alert correlation for the detection of distributed multi-step attacks," in *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, ser. SAC '18. New York, NY, USA: ACM, 2018, pp. 979–988. [*Online*]. Disponível: http://doi.acm.org/10.1145/3167132.3167239 (citado nas págs. 2, 31, 45, 47, 55 e 56)

[13] M. Siraj, M. Maarof, e S. Hashim, "Intelligent alert clustering model for network intrusion analysis," in *SOCO 2009*, 2009. [*Online*].

Disponível: https://www.researchgate.net/publication/50590287_Intelligent_Alert_Clustering_Model_for_Network_Intrusion_Analysis (citado nas págs. 2, 24, 28, 50, 54 e 78)

[14] M. Bhuyan, D. Bhattacharyya, e J. Kalita, *Network Traffic Anomaly Detection and Prevention: Concepts, Techniques, and Tools*, ser. Computer Communications and Networks. Springer International Publishing, 2017. [*Online*]. Disponível: http://link.springer.com/10.1007/978-3-319-65188-0 (citado nas págs. 2, 7, 19, 21, 22, 23, 57 e 78)

[15] N. Scarabeo, B. C. Fung, e R. H. Khokhar, "Mining known attack patterns from security-related events," *PeerJ Computer Science*, vol. 1, n. October, p. e25, 2015. [*Online*]. Disponível: https://peerj.com/articles/cs-25 (citado nas págs. 3 e 22)

[16] F. A. Bahareth e O. O. Bamasak, "Constructing Attack Scenario using Sequential Pattern Mining with Correlated Candidate Sequences," *The Research Bulletin of Jordan ACM,*, vol. 2, No.3, pp. 102–108, 2013. [*Online*]. Disponível: http://ijj.acm.org/volumes/volume2/issue3/ijjvol2no31.pdf (citado nas págs. 3 e 29)

[17] T. Shimeall e J. Spring, *Introduction to Information Security: A Strategic-Based Approach*. Syngress Publishing, 2013. (citado nas págs. 5, 7 e 10)

[18] W. Stalings, *Criptografia e segurança de redes : Princípios e práticas*, 6th ed. Pearson, 2014. (citado nas págs. 5 e 7)

[19] J. D. Howard e T. A. Longstaff, "A common language for computer security incidents," Relatório Técnico, Outubro 1998. [*Online*]. Disponível: https://doi.org/10.2172/751004 (citado nas págs. 6 e 7)

[20] P. Cichonski, T. Millar, T. Grance, e K. Scarfone, "Computer security incident handling guide : Recommendations of the national institute of standards and technology," Relatório Técnico, Agosto 2012. [*Online*]. Disponível: https://doi.org/10.6028/nist.sp.800-61r2 (citado nas págs. 6 e 7)

[21] CNCS, "Glossário," consultado em 2019/05/02. [*Online*]. Disponível: https://www.cncs.gov.pt/recursos/glossario/ (citado na pág. 6)

[22] "ISO/IEC 27000:2018," consultado em 2019/03/04. [*Online*]. Disponível: https://www.iso.org/obp/ui/#iso:std:iso-iec:27000:ed-5:v1:en (citado na pág. 6)

[23] R. H. James Graham, Ryan Olson, *Cyber Security Essentials*, 1st ed. CRC Press, 2010. (citado na pág. 6)

[24] E. Madeira, "Uma abordagem para a correlação de eventos de segurança baseada em técnica de aprendizado de máquina," Dissertação de mestrado, Universidade Estadual de Campinas, 2009. [*Online*]. Disponível: http://repositorio.unicamp.br/bitstream/REPOSIP/275828/1/Stroeh_Kleber_M.pdf (citado nas págs. 6, 12, 13, 14, 17 e 18)

[25] S. O. Al-Mamory e H. L. Zhang, "A survey on ids alerts processing techniques," in *Proceedings of the 6th WSEAS International Conference on Information Security and Privacy*, ser. ISP'07. Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS), 2007, pp. 69–78. [*Online*]. Disponível: http://dl.acm.org/citation.cfm?id=1981242.1981254 (citado nas págs. 7, 18, 20, 21 e 22)

[26] P. J. R. Leonard Kaufman, *Finding Groups in Data: An Introduction to Cluster Analysis*, 1st ed., ser. Wiley Series in Probability and Statistics. Wiley-Interscience, 2005. (citado nas págs. 8 e 49)

[27] F. Valeur, G. Vigna, C. Kruegel, e R. Kemmerer, "Comprehensive approach to intrusion detection alert correlation," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, n. 3, pp. 146–169, jul 2004. [*Online*]. Disponível: http://ieeexplore.ieee.org/document/1366134/ (citado nas págs. 8, 18, 20, 32 e 44)

[28] MIT Lincoln Laboratory, "2000 DARPA Intrusion Detection Scenario Specific Datasets," 2000, consultado em 2019/05/19. [*Online*]. Disponível: https://www.ll.mit.edu/r-d/datasets/2000-darpa-intrusion-detection-scenario-specific-datasets (citado na pág. 9)

[29] J. Deckard, *Buffer Overflow Attacks: Detect, Exploit, Prevent*, 1st ed. Syngress, 2005. (citado nas págs. 11 e 67)

[30] M. Lambort Batista, "Análise de Eventos de Segurança em Redes de Computadores Utilizando Detecção de Novidade," Dissertação de mestrado, Universidade Estadual Paulista, 2012. [*Online*]. Disponível: https://repositorio.unesp.br/bitstream/handle/11449/89342/batista_ml_me_sjrp.pdf (citado nas págs. 12, 13 e 14)

[31] H. Alaidaros, M. Mahmuddin, e A. Al mazari, "An overview of flow-based and packet-based intrusion detection performance in high-speed networks," 01 2011. [*Online*]. Disponível: https://www.researchgate.net/publication/303498189_An_Overview_of_Flow-based_and_Packet-based_Intrusion_Detection_Performance_in_High-speed_Networks (citado nas págs. 12 e 14)

[32] R. Smith, N. Japkowicz, M. Dondo, e P. Mason, "Using Unsupervised Learning for Network Alert Correlation," in *Advances in Artificial Intelligence*, S. Bergler, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 308–319. [*Online*]. Disponível: https://doi.org/10.1007/978-3-540-68825-9_29 (citado nas págs. 12, 22, 27, 28, 50, 78 e 79)

[33] S. Brito, *Serviços de Redes em Servidores Linux*. Novatec, 2017. (citado na pág. 12)

[34] V. O. U. Ferreira, "Classificação de anomalias e redução de falsos positivos em sistemas de detecção de intrusão baseados em rede utilizando métodos de agrupamento," Dissertação de mestrado, Universidade Estadual Paulista (UNESP), apr 2016. [*Online*]. Disponível: https://repositorio.unesp.br/handle/11449/138755 (citado nas págs. 13, 14 e 18)

[35] S. H. Ahmadinejad, S. Jalili, e M. Abadi, "A hybrid model for correlating alerts of known and unknown attack scenarios and updating attack graphs," *Computer Networks*, vol. 55, n. 9, pp. 2221–2240, 2011. [*Online*]. Disponível: http://www.sciencedirect.com/science/article/pii/S1389128611000983 (citado nas págs. 13, 19 e 30)

[36] Snort, "Snort - Network Intrusion Detection & Prevention System," consultado em 2019/06/27. [*Online*]. Disponível: https://www.snort.org/ (citado na pág. 13)

[37] P. Laskov, P. Düssel, C. Schäfer, e K. Rieck, "Learning Intrusion Detection: Supervised or Unsupervised?" in *Image Analysis and Processing – ICIAP 2005*, F. Roli e S. Vitulano, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 50–57. [*Online*]. Disponível: https://doi.org/10.1007/11553595_6 (citado na pág. 14)

[38] H. Hindy, D. Brosset, E. Bayne, A. Seeam, C. Tachtatzis, R. C. Atkinson, e X. J. A. Bellekens, "A taxonomy and survey of intrusion detection system

design techniques, network threats and datasets," *CoRR*, vol. 1, No.1, 2018. [*Online*]. Disponível: http://arxiv.org/abs/1806.03517 (citado na pág. 14)

[39] D. Palma, "Sistema Distribuído de Deteção de Intrusões," Dissertação de mestrado, Instituto Politécnico de Beja, 2014. (citado nas págs. 14 e 77)

[40] M. Md Siraj, "Classifying security alerts from multiple sensors based on hybrid approach," *The International Conference on Informatics and Applications (ICIA2012)*, pp. 174–181, 01 2012. [*Online*]. Disponível: https://www.researchgate.net/publication/230771723_CLASSIFYING_SECURITY_ALERTS_FROM_MULTIPLE_SENSORS_BASED_ON_HYBRID_APPROACH (citado nas págs. 15, 18, 21 e 79)

[41] M. Goodrich e R. Tamassia, *Introdução à Segurança de Computadores*. Bookman, 2013. (citado na pág. 17)

[42] A. Ribeiro, "Correlação e visualização de alertas de segurança em redes de computadores," Dissertação de mestrado, Universidade Estadual Paulista, 2015. [*Online*]. Disponível: https://repositorio.unesp.br/bitstream/handle/11449/127861/000844040.pdf (citado nas págs. 17, 18, 19 e 25)

[43] IETF, "The Intrusion Detection Message Exchange Format (IDMEF)," 2007, consultado em 2019/05/07. [*Online*]. Disponível: https://tools.ietf.org/html/rfc4765 (citado na pág. 17)

[44] R. Sadoddin e A. Ghorbani, "Alert correlation survey: framework and techniques," in *Proceedings of the 2006 International Conference on Privacy, Security and Trust Bridge the Gap Between PST Technologies and Business Services - PST '06*. New York, New York, USA: ACM Press, 2006, p. 1. [*Online*]. Disponível: http://portal.acm.org/citation.cfm?doid=1501434.1501479 (citado nas págs. 18, 20, 21, 22, 24, 50 e 78)

[45] L. Silva, "Aplicando a Frequência de Episódios na Predição de Anomalias," 2009. [*Online*]. Disponível: https://www.cin.ufpe.br/~tg/2009-2/lhvs.pdf (citado nas págs. 19, 20 e 21)

[46] S. A. Mirheidari, S. Arshad, e R. Jalili, "Alert Correlation Algorithms: A Survey and Taxonomy," in *Cyberspace Safety and Security*, G. Wang, I. Ray, D. Feng, e M. Rajarajan, Eds. Cham: Springer International Publishing, 2013, pp. 183–197. [*Online*]. Disponível: https://doi.org/10.1007/978-3-319-03584-0_14 (citado nas págs. 19, 20, 21, 22 e 34)

[47] A. Hätälä, C. Särs, R. Addams-Moring, e T. Virtanen, "Event Data Exchange and Intrusion Alert Correlation in Heterogeneous Networks," in *Proceedings of the 8th Colloquium for Information Systems Security Education (CISSE)*, 2004. [*Online*]. Disponível: https://cisse.info/resources/archives/category/2-papers?download=13:s4p01-2004 (citado nas págs. 20, 21, 22, 27 e 50)

[48] Z. Bin e A. Ghorbani, "Alert correlation for extracting attack strategies," *International Journal of Network Security*, vol. 3(3), 01 2006. [*Online*]. Disponível: https://www.researchgate.net/publication/45728853_Alert_Correlation_for_Extracting_Attack_Strategies (citado nas págs. 20, 27, 45 e 78)

[49] A. Valdes e K. Skinner, "Probabilistic Alert Correlation," in *Recent Advances in Intrusion Detection*, W. Lee, L. Mé, e A. Wespi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 54–68. [*Online*]. Disponível: https://doi.org/10.1007/3-540-45474-8_4 (citado nas págs. 20, 26, 27 e 50)

[50] M. Saikia, N. Hoque, e D. K. Bhattacharyya, "MaNaDAC: An Effective Alert Correlation Method," in *Recent Developments in Machine Learning and Data Analytics*, J. Kalita, V. E. Balas, S. Borah, e R. Pradhan, Eds. Singapore: Springer Singapore, 2019, pp. 249–260. [*Online*]. Disponível: https://doi.org/10.1007/978-981-13-1280-9_24 (citado na pág. 20)

[51] A. B. Palekar e S. S. Dhande, "Study of Alert Correlation Technique," *International Journal of Advanced Research in Computer and Communication Engineering ISO*, vol. 3297, 2017. [*Online*]. Disponível: https://doi.org/10.17148/ijarcce.2017.63150 (citado na pág. 21)

[52] F. Cuppens e R. Ortalo, "LAMBDA: A Language to Model a Database for Detection of Attacks." Springer, Berlin, Heidelberg, 2000, pp. 197–216. [*Online*]. Disponível: https://link.springer.com/chapter/10.1007%2F3-540-39945-3_13 (citado na pág. 21)

[53] S. J. Templeton e K. Levitt, "A requires/provides model for computer attacks," in *Proceedings of the 2000 Workshop on New Security Paradigms*, ser. NSPW '00. New York, NY, USA: ACM, 2000, pp. 31–38. [*Online*]. Disponível: http://doi.acm.org/10.1145/366173.366187 (citado na pág. 21)

[54] F. Cuppens e A. Miege, "Alert correlation in a cooperative intrusion detection framework," in *Proceedings 2002 IEEE Symposium on Security*

and Privacy. IEEE Comput. Soc, pp. 202–215. [*Online*]. Disponível: http://ieeexplore.ieee.org/document/1004372/ (citado na pág. 21)

[55] P. Ning, Y. Cui, e D. S. Reeves, "Constructing attack scenarios through correlation of intrusion alerts," in *Proceedings of the 9th ACM conference on Computer and communications security - CCS '02*. New York, New York, USA: ACM Press, 2002, p. 245. [*Online*]. Disponível: http://portal.acm.org/citation.cfm?doid=586110.586144 (citado na pág. 21)

[56] S. S. Sekharan e K. Kandasamy, "Profiling SIEM tools and correlation engines for security analytics," in *2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*. IEEE, mar 2017, pp. 717–721. [*Online*]. Disponível: http://ieeexplore.ieee.org/document/8299855/ (citado na pág. 21)

[57] S. T. Eckmann, G. Vigna, e R. A. Kemmerer, "Statl: An attack language for state-based intrusion detection," *J. Comput. Secur.*, vol. 10, n. 1-2, pp. 71–103, Julho 2002. [*Online*]. Disponível: http://dl.acm.org/citation.cfm?id=597917.597921 (citado na pág. 21)

[58] E. Totel, B. Vivinis, e L. Mé, "A Language Driven Intrusion Detection System for Event and Alert Correlation," in *Security and Protection in Information Processing Systems*. Boston, MA: Springer US, 2004, pp. 209–224. [*Online*]. Disponível: https://link.springer.com/chapter/10.1007%2F1-4020-8143-X_14 (citado na pág. 21)

[59] C. Geib e R. Goldman, "Plan recognition in intrusion detection systems," in *Proceedings DARPA Information Survivability Conference and Exposition II. DISCEX'01*, vol. 1. IEEE Comput. Soc, pp. 46–55. [*Online*]. Disponível: http://ieeexplore.ieee.org/document/932191/ (citado na pág. 22)

[60] J. Metz, "Interpretação de clusters gerados por algoritmos de clustering hierárquico," Dissertação de mestrado, Universidade de São Paulo, São Carlos, aug 2006. [*Online*]. Disponível: http://www.teses.usp.br/teses/disponiveis/55/55134/tde-14092006-090701/ (citado na pág. 23)

[61] L. Wang, A. Ghorbani, e L. Yao, "Automatic multi-step attack pattern discovering," *International Journal of Network Security*, vol. 10, 01 2010. [*Online*]. Disponível: https://www.researchgate.net/publication/46137457_Automatic_Multi-step_Attack_Pattern_Discovering (citado na pág. 29)

[62] D. Margarido, "Waldo, the Virtual & Intelligent Cyber Analyst," Dissertação de mestrado, Instituto Superior de Engenharia de Coimbra, 2017. [*Online*]. Disponível: https://comum.rcaap.pt/bitstream/10400.26/25336/1/Daniel-Ribeiro-Margarido.pdf (citado na pág. 30)

[63] H. Ren, N. Stakhanova, e A. A. Ghorbani, "An online adaptive approach to alert correlation," in *Detection of Intrusions and Malware, and Vulnerability Assessment*, C. Kreibich e M. Jahnke, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 153–172. [*Online*]. Disponível: https://doi.org/10.1007/978-3-642-14215-4_9 (citado na pág. 30)

[64] J. Yu, Y. V. Ramana Reddy, C. Srinivas Kankanahalli, S. Reddy, V. Jagannathan, e E. Gunel, "TRINETR: An Intrusion Detection Alert Management and Analysis System," Tese de doutoramento, West Virginia University, 2004. [*Online*]. Disponível: https://pdfs.semanticscholar.org/1d7e/a142ec97fc551476906eb4abf0737f4a77bb.pdf (citado na pág. 30)

[65] "Logstash: Collect, Parse, Transform Logs — Elastic," consultado em 2019/05/03. [*Online*]. Disponível: https://www.elastic.co/pt/products/logstash (citado na pág. 42)

[66] Elastic, "Open Source Search & Analytics · Elasticsearch — Elastic," consultado em 2019/05/31. [*Online*]. Disponível: https://www.elastic.co/pt/ (citado na pág. 42)

[67] CloudAMQP, "Part 1: RabbitMQ for beginners - What is RabbitMQ? - CloudAMQP," 2015, consultado em 2019/05/04. [*Online*]. Disponível: https://www.cloudamqp.com/blog/2015-05-18-part1-rabbitmq-for-beginners-what-is-rabbitmq.html (citado na pág. 43)

[68] "idstools: Snort and Suricata Rule and Event Utilities in Python," consultado em 2019/05/04. [*Online*]. Disponível: https://github.com/jasonish/py-idstools (citado na pág. 43)

[69] B. Pattabiraman, M. M. A. Patwary, A. H. Gebremedhin, W. Liao, e A. N. Choudhary, "Fast algorithms for the maximum clique problem on massive graphs with applications to overlapping community detection," *CoRR*, vol. abs/1411.7460, 2014. [*Online*]. Disponível: http://arxiv.org/abs/1411.7460 (citado na pág. 47)

[70] T. A. Alhaj, M. M. Siraj, A. Zainal, H. T. Elshoush, e F. Elhaj, "Feature selection using information gain for improved structural-based alert correlation," *PLOS ONE*, vol. 11, n. 11, pp. 1–18, 11 2016. [*Online*]. Disponível: https://doi.org/10.1371/journal.pone.0166017 (citado nas págs. 48, 50 e 51)

[71] L. Krippahl, "Hierarchical clustering," 2018. [*Online*]. Disponível: http://aa.ssdi.di.fct.unl.pt/files/AA-19_notes.pdf (citado na pág. 52)

[72] J. M. Żytkow e J. Rauch, Eds., *Principles of Data Mining and Knowledge Discovery*. Springer Berlin Heidelberg, 1999. [*Online*]. Disponível: https://doi.org/10.1007/b72280 (citado na pág. 53)

[73] G. W. Milligan e M. C. Cooper, "An examination of procedures for determining the number of clusters in a data set," *Psychometrika*, vol. 50, n. 2, pp. 159–179, Jun 1985. [*Online*]. Disponível: https://doi.org/10.1007/BF02294245 (citado na pág. 54)

[74] NbClust, "CRAN - Package NbClust," consultado em 2019/07/26. [*Online*]. Disponível: https://cran.r-project.org/web/packages/NbClust/index.html (citado na pág. 54)

[75] M. Charrad, N. Ghazzali, V. Boiteau, e A. Niknafs, "Determining the number of clusters using nbclust package," 03 2014. [*Online*]. Disponível: https://www.researchgate.net/publication/275463140_Determining_the_number_of_clusters_using_NbClust_package (citado na pág. 54)

[76] C.-H. Wang, , e Y.-C. Chiou, "Alert correlation system with automatic extraction of attack strategies by using dynamic feature weights," *International Journal of Computer and Communication Engineering*, vol. 5, n. 1, pp. 1–10, 2016. [*Online*]. Disponível: https://doi.org/10.17706/ijcce.2016.5.1.1-10 (citado na pág. 57)

[77] MITRE, "Groups - MITRE ATT&CK™," consultado em 2019/06/07. [*Online*]. Disponível: https://attack.mitre.org/groups/ (citado na pág. 60)

[78] SQLite, "SQLite Home Page," consultado em 2019/06/08. [*Online*]. Disponível: https://www.sqlite.org/index.html (citado na pág. 62)

[79] Flask, "Welcome — Flask (A Python Microframework)," consultado em 2019/06/02. [*Online*]. Disponível: http://flask.pocoo.org/ (citado na pág. 62)

[80] "Pure Python RabbitMQ/AMQP 0-9-1 client library," consultado em 2019/05/04. [*Online*]. Disponível: https://github.com/pika/pika (citado na pág. 63)

[81] NetworkX, "NetworkX — NetworkX," consultado em 2019/06/07. [*Online*]. Disponível: https://networkx.github.io/ (citado na pág. 63)

[82] PRELUDE SIEM, "IDMEF Library," consultado em 2019/05/07. [*Online*]. Disponível: https://github.com/jasonish/py-idstools (citado na pág. 63)

[83] Pandas, "Python Data Analysis Library — pandas: Python Data Analysis Library," consultado em 2019/06/07. [*Online*]. Disponível: https://pandas.pydata.org/ (citado na pág. 63)

[84] Rpy2, "rpy2 - R in Python," consultado em 2019/06/07. [*Online*]. Disponível: https://rpy2.bitbucket.io/ (citado na pág. 63)

[85] ReportLab, "ReportLab - Content to PDF Solutions," consultado em 2019/06/27. [*Online*]. Disponível: https://www.reportlab.com/ (citado na pág. 63)

[86] PyCharm, "PyCharm: the Python IDE for Professional Developers by JetBrains," consultado em 2019/06/29. [*Online*]. Disponível: https://www.jetbrains.com/pycharm/ (citado na pág. 64)

[87] Weka, "Weka 3 - Data Mining with Open Source Machine Learning Software in Java," consultado em 2019/06/29. [*Online*]. Disponível: https://www.cs.waikato.ac.nz/ml/weka/ (citado na pág. 64)

[88] Inundator, "inundator: because pattern matching is full of fail." consultado em 2019/06/02. [*Online*]. Disponível: http://inundator.sourceforge.net/ (citado na pág. 64)

[89] VirtualBox, "Oracle VM VirtualBox," consultado em 2019/08/23. [*Online*]. Disponível: https://www.virtualbox.org/ (citado na pág. 65)

[90] Wireshark, "Wireshark · Go Deep." consultado em 2019/08/23. [*Online*]. Disponível: https://www.wireshark.org/ (citado na pág. 66)

[91] Scapy, "Packet crafting for Python2 and Python3," consultado em 2019/06/30. [*Online*]. Disponível: https://scapy.net/ (citado na pág. 69)

[92] Famatech, "Advanced Port Scanner," consultado em 2019/06/06. [*Online*]. Disponível: http://www.advanced-port-scanner.com/br/ (citado na pág. 74)

[93] Steve, "2019 Official Annual Cybercrime Report," Relatório Técnico. [*Online*]. Disponível: https://www.herjavecgroup.com/wp-content/uploads/2018/12/CV-HG-2019-Official-Annual-Cybercrime-Report.pdf (citado na pág. 77)

[94] J. N. D. Gupta e S. K. Sharma, *Handbook of Research on Information Security and Assurance*. Hershey, PA: Information Science Reference - Imprint of: IGI Publishing, 2008. (citado na pág. 77)

[95] DRE, "Resolução do Conselho de Ministros 92/2019, 2019-06-05 - DRE," consultado em 2019/06/07. [*Online*]. Disponível: https://data.dre.pt/eli/resolconsmin/92/2019/06/05/p/dre (citado na pág. 78)

Apêndices

Apêndice I

Imagens relativas à aplicação web

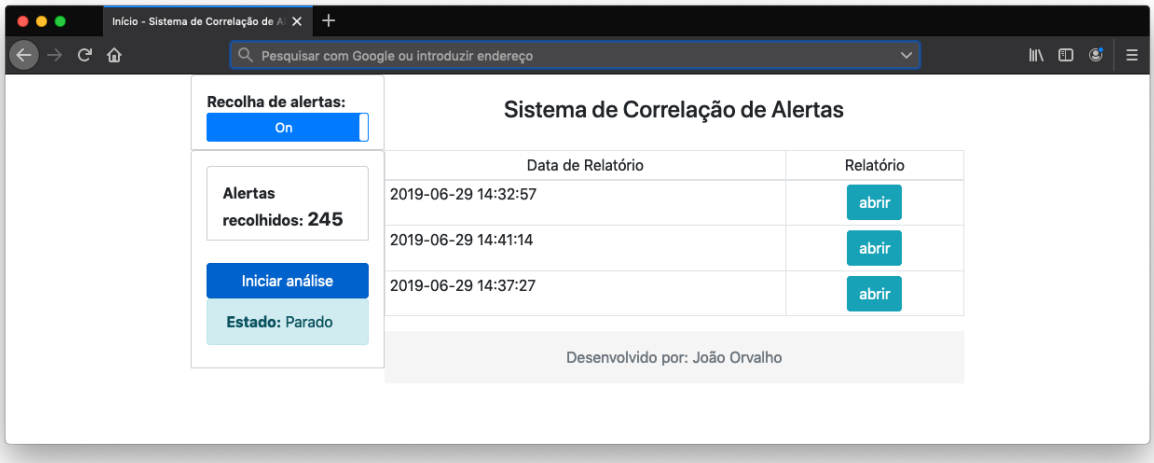

Figura I.1: Página principal da aplicação web

Instituto Politécnico de Beja — Escola Superior de Tecnologia e Gestão

Relatório de Análise

Ataques completos identificados: 1

Cenário 1

Grau de semelhança com cenário conhecido com ID= 1 : 8.33 %
Grau de semelhança com cenário conhecido com ID= 2 : 88.89 %

Alertas Únicos:

date	src_addr	src_port	dst_addr	dst_port	protocol	priority	pkt_len	class	rule_number
2019-06-23T16:07:53	192.168.0.3	52279	192.168.0.1	161	TCP	medium	44	Attempted Information Leak	1:1418:18
2019-06-23T16:07:53	192.168.0.3	52279	192.168.0.1	705	TCP	medium	44	Attempted Information Leak	1:1421:18
2019-06-23T16:07:54	192.168.0.3	52279	192.168.0.2	161	TCP	medium	44	Attempted Information Leak	1:1418:18
2019-06-23T16:07:54	192.168.0.3	52279	192.168.0.2	705	TCP	medium	44	Attempted Information Leak	1:1421:18
-	-	-	-	-	-	-	-	-	-
2019-06-23T16:09:46	192.168.0.3	8	192.168.0.1	9	ICMP	low	148	Misc activity	1:365:11
2019-06-23T16:09:46	192.168.0.3	8	192.168.0.2	9	ICMP	low	148	Misc activity	1:365:11

Figura I.2: Excerto de relatório exemplo com análise de correlação de alertas