Universidade Estadual de Londrina

CLÁUDIO TOSHIO KAWAKANI

GERAÇÃO ONLINE DE HIPERALERTAS COM BASE NO HISTÓRICO DE ESTRATÉGIAS DE ATAQUE

Londrina
2017

CLÁUDIO TOSHIO KAWAKANI

GERAÇÃO ONLINE DE HIPERALERTAS COM BASE NO HISTÓRICO DE ESTRATÉGIAS DE ATAQUE

Dissertação apresentada ao Programa de Mestrado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Bruno Bogaz Zarpelão.

Londrina
2017

CLÁUDIO TOSHIO KAWAKANI

GERAÇÃO ONLINE DE HIPERALERTAS COM BASE NO HISTÓRICO DE ESTRATÉGIAS DE ATAQUE

Dissertação apresentada ao Programa de Mestrado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Mestre em Ciência da Computação.

BANCA EXAMINADORA

Orientador: Prof. Dr. Bruno Bogaz Zarpelão.
Universidade Estadual de Londrina- UEL

Profa. Dra. Luciana Andréia Fondazzi Martimiano
UniversidadeEstadualdeMaringá - UEM

Prof. Dr. Sylvio Barbon Junior
UniversidadeEstadualdeLondrina - UEL

Prof. Dr. Mario Lemes Proença Junior
UniversidadeEstadualdeLondrina - UEL

Londrina, 27 de março de 2017.

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema de Bibliotecas da UEL

Kawakani, Cláudio Toshio.
Geração online de hiperalertas com base no histórico de estratégias de ataque / Cláudio Toshio Kawakani. - Londrina, 2017.
83 f. : il.

Orientador: Bruno Bogaz Zarpelão.
Dissertação (Mestrado em Ciência da Computação) - Universidade Estadual de Londrina, Centro de Ciências Exatas, Programa de Pós-Graduação em Ciência da Computação, 2017.
Inclui bibliografia.

1. Correlação de alertas - Tese. 2. Mineração de dados - Tese. 3. Detecção de intrusão - Tese. 4. Segurança de redes - Tese. I. Zarpelão, Bruno Bogaz. II. Universidade Estadual de Londrina. Centro de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação. III. Título.

AGRADECIMENTOS

A Deus.

Aos meus pais Oscar e Luci e a toda minha família pelo incentivo e apoio imensurável.

A todos os meus amigos, pelos momentos de diversão e alegria.

Ao meu orientador Bruno, por todas as oportunidades, pelo conhecimento, pela experiência e pelos conselhos, os quais vão além dos estudos.

Aos professores Sylvio e Miani, que sempre estiveram disponíveis para ajudar com boas ideias e sugestões.

Aos professores participantes da banca examinadora, pelas sugestões, correções e contribuições.

A todos os professores que tive até hoje, por todas as lições e conselhos.

A UEL, lugar onde estudei durante os últimos seis anos, desde o início da graduação até o final mestrado.

A Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (Capes), pelo suporte financeiro que possibilitou a realização deste trabalho.

A Universidade de Maryland, por fornecer a base de dados utilizada nos experimentos deste trabalho.

A todos que, direta ou indiretamente, participaram da minha formação.

Muito obrigado!

KAWAKANI, C. T.. **Geração online de hiperalertas com base no histórico de estratégias de ataque**. 83 p. Dissertação de Mestrado (Mestrado em Ciência da Computação) – Universidade Estadual de Londrina, Londrina–PR, 2017.

RESUMO

Para auxiliar na segurança da informação, as organizações implantam Sistemas de Detecção de Intrusão (*Intrusion Detection System* - IDS), os quais monitoram os sistemas de informação e as redes e geram alertas quando atividades de comportamento suspeito são detectadas. No entanto, esses alertas são gerados em grandes quantidades e apresentam informações muito elementares quando estudados individualmente. Logo, para entender o comportamento dos ataques, o estudo de um conjunto de alertas relacionados é mais significativo do que o estudo de alertas individuais. Portanto, a análise de alertas de IDS é necessária, porém também é uma tarefa desafiadora. Neste trabalho, é proposta uma nova abordagem que utiliza clusterização hierárquica para auxiliar a análise de alertas de intrusão. A abordagem é constituída por duas fases. Na primeira fase, denominada correlação *offline*, um histórico de alertas é correlacionado para identificar quais padrões de estratégias de ataque são normalmente utilizados contra a rede monitorada. Na segunda fase, denominada correlação *online*, conforme o IDS gera novos alertas, eles são agrupados em cenários de ataque de acordo com seus atributos de endereços IP e *timestamp*. As informações de cada cenário são extraídas e representadas na forma de hiperalertas. Cada hiperalerta é associado à uma das estratégias descobertas na primeira fase, permitindo que o analista de segurança responda a esses hiperalertas de acordo com os padrões de estratégia de ataque identificados anteriormente. Os experimentos foram realizados com uma base de dados real fornecida pela Universidade de Maryland. Os resultados mostram que a abordagem proposta foi capaz de identificar os padrões de estratégia de ataque em conjuntos de milhares de alertas. Além disso, a agregação dos alertas em hiperalertas auxilia o trabalho do analista de segurança, o qual passa a analisar cenários de ataque ao invés de alertas individuais.

Palavras-chave: Correlação de Alertas. Mineração de Dados. Detecção de Intrusão. Segurança de Redes.

KAWAKANI, C. T. **Online hyper-alert generation based on attack strategy historic**. 83 p. Master's Thesis (Master in Science in Computer Science) – State University of Londrina, Londrina–PR, 2017.

ABSTRACT

To support information security, organizations deploy Intrusion Detection Systems (IDS) that monitor and generate alerts for every suspicious behavior. However, the huge amount of alerts that an IDS triggers and their elementary information make the alerts analysis a challenging task. We propose a new approach based on hierarchical clustering that supports intrusion alert analysis in two main steps. The first step, referred to as offline correlation, correlates historical alerts to identify the most typical strategies attackers have used. In the second step, referred to as online correlation, as the IDS generate new alerts, they are separated into attack scenarios according to their IP addresses and timestamp attributes. The information of each scenario is extracted and represented in the form of hyper-alerts. Moreover, each hyper-alert is associated to one of the strategies discovered in the first step. This association allows the security analyst to respond to these hyper-alerts according to the attack strategy patterns previously identified. The experiments were performed using a real-life data set provided by the University of Maryland. The results show the proposed approach is able to identify attack strategy patterns from thousand of alerts. Furthermore, the aggregation of alerts into hyper-alerts assist the security analyst, which may replace the study of isolated alerts by the study of attack scenarios.

Keywords: Alerts Correlation. Data mining. Intrusion Detection. Network Security.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A informação é um recurso cada vez mais valioso e importante para as organizações. Um incidente de segurança da informação em uma organização pode causar perda de produtividade, recursos e reputação. Por exemplo, danos imensuráveis podem ser gerados por falhas em sistemas da área médica, pois são capazes de colocar a vida de pacientes em risco. Portanto, o gerenciamento de incidentes é uma tarefa importante para a segurança da informação, o qual define processos para detecção, análise e resposta para os incidentes [1, 2, 3].

Incidentes de segurança podem ser gerados por meio de intrusões em um sistema ou rede de computadores. Os Sistemas de Detecção de Intrusão (*Intrusion Detection Systems* - IDSs) são dispositivos que monitoram os sistemas e as redes à procura de sinais de violações de segurança. Quando um IDS detecta uma possível violação, um alerta é gerado e armazenado em uma base de dados de alertas. Uma melhor compreensão dos ataques realizados contra a rede ou sistema pode ser adquirida por meio da análise dos alertas de intrusão. Essa análise auxilia para que a resposta e tratamento dos ataques seja feita de forma mais eficiente [4].

O tempo entre a ocorrência de um incidente e a sua resposta e tratamento deve ser reduzido, diminuindo os danos e o custo para recuperação do sistema. Tendo como exemplo a base de alertas de intrusão de 2012 da Universidade de Maryland, a qual apresenta mais de 80 milhões de alertas, realizar apenas a análise manual desses dados não é recomendado, pois é considerada uma tarefa demorada e sujeita a erros. Técnicas de mineração de dados podem ser utilizadas para reduzir o tempo entre a ocorrência do evento e a resposta, pois são capazes de analisar um grande número de alertas de intrusão de forma automática, a fim de encontrar padrões de comportamento dos atacantes e resumi-los em estruturas mais adequadas [3, 4].

Neste trabalho, foi realizado um estudo de artigos sobre análise de alertas de intrusão publicados entre 2001 e 2015. Por meio deste estudo, foram identificados dois principais problemas enfrentados na área de análise de alertas de intrusão. O primeiro problema é a grande quantidade de alertas gerados pelo IDS. Para solucionar este problema, abordagens para reduzir a quantidade de alertas utilizando técnicas de filtragem e agregação foram propostas [4, 5]. O segundo problema está relacionado à informação fornecida pelos alertas. Os alertas apresentam informações muito elementares quando estudados individualmente. Para entender o comportamento dos ataques, o estudo de um conjunto de alertas correlacionados é mais favorável do que o estudo de alertas individuais. Portanto, para solucionar o segundo problema, abordagens para correlacionar os alertas a fim de extrair informações mais significativas também foram propostas [6, 7]. No entanto,

nenhum dos trabalhos revisados faz uma análise aprofundada das informações que podem ser extraídas de cenários de ataque. Neste trabalho, as informações dos cenários de ataque são extraídas e representadas por meio de hiperalertas. Um hiperalerta é uma estrutura de informação que descreve um cenário de ataque por meio de atributos que podem ser úteis para o analista de segurança. Além disso, a maioria dos trabalhos utiliza uma base de dados experimental para a validação dos métodos propostos. A baixa capacidade de uma base de dados experimental refletir situações reais tem sido amplamente criticada na literatura [8, 9, 10, 11].

Considerando os dois principais problemas enfrentados na área de análise de alertas de intrusão, uma nova abordagem foi proposta, a qual é formada por dois componentes, o correlacionador *offline* e o correlacionador *online*. O correlacionador *offline* tem o objetivo de analisar um histórico de alertas de IDS para descobrir quais padrões de estratégias de ataques são tipicamente utilizadas contra o sistema. Para isso, o correlacionador *offline* agrega os alertas históricos em cenários de ataque com o método de separação por componentes conexos, extrai os grafos de estratégia de ataque de cada cenário e utiliza clusterização hierárquica para agrupar os grafos que possuem estratégias de ataques similares. As características de ataque de cada cluster gerado são identificadas, especificando um padrão de estratégia de ataque.

O correlacionador *online*, por sua vez, organiza os alertas em cenários de ataque conforme são gerados pelo IDS. Assim como no correlacionador *offline*, os cenários de ataque são identificados utilizando o método de separação por componentes conexos. Para cada cenário de ataque encontrado, é gerado um hiperalerta. Além disso, cada hiperalerta é associado a um dos padrões de estratégia de ataque identificados anteriormente na fase *offline*. Analisando as informações do padrão de estratégia de ataque associado, é possível entender as características da intrusão que está sendo realizada no momento presente apenas com os primeiros alertas gerados para esse ataque. Dessa forma, para auxiliar o analista de segurança, pretende-se resumir uma grande quantidade de alertas com informações elementares em uma menor quantidade de hiperalertas com informações de maior significado. Os experimentos foram realizados utilizando uma base de dados real gerada por um IDS implantado em uma rede de aproximadamente 40.000 computadores na Universidade de Maryland[1].

As principais contribuições deste trabalho são: (a) uma taxonomia que organiza as técnicas utilizadas para análise de alertas de intrusão, resultante de um levantamento de artigos publicados sobre o tema entre 2001 e 2015; (b) um novo método de correlação de alertas formado pelos correlacionadores *offline* e *online*. O correlacionador *offline* organiza alertas históricos para identificar os padrões de estratégias de ataques tipicamente utili-

[1] O IDS utilizado é uma solução desenvolvida pela *Hewlett-Packard Company*. As características desta solução não serão detalhadas por questões de confidencialidade.

zados pelos atacantes. Por sua vez, o correlacionador *online* forma hiperalertas associados a um padrão de estratégia de ataque identificado na correlação *offline*.

O restante desta dissertação está organizado da seguinte forma: a Seção 2 apresenta os conceitos básicos sobre IDSs, uma taxonomia sobre análise de alertas de IDS e um breve resumo de cada trabalho estudado. A Seção 3 explica os correlacionadores *offline* e *online*. A Seção 4 descreve os experimentos e analisa os resultados. Por fim, a Seção 5 conclui este trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, é realizada uma introdução aos ataques multiestágio e aos Sistemas de Detecção de Intrusão, os quais são responsáveis por detectar intrusões e gerar alertas com informações elementares que descrevem as tentativas de intrusão. Em seguida, são apresentadas as técnicas utilizadas para análise de alertas de intrusão em trabalhos publicados do ano de 2001 até o ano de 2015. Essas técnicas foram organizadas em uma taxonomia.

2.1 Ataques Multiestágio

Uma intrusão é um evento (ou uma combinação de múltiplos eventos) no qual um invasor tenta ganhar acesso a um sistema ou interromper o seu funcionamento normal [12]. Portanto, as intrusões de segurança ameaçam a integridade, confidencialidade e disponibilidade de um sistema. Garantir a integridade significa assegurar que os dados não sofram modificações não autorizadas. Garantir a confidencialidade, por sua vez, significa assegurar que os dados não estarão disponíveis para usuários não autorizados. Já a disponibilidade indica que os dados sempre estarão disponíveis para usuários autorizados.

Existem três principais classes de ataque contra a rede que são consideradas intrusões: ataques de reconhecimento, acesso e negação de serviço (DoS) [13, 14]. Os ataques da classe de reconhecimento são utilizados para coletar informações sobre uma rede ou *host* alvo. As informações coletadas são, por exemplo, endereços IP, portas e aplicações que estão ativas na rede. Dessa forma, o atacante consegue explorar a vulnerabilidade de alguma aplicação de um determinado *host*. Esta classe de ataque é geralmente utilizada para reunir informações necessárias para realizar ataques da classe de acesso ou ataques de negação de serviço. Os ataques da classe de acesso ocorrem por meio da exploração das vulnerabilidades de aplicações e sistemas operacionais. São responsáveis por acessar informações confidenciais e utilizar um *host* alvo de forma ilegítima. *Password attack*, *buffer overflow* e *malwares* são exemplos de intrusões da classe de acesso. Já os ataques de negação de serviço buscam impedir o uso de um serviço ou sistema. A negação de um serviço pode acontecer por vários motivos, como por exemplo, por causa de configurações indevidas, arquivos corrompidos, ou sobrecarga de requisições [13, 14].

Para atingir um objetivo final, um mesmo atacante, controlando um ou mais *hosts*, realiza uma série de passos de intrusão contra um ou mais alvos. Esta sequência de intrusões é conhecida como ataque multiestágio. Um atacante ou ferramenta de ataque apresentam certas características próprias ao realizar um ataque multiestágio. Por exemplo, um comportamento comum é sempre realizar ataques de reconhecimento a fim de en-

contrar *hosts* ativos com aplicações vulneráveis para, depois, realizar ataques de negação de serviço ou de acesso [15, 13]. Caso certa característica de ataque se repita frequentemente, ela torna-se um padrão de ataque. Por meio dos padrões de ataque, é possível, por exemplo, identificar ferramentas de ataques automáticas sendo utilizadas contra a rede [16].

Portanto, um ataque multiestágio pode envolver vários *hosts* e vários passos de intrusão. Assim, para retratar um ataque multiestágio, este trabalho utiliza a definição de cenário de ataque, o qual representa todos os *hosts* e passos de intrusão envolvidos em um mesmo ataque.

2.2 Sistemas de Detecção de Intrusão

Alguns mecanismos de segurança são utilizados para proteger as redes e *hosts* contra intrusões, como a autenticação de usuário, o controle de acesso, os *firewalls* e o Sistema de Detecção de Intrusão (*Intrusion Detection System* - IDS). Um IDS é um software (ou hardware) implantado em *hosts* ou redes para detectar intrusões. Um IDS baseado em *host* monitora objetos do sistema, processos e regiões de memória de um único *host*. Já um IDS baseado em rede é implantado em pontos estratégicos da rede para monitorar o tráfego de pacotes [17]. Quando o IDS detecta uma atividade suspeita, um alerta com informações desta atividade é armazenado em uma base de dados de alertas. Posteriormente, um analista de segurança pode utilizar essas informações para aprimorar a segurança do sistema [18, 19, 12, 20].

No entanto, uma grande quantidade de alertas tende a ser gerada quando um IDS é implantado em uma rede de médio ou grande porte. Além disso, as informações contidas em um alerta de IDS isolado apresentam apenas informações elementares, como endereço IP de origem, endereço IP de destino, *timestamp*, prioridade e tipo da intrusão. A Figura 1 mostra um exemplo de alerta gerado pelo IDS Snort[1].

```
12/04-16:20:29.371181  [**] [1:469:4] ICMP PING NMAP [**] [Classification: Attempted Information Leak]
[Priority: 2] {ICMP} 10.100.1.1 -> 10.100.102.13
```

Figura 1 – Exemplo de alerta gerado pelo Snort.

Por meio da análise deste alerta, é possível identificar algumas informações básicas, como:

1. $12/04 - 16:20:29.371181$: data e hora que o alerta foi gerado.

[1] https://www.snort.org/

2. [1:469:4] *ICMP Ping Nmap*: identificador da regra por meio da qual foi detectada a intrusão e o tipo da intrusão.

3. [*Classification: Attempted Information Leak*]: classificação mais generalizada da intrusão.

4. [*Priority:* 2]: indica a prioridade do alerta. A prioridade do alerta assume desde o valor 1 (alta) até 4 (muito baixa). Alertas que retratam a tentativa de ganhar privilégio de administrador ou alertas gerados pela detecção de código executável são exemplos de alertas com prioridade alta (1). Já os alertas gerados por tentativa de vazamento de informação e negação de serviço são consideradas de prioridade média (2). Os alertas de prioridade baixa (3) são gerados, por exemplo, devido à detecção de tráfego desconhecido. Por fim, os alertas de prioridade muito baixa (4) são gerados por causa da detecção de uma conexão TCP suspeita.

5. 10.100.1.1: refere-se ao endereço IP de origem da intrusão.

6. 10.100.102.13: refere-se ao endereço IP de destino da intrusão.

Em um ataque multiestágio, considera-se que um atacante utiliza uma estratégia de ataque composta por uma sequência de passos de ataque para atingir seu objetivo [15]. Ainda, considera-se que mais de um *host* atacante ou mais de um *host* alvo podem ser envolvidos em um mesmo cenário de ataque como, por exemplo, o ataque de negação de serviço distribuído e o ataque de reconhecimento. No entanto, a informação trazida por um único alerta é restrita a apenas um tipo de intrusão, um endereço IP de origem e um endereço IP de destino. Logo, um alerta, individualmente, não é capaz de representar um ataque multiestágio de forma apropriada. Portanto, a informação de um alerta isolado é considerada de baixo nível. Dessa forma, a análise de um conjunto de alertas correlacionados torna-se mais favorável do que a análise individual dos alertas.

Os IDSs detectam as intrusões utilizando métodos como a detecção baseada em assinatura ou anomalia. Uma assinatura representa características de uma ameaça conhecida [19]. A detecção baseada em assinatura tenta detectar as possíveis ameaças por meio da comparação das assinaturas conhecidas com os eventos observados [19, 21]. Se a assinatura equivale ao evento observado, um alerta é gerado [22]. Por exemplo, o Snort possui uma assinatura em sua base de dados chamada de *ICMP Ping Nmap*. Uma instância do Snort gera um alerta com esta assinatura quando detecta um pacote ICMP com tamanho de dados igual a zero. Essa assinatura indica uma possível tentativa de reconhecimento da rede. O método de detecção baseada em assinatura é eficiente para detectar ameaças conhecidas [19, 23] e tende a gerar menor taxa de falsos positivos que as abordagens baseadas em anomalia [22, 21]. Porém, este método não é efetivo para detectar novos tipos de

ataque ou variações de um ataque conhecido. Além disso, a base de dados de assinaturas precisa ser constantemente atualizada [22, 24, 23].

Assumindo que atividades maliciosas apresentam comportamentos diferentes do comportamento normal do sistema, na detecção baseada em anomalias o IDS aprende qual o comportamento normal do sistema. Consequentemente, ele é capaz de detectar e gerar alertas quando um evento anormal acontece. Dessa forma, este método consegue detectar ameaças desconhecidas. Porém, aprender o comportamento normal do sistema não é uma tarefa trivial, já que um comportamento considerado normal para o sistema pode mudar constantemente. Além disso, anomalias não são necessariamente intrusões, pois podem significar apenas situações atípicas. Ainda, os IDSs baseados em anomalia também podem apresentar alta taxa de falsos positivos [19, 22, 24, 21].

2.3 Análise de Alertas de Intrusão

Esta seção apresenta os principais conceitos relacionados à análise de alertas de intrusão. As informações apresentadas são baseadas no estudo de artigos que aplicam diferentes técnicas para análise de alertas de intrusão. As técnicas foram organizadas na taxonomia proposta nesse trabalho, representada pela Figura 2.

Nas próximas subseções, a taxonomia é apresentada em quatro ramos separados para melhor visualização. Cada ramo representa um objetivo que auxilia a análise de alertas de intrusão. A raiz de cada ramo representa um objetivo distinto dentro da área de análise de alertas de intrusão: redução de dados, identificação dos cenários de ataque, identificação das estratégias de ataque ou análise das estratégias de ataque. Os nós internos da taxonomia representam os meios para atingir o objetivo. Já os nós folha representam as técnicas utilizadas para atingir o objetivo.

O primeiro ramo da taxonomia (Seção 2.3.1) organiza as técnicas aplicadas para atingir o objetivo de redução de dados. A redução de dados é um dos objetivos mais encontrados nos trabalhos relacionados. As técnicas para redução de dados tentam reduzir a grande quantidade de alertas de IDS por meio de filtragem ou agregação. O segundo ramo da taxonomia (Seção 2.3.2) organiza as técnicas utilizadas para identificação de cenários de ataque, os quais agrupam alertas relacionados com base nos endereços IP de origem e destino. O terceiro ramo da taxonomia (Seção 2.3.3) organiza as técnicas usadas para identificar as estratégias de ataque (por exemplo, as sequências de assinaturas dos alertas). Essas técnicas podem ser aplicadas para extrair estratégias de ataque a partir de cenários de ataque. O último ramo da taxonomia (Seção 2.3.4) organiza as técnicas aplicadas para a análise das estratégias de ataque. Portanto, organiza técnicas para medir a similaridade entre duas estratégias de ataque, identificar padrões de estratégias de ataque, verificar a importância de uma estratégia e predizer os próximos passos de ataques futuros.

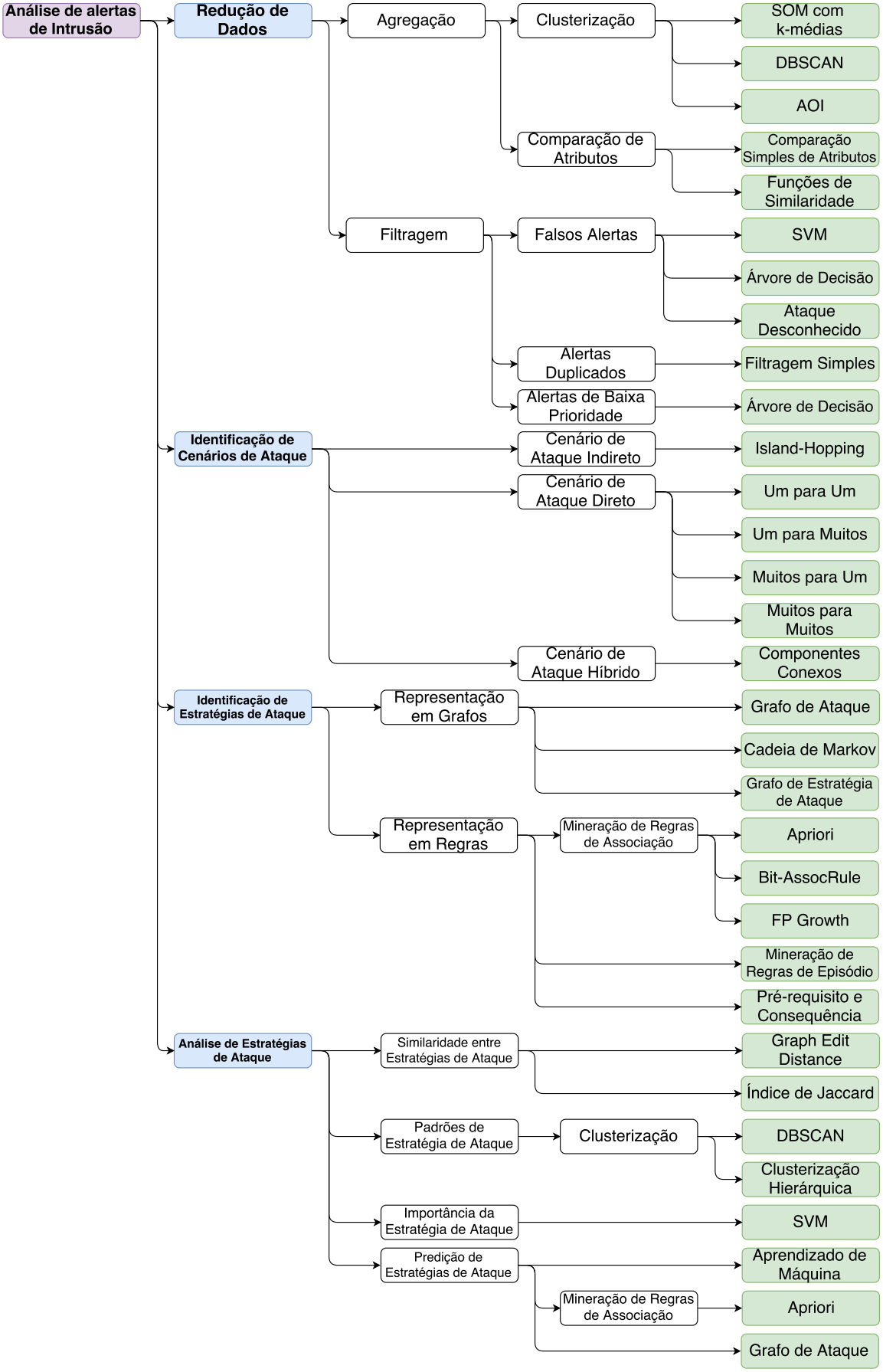


Figura 2 – Taxonomia para Análise de Alertas de Intrusão.

2.3.1 Redução de Dados

A redução na quantidade de alertas permite uma melhor performance na sua análise. Para reduzir a quantidade de alertas, os pesquisadores realizam a agregação de dados e filtragem de dados. A agregação de dados visa agrupar alertas similares com a utilização de técnicas de clusterização ou comparação de atributos. Por outro lado, a de filtragem dos dados é realizada para eliminar alertas duplicados, falsos positivos e alertas de baixa prioridade. A Figura 3 mostra o ramo da taxonomia para técnicas de redução de dados.

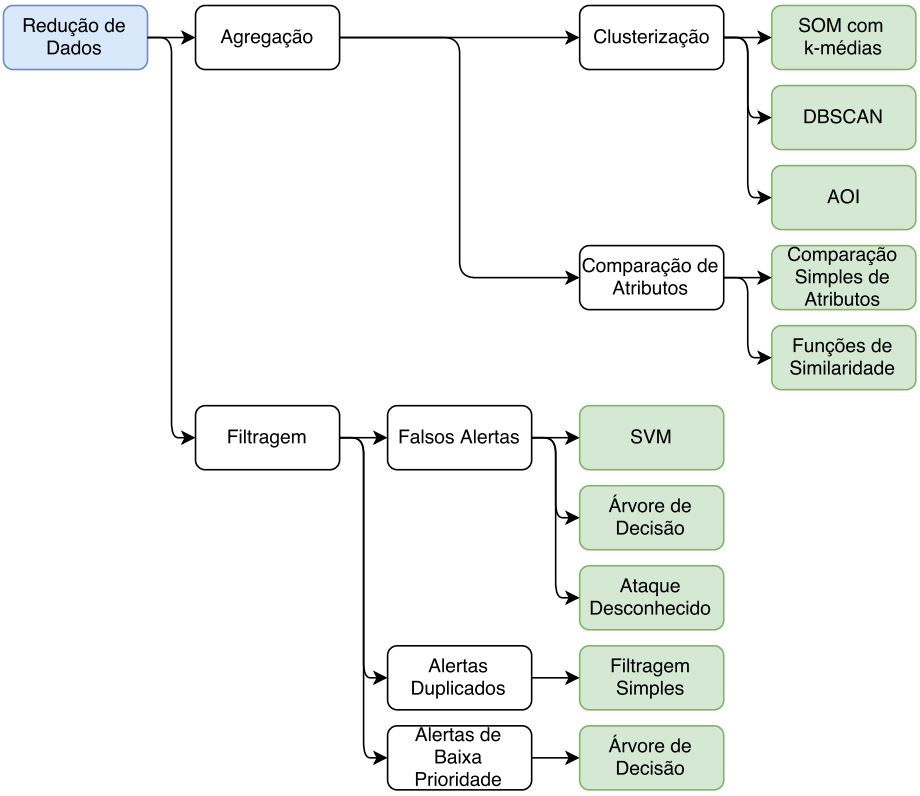


Figura 3 – Ramo da taxonomia com técnicas de redução de dados.

A agregação de dados é aplicada para unir alertas similares, reduzindo a quantidade de alertas redundantes a ser investigada. Para este propósito, a clusterização de dados e comparação de atributos são aplicadas.

A clusterização de alertas de IDS pretende agrupar alertas que são similares e separar daqueles que são dissimilares. Desta forma, apenas um hiperalerta representará um cluster de vários alertas similares, reduzindo a quantidade de dados para se analisar. Mapas Auto-Organizáveis (*Self-Organizing Map* - SOM) com k-médias [25], *Density-Based Spatial Clustering of Applications with Noise* (DBSCAN) [26] e Indução Orientada a Atributo (*Attribute-Oriented Induction* - AOI) [16, 4] são técnicas aplicadas para clusterização de alertas e obtenção de hiperalertas.

A comparação de atributos é uma estratégia mais simplificada quando comparada

com a clusterização. A técnica de comparação de atributos simplificada agrupa alertas se eles apresentarem o mesmo valor para determinados atributos. Por exemplo, se dois alertas apresentarem endereços IP de origem, endereços IP de destino e assinaturas idênticas, então eles são agrupados. É possível utilizar um novo atributo para indicar a quantidade de alertas que foram agrupados por meio desta técnica [27, 28].

O uso das funções de similaridade é uma técnica mais robusta para agrupar alertas de IDS através da comparação de atributos. Valdes e Skinner [29] propuseram um método probabilístico para agrupar alertas por meio da similaridade entre atributos. Uma função de similaridade para cada atributo do alerta é definida e retorna um valor de 0 (baixa similaridade) até 1 (alta similaridade). Se a similaridade entre os atributos do alerta forem maior que um certo limiar, então eles são agrupados.

A filtragem de alertas é aplicada para remover tipos de alertas específicos que não são de interesse para a análise. Falsos alertas, alertas duplicados e alertas de baixa prioridade são exemplos de dados que são filtrados.

Os falsos alertas são gerados quando o IDS classifica uma atividade normal como maliciosa [19]. A sua filtragem é realizada utilizando técnicas de aprendizado de máquina como Árvore de Decisão e Máquina de Vetor de Suporte (*Support Vector Machine* - SVM). Essas técnicas aprendem o comportamento dos alertas falsos e dos alertas verdadeiros para realizar uma classificação de alertas desconhecidos. Dessa forma, os alertas classificados como falsos podem ser descartados. A Árvore de Decisão e o SVM são técnicas supervisionadas para classificação. Portanto, os dados de treinamento precisam ser previamente rotulados, indicando se cada alerta é verdadeiro ou falso. Isso torna a detecção de falsos alertas uma tarefa árdua em situações reais, uma vez que é preciso identificar e rotular manualmente uma quantidade suficiente de alertas para treinar o modelo de aprendizado de máquina [25]. Outra técnica para encontrar e filtrar falsos alertas é conhecer os padrões de ataques. Alertas que não correspondem a nenhum padrão conhecido podem ser considerados falsos. Porém, existe o risco de descartar acidentalmente alertas verdadeiros que podem estar relacionados a estratégias de ataque novas ou desconhecidas [6].

A filtragem de alertas duplicados é feita por meio da técnica de filtragem simples. Por meio desta técnica, pretende-se comparar e remover de alertas que compartilham os mesmos valores de atributos. Esses alertas são considerados redundantes e são gerados pela implantação de múltiplos IDSs na mesma rede [30, 31].

Os alertas de baixa prioridade são filtrados de acordo com a definição de prioridade de alertas. Esta definição varia de acordo com o contexto. Por exemplo, alertas de alta prioridade podem estar relacionados aos ataques contra elementos importantes de um sistema [32] ou aos ataques com alta taxa de sucesso [30]. As Árvores de Decisão são técnicas usadas para verificar a prioridade dos alertas [32].

2.3.2 Identificação de Cenários de Ataque

Os cenários de ataque representam a relação entre *hosts* atacantes e *hosts* alvos por meio de seus endereços IP. Como existe a possibilidade de vários *hosts* não relacionados atacarem simultaneamente uma mesma rede, os alertas de IDS de diferentes cenários podem ser gerados de forma intercalada na base de dados. A Figura 4 mostra o ramo da taxonomia para as técnicas de identificação de cenários de ataque.

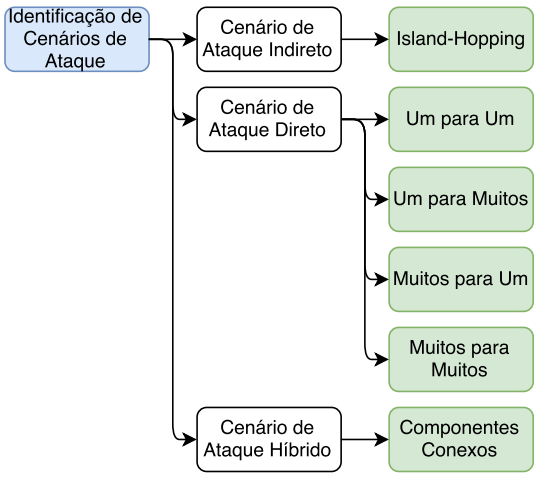


Figura 4 – Ramo da taxonomia com técnicas para identificação de cenários de ataque.

A identificação de cenários de ataque torna possível realizar a análise de diferentes tipos de cenário, como os cenários de ataque indiretos (*Island-Hopping*), os cenários de ataque diretos (um para um, um para muitos, muitos para um e muitos para muitos) e cenários de ataque híbridos (componentes conexos).

Os cenários de ataque indiretos são detectados por meio da técnica de identificação de cenários de ataque *Island-Hopping*. Esses cenários ocorrem quando um atacante invade um alvo e o controla para realizar novos ataques indiretamente [30, 33]. Dessa forma, um endereço IP é considerado alvo e atacante ao mesmo tempo. A Figura 5 mostra um exemplo de cenário de ataque *Island-Hopping*, em que o *host* 10.0.0.1 controla o *host* 10.0.0.2 para atingir o alvo final 10.0.0.3.

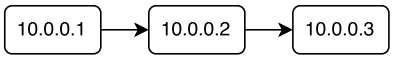


Figura 5 – Exemplo de cenário de ataque do tipo *Island-Hopping* .

Por outro lado, os cenários de ataque diretos acontecem quando um *host* de origem direciona um ataque à um *host* alvo diretamente. Cenários *um para um* ocorrem quando um *host* realiza um ataque contra um alvo. A técnica de identificação de cenários de ataque *um para um* auxilia na compreensão da sequência de ataques que um *host* específico realiza

contra um alvo específico. Os cenários de ataque do tipo *muitos para muitos* acontecem quando mais de um *host* ataca mais de um alvo. As técnicas para identificação deste tipo de cenário pode revelar características de, por exemplo, um *worm* ou vírus atacando a rede [31]. Cenários de ataque do tipo *muitos para um* são definidos por ataques de múltiplos *hosts* contra apenas um alvo. A técnica para identificação deste tipo de cenário de ataque é utilizada para concentrar a análise em um alvo específico, como, por exemplo, um servidor importante da rede [30]. Além disso, esta técnica pode revelar tipos de ataque como DDoS [31]. Por outro lado, os cenários de ataque do tipo *um para muitos* consistem em um *host* direcionando ataques contra múltiplos alvos. A técnica para identificação deste tipo de cenário de ataque é utilizada para concentrar a análise em um atacante específico. A análise deste tipo de cenário de ataque pode revelar ataques de reconhecimento [30, 31].

Já um cenário de ataque híbrido apresenta ambos os ataques diretos e indiretos. A técnica de identificação de componentes conexos é utilizada para identificar tais cenários. Os alertas de IDS são representados como um ou mais componentes conexos, os quais são grafos conexos onde os vértices representam os *hosts* e as arestas direcionadas representam a direção do ataque, de um *host* de origem para um *host* de destino. Se algum *host* realiza ataque contra um outro *host*, então eles pertencem ao mesmo componente conexo. Senão, eles são separados em componentes conexos diferentes.

Assim, a identificação de cenários de ataque nos alertas de IDS encontra conjuntos de alertas relacionados pelos seus endereços IP de origem e destino. Dessa forma, os conjuntos de alertas relacionados são estudados separadamente, melhorando o foco da análise [6]. Ainda, é possível realizar um estudo mais aprofundado de cada cenário de ataque para identificar as estratégias de ataque utilizada em cada um deles [34].

2.3.3 Identificação de Estratégias de Ataque

Em geral, um ataque é composto por uma sequência de passos ou fases de ataque. Por exemplo, primeiro é realizado um passo de reconhecimento da rede em busca de vulnerabilidades, depois ocorre a exploração das vulnerabilidades e, por fim, o atacante atinge seu objetivo [15]. A identificação de estratégias de ataque busca descobrir estas sequências de passos de ataque, as quais encontram-se abstratas nos alertas de IDS. A Figura 6 mostra o ramo da taxonomia para identificação de estratégias de ataque.

As estratégias de ataque são representadas em forma de grafos ou regras. Grafo de ataque, cadeia de Markov e grafos de estratégia de ataque são técnicas baseadas em grafos para identificação de estratégias de ataque.

Um grafo de ataque é um grafo direcionado que detalha as condições necessárias para explorar uma vulnerabilidade, já que a exploração de uma ou mais vulnerabilidades específicas pode ser a condição necessária para permitir a exploração de outra vulnerabilidade. Por exemplo, realizar um ataque de reconhecimento para identificar falhas na

segurança do *host* alvo pode ser a condição necessária para explorar tais falhas, ou controlar vários *hosts* alvos pode ser a condição necessária para realizar um ataque de DDoS [35]. Os grafos de ataque não são necessariamente relacionados aos alertas de IDS, pois são gerados pelo estudo e análise das vulnerabilidades de um sistema. Os alertas de IDS são utilizados para complementar e atualizar os grafos de ataque [36, 35]. Já a cadeia de Markov é uma técnica que identifica e representa estratégias de ataque por meio de um grafo direcionado, onde os vértices representam os passos de ataque e as arestas representam a probabilidade de um passo de ataque acontecer após outro passo de ataque. Portanto, dado um passo de ataque, esta técnica indica a probabilidade de outros passos de ataque ocorrem como consequência [37]. Por fim, os grafos de estratégia de ataque, diferentes dos grafos de ataque, são extraídos diretamente dos alertas de IDS, ao invés de serem apenas atualizados por eles. Após separar os alertas de IDS em cenários de ataque, o grafo de estratégia de ataque de cada cenário é identificado por meio da sequência de assinaturas dos alertas ordenados por *timestamp*. Portanto, um grafo de estratégia de ataque é um grafo direcionado, onde cada vértice representa uma assinatura e cada aresta representa a relação sequencial entre duas assinaturas [34].

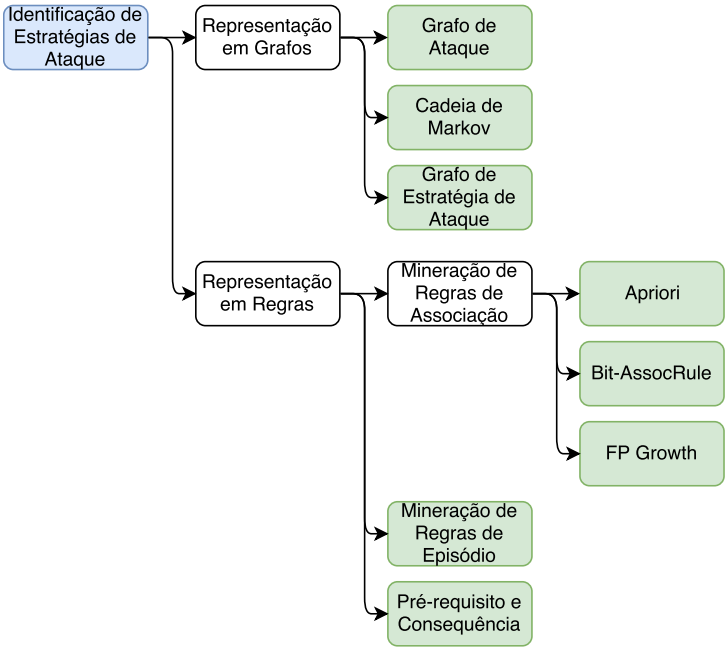


Figura 6 – Ramo da taxonomia com técnicas para identificação de estratégias de ataque.

Apriori, Bit-AssocRule, FP Growth, mineração de regras de episódios e pré-requisito e consequência são exemplos de técnicas baseadas em regras, as quais são utilizadas para identificação e representação de estratégias de ataque.

A mineração de regras de associação é aplicada para encontrar regras que descrevem as assinaturas dos alertas de IDS. As regras indicam quais e com qual frequência as assinaturas aparecem relacionadas na base de dados. Por exemplo, uma regra pode

indicar a frequência com que duas assinaturas específicas aparecem em um mesmo cenário de ataque [6]. O Apriori é um exemplo de técnica para mineração de regras de associação. Esta técnica apresenta a desvantagem de exigir múltiplas leituras da base de dados. O Bit-AssocRule é uma alternativa utilizada para mineração de regras de associação que trabalha em nível de bit para superar a performance do Apriori [38]. Por fim, o *FP_Growth* é outra técnica para mineração de regras de associação aplicada em alertas de IDS [39]. Sua principal característica é o uso de uma estrutura de dados compacta conhecida como árvore de padrões frequentes (*FP_Tree*), a qual elimina a necessidade de múltiplas leituras da base de dados [39].

A mineração de regras de episódios é uma técnica que encontra padrões em sequências de eventos [16]. Considerando os alertas de IDS, ordenados pelo atributo de *timestamp*, a mineração de regras de episódios identifica a porcentagem dos casos em que uma assinatura as_1 é seguida por uma outra assinatura as_2. Quando o atacante utiliza um *script* ou uma ferramenta de ataque, é provável que seja produzida a mesma sequência de alertas para diferentes alvos. Desta forma, aumenta-se a proporção de certas sequências de assinatura na base de dados. Portanto, esta técnica é útil para identificar ataques realizados por *scripts* ou ferramentas de ataque.

Por fim, a técnica de pré-requisito e consequência dos ataques também é utilizada para identificar estratégias de ataque em forma de regras [40]. O pré-requisito é uma condição para uma intrusão ter sucesso. Já a consequência refere-se às possíveis consequências causadas pela intrusão. Por exemplo, quando um sistema possui uma vulnerabilidade p como pré-requisito e um ataque de *buffer overflow* é realizado com sucesso, então, o ganho de acesso ao alvo ocorre como consequência. A descrição desta técnica se assemelha ao dos grafos de ataque. A diferença é que, ao invés de um grafo, são utilizadas combinações de predicados lógicos para representar as regras [40].

2.3.4 Análise de Estratégias de Ataque

Ao investigar um conjunto de alertas de intrusão, um objetivo a ser alcançado é a identificação das estratégias de ataque empregadas pelos atacantes, como explicado na Seção 2.3.3. Após descobrir essas estratégias, pode ser necessário fazer análises mais profundas sobre cada uma delas. É disso que trata o objetivo "análise de estratégias de ataque". Como múltiplas estratégias similares podem ser realizadas em um curto período de tempo, a clusterização pode ser aplicada para agrupar estratégias similares, representando-as em padrões de estratégia de ataque. Para isso, técnicas para medir a similaridade entre grafos de estratégia de ataque devem ser definidas. Além disso, técnicas de aprendizado de máquina como o SVM podem ser utilizadas para avaliar a importância de uma estratégia de ataque automaticamente. Ainda, considerando que uma estratégia de ataque utilizada anteriormente possa ser empregada novamente no futuro, é possível realizar a predição

dos próximos passos de ataque. A Figura 7 mostra o ramo da taxonomia com técnicas para análise de estratégia de ataque.

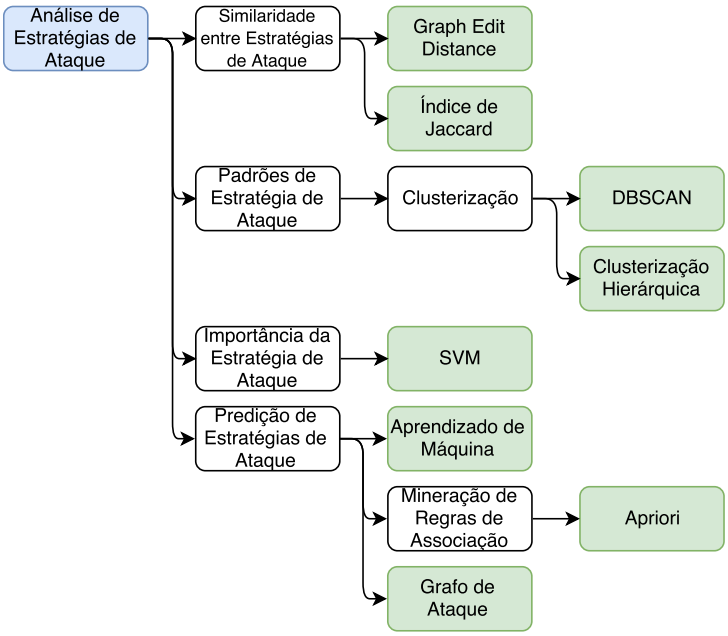


Figura 7 – Ramo da taxonomia com técnicas para análise de estratégias de ataque.

Devido à natureza repetitiva dos ataques, muitos cenários de ataque diferentes podem apresentar estratégias de ataques similares. Portanto, o cálculo da similaridade e a clusterização são realizados para identificar os padrões de estratégias de ataque. Dessa forma, uma base de dados contendo dezenas de cenários de ataque diferentes, cada qual com a sua estratégia de ataque, pode ser resumida em apenas alguns padrões de estratégia de ataque [34]. A técnica *Graph Edit Distance* (GED) é uma métrica de distância que identifica quantas operações de edição (adição, remoção e substituição de arestas e vértices) são necessárias para transformar um grafo em outro. Nesse trabalho, utilizamos uma técnica baseada no índice de Jaccard para medir a similaridade entre estratégias de ataque, que será explicada na Seção 3. Com o método de similaridade definido, as técnicas de clusterização de grafos de estratégia de ataque podem ser aplicadas. O DBSCAN [41] e a clusterização hierárquica [34] são aplicados como técnicas para clusterização de estratégias de ataque. Cada cluster composto por estratégias similares representam um padrão de estratégia de ataque.

A importância de uma estratégia de ataque é estimada automaticamente utilizando a técnica SVM.Um conjunto de estratégias de ataques rotuladas como “interessante” ou “não interessante” pode ser utilizado para treinar o classificador SVM, o qual avalia automaticamente as próximas estratégias de ataque [42]. A identificação da importância das estratégias de ataque torna possível a filtragem daquelas de menor interesse.

Já a predição de estratégias de ataque é realizada utilizando técnicas baseadas em

aprendizado de máquina. O Nexat [33] é um exemplo de técnica baseada em aprendizado de máquina, o qual utiliza informações de alertas de IDS para predizer os próximos passos do atacante. Dessa forma, as perdas que os ataques futuros podem causar são reduzidas.

As estratégias de ataque na forma de regras ou grafos também são utilizadas como auxílio para predizer ataques. Por exemplo, a mineração de regras de associação por meio da técnica Apriori gera regras que podem ser analisadas para predizer os próximos passos do atacante, pois elas indicam os passos de ataque que acontecem juntos, com uma certa confiança e suporte [28, 38]. Ainda, os grafos de ataque atualizados com informações de alertas de IDS podem ser usados para predizer ataques, já que eles representam uma cadeia de condições e explorações de vulnerabilidade necessárias para executar um ataque completo [35, 36].

2.4 Soluções Propostas para Análise de Alertas de Intrusão

Esta seção apresenta os trabalhos estudados, os quais abordam o tema de análise de alertas de intrusão e foram publicados entre 2001 e 2015. As seguintes subseções organizam os trabalhos de acordo com seus principais objetivos e um breve resumo de cada um deles é apresentado. Dentro de cada subseção, os trabalhos são descritos de forma cronológica. Nota-se que alguns trabalhos não se restringem a apenas um objetivo. Isso ocorre, pois os objetivos podem ser relacionados, por exemplo, pode-se reduzir a quantidade de alertas de IDS para melhorar a performance das próximas operações a serem realizadas. Assim, é possível organizar os alertas restantes em cenários de ataque, representando grupos de alertas gerados por *hosts* cujos endereços IP são relacionados. Após isso, é possível identificar as estratégias de ataque utilizadas em cada cenário de ataque. E, por fim, tendo as estratégias de ataque identificadas, é possível realizar uma análise mais aprofundada de cada uma delas. Portanto, alguns trabalhos buscam focar em apenas um dos objetivos, fornecendo ferramentas que auxiliam em trabalhos futuros, e outros trabalhos procuram combinar mais de um objetivo, a fim de oferecer ferramentas mais completas para a análise de alertas de IDS. Por exemplo, o trabalho de Valdes e Skinner [29] apresenta o foco apenas em redução de dados. Esse trabalho foi utilizado posteriormente para melhorar performance do método proposto por Lee et al. [31], cujos objetivos são redução de dados e identificação de cenários de ataque.

2.4.1 Redução de Dados

Os trabalhos que possuem redução de dados como principal objetivo são apresentados nesta subseção. A Tabela 1 mostra as técnicas utilizadas em cada trabalho.

Tabela 1 – Trabalhos que apresentam redução de dados como principal objetivo.

Trabalho	**Técnicas**
Valdes e Skinner (2001) [29]	◇ Agregação → Comparação de Atributos → Funções de Similaridade
Julisch (2003) [4]	◇ Agregação → Clusterização → AOI
Soleimani e Ghorbani (2008) [32]	◇ Filtragem → Alertas de Baixa Prioridade → Árvore de Decisão
Dasireddy et al. (2010) [27]	◇ Agregação → Comparação de Atributos → Comparação Simples de Atributos
Fatma e Mohamed (2013) [25]	◇ Agregação → Clusterização → SOM com k-médias ◇ Filtragem → Falsos Alertas → SVM e Árvore de Decisão
Spathoulas e Katsikas (2013) [5]	◇ Filtragem → Alertas Duplicados → Filtragem Simples ◇ Agregação → Comparação de Atributos → Comparação Simples de Atributos e Funções de Similaridade
Ghasemigol e Ghaemi-Bafghi (2015) [26]	◇ Agregação de Dados → Clusterização → DBSCAN

Em 2001, Valdes e Skinner [29] propuseram uma abordagem probabilística para auxiliar a análise de alertas de IDS. Esta abordagem tem como objetivo reduzir a quantidade de alertas agrupando aqueles que são similares. Para isso, funções de similaridade foram desenvolvidas para medir a similaridade entre cada atributo dos alertas. A abordagem proposta foi testada com uma base de dados real e uma base de dados experimental. Os resultados mostraram uma redução de 50% até 66% na base de dados real. Já a base de dados experimental pôde ser reduzida em até cinquenta vezes.

Em 2003, Julisch [4] propôs uma abordagem para análise de causas raízes. Uma causa raiz refere-se ao motivo da geração de um alerta. De acordo com o autor, 90% dos alertas são gerados devido a apenas algumas causas raízes. Para identificar as causas raízes, o autor propôs uma adaptação da técnica de clusterização conceitual AOI para agrupar alertas de IDS considerados similares. A técnica foi avaliada com uma base de dados real composta por 156.380 alertas gerados durante um mês. Como resultado, foi descoberto que as causas raízes são persistentes e continuam gerando alertas enquanto não são tratadas. Portanto, a identificação e remoção de uma causa raiz geraria uma redução na quantidade de alertas futuros.

Em 2008, Soleimani e Ghorbani [32] propuseram um método para reduzir a quantidade de alertas de IDS por meio da filtragem dos que possuíam baixa prioridade. Para diferenciar os alertas críticos dos alertas de baixa prioridade, os autores definiram o que seria um alerta crítico. Então, utilizando uma árvore de decisão, o método proposto pôde identificar os alertas críticos e filtrar todos os outros alertas de baixa prioridade. As bases de dados utilizadas para os testes foram DARPA 2000 e uma base de dados experimental gerada por uma sequência conhecida de passos de ataque. Os resultados mostraram que o método proposto reduziu a quantidade de alertas em 90%.

Em 2010, Dasireddy et al. [27] propuseram uma abordagem para agrupar os alertas

utilizando comparação simples de atributos. Os alertas são agrupados se eles apresentam a mesma causa raiz, mesmo endereço IP de origem e destino e possuem uma diferença de tempo menor que um segundo. Os resultados mostraram uma alta taxa de redução de alertas (de 150.000 para 3.000). A base de dados utilizada nos experimentos foi gerada em uma rede de 300 nós.

Em 2013, Fatma e Mohamed [25] propuseram um sistema composto por dois componentes para redução de alertas. O primeiro componente gerou meta-alertas aplicando técnicas para clusterização como SOM com k-médias. O segundo componente aplicou técnicas de classificação como SVM e árvore de decisão para classificar alertas como verdadeiros ou falsos positivos. O sistema proposto foi avaliado com a base de dados DARPA 1999 e os falsos positivos foram detectados e filtrados em até 90%.

Em 2013, Spathoulas e Katsikas [5] propuseram um sistema para gerar uma representação de alto nível dos alertas. O sistema proposto é composto por três fases. A primeira fase aplica comparação simples de atributos para agrupar todos os alertas que pertencem ao mesmo evento (alertas com mesmo endereço IP de origem e destino, mesma assinatura e que ocorreram próximos no tempo). Além disso, os alertas duplicados foram filtrados. A segunda fase agrupa os alertas relacionados utilizando as funções de similaridade propostas por Valdes e Skinner [29]. A última fase gera uma representação visual para cada grupo encontrado na fase anterior. Os experimentos foram realizados utilizando a base de dados DARPA 2000 e uma base de dados experimental gerada por simulações de ataque. Os resultados mostraram que o sistema proposto pôde representar visualmente os alertas de IDS.

Em 2015, Ghasemigol e Ghaemi-Bafghi [26] propuseram um sistema para correlação com base na entropia dos alertas. Para isso, a entropia dos alertas de IDS foi calculada, gerando uma matriz de entropia. A técnica de clusterização DBSCAN foi aplicada para agrupar alertas de acordo com a matriz de entropia. Por fim, foi gerada uma representação visual para os clusters. O sistema foi validado utilizando a base de dados DARPA 2000. Os resultados mostraram uma redução de 99,98% dos dados.

2.4.2 Identificação de Estratégias de Ataque

Em 2002, Ning et al. [40] propuseram um *framework* para correlação de alertas de IDS. Os autores consideram que um *host* atacante realiza passos de ataque como preparativos para próximos passos de ataque. Portanto, o *framework* proposto utiliza a técnica de pré-requisito e consequência para analisar os passos de ataque, identificando as estratégias de ataque utilizadas pelo *host* atacante. Os experimentos foram realizados utilizando a base de dados DARPA 2000. Os resultados mostraram que o *framework* conseguiu correlacionar os alertas para gerar uma representação intuitiva das estratégias de ataque.

2.4.3 Análise de Estratégias de Ataque

Os trabalhos que apresentam a análise de estratégias de ataque como principal objetivo são apresentados nesta subseção. A Tabela 2 mostra as técnicas utilizadas nesses trabalhos.

Tabela 2 – Trabalhos que apresentam a análise de estratégias de ataque como principal objetivo.

Trabalho	**Técnicas**
Cipriano et al. (2011) [33]	◇ Predição de Estratégias de Ataque → Aprendizado de Máquina
Zomlot et al. (2013) [42]	◇ Importância da Estratégia de Ataque → SVM

Em 2011, Cipriano et al. [33] propuseram uma abordagem baseada em técnicas de aprendizado de máquina para predizer ataques. Essa abordagem agrupa os alertas de IDS em sequências chamadas de sessões de ataque. Então, as informações das sessões de ataque são analisadas para gerar uma base de conhecimento. A base de conhecimento é utilizada para a predição de ataques. A base de dados da competição hacker *International Capture The Flag* (iCTF) de 2008 foi utilizada para validar o sistema. Os resultados mostraram que a abordagem proposta pôde predizer os ataques com uma média de 94% de acurácia.

Em 2013, Zomlot et al. [42] propuseram um método para avaliar automaticamente a saída do *framework* de análise de intrusão SnIPS. O SnIPS tem como saída os grafos de estratégia de ataque formados pela correlação de alertas. Os autores consideram que algumas dessas saídas não são interessantes para análises mais profundas. Portanto, o método utiliza a técnica de aprendizado de máquina SVM para avaliar e classificar automaticamente os grafos de saída do SnIPS como “interessante” ou “não interessante”. A definição de uma estratégia “interessante” para análises posteriores é estabelecida pelo usuário do método proposto. A necessidade de rotular manualmente uma quantidade suficiente de grafos para treinar a SVM é uma desvantagem desta abordagem. Uma base de dados real foi utilizada para testar a abordagem proposta. Os resultados mostraram que o método proposto consegue automatizar o trabalho de selecionar os grafos relevantes.

2.4.4 Redução de Dados e Identificação de Cenários de Ataque

Os trabalhos que combinam a redução de dados e identificação de cenários de ataque como principais objetivos são apresentados nesta subseção. Nota-se que, para melhorar a eficiência de suas propostas, os autores buscam reduzir a maior quantidade de alertas possível já nas primeiras fases. Dessa forma, uma menor quantidade de alertas precisará ser processada pelas próximas fases dos sistemas propostos [31, 30]. A Tabela 3 mostra as técnicas utilizadas nesses trabalhos.

Tabela 3 – Trabalhos que combinam a redução de dados e identificação de cenários de ataque como principais objetivos.

Trabalho	**Objetivos**	**Técnicas**
Lee et al. (2006) [31]	Redução de Dados	◇ Filtragem → Alertas Duplicados → Filtragem Simples ◇ Agregação → Comparação de Atributos → Funções de Similaridade
	Identificação de Cenários de Ataque	◇ Cenário de Ataque Direto → Um para Muitos, Muitos para Um e Muitos para Muitos
Taha et al. (2010) [30]	Redução de Dados	◇ Filtragem → Alertas Duplicados → Filtragem Simples
	Identificação de Cenários de Ataque	◇ Cenário de Ataque Direto → Um para Um, Um para Muitos e Muitos para Um ◇ Cenário de Ataque Indireto → Island-Hopping

Em 2006, Lee et al. [31] propuseram um sistema para auxiliar o administrador de segurança a analisar grandes quantidade de alertas. O sistema proposto filtra alertas duplicados e os agrupa utilizando as funções de similaridade de Valdes e Skinner [29]. Então, um componente do sistema, chamado de *Situator*, analisa as relações entre os endereços IP dos alertas para identificar os cenários de ataque. O *Situator* identifica cenários de ataque do tipo um para muitos, muitos para um e muitos para muitos. O sistema foi avaliado utilizando dados experimentais gerados por ataques simulados de DDoS. Os resultados mostraram que o sistema proposto foi capaz de reduzir a quantidade de alertas redundantes sem perder informação e identificar os cenários de ataque.

Em 2010, Taha et al. [30] propuseram uma abordagem para aprimorar o tempo de correlação de um sistema desenvolvido anteriormente por Valeur et al. [43]. O sistema de Valeur et al. [43] tem como objetivo reduzir a quantidade de alertas de IDS e identificar os cenários de ataque. O sistema apresenta seis componentes de correlação, responsáveis por remover alertas duplicados, determinar o sucesso de um ataque e identificar os cenários de ataque. A abordagem proposta por Taha et al. assimila as características dos alertas e define quais componentes de correlação do sistema de Valeur et al. devem ser utilizados. Além disso, os componentes de correlação são reordenados a fim de otimizar sua performance, pois a ordem em que cada componente de correlação é aplicado influencia a performance do sistema. Várias bases de dados foram utilizadas para validar a proposta, incluindo o DARPA 2000 e o DARPA 1999. Os resultados mostraram que a abordagem proposta por Taha et al. pôde correlacionar os alertas de forma mais rápida e preservar a acurácia do sistema de Valeur et al.

2.4.5 Redução de Dados e Identificação de Estratégias de Ataque

Em 2002, Julisch e Dacier [16] propuseram e avaliaram duas técnicas separadamente para auxiliar a análise de alertas de IDS. A primeira técnica é a mineração de regras de episódios, a qual é aplicada para identificar as estratégias de ataque em forma de re-

gras. Utilizando esta técnica, foi possível identificar o uso de ferramentas de ataque, as quais tendem a gerar alertas com a mesma sequência de assinaturas para diferentes *hosts* alvos. A segunda técnica é a clusterização conceitual AOI. Esta técnica foi aplicada para agrupar os alertas que apresentam atributos similares e reduzir a quantidade de dados. Os experimentos foram realizados utilizando uma base de dados real gerada por dezesseis IDSs. A técnica de mineração de regras de episódios conseguiu identificar as estratégias de ataque em meio a grande quantidade de alertas de IDS. Já a técnica de clusterização conceitual conseguiu reduzir a quantidade de alertas em uma média de 75%.

2.4.6 Identificação de Cenários e Estratégias de Ataque

Os trabalhos que combinam a identificação de cenários de ataque e a identificação de estratégias de ataque como principais objetivos são apresentados nesta subseção. Observa-se que a identificação de cenários de ataque é um processo que geralmente ocorre antes da identificação da estratégia de ataque. Isto ocorre, pois um cenário de ataque é uma representação dos alertas gerados pelos mesmos *hosts* de origem, para os mesmos *hosts* de destino. Portanto, a organização dos alertas em cenários de ataque permite o estudo isolado das estratégias de ataque, uma para cada conjunto de alertas que apresentam endereços IP relacionados. A Tabela 4 mostra as técnicas utilizadas nesses trabalhos.

Tabela 4 – Trabalhos que combinam a identificação de cenários de ataque e a identificação de estratégias de ataque como principais objetivos.

Trabalho	**Objetivos**	**Técnicas**
Sadoddin e Ghorbani (2009) [44]	Identificação de Cenários de Ataque	◇ Cenário de Ataque Direto → Um para Um, Um para Muitos, Muitos para Um e Muitos para Muitos
	Identificação de Estratégias de Ataque	◇ Representação em Regras → Mineração de Regras de Associação → *FP Growth*
Lagzian et al. (2012) [38]	Identificação de Cenários de Ataque	◇ Cenário de Ataque Direto → Um para Muitos e Muitos para Um
	Identificação de Estratégias de Ataque	◇ Representação em Regras → Mineração de Regras de Associação → Bit-AssocRule
Xuewei et al. (2014) [37]	Identificação de Cenários de Ataque	◇ Cenário de Ataque Híbrido → Componentes Conexos
	Identificação de Estratégias de Ataque	◇ Representação em Grafos → Cadeia de Markov

Em 2009, Sadoddin e Ghorbani [44] propuseram uma abordagem para agrupar os alertas em cenários de ataque do tipo um para um, um para muitos, muitos para um e muitos para muitos. Então, utilizando a técnica *FP_Growth*, as estratégias de ataque mais frequentes nos cenários foram identificadas. Desta forma, uma melhor compreensão das

intrusões foi adquirida. Os experimentos foram realizados com a base de dados DARPA 2000 e uma base de dados real. Os resultados mostraram que a abordagem proposta foi capaz de identificar as estratégias de ataque presentes nesta base de dados.

Em 2012, Lagzian et al. [38] desenvolveram um método para identificar estratégias de ataque por meio de correlação de alertas em tempo real. O método consiste em três fases: pré-processamento de alertas, identificação dos cenários de ataque e identificação das estratégias de ataque. A fase de pré-processamento normaliza e converte os alertas para um mesmo formato. Então, os alertas foram agrupados em cenários de ataque do tipo um para muito e muitos para um. Por fim, a técnica Bit-AssocRule foi aplicada nos cenários de ataque para identificar as estratégias de ataques mais frequentes. A base de dados DARPA 2000 foi utilizada para validar o método proposto e os resultados mostraram que as estratégias de ataque puderam ser encontradas em tempo real.

Em 2014, Xuewei et al. [37] propuseram um método para identificar estratégias de ataque. O método proposto consiste em duas fases. A primeira fase tem como objetivo identificar os cenários de ataque. Portanto, os alertas foram agrupados em cenários de ataque por meio da técnica de separação por componentes conexos. Já a segunda fase tem como objetivo identificar as estratégias de ataque de cada cenário. Logo, a técnica de cadeia de Markov foi aplicada. Dessa forma, os grafos foram gerados, onde cada vértice representa um passo de ataque e cada aresta direcionada apresenta as probabilidades de um passo de ataque acontecer após outro passo de ataque. O método proposto foi testado utilizando a base de dados DARPA 2000. Como resultado, o método proposto mostrou ser capaz de identificar os cenários de ataque de forma apropriada.

2.4.7 Identificação e Análise de Estratégias de Ataque

Os trabalhos que combinam a identificação de estratégias de ataque e análise de estratégias de ataque como principais objetivos são apresentados nesta subseção. Além de identificar as estratégias de ataque, os autores também propõem métodos para analisá-las. A análise compreende técnicas para verificar a similaridade entre estratégias de ataque e clusterização, responsável por identificar padrões de estratégias de ataque. Ainda, é apresentada uma técnica para predição de estratégias de ataque por meio de grafos de ataque e aprendizado de máquina. A Tabela 5 mostra as técnicas utilizadas nesses trabalhos.

Em 2003, Ning e Xu [45] propuseram uma técnica para identificar as estratégias de ataque dos alertas de IDS automaticamente. Primeiro, os autores identificaram os grafos de estratégia de ataque presentes nos alertas. Os vértices de cada grafo representam as assinaturas dos alertas e as arestas representam a relação entre elas. Então, a técnica *Graph Edit Distance* foi utilizada para calcular a dissimilaridade entre os grafos de estratégia. As bases de dados utilizadas para os testes incluem o DARPA 2000 e uma base de dados experimental formada por três sequências de ataques contra uma rede isolada. Os

resultados mostraram que foi possível medir a dissimilaridade entre grafos com menos de dez vértices (estratégias com menos de dez assinaturas).

Tabela 5 – Trabalhos que combinam a identificação de estratégias de ataque e análise de estratégias de ataque como principais objetivos.

Trabalho	**Objetivos**	**Técnicas**
Ning e Xu (2003) [45]	Identificação de Estratégias de Ataque	◇ Representação em Grafos → Grafo de Estratégia de Ataque
	Análise de Estratégias de Ataque	◇ Similaridade entre Estratégias de Ataque → Graph Edit Distance
Ahmadinejad et al. (2011) [35]	Identificação de Estratégias de Ataque	◇ Representação em Grafos → Grafo de Ataque
	Análise de Estratégias de Ataque	◇ Predição de Estratégias de Ataque → Grafo de Ataque ◇ Predição de Estratégias de Ataque → Aprendizado de Máquina
Shittu et al. (2015) [41]	Identificação de Estratégias de Ataque	◇ Representação em Grafos → Grafo de Estratégia de Ataque
	Análise de Estratégias de Ataque	◇ Similaridade entre Estratégias de Ataque → Graph Edit Distance ◇ Padrões de Estratégia de Ataque → DBSCAN

Em 2011, Ahmadinejad et al. [35] propuseram um método composto de dois módulos para identificação e análise de estratégias de ataque. O primeiro módulo utiliza grafos de ataque para identificar os ataques já realizados contra o sistema monitorado e, por meio deles, predizer os próximos passos do atacante. Já o segundo módulo utiliza técnicas de aprendizado de máquina para identificar os ataques desconhecidos. As informações sobre os novos ataques identificados pelo segundo módulo são utilizados para atualizar o primeiro módulo. O método proposto foi validado com a base de dados DARPA 2000. Os resultados mostraram que ele foi capaz de correlacionar os alertas de IDS corretamente.

Em 2015, Shittu et al. [41] propuseram um *framework* para identificação de estratégias de ataque utilizando métodos de correlação. Primeiro, o *framework* proposto correlaciona os alertas em meta-alertas (conhecido como grafos de correlação) por meio da probabilidade de uma assinatura de alerta ocorrer após a outra. A prioridade dos meta-alertas foram computadas e eles foram clusterizados utilizando a técnica DBSCAN com *Graph Edit Distance* como medida de similaridade. Então, as características dos clusters foram identificadas para auxiliar na análise de alertas de intrusão. Uma base de dados experimental foi utilizada para testar o *framework* e os resultados mostraram que ele foi capaz de organizar os alertas para auxiliar a análise de alertas de intrusão.

2.4.8 Redução de Dados e Identificação de Cenários e Estratégias de Ataque

Os trabalhos que combinam a redução de dados, identificação de cenários de ataque e identificação de estratégias de ataque como principais objetivos são apresentados nesta subseção. Nesta subseção, nota-se um uso mais abrangente dos objetivos, compreendendo a redução de dados para melhorar a eficiência do sistema e a identificação de cenários de ataque para permitir um estudo isolado das estratégias de ataque presentes na base de dados de alertas de IDS. A Tabela 6 mostra as técnicas utilizadas nesses trabalhos.

Tabela 6 – Trabalhos que combinam a redução de dados, identificação de cenários de ataque e identificação de estratégias de ataque como principais objetivos.

Trabalho	**Objetivos**	**Técnicas**
Treinen e Thurimella (2006) [6]	Redução de Dados	◇ Filtragem → Falsos Alertas → Ataque Desconhecido
	Identificação de Cenários de Ataque	◇ Cenário de Ataque Híbrido → Componentes Conexos
	Identificação de Estratégias de Ataque	◇ Representação em Regras → Mineração de Regras de Associação → Não especificado
Liu et al. (2010) [7]	Redução de Dados	◇ Filtragem → Falsos Alertas → Não especificado
	Identificação de Cenários de Ataque	◇ Cenário de Ataque Direto → Um para Um, Um para Muitos e Muitos para Um
	Identificação de Estratégias de Ataque	◇ Representação em Grafos → Grafo de Estratégia de Ataque

Em 2006, Treinen e Thurimella [6] propuseram um *framework* para identificar estratégias de ataques e auxiliar na filtragem de falsos alertas (alertas relacionados às estratégias desconhecidas). O objetivo é alimentar o *Enterprise Security Manager* (ESM) com essas estratégias de ataque para auxiliar a construção automática de regras para correlação de alertas. Primeiro, os alertas de IDS foram organizados em cenários de ataque por meio da técnica de componentes conexos. Então, foi aplicada a mineração de regras de associação em todos os cenários de ataque para encontrar associações entre as assinaturas dos alertas. Os experimentos foram realizados com uma base de dados real gerada durante 24 horas. Os resultados mostraram que o *framework* foi capaz de identificar novas regras de associação com alta confiança.

Em 2010, Liu et al. [7] propuseram um método para identificar os cenários e suas estratégias de ataque. Além disso, o método proposto providencia uma melhor visualização para os cenários e estratégias de ataque. Primeiro, os alertas foram colocados em um mesmo formato e os falsos alertas foram filtrados. Então, três tipos de cenários de ataque foram explorados: um para muitos, muitos para um e um para um. Os passos de ataque dos cenários do tipo um para um foram extraídos e representados em grafos de estratégia

de ataque. Os experimentos foram realizados utilizando uma base de dados experimental gerada por dois ataques simulados. Os resultados mostraram que o método proposto foi capaz de gerar uma melhor organização e visualização para os alertas.

2.4.9 Redução de Dados e Identificação e Análise de Estratégias de Ataque

Os trabalhos que combinam a redução de dados, identificação de estratégias de ataque e análise de estratégias de ataque como principais objetivos são apresentados nesta subseção. Diferentemente da subseção anterior, nesta subseção os autores optam por técnicas que permitem identificar as estratégias de ataque sem a necessidade de separação dos alertas em cenários de ataque. Ainda, duas formas para predição de estratégias de ataque são apresentadas, uma por meio da mineração de regras de associação [28] e outra por meio de grafos de ataque [36]. A Tabela 7 mostra as técnicas utilizadas nesses trabalhos.

Tabela 7 – Trabalhos que combinam a redução de dados, identificação de estratégias de ataque e análise de estratégias de ataque como principais objetivos.

Trabalho	**Objetivos**	**Técnicas**
Yang et al. (2010) [28]	Redução de Dados	◇ Agregação → Comparação de Atributos → Comparação Simples de Atributos
	Identificação de Estratégias de Ataque	◇ Representação em Regras → Mineração de Regras de Associação → Apriori
	Análise de Estratégias de Ataque	◇ Predição de Estratégias de Ataque → Mineração de Regras de Associação → Apriori
Fayyad e Meinel (2013) [36]	Redução de Dados	◇ Filtragem → Falsos Alertas → Não especificado ◇ Filtragem → Alertas Duplicados → Filtragem Simples
	Identificação de Estratégias de Ataque	◇ Representação em Grafos → Grafo de Ataque
	Análise de Estratégias de Ataque	◇ Predição de Estratégias de Ataque → Grafo de Ataque

Em 2010, Yang et al. [28] propuseram um sistema para analisar e visualizar os alertas de IDS. Primeiro, os alertas de IDS são agrupados utilizando comparação simples de atributos. Os alertas foram agrupados se eles apresentavam o mesmo endereço IP de origem, mesma assinatura e ocorreram com uma diferença de tempo menor que um segundo. Então, um novo atributo foi computado. Este atributo refere-se à quantidade de alertas agrupados. Após isso, a técnica Apriori foi aplicado nos grupos de alertas para identificar as estratégias de ataque em forma de regras de associação. Com as regras de associação, o sistema é capaz de predizer os próximos passos do atacante. Os experimentos foram realizados com uma base de dados gerada em uma rede de 300 nós. Os resultados

mostraram que o sistema proposto é capaz de identificar as estratégias de ataque de forma eficiente.

Em 2013, Fayyad e Meinel [36] propuseram um sistema para predizer as estratégias de ataque utilizando grafos de ataque atualizados com alertas de IDS. Os grafos de ataque atualizados com alertas de IDS são chamados de modelo de predição. O sistema proposto é composto por três fases. A primeira fase consiste em filtrar falsos positivos e alertas duplicados. A segunda fase identifica as sequências de alertas, as quais são mapeadas no grafo de ataque para atualizar o modelo de predição. Por fim, o modelo de predição pode ser utilizado para predizer as próximas estratégias de ataque em tempo real.

2.5 Discussão das Soluções Propostas

Pesquisadores têm proposto vários métodos para a análise e correlação de alertas de IDS nos últimos anos. Nota-se que a maioria dos trabalhos utiliza uma base de dados experimental para a validação dos métodos propostos. A baixa capacidade de uma base de dados experimental refletir situações reais tem sido amplamente criticada na literatura [8, 9, 10, 11].

Os métodos propostos têm como objetivo reduzir a grande quantidade de alertas por meio de técnicas de filtragem e agregação de alertas similares, organizar os alertas em cenários de ataque utilizando técnicas que separam os alertas por meio de endereços IP relacionados, identificar as estratégia de ataque representando-as em forma de grafos e regras, e analisar as estratégias encontradas. Foi observado que tais objetivos podem ser combinados, formando um sistema mais completo, que busca desde reduzir a quantidade de alertas até analisar as estratégias de ataques identificadas.

Considera-se que mais de um atacante pode realizar ataques contra uma mesma rede, gerando uma base de dados cujos alertas pertencem a diferentes cenários de ataque. Portanto, a organização dos alertas em cenários de ataque é importante, pois permite um estudo de cada cenário separadamente. Apenas sete entre vinte e três soluções analisadas consideram o objetivo de identificação de cenários de ataques [31, 30, 44, 38, 37, 6, 7]. Além disso, nenhuma solução utiliza a identificação de cenários de ataque combinada com a identificação de estratégias de ataque e análise de estratégias de ataque. A união desses três objetivos é possível, pois, por meio da identificação de estratégias de ataque, descobre-se qual estratégia de ataque foi utilizada em cada cenário de ataque. Então, sabendo que uma mesma estratégia de ataque pode ser utilizada múltiplas vezes, a análise de estratégias de ataque torna-se útil, já que, por meio dela, é possível descobrir, por exemplo, quantos tipos diferentes de estratégias de ataque foram utilizadas contra a rede (os padrões de estratégias de ataque) e quais estratégias de ataque são utilizadas mais frequentemente nos diferentes cenários de ataque.

Nesse trabalho, diferentemente dos trabalhos analisados, combinamos esses três objetivos, a identificação de cenários de ataque, a identificação de estratégias de ataque e a análise de estratégias de ataque. Em essência, buscamos separar a base de dados de alertas em cenários de ataque e, de cada um deles, extrair informações que podem ser úteis para a segurança da rede. Portanto, este trabalho não se restringe a apenas um estudo das estratégias de ataque, pois a estratégia de ataque é apenas um dos atributos encontrados dentro de um cenário de ataque. Além de informações de estratégia de ataque, consideramos também extrair atributos temporais, informações de endereços IP envolvidos, informações das assinaturas de ataque e características do ataque. Logo, buscamos gerar estruturas de informação que descrevem os cenários de ataques, as quais denominamos de hiperalertas.

Portanto, tendo como base o histórico de alertas que foram gerados em períodos anteriores, neste trabalho é proposto um modelo de correlação *offline* que organiza um histórico de alertas em cenários de ataque e identifica as estratégias de ataque. Então, agrupa as estratégias de ataques similares por meio da clusterização hierárquica aglomerativa utilizando o método Ward [46, 47]. Dessa forma, os padrões de estratégia de ataque utilizados contra o sistema são identificados. Além disso, é proposto um modelo de correlação *online*, que gera e atualiza os hiperalertas conforme os alertas são produzidos por um IDS. Cada hiperalerta apresenta diversas informações sobre um cenário de ataque e é associado a um dos padrões de estratégia de ataque descoberto na fase *offline*. É importante ressaltar que foi utilizada uma base de dados real para a validação da abordagem proposta.

3 GERAÇÃO DE HIPERALERTAS COM BASE NO HISTÓRICO DE ESTRATÉGIAS DE ATAQUE

Neste trabalho, é proposto um método cujo objetivo é solucionar dois dos principais desafios relacionados à análise de alertas de intrusão, em essência, a grande quantidade de alertas gerada pelos IDSs e a informação de baixo nível fornecida. O método proposto gera hiperalertas com base em estratégias de ataque identificadas em um histórico de alertas gerados pelo IDS. Para tanto, a proposta é formada por dois correlacionadores: o correlacionador *offline* e o correlacionador *online*. Ambos são representados pela Figura 8. O correlacionador *offline* recebe como entrada um histórico de alertas de IDS gerados no passado e é responsável por processá-los a fim de encontrar os padrões de estratégia de ataque utilizados previamente pelos atacantes. Por outro lado, tomando como base os padrões de estratégia de ataque revelados pelo correlacionador *offline*, o correlacionador *online* analisa os alertas que são gerados pelo IDS no momento presente, a fim de separá-los em cenários de ataque. Então, é gerado um hiperalerta para descrever cada cenário de ataque. Os hiperalertas são estruturas de informação que descrevem os cenários de ataques por meio de novos atributos. Os correlacionadores *offline* e *online* são detalhados nas próximas subseções.

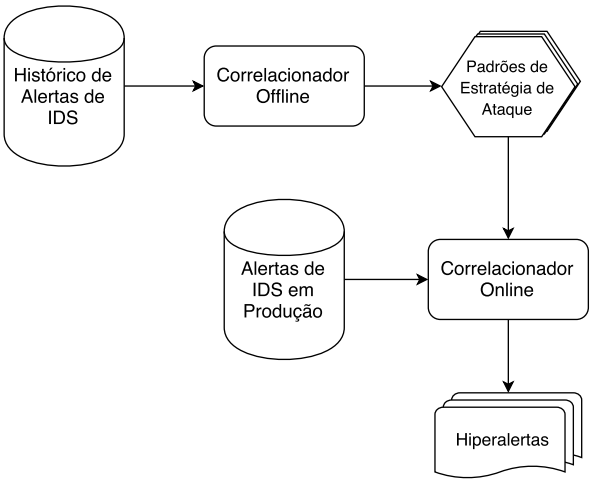


Figura 8 – Correlacionadores *offline* e *online*.

3.1 Correlacionador *Offline*

Considerando o histórico de alertas gerado por um IDS (Figura 9a), o correlacionador *offline* tem como objetivo organizá-los a fim de identificar as estratégias de ataque empregadas pelos atacantes contra a rede protegida pelo IDS. Uma estratégia de ataque é uma sequência de intrusões que um atacante realiza para atingir seu objetivo final.

Para isso, o primeiro passo (fase de agregação) consiste em aplicar a técnica de separação por componentes conexos para agrupar os alertas em cenários de ataque, com base nos endereços IP e proximidade no tempo entre os alertas (Figura 9b). Depois, para cada cenário de ataque, um grafo de estratégia de ataque é extraído (Figura 9c). Em seguida, considerando que podem ser encontrados grafos similares de estratégia de ataque, estes grafos são clusterizados (fase de clusterização), com o objetivo de unir todas as estratégias similares no mesmo cluster, separando as estratégias diferentes. Com isso, os padrões de estratégia de ataque diferentes são identificados (cada cluster representará um padrão - Figura 9d). A técnica utilizada para a fase de clusterização é a clusterização hierárquica com uma métrica de similaridade baseada no índice de Jaccard para comparação de grafos de estratégia de ataque. A Figura 9 apresenta a visão geral do correlacionador *offline*.

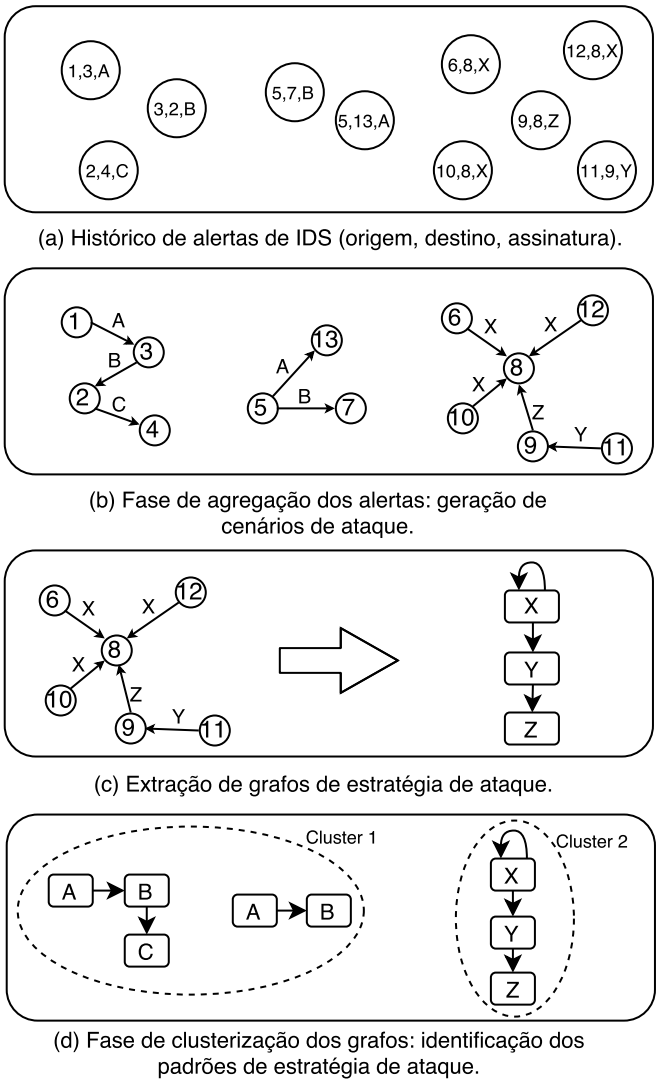


Figura 9 – Visão geral do correlacionador *offline*.

Neste trabalho, um alerta A é definido como a 4-tupla $A = \langle s, d, t, k \rangle$, onde s é o

endereço IP de origem, d é o endereço IP de destino, t é o *timestamp* e k é a assinatura do alerta. Na fase de agregação, o correlacionador *offline* utiliza os endereços IP e o *timestamp* dos alertas para organizá-los com o método de separação por componentes conexos, formando os cenários de ataque. Os cenários de ataque são grupos de alertas que apresentam endereços IP relacionados e ocorreram próximos no tempo. Portanto, considerando dois alertas, A_i e A_j, se $A_i.s = A_j.s$ ou $A_i.d = A_j.d$ ou $A_i.s = A_j.d$ ou $A_i.d = A_j.s$, então A_i e A_j apresentam endereços IP relacionados. Ainda, os dois alertas devem possuir uma diferença de *timestamp* menor do que um limiar denotado por x. Isto é, se $|A_j.t - A_i.t| < x$, então A_i e A_j ocorreram próximos no tempo. Logo, cada grupo de alertas formado pela separação por componentes conexos representa um cenário de ataque, cujos alertas possuem endereços IP relacionados e ocorreram próximos no tempo.

Um componente conexo é a representação visual de um cenário de ataque. O componente conexo é definido como um grafo direcionado e conexo $CC = (V, E)$, onde o conjunto de vértices V representa os endereços IP e o conjunto de arestas E representa a direção do ataque entre os endereços IP de um determinado cenário de ataque [6].

Como primeiro exemplo, considere o conjunto de alertas da Tabela 8. Ao aplicar o método de separação por componentes conexos, dois cenários de ataque podem ser formados desses alertas. O primeiro cenário de ataque é composto pelos três primeiros alertas e reúne os endereços IP 10.0.0.1, 10.0.0.2 e 10.0.0.3. Já o segundo cenário de ataque é composto pelos quatro últimos alertas e reúne os endereços IP 10.0.0.64 e 10.0.0.65. A Figura 10 mostra a representação visual dos dois cenários de ataques por meio de componentes conexos, onde cada nó representa um endereço IP e cada aresta direcionada representa a direção do ataque (da origem para o destino).

Tabela 8 – Alertas de IDS para o primeiro exemplo.

Timestamp	Origem	Destino	Assinatura
01/01/2016 00:30:00	10.0.0.0	10.0.0.1	A
01/01/2016 00:35:00	10.0.0.1	10.0.0.2	B
01/01/2016 00:40:00	10.0.0.1	10.0.0.3	C
01/01/2016 00:45:00	10.0.0.64	10.0.0.65	D
01/01/2016 00:50:00	10.0.0.64	10.0.0.65	D
01/01/2016 00:55:00	10.0.0.64	10.0.0.65	E
01/01/2016 01:00:00	10.0.0.64	10.0.0.65	F

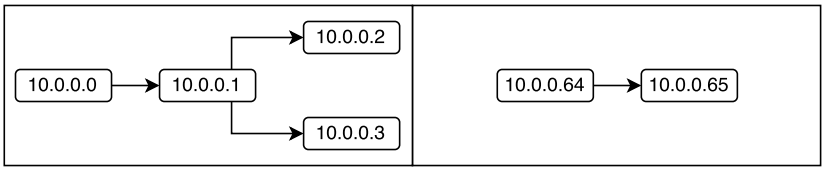


Figura 10 – Componentes conexos do primeiro exemplo.

Um segundo exemplo é baseado nos alertas apresentados na Tabela 9. Neste exemplo, é destacada a importância da condição relacionada à diferença de tempo dos alertas. Todos os alertas na Tabela 9 possuem endereços IP relacionados. Porém, os três últimos alertas foram gerados algumas horas depois dos três primeiros. Considerando um limiar $x = 60$ minutos, os três primeiros alertas formam um cenário de ataque e os três últimos alertas formam um outro cenário de ataque, como mostra a Figura 11.

Tabela 9 – Alertas de IDS para o segundo exemplo.

Timestamp	Origem	Destino	Assinatura
01/01/2016 00:00:00	10.0.0.0	10.0.0.1	A
01/01/2016 00:40:00	10.0.0.1	10.0.0.2	B
01/01/2016 01:20:00	10.0.0.1	10.0.0.3	B
01/01/2016 10:00:00	10.0.0.0	10.0.0.4	A
01/01/2016 10:40:00	10.0.0.4	10.0.0.5	B
01/01/2016 11:20:00	10.0.0.4	10.0.0.6	B

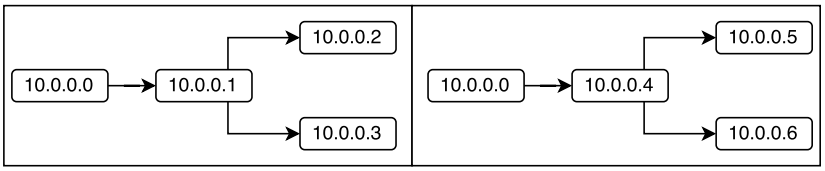


Figura 11 – Componentes conexos do segundo exemplo.

O próximo passo do método proposto considera que, para um atacante atingir seu objetivo, ele realiza uma sequência de ações maliciosas (ataque multiestágio), como, por exemplo, ataques de reconhecimento antes de ataques de exploração de vulnerabilidades. Portanto, as estratégias de ataque foram representadas em forma de grafos, os quais apresentam a sequência de assinaturas que ocorreu em cada cenário de ataque.

Analisando a ordem temporal (por meio do *timestamp*) e as assinaturas dos alertas, é possível determinar a sequência de assinaturas geradas em cada cenário de ataque. A sequência de assinaturas é representada por meio de um grafo direcionado $ASG = (V, E)$, onde o conjunto de vértices V representa as assinaturas dos alertas e o conjunto de arestas E representa a relação de sequência entre as assinaturas dos alertas. Logo, uma aresta e que se origina em v_i e termina em v_j indica que um alerta com a assinatura representada por v_j ocorreu na sequência de um alerta com a assinatura representada por v_i. Este grafo é denominado de grafo de estratégia de ataque, e é extraído de cada cenário de ataque. A Figura 12 mostra o grafo de estratégia de ataque extraído de cada cenário de ataque da Tabela 8. Já a Figura 13 mostra o grafo de estratégia de ataque extraído de cada cenário de ataque da Tabela 9.

Devido à possível existência de grafos de estratégia de ataque similares ou idênticos (como visto na Figura 13), o próximo passo do correlacionador *offline* é agrupar grafos com estratégias similares. Assim, uma mesma estratégia de ataque que ocorreu diversas vezes na base de dados pode ser identificada e considerada como um padrão de estratégia de ataque. Para isso, técnicas de comparação de grafos foram estudadas.

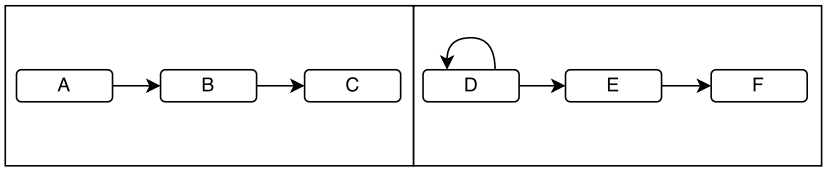


Figura 12 – Grafos de estratégia de ataque dos alertas da Tabela 8.

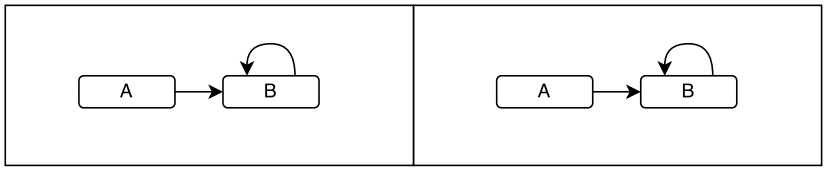


Figura 13 – Grafos de estratégia de ataque dos alertas da Tabela 9.

Para comparar grafos produzidos durante a correlação de alertas, Ning e Xu [45] e Shittu et al. [41] utilizaram a técnica *Graph Edit Distance*. Esta técnica verifica as operações de edição (inserção, remoção ou substituição de vértices ou arestas) que geram um menor custo para transformar um grafo em outro. Quanto maior o custo, mais distante são dois grafos. O custo de cada operação de edição é definido manualmente. Porém, o problema do *Graph Edit Distance* é NP-completo [48]. Vários estudos recentes propuseram algoritmos aproximados. No entanto, essas aproximações garantem apenas a distância mínima local [48]. Logo, o GED não é viável para comparar os grafos gerados pelo método proposto neste trabalho, os quais podem apresentar números elevados de vértices e arestas.

Além disso, os resultados do *Graph Edit Distance* para grafos pequenos são sempre baixos, mesmo se os dois grafos forem completamente diferentes. Isso ocorre pois grafos pequenos possuem poucos vértices e arestas, logo, menos operações de edição são necessárias. A Figura 14 ilustra este problema. O custo da comparação entre os dois grafos da situação (a) é igual ao custo da comparação entre os dois grafos da situação (b): uma substituição de nó (X por Y) em ambos os casos. No entanto, para o problema de comparação de grafos de estratégias de ataque, a comparação entre os dois grafos da situação (a) deveria gerar uma similaridade maior do que na situação (b), pois (a) apresenta vértices similares (A e B) e (b) apresentam vértices completamente diferentes. Logo, se dois grafos

possuírem uma grande quantidade de vértices similares e apenas um nó diferente, o *Graph Edit Distance* vai considerar apenas o nó diferente.

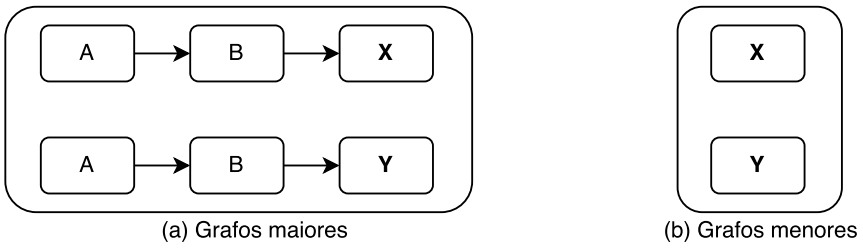


Figura 14 – Exemplo de grafos de estratégia de ataque para o *Graph Edit Distance* e para o método baseado no índice de Jaccard.

Considerando os problemas apresentados pelo *Graph Edit Distance*, Shittu [14] utilizou o índice de Jaccard para calcular a dissimilaridade entre dois grafos. Assim como em Shittu [14], nós utilizamos um método com base no índice de Jaccard para medir a similaridade de grafos de estratégia de ataque, já que esse índice é capaz de determinar a similaridade entre dois conjuntos finitos. Considerando M e N dois conjuntos finitos, o índice de Jaccard entre M e N é definido como o tamanho (cardinalidade) da interseção de M e N, dividido pelo tamanho da união de M e N (Equação 3.1) [49].

$$J(M,N) = \frac{|M \cap N|}{|M \cup N|} \tag{3.1}$$

Um grafo de estratégia de ataque pode ser representado por dois conjuntos finitos: um conjunto de vértices $V = \{v_1, v_2, v_3, ..., v_n\}$, o qual contém as assinaturas presentes no grafo de estratégia de ataque; e por um conjunto de arestas direcionadas $P = \{p_1, p_2, p_3, ..., p_m\}$, o qual contém pares de assinaturas que são adjacentes no grafo de estratégia de ataque, ou seja, um par p_k denota a existência de uma aresta direcionada entre os vértices de uma assinatura v_i e uma assinatura v_j.

Para medir a similaridade entre dois grafos de estratégia de ataque, dois índices de Jaccard são calculados: um índice J_1 para a comparação de vértices e outro índice J_2 para a comparação de pares de vértices (representando as arestas). A fim de atribuir a mesma influência às similaridade entre vértices e arestas, a média dos dois resultados foi considerada como a similaridade final S entre dois grafos de estratégia de ataque (Equação 3.2). Por exemplo, a similaridade entre os dois grafos da situação (a) na Figura 14 pode ser obtida da seguinte forma:

1. Primeiro é calculada a similaridade J_1 dos vértices, a qual é obtida por meio da divisão do tamanho da interseção dos vértices, neste caso $\{A, B\}$, pelo tamanho da união dos vértices, neste caso $\{A, B, X, Y\}$. Logo, $J_1 = 2/4 = 0,5$.

2. Depois, é calculada a similaridade J_2 dos pares de vértices (arestas), a qual é obtida por meio da divisão da interseção das arestas, neste caso $\{A \rightarrow B\}$, pelo tamanho da união das arestas, neste caso $\{A \rightarrow B, B \rightarrow X, B \rightarrow Y\}$. Logo, $J_2 = 1/3 = 0,\overline{3}$.

3. A similaridade final S é dada pela média entre J_1 e J_2, a qual equivale aproximadamente a 0,4.

$$S = \frac{J_1 + J_2}{2}. \tag{3.2}$$

Existe a possibilidade de o conjunto de arestas ser vazio (grafo com apenas um nó). Se os dois grafos comparados apresentam conjuntos vazios de arestas, a similaridade final S é definida apenas por J_1 (Equação 3.3), pois, já que ambos os grafos são compostos apenas por um nó, a similaridade deve ser 1 caso sejam iguais e 0 caso sejam diferentes.

$$S = J_1. \tag{3.3}$$

Com esta métrica, nota-se que a similaridade entre os grafos da situação (b) da Figura 14 será zero. Logo, ela é capaz de indicar que os grafos da situação (a) são mais similares entre si do que os grafos da situação (b), que são completamente diferentes. Dessa forma, o problema do *Graph Edit Distance* exemplificado pela Figura 14 é resolvido, pois todos os vértices e arestas são levados em consideração (não apenas os que são diferentes). Além disso, esta métrica não apresenta os mesmos problemas de performance do *Graph Edit Distance*.

Utilizando a métrica apresentada, todos os grafos de estratégia de ataque podem ser comparados, gerando uma matriz de similaridade. A matriz de similaridade é uma matriz quadrada de tamanho $n \times n$, onde n é a quantidade de grafos de estratégias de ataque e o conteúdo da matriz é a similaridade (de 0 até 1) entre o grafo da coluna e o grafo da linha correspondente.

A Figura 15 exemplifica uma matriz de similaridade, a qual indica a similaridade entre os grafos denominados $G1$, $G2$, $G3$ e $G4$. Por meio desta matriz, pode-se identificar, por exemplo, que os grafos $G1$ e $G2$ são similares entre si e diferentes dos grafos $G3$ e $G4$. Nota-se que a diagonal da matriz é igual a 1, pois quando um grafo é comparado com ele mesmo, a similaridade é de 100%.

	G1	G2	G3	G4
G1	1	0.83	0	0
G2	0.83	1	0	0
G3	0	0	1	0.32
G4	0	0	0.32	1

Figura 15 – Exemplo de matriz de similaridade.

A matriz de similaridade é utilizada como entrada para a técnica de clusterização hierárquica com o método Ward, a qual visa agrupar todos os grafos similares [46, 47]. O método Ward tem como objetivo verificar o erro interno dos clusters para auxiliar a clusterização hierárquica nos passos de agrupamento. Considerando z_{ik} os elementos de um cluster k (que possui n_k elementos), e considerando $\bar{z}_k$ a média dos elementos, o erro E_k do cluster k é dado pela Equação 3.4.

$$E_k = \sum_{i=1}^{n_k} \parallel z_{ik} - \bar{z}_k \parallel^2 . \tag{3.4}$$

Como, neste trabalho, os elementos de um cluster k são grafos, não é possível encontrar a média $\bar{z}_k$ dos elementos. Dessa forma, a Equação 3.4 não pode ser aplicada. Utilizando a prova apresentada em [50], a Equação 3.4 pode ser reescrita como a soma de todas as distâncias ao quadrado entre os pares de elementos, dividido pelo número de elementos n (Equação 3.6). Neste caso, foi utilizada a dissimilaridade ao invés de distâncias entre os elementos. Logo, considerando S_{ij} o valor de similaridade entre os elementos i e j dado pela matriz de similaridade, o valor de dissimilaridade D_{ij} entre tais elementos é adquirido por meio da Equação 3.5.

$$D_{ij} = 1 - S_{ij}. \tag{3.5}$$

Então, é possível utilizar a Equação 3.6 para o cálculo do erro interno E_k de um cluster, onde D_{ij} é o valor de dissimilaridade entre os elementos i e j, e n é o número de elementos do cluster.

$$E_k = \frac{1}{n} \sum_{i} \sum_{j>i} D_{ij}^2. \tag{3.6}$$

Por fim, utilizando E_k, definido pelas Equações 3.4 e 3.6, o método Ward calcula o erro total E dos clusters para todos os g clusters por meio da Equação 3.7.

$$E = \sum_{k=1}^{g} E_k. \tag{3.7}$$

A clusterização hierárquica aglomerativa começa com n clusters que contêm um único elemento (onde n é o número de elementos). Após isso, realiza $n - 1$ passos de agrupamento (*merging steps*). Em cada passo de agrupamento, são mesclados os dois clusters que geram o menor aumento no erro total E (obtido com o método Ward, Equação 3.7). O método termina com apenas um cluster contendo todos os n elementos. O resultado dos passos de agrupamento pode ser representado por um dendrograma. Um dendrograma é uma árvore que representa todo o processo da clusterização hierárquica (Figura 16). A altura ideal de corte no dendrograma para definir a quantidade de clusters é dependente do problema [51, 52]. Portanto, neste trabalho, a altura do corte é definida como o valor do terceiro quartil do conjunto ordenado de alturas do dendrograma, pois o terceiro quartil separa os 75% das menores alturas dos 25% das maiores alturas [53].

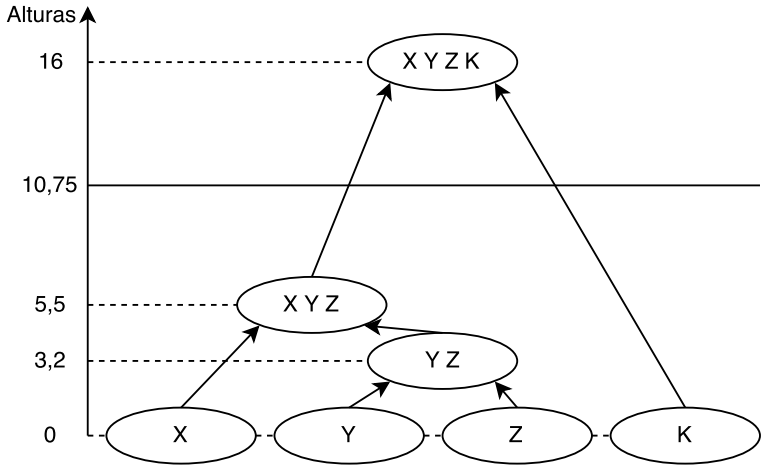


Figura 16 – Exemplo de dendrograma para a clusterização hierárquica aglomerativa com o método Ward.

A Figura 16 mostra um exemplo da técnica de clusterização hierárquica utilizada, onde cada altura representa a raiz quadrada do erro total E dos clusters [47]. Neste exemplo, o número de elementos é quatro. Logo, três $(n - 1)$ passos de agrupamento são necessários. O primeiro passo de agrupamento une os clusters Y e Z, formando o novo cluster YZ. O segundo passo de agrupamento une os clusters X e YZ, formando o novo cluster XYZ. O último passo de agrupamento une os clusters XYZ e K, formando o novo cluster $XYZK$. A altura de corte 10,75 seria utilizada, pois é o terceiro quartil das alturas 0, 3,2, 5,5 e 16. Dessa forma, dois clusters seriam gerados: o primeiro cluster composto pelos elementos X, Y e Z e o segundo cluster composto pelo elemento K.

Neste trabalho, os elementos X, Y, Z e K são grafos de estratégias de ataque. Portanto, este resultado indicaria que os grafos de estratégias de ataque X, Y e Z são

similares entre si (formando um padrão de estratégia de ataque), e diferentes do grafo K (formando outro padrão de estratégia de ataque). Um padrão de estratégia de ataque é definido pelas características de um cluster que reúne estratégias similares.

Com isso, uma visão generalizada de cada padrão de estratégia de ataque presente no histórico de alertas de IDS é obtida com a correlação *offline*. Os padrões de estratégia de ataque identificados mostram, para um dado período de histórico de alertas, como os atacantes se comportaram ao tentarem atacar a rede. Eles são utilizados para auxiliar o correlacionador *online* na análise dos alertas futuros e construção dos hiperalertas.

3.2 Correlacionador *Online*

Uma das principais diferenças do correlacionador *online* para o correlacionador *offline* está na forma em que os dados são apresentados para o correlacionador. No *offline*, todo um conjunto histórico de alertas é analisado de uma só vez. Já no *online*, os alertas são apresentados um a um, conforme são gerados pelo IDS. O objetivo do correlacionador *online* é organizar esses alertas em cenários de ataque e, para cada cenário de ataque, formar um hiperalerta com atributos que o descrevam. Dessa forma, o problema do estudo de vários alertas é simplificado para o estudo de apenas alguns hiperalertas.

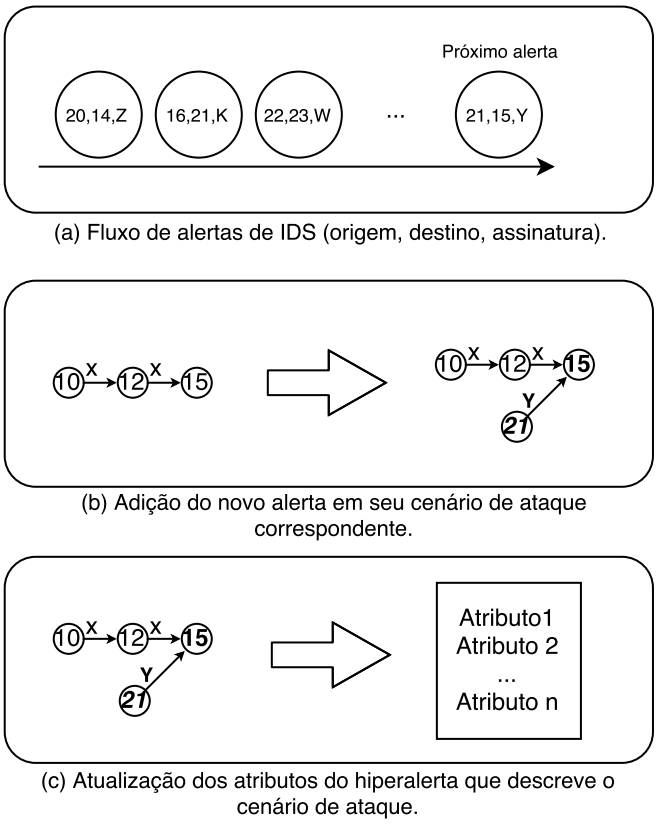


Figura 17 – Visão geral do correlacionador *online*.

A Figura 17 mostra o funcionamento geral do correlacionador *online*. A Figura 17a ilustra a chegada de um novo alerta A com endereço IP de origem representado pelo número 21, endereço IP de destino representado pelo número 15 e assinatura Y. A Figura 17b mostra a inclusão desse alerta no cenário de ataque apropriado. Por fim, a Figura 17c representa a atualização de todos os atributos do hiperalerta que representa o cenário de ataque atualizado. Como o correlacionador opera de forma *online*, nota-se que um cenário de ataque pode receber novos alertas constantemente. Portanto, as informações de um hiperalerta são atualizadas sempre que um novo alerta é adicionado em seu cenário de ataque.

As informações de um cenário de ataque são identificadas e representadas por meio de um hiperalerta com quatro categorias de atributos: atributos gerais, atributos de cluster associado, atributos de *hosts* atacantes e alvos e atributos de estratégia de ataque. A Figura 18 mostra um hiperalerta A com informações sobre um cenário de ataque A. Cada uma dessas categorias é explicada nos próximos parágrafos.

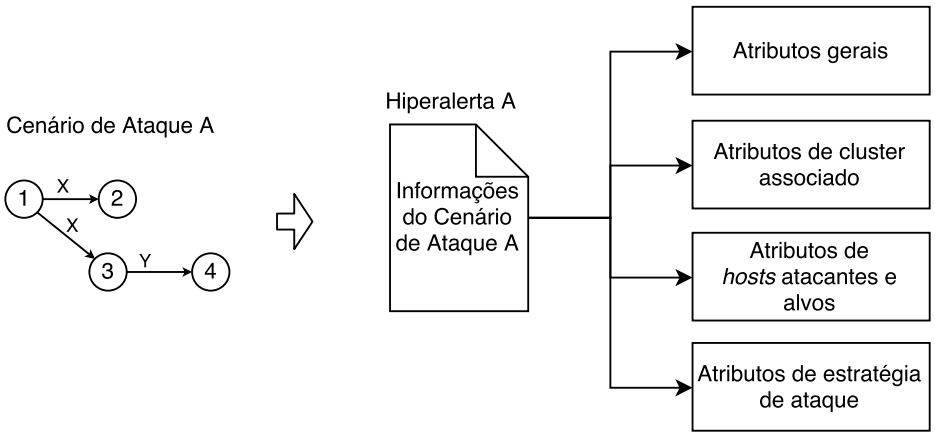


Figura 18 – Hiperalerta com informações sobre o cenário de ataque A.

Atributos gerais. Esta categoria de atributos do hiperalerta descreve as informações gerais de um cenário de ataque. Os atributos nesta categoria são: situação, tempo inicial, tempo final, duração e quantidade de alertas. O atributo de situação recebe o valor de "concluído" se nenhum alerta for adicionado no cenário de ataque em um período de tempo maior que x, caso contrário a situação é considerada "em formação". Se o cenário de ataque representado pelo hiperalerta está concluído, então isto indica que ele está em sua forma final (não será mais atualizado com novos alertas). Os atributos de tempo inicial e final informam o *timestamp* do primeiro e último alerta adicionado no cenário de ataque até então. O atributo de duração indica a diferença de tempo entre o tempo inicial e o tempo final. Tais atributos temporais podem ser úteis, pois, por meio deles, é possível perceber padrões no comportamento do atacante, como, por exemplo, sempre realizar ataques em certos períodos do dia ou em certos dias da semana. Já o atributo

de quantidade de alertas, representa a quantidade de alertas que compõem o cenário de ataque. Este atributo pode ser útil para verificar quais cenários de ataque são responsáveis pela maior quantidade de alertas.

Atributos de cluster associado. Esta categoria de atributos do hiperalerta interage com os resultados encontrados pelo correlacionador *offline*. Por meio dos atributos dessa categoria, os hiperalertas são associados aos padrões de estratégia de ataque identificados na correlação *offline*. Já que cada cluster identificado pela fase *offline* representa um padrão de estratégia de ataque tipicamente utilizados pelos atacantes contra a rede, a associação do hiperalerta a um desses clusters permite identificar a natureza do ataque que está ocorrendo no momento presente. Dessa forma, a associação busca facilitar a definição das medidas a serem tomadas pelos analistas de segurança. Os atributos nesta categoria são: melhor cluster e aumento de erro. O melhor cluster indica qual é o cluster identificado na fase *offline* que apresenta o padrão de estratégia de ataque mais similar à estratégia de ataque do hiperalerta em questão. Para isso, sabendo que os elementos de cada cluster da fase *offline* são grafos de estratégia de ataque, o correlacionador *online* aplica o método Ward para verificar o erro interno de cada um dos clusters da fase *offline*. Em seguida, aplica o método Ward para verificar o erro interno de cada um dos clusters com a adição de um novo elemento: a estratégia de ataque do hiperalerta em questão. Dessa forma, a diferença entre os erros internos do cluster com e sem o novo elemento é calculada e denominada de aumento de erro. O cluster que gerar o menor aumento de erro ao receber a nova estratégia de ataque é aquele escolhido como melhor cluster para o hiperalerta. O atributo de aumento de erro indica a força de associação entre a estratégia de ataque do hiperalerta e do cluster mais similar. Portanto, quanto maior o aumento de erro, mais fraca é a associação.

Atributos de *hosts* atacantes e alvos. Esta categoria de atributos tem como objetivo retratar a interação dos *hosts* envolvidos no cenário de ataque. Portanto, os atributos que compõem essa categoria são: lista de endereços IP atacantes, lista de endereços IP alvos, quantidade de atacantes distintos, quantidade de alvos distintos, lista de relações, média de atacantes por alvo, média de alvos por atacante e cenário de ataque representado como componente conexo. A lista de endereços IP atacantes reúne todos os endereços IP atacantes presentes no cenário de ataque e a quantidade de alertas nos quais cada um desses endereços IP apareceram como atacantes. A lista de endereços IP alvos é similar à lista de endereços IP atacantes, porém para alvos. A quantidade de atacantes distintos e quantidade de alvos distintos representam a contagem de endereços IP atacantes distintos e endereços IP alvos distintos do cenário de ataque. O atributo de lista de relações denota uma lista, na qual cada elemento representa uma relação diferente de endereços IP. Cada relação é formada por um determinado endereço IP de origem, um determinado endereço IP de destino e a quantidade de alertas nos quais esses dois endereços aparecem juntos no cenário de ataque. Este atributo pode ser útil para indicar quais *hosts* atacantes mais

geram alertas contra determinados *hosts* alvos. A média de atacantes por alvo é dada pela Equação 3.8. Já a média de alvos por atacantes é dada pela Equação 3.9. Estes dois atributos indicam a característica do ataque. Se a média de alvos por atacante é alta, o ataque tem, por exemplo, características de reconhecimento. Por outro lado, a média alta de atacantes por alvo pode ser indício de um ataque de DDoS. Por fim, o atributo de componente conexo, definido na Seção 3.1, refere-se a uma representação visual do cenário de ataque.

$$\textit{Média de atacantes por alvo} = \frac{\textit{Quantidade de relações distintas}}{\textit{Quantidade de alvos distintos}}. \tag{3.8}$$

$$\textit{Média de alvos por atacante} = \frac{\textit{Quantidade de relações distintas}}{\textit{Quantidade de atacantes distintos}}. \tag{3.9}$$

Atributos de estratégia de ataque. Esta categoria de atributos do hiperalerta tem como objetivo descrever e representar a estratégia de ataque do cenário de ataque. Esta categoria é composta pelos seguintes atributos: lista de assinaturas, quantidade de assinaturas distintas e grafo de estratégia de ataque. O atributo de lista de assinaturas denota uma lista, cujos elementos representam os nomes das assinaturas e a quantidade de alertas do cenário de ataque nos quais cada uma delas apareceu. O grafo de estratégia de ataque, definido na Seção 3.1, é a representação visual da estratégia de ataque presente no cenário de ataque. Os atributos desta categoria buscam revelar os passos do atacante na rede, conforme são realizados.

O funcionamento do correlacionador *online* é detalhado a seguir. No correlacionador *online*, assim como no correlacionador *offline*, um alerta é representado por uma 4-tupla (s, d, t, k), onde s é o endereço IP de origem, d é o endereço IP de destino, t é o *timestamp* e k é a assinatura do alerta. Já um cenário de ataque C é representado por um conjunto de alertas que possuem endereços IP relacionados e ocorreram próximos no tempo.

Para cada novo alerta A_i, o correlacionador *online* identifica se existem um ou mais cenários de ataque que possuem pelo menos um alerta A_j que ocorreu próximo no tempo e que apresenta endereço IP relacionado ao do alerta A_i. Essa verificação é realizada por meio da técnica de separação por componentes conexos, ou seja, se os dois alertas ocorrem próximos no tempo ($|A_i.t - A_j.t| < x$) e se $A_i.s = A_j.s$ ou $A_i.d = A_j.d$ ou $A_i.s = A_j.d$ ou $A_i.d = A_j.s$, então A_i pertence ao mesmo cenário de ataque do alerta A_j.

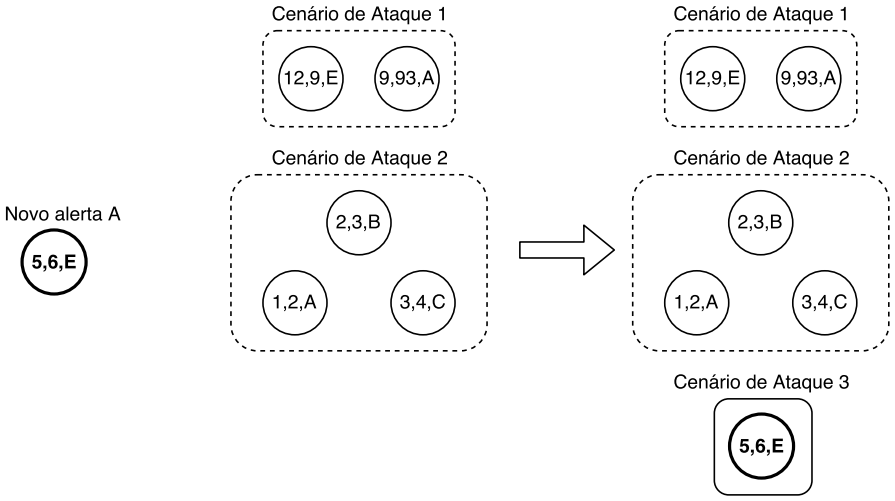


Figura 19 – Exemplo de agrupamento do novo alerta A quando nenhum cenário de ataque relacionado é encontrado.

Se nenhum cenário de ataque for identificado pela técnica de separação por componentes conexos, um novo cenário de ataque contendo A_i é criado. Considerando que todos os alertas ocorreram próximos no tempo, a Figura 19 mostra um exemplo de agrupamento do novo alerta A quando nenhum cenário de ataque relacionado é encontrado (não satisfaz nenhuma das condições citadas anteriormente). Nesse caso, o novo Cenário de Ataque 3 foi criado para acomodar o alerta A.

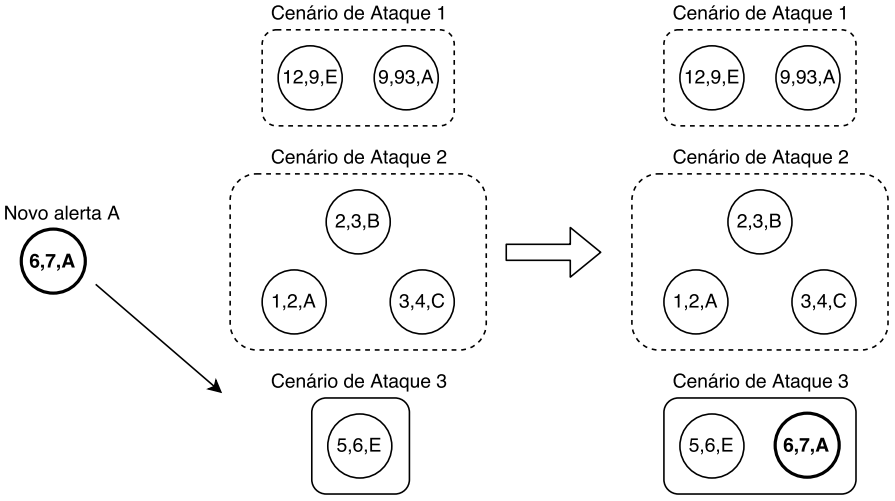


Figura 20 – Exemplo de agrupamento do novo alerta A quando apenas um cenário de ataque relacionado é encontrado.

Se apenas um cenário de ataque for identificado pela técnica de separação por componentes conexos, o alerta é adicionado neste cenário. Considerando que todos os alertas ocorreram próximos no tempo, a Figura 20 mostra um exemplo do novo alerta

A, cujo endereço IP de origem (representado pelo número 6) é igual ao endereço IP de destino de um alerta pertencente ao Cenário de Ataque 3. Dessa forma, é satisfeita a condição $A_i.s = A_j.d$, e o alerta A é adicionado no Cenário de Ataque 3.

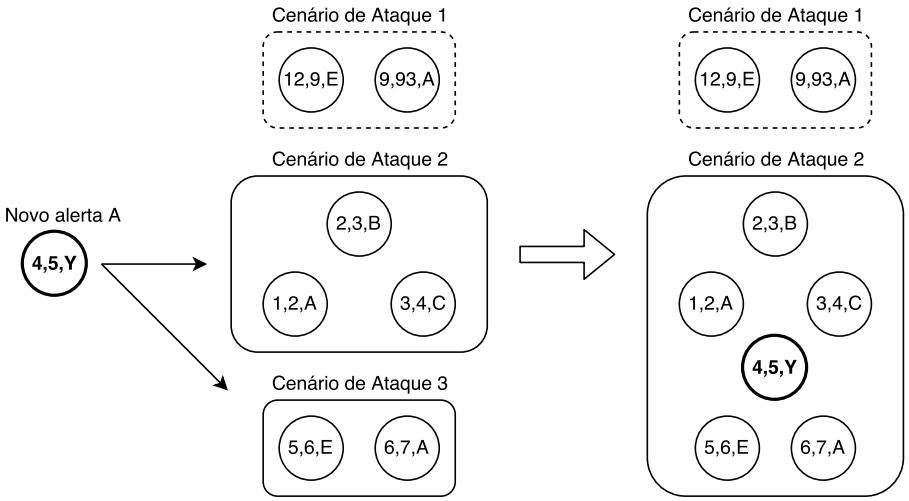


Figura 21 – Exemplo de agrupamento do novo alerta A quando dois cenários de ataque relacionados são encontrados.

Se mais de um cenário de ataque é identificado, eles são agrupados em um novo cenário de ataque, e o alerta A_i é adicionado neste novo cenário. Considerando que todos os alertas ocorreram próximos no tempo, a Figura 21 mostra um exemplo do novo alerta A, cujo endereço IP de origem (representado pelo número 4) é igual ao endereço IP de destino de um alerta pertencente ao Cenário de Ataque 2 (satisfazendo a condição $A_i.s = A_j.d$). Além disso, apresenta endereço IP de destino (representado pelo número 5) igual ao endereço IP de origem de um alerta pertencente ao Cenário de Ataque 3 (satisfazendo a condição $A_i.d = A_j.s$). Portanto, o Cenário de Ataque 2 e o Cenário de Ataque 3 são agrupados em um mesmo cenário. Por fim, o novo alerta A é adicionado, resultando no novo Cenário de Ataque 2.

Cada cenário de ataque é descrito por um hiperalerta. Portanto, ao determinar a qual cenário de ataque este novo alerta pertence, todos os atributos do hiperalerta que descreve este cenário de ataque são atualizados. Como o grafo de estratégia de ataque é um atributo do hiperalerta, este é atualizado com o novo alerta A. Depois, utilizando o método Ward, o correlacionador *online* faz uma nova associação do grafo de estratégia de ataque atualizado com um dos clusters gerados pelo correlacionador *offline*. Dessa forma, os atributos de *melhor cluster* e *erro interno do cluster* do hiperalerta são atualizados.

A Figura 22 mostra um exemplo de atualização do atributo de melhor cluster, onde um cenário de ataque recebe um alerta com endereço IP de origem representado pelo número 21, endereço IP de destino representado pelo número 15 e assinatura Y.

Então, o atributo de grafo de estratégia de ataque é atualizado com a nova assinatura Y (Figura 22b). Por fim, utilizando o método Ward, é feita uma nova associação do novo grafo de estratégia de ataque com o Cluster 2 (Figura 22c), identificado anteriormente pelo correlacionador *offline*. Por meio desta associação, nota-se que, ao receber novos alertas, a estratégia de ataque do cenário em questão tem suas características cada vez mais próximas das características da estratégia de ataque mostrada pelo Cluster 2, antecipando um possível ataque com a assinatura Z.

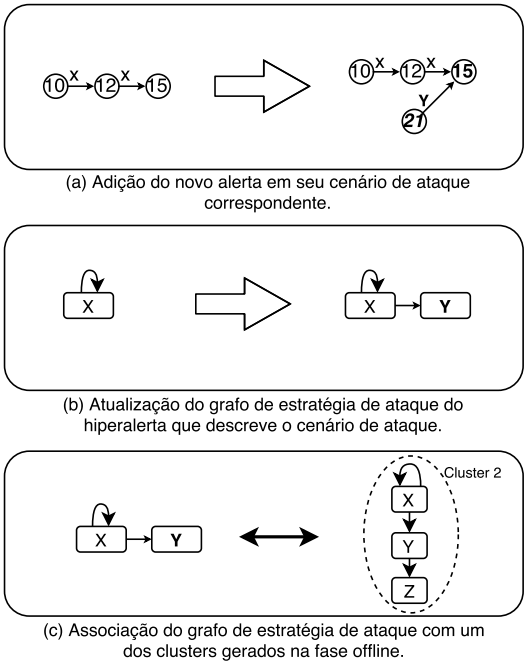


Figura 22 – Exemplo de atualização do atributo de melhor cluster: associação do grafo de estratégia de ataque com um cluster encontrado pela fase *offline*.

Como resultado, o correlacionador *online* cria e sempre mantem os hiperalertas atualizados. Cada hiperalerta oferece uma descrição detalhada de cada cenário de ataque por meio de seus atributos. Dessa forma, uma melhor consciência situacional sobre a rede monitorada é alcançada com essas novas informações, o que pode auxiliar na velocidade de reação contra os ataques realizados na rede.

4 EXPERIMENTOS E RESULTADOS

Para a aplicação da solução proposta, os alertas de IDS precisam apresentar quatro atributos: *timestamp*, endereço IP de origem, endereço IP de destino e tipo do ataque. Os alertas de IDS utilizados nos experimentos foram gerados por um IDS comercial baseado em rede implantado na rede da Universidade de Maryland. Dois experimentos foram realizados. O primeiro experimento (apresentado na Seção 4.2) foi realizado para analisar o comportamento geral da solução proposta. Dessa forma, pretende-se verificar se o correlacionador *offline* é capaz de identificar padrões de estratégias de ataque em meio a milhares de alertas. Além disso, procura-se observar como as informações dos hiperalertas gerados pelo correlacionador *online* podem ser interpretadas. Por fim, busca-se verificar se o comportamento dos atacantes é cíclico, fazendo com que os padrões de estratégia de ataque descobertos no passado se repitam no futuro. O segundo experimento (apresentado na Seção 4.3) investiga como o tamanho da amostra de alertas utilizadas no correlacionador *offline* afeta as associações feitas pelo correlacionador *online*.

4.1 Ferramentas Utilizadas

Os experimentos descritos nessa seção foram realizados no sistema operacional OS X 10.10.5, em um computador Intel Core i5 com 8GB de memória. Os correlacionadores *offline* e *online* foram implementados utilizando a linguagem de programação Java[1], versão 1.8.0_20. A linguagem de programação R[2], versão 3.1.0, foi utilizada para realizar a clusterização hierárquica aglomerativa com o método Ward. A clusterização foi feita utilizando a função *hclust* do pacote *stats* versão 3.1.0.

Como entrada para os correlacionadores, foram utilizados os alertas de uma base de dados gerados por um IDS comercial implantado em uma rede de aproximadamente 40.000 computadores, durante o ano de 2012, na Universidade de Maryland. Os alertas desta base de dados apresentam os atributos necessários para seu processamento nos correlacionadores *offline* e *online*: *timestamp*, endereço IP de origem, endereço IP de destino e tipo da intrusão. Por questões de confidencialidade, os endereços IP de origem e destino foram substituídos por um código *hash* e, na descrição deste experimento, representados como valores inteiros. Os alertas foram armazenados no sistema de gerenciamento de banco de dados MySQL[3].

Como saída, o correlacionador *offline* apresenta os padrões de estratégia de ataque, formado por grafos de estratégias de ataques similares. Cada grafo, no formato de lista

[1] https://www.oracle.com/br/java/index.html
[2] https://www.r-project.org/
[3] https://www.mysql.com/

de arestas, foi armazenado em um arquivo texto, a fim de ser utilizado posteriormente no correlacionador *online*. Além disso, um arquivo de imagem no formato JPEG foi gerado para cada grafo de estratégia de ataque e para cada padrão de estratégia de ataque, a fim de auxiliar a avaliação dos resultados. Por sua vez, o correlacionador *online* tem como saída os hiperalertas, os quais apresentam diversos atributos descritos na Seção 3.2. Um arquivo texto foi utilizado para armazenar cada hiperalerta, contendo a descrição de todos os seus atributos. Arquivos de imagem no formato JPEG também foram gerados para representar visualmente os atributos de componente conexo e grafo de estratégia de ataque de cada hiperalerta.

4.2 Primeiro Experimento

Este experimento buscou avaliar a saída dos correlacionadores propostos. Portanto, foram discutidos os principais clusters identificados pelo correlacionador *offline* e os principais hiperalertas gerados pelo correlacionador *online*. Primeiramente, a solução proposta foi aplicada a um conjunto de alertas gerados no mês de julho de 2012, pois este mês apresenta uma quantidade de alertas que permitiu uma melhor interpretação de resultados. A avaliação baseou-se na análise manual dos padrões de estratégia de ataque e dos hiperalertas gerados. Em seguida, foi realizado outro teste semelhante com os dados de maio de 2012, cuja quantidade de alertas é maior do que a quantidade de alertas gerados no mês de julho de 2012.

4.2.1 Dados do mês de julho

Os alertas do mês de julho foram separados em duas partes. Os alertas gerados nos primeiros 14 dias de julho de 2012 foram utilizados como entrada para o correlacionador *offline* a fim de encontrar os padrões de estratégia de ataque que ocorreram nesses 14 dias. Em seguida, os alertas gerados nos demais dias do mês (do dia 15 até o dia 31 de julho de 2012) foram utilizados para testar o correlacionador *online*. A diferença máxima de tempo entre dois alertas dentro de um cenário de ataque foi definida pelo limiar $x = 60$ minutos para os correlacionadores *offline* e *online*.

Um total de 20.509 alertas serviu de entrada para o correlacionador *offline*. O passo de agregação separou esses 20.509 alertas em 895 cenários de ataque. Então, os cenários de ataque que apresentaram situações excepcionais foram removidos. Um cenário de ataque é considerado excepcional caso contenha somente alertas com a mesma assinatura. Essas situações mostram um fluxo de ataque com apenas uma assinatura e não produzem informações suficientes para definir o comportamento de um atacante. Um exemplo de cenário de ataque considerado excepcional é composto por dezenas de alertas, todos com a assinatura de *Portscan*. Esta filtragem é opcional e foi realizada pois o foco deste experimento é a análise das estratégias de ataque com mais de uma assinatura. Com esta

filtragem, 3.796 alertas foram removidos, sobrando 16.713 alertas. Esses 16.713 alertas foram organizados em 57 cenários de ataque. Para cada um dos 57 cenários de ataque, um grafo de estratégia de ataque foi gerado e a matriz de similaridade de tamanho 57×57 foi calculada utilizando a métrica baseada no índice de Jaccard. Por fim, os 57 grafos de estratégia de ataque foram separados em clusters utilizando a técnica de clusterização hierárquica.

Um total de 12 clusters foi gerado, indicando que 12 padrões de estratégia de ataque diferentes foram utilizados nos 14 primeiros dias de julho de 2012. Portanto, o problema da análise de 16.713 alertas com informações elementares foi simplificado para a análise de 12 clusters com informações mais significativas. Os dois clusters (Cluster 1 e Cluster 7) que representam a maioria dos alertas foram escolhidos para serem explicados neste experimento. O Cluster 1 cobre 14.138 alertas (84,59% dos 16.713 alertas) e é composto por 10 estratégias de ataque similares. O Cluster 7 cobre 2.317 alertas (13,86% dos 16.713 alertas) e possui 12 estratégias de ataque similares. Isso significa que 98,46% dos 16.713 alertas são representados por esses dois clusters.

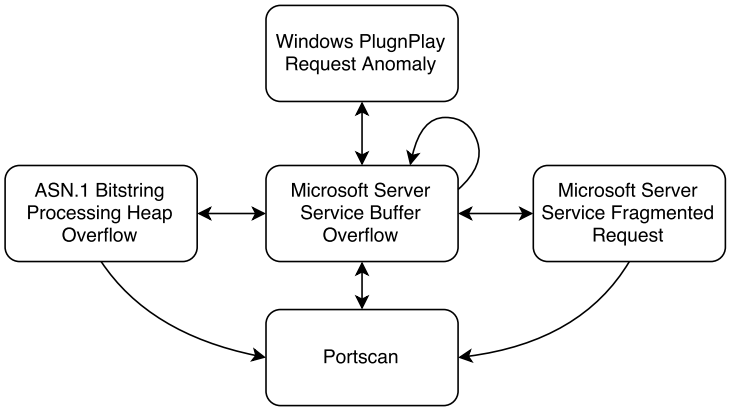


Figura 23 – Exemplo de estratégia de ataque pertencente ao Cluster 1.

O Cluster 1 é caracterizado pelo padrão de ataques de *buffer overflow*. Um *buffer overflow* é uma anomalia que ocorre quando um programa ultrapassa o limite do *buffer* e sobrescreve os locais de memória adjacentes. Esta anomalia pode ser utilizada por atacantes para prejudicar um sistema. A Figura 23 mostra um exemplo de grafo de estratégia de ataque pertencente ao Cluster 1. Nesta figura, nota-se a presença de cinco assinaturas. A assinatura *Microsoft Server Service Buffer Overflow* ocorre quando um atacante realiza ataques de *buffler overflow* contra o serviço *Microsoft Server*. Já a assinatura *ASN.1 Bitstring Processing Heap Overflow* ocorre quando um ataque de *buffler overflow* é realizado contra a biblioteca *Microsoft ASN.1*. O ASN.1 é um padrão internacional utilizado para descrever e transmitir pacotes de dados entre aplicações pela rede. Por sua vez, a assinatura *Windows PlugnPlay Request Anomaly* ocorre quando um atacante realiza ataques de *buffler overflow* contra o serviço *Microsoft Plug and Play*. Este serviço permite que

o sistema operacional detecte automaticamente um novo *hardware*, quando este é instalado no sistema. Quando o ataque de *buffer overflow* é realizado com sucesso contra um desses serviços vulneráveis, o atacante torna-se capaz de executar remotamente códigos maliciosos com privilégios de sistema e adquirir o controle completo do sistema. Dessa forma, o atacante pode instalar programas, ver, alterar ou remover dados, criar contas com privilégio total e realizar ataques de negação de serviço. O ataque contra o serviço *Microsoft Server*, por exemplo, ocorre por meio de portas específicas (139 e 445). Possivelmente, foi por esse motivo que o *Portscan* também foi utilizado, o qual é responsável por identificar quais portas estão ativas na rede. Por fim, a assinatura *Microsoft Server Service Fragmented Request* refere-se a tentativas de ataques de negação de serviço por meio de requisições inválidas [54, 55, 56, 57].

Já o Cluster 7 é caracterizado pelo padrão de ataques de reconhecimento. Neste cluster, nota-se a presença de ataques de reconhecimento como o *Possible Nmap Scan (XMAS (FIN PSH URG))*, que realiza varredura de portas e o *Fingerprinting Probe*, o qual tenta identificar o sistema operacional do alvo. A Figura 24 mostra um exemplo de grafo de estratégia de ataque pertencente ao Cluster 7.

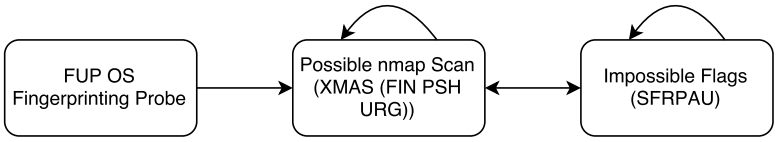


Figura 24 – Exemplo de estratégia de ataque pertencente ao Cluster 7.

Para representar visualmente o padrão de estratégia de ataque, é feita a união de todos os grafos de estratégia de ataque do cluster. A Figura 25 mostra o padrão de estratégia de ataque completo do Cluster 7.

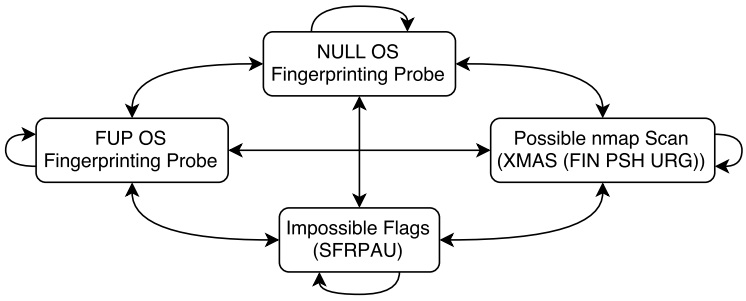


Figura 25 – Grafo de estratégia de ataque do Cluster 7.

Os alertas gerados do dia 15 até o dia 31 de julho de 2012 foram aplicados no correlacionador *online*. Neste período, 19.933 alertas foram gerados. Os alertas foram liberados um a um para testar o correlacionador *online*. Como explicado na Seção 3, o objetivo do correlacionador *online* é gerar e atualizar hiperalertas que descrevem os

cenários de ataques. Para tanto, verifica-se a qual cenário de ataque o novo alerta pertence. Após inserir o novo alerta no cenário de ataque correspondente, o hiperalerta que descreve este cenário tem seus atributos atualizados. Dessa forma, o grafo de estratégia de ataque deste hiperalerta é atualizado e reassociado a um dos clusters gerados pelo correlacionador *offline*.

Portanto, este é um processo no qual o cenário de ataque (e consequentemente o grafo de estratégia de ataque) e todos os demais atributos do hiperalerta são atualizados sempre que um novo alerta é adicionado. Logo, o grafo de estratégia de ataque de um hiperalerta pode ser associado à um cluster x no início e, após receber mais alertas, ele pode mudar sua associação para um outro cluster y, pois quanto mais alertas são usados para atualizar o hiperalerta, mais informações são disponíveis para a associação.

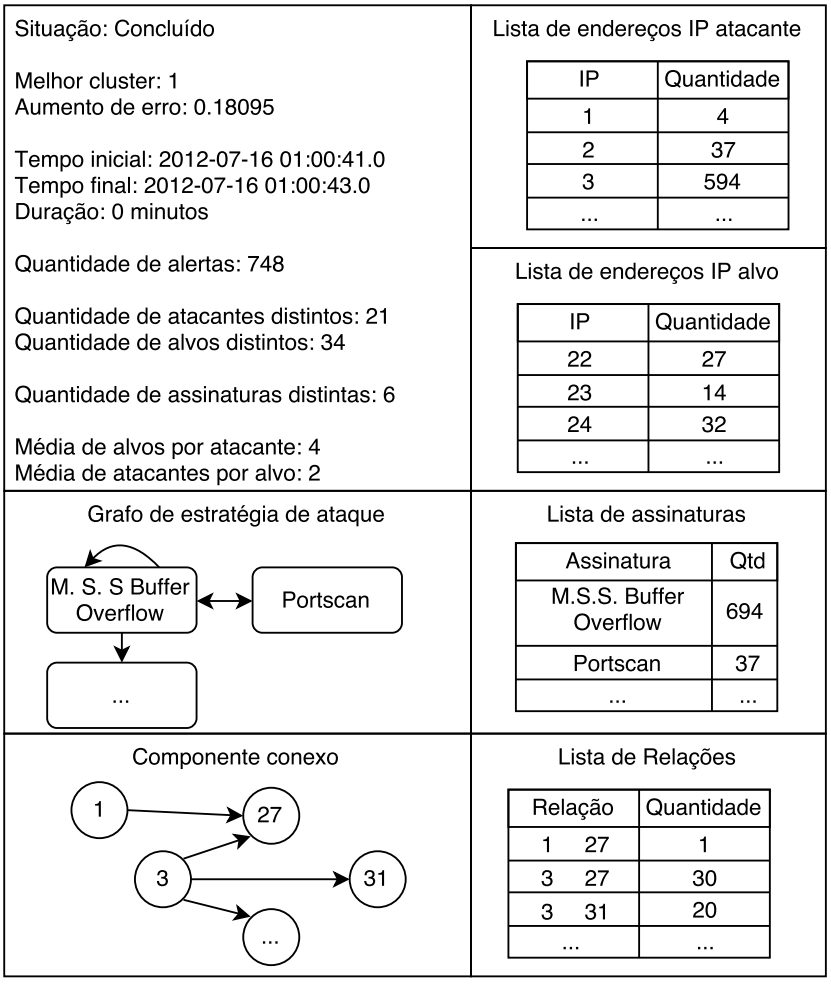


Figura 26 – Exemplo de hiperalerta associado ao Cluster 1.

Como a diferença máxima de tempo entre dois alertas dentro de um cenário de ataque foi definida pelo limiar $x = 60$ minutos, um hiperalerta é considerado *fechado* quando nenhum alerta é adicionado a ele durante 60 minutos. Consequentemente, nenhum alerta pode ser adicionado a um hiperalerta fechado. Quando o hiperalerta é considerado

fechado, significa que os atributos estão na sua forma final (não será mais modificado com atualizações). O cluster associado ao hiperalerta *fechado* é denominado de *cluster final*.

Durante os experimentos, os hiperalertas mostraram rápida convergência para encontrar o *cluster final*: foi observado que o hiperalerta é associado corretamente ao *cluster final* após uma média de 1,54 associações com um desvio padrão de 0,7. A média de associações realizadas em um hiperalerta até ele ser fechado é de 8,73 com um desvio padrão de 10,05 (o valor do desvio padrão ocorre devido à diferença do tamanho dos grafos de estratégia de ataque). Cada nova associação é realizada após a adição de um novo alerta que modifica o grafo de estratégia de ataque de um hiperalerta. Dessa forma, um hiperalerta formado apenas pelos primeiros alertas pode ser capaz de informar o padrão de estratégia de ataque utilizado pelo atacante.

Como resultado, um total de 59 hiperalertas foram identificados representando 14.099 alertas (assim como na fase *offline*, as situações excepcionais foram desconsideradas). Dezesseis dos cinquenta e nove hiperalertas foram associados ao Cluster 1 e possuíam entre 748 e 1209 alertas cada. A Figura 26 representa um dos 16 hiperalertas associados ao Cluster 1. Analisando o atributo de tempo dos hiperalertas associados ao Cluster 1, percebeu-se que todos os 16 hiperalertas foram gerados de noite no mesmo horário (um hiperalerta gerado a cada noite, entre os dias 15 e 31 de julho, com exceção do dia 29). Analisando os atributos de endereço dos 16 hiperalertas, identificou-se que o mesmo endereço IP foi responsável por gerar a maioria dos alertas. Ainda, analisando os atributos de assinatura, observou-se que a maioria dos alertas possui a assinatura *Microsoft Server Service Buffer Overflow*. Possivelmente, o atacante utilizou um *script* que busca explorar vulnerabilidades em máquinas Windows para realizar ataques de *buffer overflow* no mesmo horário em dias diferentes. Utilizando a abordagem proposta, um analista de segurança poderia ter identificado este padrão logo nos primeiros dias, reduzindo significativamente a quantidade de alertas futuros com a mitigação deste problema.

Treze hiperalertas foram associados ao Cluster 7 e possuem entre 6 e 32 alertas. Um padrão relevante foi encontrado nesses 13 hiperalertas: a presença de um único endereço IP atacante com múltiplos endereços IP alvo. A Figura 27 retrata a evolução de um hiperalerta associado ao Cluster 7, mostrando seus atributos de quantidade de alertas, cluster associado, cenário de ataque e grafo de estratégia de ataque. Os endereços IP dos cenários de ataque foram trocados para os rótulos de 1 até 8 por motivos de privacidade. Quando o hiperalerta possuía apenas 3 alertas, ele estava associado ao Cluster 2. Após a adição de mais um alerta, a associação do hiperalerta mudou para o Cluster 7 e permaneceu neste cluster até o fim (quando o hiperalerta foi fechado com 7 alertas). Este hiperalerta mostra um padrão quando seu componente conexo é analisado: apenas um endereço IP de origem está atacando outros endereços IP. A média de alvos por atacante é 7, que reforça a característica de ataque de reconhecimento. O grafo de estratégia de

ataque deste hiperalerta é similar ao grafo do Cluster 7 (Figura 25). Portanto, com 4 alertas foi possível inferir que este hiperalerta possuía o mesmo comportamento de ataques do Cluster 7 (ataques de reconhecimento).

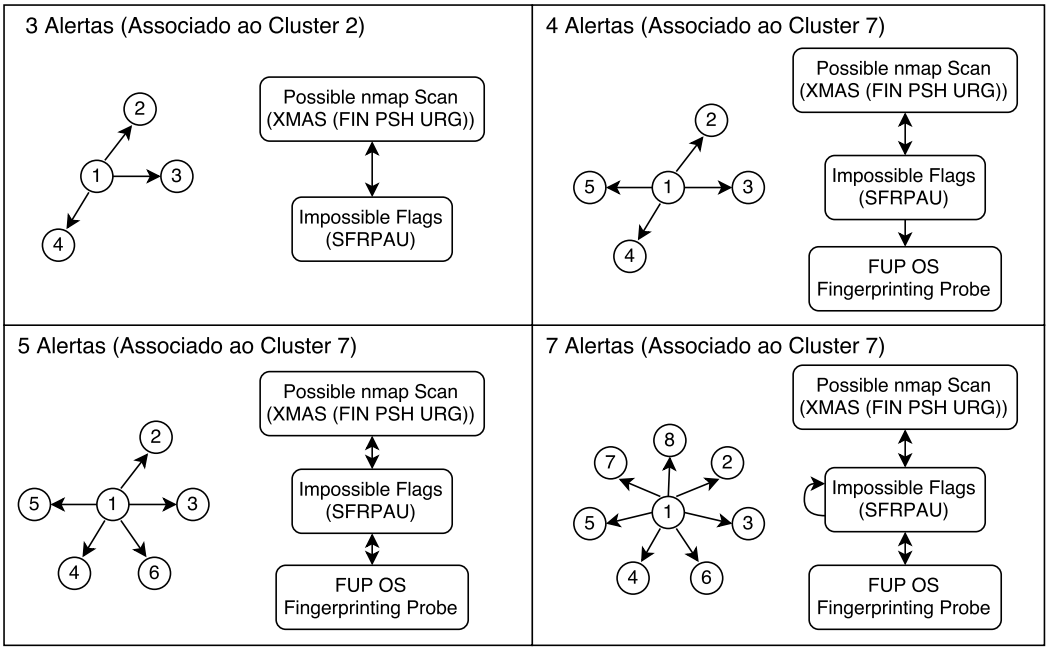


Figura 27 – Evolução de um hiperalerta associado ao Cluster 7.

Ainda, foi observado que 74,58% dos hiperalertas (44 dos 59) apresentaram um erro de associação de cluster menor do que 0,39 (terceiro quartil dos 59 erros de associação). O erro de associação é utilizado para medir o nível de dissimilaridade entre um padrão de estratégia de ataque (cluster) e uma estratégia de ataque de um hiperalerta. Quanto maior o erro de associação, maior é o nível de dissimilaridade entre a estratégia de ataque e o seu respectivo cluster associado. Esses 44 hiperalertas cobrem 95,52% dos alertas (13468 de 14099 alertas). Esse resultado indica que 74,58% dos hiperalertas (consequentemente 95,52% dos alertas) apresentaram estratégias de ataque similares aos padrões de estratégia de ataque já identificados na fase *offline*. A Figura 28 mostra um exemplo de estratégia de ataque associada ao Cluster 7 com erro de 0,42. Nota-se que, mesmo com erro acima de 0,39, a estratégia de ataque em questão ainda se assemelha ao Cluster 7 (Figura 25). Portanto, é reforçada a hipótese de que uma estratégia de ataque utilizada anteriormente tem chances de se repetir em ataques futuros.

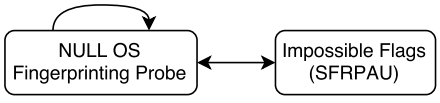


Figura 28 – Estratégia de ataque associada ao Cluster 7 apresentando erro acima de 0,39.

Desta forma, o correlacionador *offline* foi capaz de organizar os alertas históricos e identificar os padrões de estratégia de ataque por meio da técnica de clusterização. Assim, o problema da análise de uma grande quantidade de alertas foi simplificado para a análise de poucos clusters com informações relevantes. Os dois principais clusters, que representam a maioria dos alertas (98,46%), indicam que a maioria dos alertas foi gerada por apenas dois padrões de estratégia de ataque diferentes. Já o correlacionador *online* foi capaz de correlacionar os novos alertas, gerando os hiperalertas com informações que podem auxiliar na resposta contra as ameaças de segurança. Por meio da análise dos hiperalertas, foi possível encontrar padrões no comportamento dos ataques, o que não seria possível com a análise manual dos alertas não correlacionados. Além disso, examinando o cluster associado ao hiperalerta, foi possível identificar o padrão de estratégia de ataque utilizado pelo atacante com apenas os primeiros alertas.

4.2.2 Dados do mês de maio

O teste com os alertas de maio de 2012 foi realizado por dois principais motivos: (a) interpretar a saída do correlacionador com uma quantidade maior de alertas e (b) verificar se as estratégias de ataque também são repetitivas neste mês. Para isso, os alertas do mês de maio de 2012 foram separados da seguinte forma: os primeiros 7 dias com 50.516 alertas para o correlacionador *offline* e os últimos 24 dias com 282.293 alertas para o correlacionador *online*.

Os alertas utilizados na fase *offline* foram separados em 890 cenários de ataque. Os cenários que formavam grafos de estratégias de ataque de apenas um nó foram filtrados (situações excepcionais). Dessa forma, restaram 142 cenários de ataque, totalizando 22.129 alertas. Os 142 cenários de ataque foram separados em 18 clusters. Portanto, um total de 18 padrões de estratégia de ataque diferentes foi utilizado nos primeiros 7 dias do mês de maio. Dessa forma, o problema da análise de 22.129 alertas foi simplificado para a análise de 18 padrões de estratégia de ataque.

16.375 dos 22.129 alertas se concentram apenas no Cluster 17. Este cluster apresenta uma estratégia de ataque composta por duas assinaturas: 1) *The Onion Router (Directory Server)* e 2) *Certificate Exchange*. O *The Onion Router* (Tor) é um *software* que proporciona comunicação anônima ao utilizar a rede [4]. Tentativas de comunicação feitas por fontes anônimas como a rede Tor são consideradas suspeitas e são detectadas pelo IDS. Ambas as duas assinaturas são geradas devido à comunicação anônima do Tor e foram correlacionadas no mesmo cluster. A primeira assinatura é gerada quando o IDS detecta verificações nos servidores de diretório do Tor. Esta verificação é considerada suspeita, pois, por meio dela, a rede Tor oferece serviços escondidos, como acesso a domínios não reconhecidos por servidores oficiais de DNS. Já a segunda assinatura é gerada quando

[4] <https://www.torproject.org/about/overview.html.en>

o IDS detecta a troca de certificados que ocorre quando um cliente se conecta à rede Tor.

Já o correlacionador *online* separou 127.379 alertas em 1.483 cenários de ataque (assim como no correlacionador *offline*, as situações excepcionais foram desconsideradas). 925 dos 1.483 hiperalertas que representam esses cenários foram associados ao Cluster 17, com uma média de erros de 0,026 e desvio padrão de 0,034. Esses valores indicam que tais hiperalertas apresentam estratégias de ataque muito similares ao padrão de estratégia de ataque representado pelo Cluster 17. Esses 925 hiperalertas reúnem 123.518 alertas (96,97% dos 127.379 alertas). Portanto, considerando que os 925 cenários retratam problemas semelhantes, a solução de apenas um dos 925 cenários implicaria na possível solução para todos os 925 cenários. A Figura 29 mostra um exemplo de estratégia de ataque de um dos 925 hiperalertas associados ao Cluster 17. Este hiperalerta reúne 74 alertas e apresenta erro de associação igual a 0,042. Logo, mesmo com um erro acima da média, a estratégia de ataque em questão ainda se assemelha ao padrão de estratégia de ataque representado pelo Cluster 17.

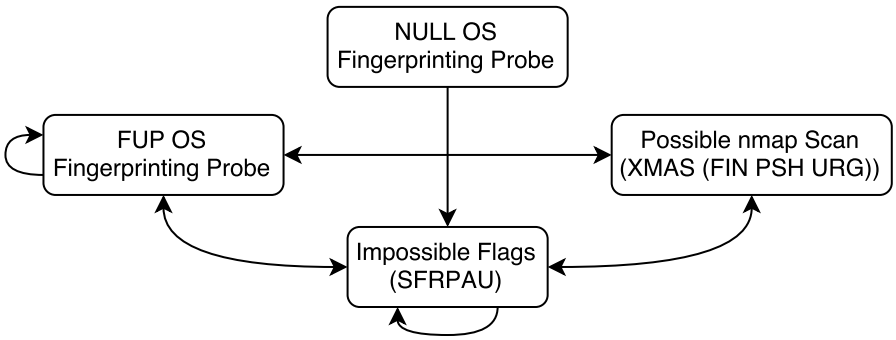
(a) Grafo do Cluster 17

(b) Estratégia de ataque associada ao Cluster 17 com erro igual à 0,042

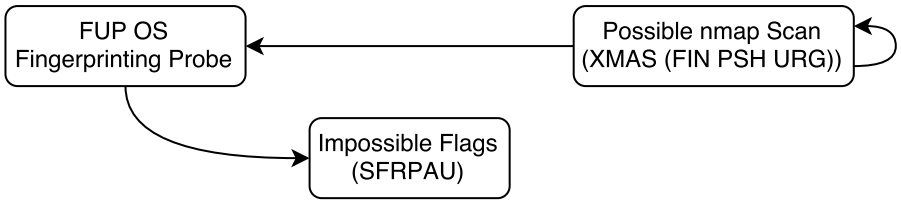
Figura 29 – Grafo de estratégia de ataque do Cluster 17 e um exemplo de estratégia de ataque associada a este cluster.

Dessa forma, restariam 3.861 dos 127.379 alertas para serem estudados. Desses 3.861 alertas, 1.707 formaram cenários de ataque cujos hiperalertas produziram erro acima de 0,140. A Figura 30 mostra um exemplo de estratégia de ataque de um hiperalerta associado ao Cluster 13, a fim de retratar uma associação com erro igual à 0,140. Portanto, alguns dos hiperalertas que produziram erros acima de 0,140 podem representar estratégias inéditas de ataque. Já os demais alertas formam estratégias similares a algum padrão já identificado pelo correlacionador *offline*, semelhante ao que ocorreu com os cenários cujos hiperalertas foram associados ao Cluster 17.

Ainda, observando a Figura 30, percebe-se que o Cluster 13 apresenta um padrão de estratégia de ataque semelhante a aquele visto nos testes da base de dados de julho de 2012, retratado pela Figura 25 da Subseção 4.2.1. Isso indica, novamente, uma repetição das estratégias de ataques, mesmo entre bases de dados de meses diferentes.



(a) Grafo do Cluster 13



(b) Estratégia de ataque associada ao Cluster 13 com erro igual à 0,140

Figura 30 – Grafo de estratégia de ataque do Cluster 13 e um exemplo de estratégia de ataque associada a este cluster com erro de 0,140.

4.3 Segundo Experimento – Variação no tamanho da amostra utilizada no correlacionador *offline*

Considera-se que, quanto menor o tamanho da amostra utilizada no correlacionador *offline*, menor é a quantidade de informação disponível para a identificação dos padrões de estratégia de ataque. Portanto, este experimento busca investigar como a variação no tamanho da amostra utilizada no correlacionador *offline* afeta a associação feita entre os padrões de estratégia de ataque (encontrados na fase *offline*) e os hiperalertas (gerados pelo correlacionador *online*).

A princípio, seis testes foram realizados. A Figura 31 mostra como os seis testes foram organizados: os alertas gerados em julho de 2012 foram separados em quatro semanas (A, B, C e D). Em todos os seis testes, a última semana (D) foi utilizada no correlacionador *online* para uma comparação justa entre todos os testes. Já para o correlacionador *offline*, o teste $1a$ utilizou três semanas (A, B, C). Os testes $1b$ e $1c$ utilizaram duas semanas: (A, B) e (B,C) respectivamente. Os outros testes utilizaram uma semana: (A), (B) e (C) respectivamente. Esta organização pretende testar como a quantidade e o período de geração dos alertas utilizados na fase *offline* afeta o erro de associação dos hiperaletas da fase *online*.

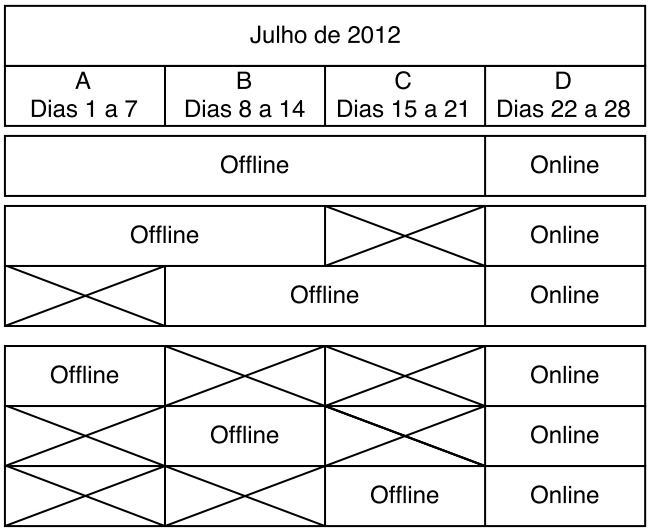


Figura 31 – Organização dos alertas de IDS de julho de 2012 para cada teste.

Como exemplo, o teste 1*a* foi realizado da seguinte forma: os 23.121 alertas capturados nas semanas A, B e C foram utilizados no correlacionador *offline* para descobrir os padrões de estratégia de ataque tipicamente utilizados pelos atacantes nesse período. Os 23.121 alertas foram separados em 91 cenários de ataque utilizando o método de separação por componentes conexos. Esses 91 cenários de ataques geraram 18 clusters de estratégias similares. Portanto, 18 padrões de estratégia de ataque foram identificados nas três semanas. Então, os alertas da semana D foram utilizados no correlacionador *online*. A semana D possui 5.908 alertas distribuídos em 10 cenários de ataque e, consequentemente, 10 hiperalertas. Utilizando o método Ward para computar o aumento de erro de associação (apresentado na Seção 3.2), cada um dos 10 hiperalertas foi associado ao padrão de estratégia de ataque mais similar. Um baixo aumento de erro indica que o hiperalerta poderia ser apropriadamente combinado com um dos 18 padrões de estratégia de ataque. Dessa forma, um erro de associação foi identificado para cada hiperalerta. A média dos erros de associação e seus desvios padrões para os 10 hiperalertas foram computados. Este processo foi realizado para todos os seis testes, gerando os resultados da Tabela 10.

Tabela 10 – Resultados para os seis testes.

Teste	**Semanas usadas no *offline***	**Contagem de Alertas**	**Contagem de cenários de ataque**	**Contagem de clusters gerados**	**Média dos dez erros de associação**	**Desvio padrão dos erros**
1a	ABC	23.121	91	18	0,15	0,08
1b	AB	16.713	57	13	0,10	0,07
1c	BC	12.659	65	16	0,12	0,08
1d	A	10.462	26	7	0,11	0,08
1e	B	6.251	31	10	0,11	0,07
1f	C	6.408	34	9	0,13	0,09

A Tabela 10 mostra os resultados para os seis testes. Para cada teste, esta tabela

mostra o nome do teste, a quantidade de alertas, a quantidade de cenários de ataques que os alertas foram separados e o número de clusters que os cenários de ataque foram distribuídos no correlacionador *offline*. As últimas duas colunas indicam a média e o desvio padrão dos erros de associação dos dez hiperalertas encontrados na semana D pelo correlacionador *online*.

A maior média de erros de associação foi identificada no teste $1a$, utilizando todas as três semanas (A, B, C) como entrada para o correlacionador *offline*. Isso significa que utilizar um período de tempo mais longo para a fase *offline* não necessariamente gera um erro de associação menor entre os hiperalertas e os padrões de estratégia de ataque.

Além disso, analisando os três últimos testes ($1d$, $1e$ e $1f$), o teste que utilizou apenas a semana A (alertas mais antigos) mostrou uma média de erros menor do que o teste utilizando a semana C (com alertas mais recentes). Portanto, foi observado que utilizar alertas mais recentes no correlacionador *offline* também não necessariamente gera erros de associação menores.

Nota-se que a diferença de erro entre os seis testes é baixa. Isso ocorre pois, dos 10 hiperalertas gerados pelo correlacionador *online*, 7 hiperalertas apresentavam o padrão de ataques de *buffer overflow* e 3 hiperalertas apresentavam o padrão de ataques de reconhecimento. Ambos os padrões estão presentes nas semanas A, B e C. Então, nenhum hiperalerta da semana D apresentou novos padrões de estratégia de ataque. Logo, todos os 10 hiperalertas puderam ser associados de forma apropriada com um dos padrões de estratégia de ataque com baixo erro de associação, sem depender da semana utilizada no correlacionador *offline*.

Devido aos erros baixos de associação para os seis testes anteriores, foram realizados mais dois novos testes ($2a$ e $2b$), utilizando apenas um ou dois dias, ao invés de semanas. O objetivo desses dois novos testes é utilizar uma quantidade pequena de alertas e verificar a influência do dia de geração dos alertas no erro de associação. O teste $2a$ foi realizado utilizando os alertas do primeiro dia de julho de 2012, totalizando 1.917 alertas mais antigos. O teste $2b$ foi realizado utilizando os alertas dos dias 20 e 21 de julho de 2012, totalizando 1.820 alertas mais recentes. Os dias entre 22 e 28 de julho foram utilizados no correlacionador *online* (os mesmos dias da semana D, utilizada nos seis testes anteriores).

A Tabela 11 mostra os resultados para ambos os testes. Foi observado que em ambos os testes, a fase *offline* identificou os padrões de ataque de *buffer overflow* e reconhecimento. Portanto, mesmo com uma quantidade menor de alertas no correlacionador *offline*, os resultados ainda continuam apresentando baixo erro de associação (0,2 para o primeiro dia e 0,14 para os dias de 20 a 21). Isso indica que o correlacionador proposto é capaz de refletir de forma apropriada o que ocorre nesta base de dados: a repetição dos ataques de *buffer overflow* e reconhecimento.

Tabela 11 – Resultados dos testes $2a$ e $2b$.

Teste	**Dias usados no *offline***	**Contagem de Alertas**	**Contagem de cenários de ataque**	**Contagem de clusters gerados**	**Média dos dez erros de associação**	**Desvio padrão dos erros**
2a	1	1.917	6	3	0,20	0,07
2b	20 e 21	1.820	11	4	0,14	0,04

Mesmo com uma pequena quantidade de alertas, o correlacionador *offline* conseguiu identificar os padrões de estratégia de ataque de forma apropriada. Devido ao comportamento repetitivo dos atacantes, os padrões identificados foram suficientes para antecipar a natureza do ataque de todos os próximos hiperalertas gerados pelo correlacionador *online*. Portanto, utilizando dados de uma grande rede de computadores, este experimento mostra que ter um conhecimento prévio das estratégias de ataque utilizadas no passado pode auxiliar a análise de alertas futuros.

4.4 Discussão Sobre os Resultados

O primeiro experimento foi dividido em dois testes. O primeiro teste foi realizado com a base de dados de julho de 2012, a qual apresenta uma quantidade menor de alertas quando comparada com a base de dados de maio de 2012 (utilizada no segundo teste). O teste realizado na base de dados de julho de 2012 mostrou os seguintes resultados. A fase *offline* resumiu um total de 16.713 alertas em apenas 12 padrões de estratégia de ataque, dos quais, apenas dois padrões de estratégia de ataque representavam 98,46% desses alertas. Os dois padrões de estratégia de ataque puderam ser interpretados, um indicando padrões de ataque de *buffer overflow* e outro indicando padrões de ataque de reconhecimento. Os hiperalertas identificados na fase *online* mostraram novas informações sobre os cenários de ataque por meio de seus atributos. Por exemplo, foram identificadas tentativas de ataque com o padrão de *buffer overflow* durante 16 dias, sempre no mesmo horário. O teste realizado com a base de dados de maio de 2012 buscou avaliar o correlacionador proposto utilizando uma maior quantidade de dados. A fase *offline* conseguiu resumir um total de 22.129 alertas em 18 padrões de estratégia de ataque. Novamente, a maioria dos alertas se concentravam em apenas um padrão de estratégia de ataque. Este padrão de estratégia de ataque pôde ser analisado e mostrou características relacionadas a acessos à rede de anonimato Tor. Já o correlacionador *online*, recebeu como entrada 127.379 alertas e os representou em 1.483 hiperalertas. A princípio, 1.483 é uma grande quantidade de hiperalertas para se analisar. Porém, 925 dos 1.483 hiperalertas foram associados ao padrão de ataque Tor, com uma média de erro de associação muito baixa. Isso indica que, a solução para um dos 925 hiperalertas poderia ser aplicada para a maioria dos 925 hiperalertas de comportamento semelhante. Nota-se que 96,97% dos alertas estão distribuídos nesses 925 hiperalertas. Portanto, o primeiro experimento mostra que o correlacionador *offline* foi capaz de simplificar o problema da análise de muitos alertas,

transformando-o na análise de apenas alguns padrões de estratégia de ataque. Além disso, mostra que o correlacionador *online* é capaz de representar os próximos alertas em uma quantidade menor de hiperalertas, os quais são estruturas de informação com atributos que auxiliam a análise de alertas de IDS.

O segundo experimento foi realizado para investigar como o tamanho da amostra de alertas utilizadas no correlacionador *offline* afeta as associações dos hiperalertas no correlacionador *online*. A princípio, seis testes foram realizados alterando a quantidade de semanas e o período de geração dos alertas utilizados na fase *offline*. A última semana do mês de julho sempre foi utilizada para o correlacionador *online* para uma comparação justa entre todos os testes. Os resultados para os seis primeiros testes mostraram que a diferença no erro de associação de todos os testes era baixa, independentemente da quantidade de semanas e do período em que os alertas foram gerados. Portanto, dois novos testes utilizando dias ao invés de semanas foram realizados. Novamente, ambos os testes produziram erros baixos de associação. Os resultados indicam um comportamento bastante repetitivo dos atacantes, pois os padrões de estratégia de ataque identificados na fase *offline* em todos os oito testes, utilizando dias ou semanas, foram suficientes para associar os hiperalertas identificados na fase *online*. Portanto, considerando os alertas gerados por um IDS implantado em uma grande rede de computadores, este experimento mostra que o conhecimento prévio das estratégias de ataque realizadas no passado pode auxiliar a análise de alertas futuros.

5 CONCLUSÃO

Para auxiliar a gerência de segurança da informação, este trabalho abordou o problema da correlação e análise de grandes quantidades de alertas de intrusão gerados por IDSs. Primeiramente, para melhor compreensão do estado da arte, foi realizado um estudo de trabalhos sobre análise de alertas de intrusão e suas técnicas foram organizadas em uma taxonomia. Por meio deste estudo, foram identificados dois principais problemas: a grande quantidade de alertas gerados pelos IDSs e o baixo nível de informação oferecido por eles. Para auxiliar na solução desses problemas, os autores têm perseguido quatro principais objetivos na área de análise de alertas de intrusão: reduzir a grande quantidade de alertas, organizar os alertas em cenários de ataque, identificar as estratégias de ataque e analisar as estratégias encontradas. Foi observado que tais objetivos podem ser combinados, formando um sistema mais completo. Porém, nota-se que a maioria dos trabalhos não explora todas as informações disponíveis em cenários de ataque e, além disso, utiliza bases de dados experimentais para a validação dos métodos propostos.

Para lidar com os desafios relacionados à grande quantidade de alertas gerados pelos IDSs e o baixo nível de informação oferecido, neste trabalho foi proposta uma nova abordagem para correlacionar alertas em duas fases: *offline* e *online*. Na fase *offline*, foi utilizado o método de separação por componentes conexos para agrupar um histórico de alertas em cenários de ataque. Então, foi possível identificar a estratégia de cada cenário de ataque e clusterizá-las, a fim de definir os padrões de estratégia de ataque. Na fase *online*, os alertas foram separados em cenários de ataque conforme foram gerados pelo IDS. Em seguida, para descrever cada cenário de ataque, uma estrutura de informação com diversos atributos, denominada de hiperalerta, foi gerada. Ainda, tendo como base a hipótese de que uma estratégia de ataque utilizada anteriormente tem chances de se repetir no futuro, cada hiperalerta foi associado a um dos padrões de estratégia de ataque já identificados e estudados na correlação *offline*. Esta associação foi realizada pois, caso o hiperalerta seja similar a um padrão já conhecido, as informações sobre esse padrão podem ser utilizadas para antecipar a solução do hiperalerta em questão.

A abordagem foi avaliada utilizando uma base de dados real de 2012 disponibilizada pela Universidade de Maryland. Os resultados do primeiro experimento mostraram que, para o mês de julho, a fase *offline* conseguiu resumir aproximadamente 17 mil alertas em apenas 12 padrões de estratégia de ataque. Já a fase *online* foi capaz de identificar informações sobre os cenários de ataque que antes não eram visíveis nos alertas não correlacionados. Por exemplo, foram identificadas tentativas de ataque com o padrão de ataque de *buffer overflow* durante 16 dias no mesmo horário. Este padrão de estratégia de ataque já havia sido identificado na fase *offline*, portanto, a sua solução já poderia ser conhecida

pelo analista de segurança. Para o mês de maio, descobriu-se um grande número de violações relacionadas a acessos à rede Tor. Ainda, nota-se que uma parte significativa dos hiperalertas foram associados a no máximo dois clusters, o que indica um comportamento bastante repetitivo dos atacantes nesses meses.

Um segundo experimento foi realizado com os alertas de julho de 2012 para verificar como a variação no tamanho e período de geração da amostra utilizada no correlacionador *offline* afeta as associações dos hiperalertas a padrões de estratégia de ataque realizadas pela fase *online*. Os resultados mostraram que até mesmo uma amostra gerada no primeiro dia do mês foi suficiente para formar padrões de estratégia de ataque capazes de representar o comportamento de todos os hiperalertas gerados na última semana do mesmo mês pelo correlacionador *online*.

Assim, o correlacionador *offline* foi capaz de simplificar o problema da análise de muitos alertas, transformando-o na análise de apenas alguns clusters. Além disso, o correlacionador *online* organizou e resumiu os próximos alertas na forma de hiperalertas, os quais são estruturas de informação que descrevem os cenários de ataque por meio de novos atributos.

Como trabalhos futuros, pretende-se: (a) eliminar a necessidade de execução periódica da fase *offline*. Isso é possível se a fase *online* atualizar automaticamente as informações da fase *offline*; (b) explorar novos métodos estatísticos para eliminar a necessidade de parâmetros fixos; (c) desenvolver mecanismos de priorização de hiperalertas. Dessa forma, aqueles com maior prioridade poderiam receber atenção diferenciada, independentemente da quantidade de hiperalertas existentes e (d) desenvolver soluções de resposta automatizada a intrusões utilizando os hiperalertas como base.

REFERÊNCIAS

[1] SHAMELI-SENDI, A.; AGHABABAEI-BARZEGAR, R.; CHERIET, M. Taxonomy of information security risk assessment (isra). *Computers & Security*, v. 57, p. 14 – 30, 2016. ISSN 0167-4048. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167404815001650>.

[2] AHMAD, A.; HADGKISS, J.; RUIGHAVER, A. B. Incident response teams - challenges in supporting the organisational security function. *Comput. Secur.*, Elsevier Advanced Technology Publications, Oxford, UK, UK, v. 31, n. 5, p. 643–652, jul. 2012. ISSN 0167-4048. Disponível em: <http://dx.doi.org/10.1016/j.cose.2012.04.001>.

[3] RUEFLE, R. et al. Computer security incident response team development and evolution. *Security Privacy, IEEE*, v. 12, n. 5, p. 16–26, Sept 2014. ISSN 1540-7993.

[4] JULISCH, K. Clustering intrusion detection alarms to support root cause analysis. *ACM Trans. Inf. Syst. Secur.*, ACM, New York, NY, USA, v. 6, n. 4, p. 443–471, nov. 2003. ISSN 1094-9224. Disponível em: <http://doi.acm.org/10.1145/950191.950192>.

[5] SPATHOULAS, G. P.; KATSIKAS, S. K. Enhancing ids performance through comprehensive alert post-processing. *Comput. Secur.*, Elsevier Advanced Technology Publications, Oxford, UK, UK, v. 37, p. 176–196, set. 2013. ISSN 0167-4048. Disponível em: <http://dx.doi.org/10.1016/j.cose.2013.03.005>.

[6] TREINEN, J. J.; THURIMELLA, R. A framework for the application of association rule mining in large intrusion detection infrastructures. In: *Proceedings of the 9th International Conference on Recent Advances in Intrusion Detection*. Berlin, Heidelberg: Springer-Verlag, 2006. (RAID'06), p. 1–18. ISBN 3-540-39723-X, 978-3-540-39723-6. Disponível em: <http://dx.doi.org/10.1007/11856214_1>.

[7] LIU, L.; ZHENG, K.; YANG, Y. An intrusion alert correlation approach based on finite automata. In: *Communications and Intelligence Information Security (ICCIIS), 2010 International Conference on*. [S.l.: s.n.], 2010. p. 80–83.

[8] MCHUGH, J. Testing intrusion detection systems: A critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory. *ACM Trans. Inf. Syst. Secur.*, ACM, New York, NY, USA, v. 3, n. 4, p. 262–294, nov. 2000. ISSN 1094-9224. Disponível em: <http://doi.acm.org/10.1145/382912.382923>.

[9] BROWN, C. et al. Analysis of the 1999 darpa/lincoln laboratory ids evaluation data with netadhict. In: *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*. [S.l.: s.n.], 2009. p. 1–7. ISSN 2329-6267.

[10] SHIRAVI, A. et al. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Computers & Security*, v. 31, n. 3, p. 357 – 374, 2012. ISSN 0167-4048. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167404811001672>.

[11] ZUECH, R.; KHOSHGOFTAAR, T. M.; WALD, R. Intrusion detection and big heterogeneous data: a survey. *Journal of Big Data*, v. 2, n. 1, p. 1–41, 2015. ISSN 2196-1115. Disponível em: <http://dx.doi.org/10.1186/s40537-015-0013-4>.

[12] STALLINGS, W.; BROWN, L. *Computer Security: Principles and Practice*. 1st. ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2007. ISBN 0136004245, 9780136004240.

[13] Chapter 1 - introduction to intrusion detection systems. In: BURTON, J. et al. (Ed.). *Cisco Security Professional's Guide to Secure Intrusion Detection Systems*. Burlington: Syngress, 2003. p. 1 – 38. ISBN 978-1-932266-69-6. Disponível em: <http://www.sciencedirect.com/science/article/pii/B9781932266696500215>.

[14] SHITTU, R. O. *Mining Intrusion Detection Alert Logs to Minimise False Positives & Gain Attack Insight*. Tese (Doutorado) — City University London, 2016.

[15] LIU, Z.; WANG, C.; CHEN, S. Correlating multi-step attack and constructing attack scenarios based on attack pattern modeling. In: *Information Security and Assurance, 2008. ISA 2008. International Conference on*. [S.l.: s.n.], 2008. p. 214–219.

[16] JULISCH, K.; DACIER, M. Mining intrusion detection alarms for actionable knowledge. In: *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2002. (KDD '02), p. 366–375. ISBN 1-58113-567-X. Disponível em: <http://doi.acm.org/10.1145/775047.775101>.

[17] VACCA, J. R. *Computer and Information Security Handbook*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2009. ISBN 0123743540, 9780123743541.

[18] PATEL, A.; QASSIM, Q.; WILLS, C. A survey of intrusion detection and prevention systems. *Information Management & Computer Security*, Emerald, v. 18, n. 4, p. 277–290, oct 2010.

[19] SCARFONE, K.; MELL, P. *Guide to Intrusion Detection and Prevention Systems (IDPS)*. [S.l.], 2007. Special Publication 800-94.

[20] STAVROULAKIS, P. P.; STAMP, M. (Ed.). *Handbook of Information and Communication Security*. Springer Science & Business Media, 2010. ISBN 978-3-642-04116-7. Disponível em: <http://dx.doi.org/10.1007/978-3-642-04117-4>.

[21] MITCHELL, R.; CHEN, I.-R. A survey of intrusion detection in wireless network applications. *Computer Communications*, v. 42, n. 0, p. 1 – 23, 2014. ISSN 0140-3664. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0140366414000280>.

[22] DEBAR, H. An introduction to intrusion-detection systems. In: *Proceedings of Connect'2000*. [S.l.: s.n.], 2002.

[23] LIAO, H.-J. et al. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, v. 36, n. 1, p. 16 – 24, 2013. ISSN 1084-8045. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1084804512001944>.

[24] VACCA, J. *Computer and information security handbook*. Amsterdam: Morgan Kaufmann, 2013. ISBN 9780123943972.

[25] FATMA, H.; MOHAMED, L. A two-stage technique to improve intrusion detection systems based on data mining algorithms. In: *Modeling, Simulation and Applied Optimization (ICMSAO), 2013 5th International Conference on*. [S.l.: s.n.], 2013. p. 1–6.

[26] GHASEMIGOL, M.; GHAEMI-BAFGHI, A. E-correlator: an entropy-based alert correlation system. *Security and Communication Networks*, v. 8, n. 5, p. 822–836, 2015. ISSN 1939-0122. Disponível em: <http://dx.doi.org/10.1002/sec.1039>.

[27] DASIREDDY, S. et al. Alerts visualization and clustering in network-based intrusion detection. In: *Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research*. New York, NY, USA: ACM, 2010. (CSIIRW '10), p. 41:1–41:4. ISBN 978-1-4503-0017-9. Disponível em: <http://doi.acm.org/10.1145/1852666.1852712>.

[28] YANG, L. et al. Alerts analysis and visualization in network-based intrusion detection systems. In: *Social Computing (SocialCom), 2010 IEEE Second International Conference on*. [S.l.: s.n.], 2010. p. 785–790.

[29] VALDES, A.; SKINNER, K. Probabilistic alert correlation. In: _____. *Recent Advances in Intrusion Detection: 4th International Symposium, RAID 2001 Davis, CA, USA, October 10–12, 2001 Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001. p. 54–68. ISBN 978-3-540-45474-8. Disponível em: <http://dx.doi.org/10.1007/3-540-45474-8_4>.

[30] TAHA, A. et al. Agent based correlation model for intrusion detection alerts. In: *Intelligence and Security Informatics (ISI), 2010 IEEE International Conference on*. [S.l.: s.n.], 2010. p. 89–94.

[31] LEE, S. et al. Real-time analysis of intrusion detection alerts via correlation. *Computers & Security*, v. 25, n. 3, p. 169 – 183, 2006. ISSN 0167-4048. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167404805001549>.

[32] SOLEIMANI, M.; GHORBANI, A. Critical episode mining in intrusion detection alerts. In: *Communication Networks and Services Research Conference, 2008. CNSR 2008. 6th Annual*. [S.l.: s.n.], 2008. p. 157–164.

[33] CIPRIANO, C. et al. Nexat: A history-based approach to predict attacker actions. In: *Proceedings of the 27th Annual Computer Security Applications Conference*. New York, NY, USA: ACM, 2011. (ACSAC '11), p. 383–392. ISBN 978-1-4503-0672-0. Disponível em: <http://doi.acm.org/10.1145/2076732.2076787>.

[34] KAWAKANI, C. T. et al. Intrusion alert correlation to support security management. In: *XII Brazilian Symposium on Information Systems - Information Systems in the Cloud Computing Era*. [S.l.: s.n.], 2016. p. 313–320.

[35] AHMADINEJAD, S. H.; JALILI, S.; ABADI, M. A hybrid model for correlating alerts of known and unknown attack scenarios and updating attack graphs. *Comput. Netw.*, Elsevier North-Holland, Inc., New York, NY,

USA, v. 55, n. 9, p. 2221–2240, jun. 2011. ISSN 1389-1286. Disponível em: <http://dx.doi.org/10.1016/j.comnet.2011.03.005>.

[36] FAYYAD, S.; MEINEL, C. Attack scenario prediction methodology. In: *Information Technology: New Generations (ITNG), 2013 Tenth International Conference on*. [S.l.: s.n.], 2013. p. 53–59.

[37] XUEWEI, F. et al. An approach of discovering causal knowledge for alert correlating based on data mining. In: *Dependable, Autonomic and Secure Computing (DASC), 2014 IEEE 12th International Conference on*. [S.l.: s.n.], 2014. p. 57–62.

[38] LAGZIAN, S. et al. Frequent item set mining-based alert correlation for extracting multi-stage attack scenarios. In: *Telecommunications (IST), 2012 Sixth International Symposium on*. [S.l.: s.n.], 2012. p. 1010–1014.

[39] HAN, J.; PEI, J.; YIN, Y. Mining frequent patterns without candidate generation. In: *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*. New York, NY, USA: ACM, 2000. (SIGMOD '00), p. 1–12. ISBN 1-58113-217-4. Disponível em: <http://doi.acm.org/10.1145/342009.335372>.

[40] NING, P.; CUI, Y.; REEVES, D. S. Constructing attack scenarios through correlation of intrusion alerts. In: *Proceedings of the 9th ACM Conference on Computer and Communications Security*. New York, NY, USA: ACM, 2002. (CCS '02), p. 245–254. ISBN 1-58113-612-9. Disponível em: <http://doi.acm.org/10.1145/586110.586144>.

[41] SHITTU, R. et al. Intrusion alert prioritisation and attack detection using post-correlation analysis. *Computers & Security*, v. 50, p. 1 – 15, 2015. ISSN 0167-4048. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167404814001837>.

[42] ZOMLOT, L. et al. Aiding intrusion analysis using machine learning. In: *Machine Learning and Applications (ICMLA), 2013 12th International Conference on*. [S.l.: s.n.], 2013. v. 2, p. 40–47.

[43] VALEUR, F. et al. A comprehensive approach to intrusion detection alert correlation. *IEEE Trans. Dependable Secur. Comput.*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 1, n. 3, p. 146–169, jul. 2004. ISSN 1545-5971. Disponível em: <http://dx.doi.org/10.1109/TDSC.2004.21>.

[44] SADODDIN, R.; GHORBANI, A. A. An incremental frequent structure mining framework for real-time alert correlation. *Computers & Security*, v. 28, n. 3–4, p. 153 – 173, 2009. ISSN 0167-4048. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167404808001144>.

[45] NING, P.; XU, D. Learning attack strategies from intrusion alerts. In: *Proceedings of the 10th ACM Conference on Computer and Communications Security*. New York, NY, USA: ACM, 2003. (CCS '03), p. 200–209. ISBN 1-58113-738-9. Disponível em: <http://doi.acm.org/10.1145/948109.948137>.

[46] JR, J. H. W. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, Taylor & Francis, v. 58, n. 301, p. 236–244, 1963.

[47] MACFARLANE, P. A. *Kansas Geological Survey, Dakota Aquifer Program - Ward's Method*. 1996. Disponível em: <http://www.kgs.ku.edu/Dakota/vol1/geo/hodge4.htm>.

[48] FISCHER, A.; RIESEN, K.; BUNKE, H. Improved quadratic time approximation of graph edit distance by combining hausdorff matching and greedy assignment. *Pattern Recognition Letters*, p. –, 2016. ISSN 0167-8655. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0167865516301386>.

[49] NIWATTANAKUL, S. et al. Using of jaccard coefficient for keywords similarity. In: *Proceedings of the International MultiConference of Engineers and Computer Scientists*. [S.l.: s.n.], 2013. v. 1, p. 6.

[50] HOPCROFT, J.; KANNAN, R. *Computer Science Theory for the Information Age*. 2012.

[51] JAIN, A. K.; MURTY, M. N.; FLYNN, P. J. Data clustering: A review. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 31, n. 3, p. 264–323, set. 1999. ISSN 0360-0300. Disponível em: <http://doi.acm.org/10.1145/331499.331504>.

[52] XU, R.; WUNSCH D., I. Survey of clustering algorithms. *Neural Networks, IEEE Transactions on*, v. 16, n. 3, p. 645–678, May 2005. ISSN 1045-9227.

[53] KERNS, G. J. *Introduction to Probability and Statistics Using R*. first. [S.l.]: Free Software Foundation, 2011.

[54] CVE. Common Vulnerabilities and Exposures 2006-3439. 2006. <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-3439>. Acesso: Fevereiro, 2017.

[55] CVE. Common Vulnerabilities and Exposures 2003-0818. 2003. <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-0818>. Acesso: Fevereiro, 2017.

[56] CVE. Common Vulnerabilities and Exposures 2005-1983. 2005. <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2005-1983>. Acesso: Fevereiro, 2017.

[57] CVE. Common Vulnerabilities and Exposures 2002-1141. 2002. <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2002-1141>. Acesso: Fevereiro, 2017.

TRABALHOS PUBLICADOS PELO AUTOR

Trabalhos publicados pelo autor durante o programa.

1. Cláudio Toshio Kawakani, Sylvio Barbon Junior, Rodrigo Sanches Miani, Michel Cukier, Bruno Bogaz Zarpelão. **Intrusion Alert Correlation to Support Security Management**. In: Proceedings of the XII Brazilian Symposium on Information Systems, p. 313-320. Florianópolis, Santa Catarina, Brazil. May 2016 (Best paper candidate). (Qualis CC Conferências 2012: B4).

2. Cláudio Toshio Kawakani, Sylvio Barbon Junior, Rodrigo Sanches Miani, Michel Cukier, Bruno Bogaz Zarpelão. **Discovering Attackers Past Behavior to Generate Online Hyper-Alerts**. iSys - Revista Brasileira de Sistemas de Informação. Editorial Vol. 10, Nº 1. 2017. (Qualis CC Periódicos 2015: B3).

3. Bruno Bogaz Zarpelão, Rodrigo Sanches Miani, Cláudio Toshio Kawakani, Sean Carlisto de Alvarenga. **A Survey of Intrusion Detection in Internet of Things**. Journal of Network and Computer Applications. Volume 84, 15 April 2017, Pages 25–37. (Qualis CC Periódicos 2015: A2).