UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

WESLEY FRANCO FERREIRA

APLICAÇÃO DE CONCEITOS DE SOC: IMPLEMENTAÇÃO DE SIEM COM ELASTICSEARCH

CAMPO MOURÃO

2023

WESLEY FRANCO FERREIRA

APLICAÇÃO DE CONCEITOS DE SOC: IMPLEMENTAÇÃO DE SIEM COM ELASTICSEARCH

Application of SOC concepts: SIEM implementation with Elasticsearch

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Ciência da Computação do Curso de Bacharelado em Ciência da Computação da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Luiz Arthur Feitosa dos Santos

CAMPO MOURÃO

2023

WESLEY FRANCO FERREIRA

APLICAÇÃO DE CONCEITOS DE SOC: IMPLEMENTAÇÃO DE SIEM COM ELASTICSEARCH

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Ciência da Computação do Curso de Bacharelado em Ciência da Computação da Universidade Tecnológica Federal do Paraná.

Data de aprovação: 15/junho/2023

Geazzy B. Marçal Zanoni
Especialista
Universidade Tecnologica Federal do Paraná

Rodrigo Campiolo
Doutorado
Universidade Tecnológica Federal do Paraná

Luiz Arthur Feitosa dos Santos
Doutorado
Universidade Tecnológica Federal do Paraná

CAMPO MOURÃO

2023

AGRADECIMENTOS

Agradeço principalmente ao meu orientador, Luiz Arthur Feitosa dos Santos, por todo conhecimento, dedicação nesse período de orientação, principalmente por não desistir de mim mesmo com todo atraso no desenvolvimento, aos membros da banca Geazzy B. Marçal Zanoni e Rodrigo Campiolo por aceitarem fazer parte da banca e de minha jornada acadêmica juntamente a todos os professores do Departamento Acadêmico de Computação (DACOM).

Por fim sou extremamente grato aos meus amigos, em especial ao Emanuel Felipe Giroldo Mazzer, Lucas Gabriel da Silva, Renan Viana Hoshi e Vinicius Bosa Petris por sempre me incentivarem, aos meus amigos e colegas de trabalho Antonio Edmilson do Amaral Junior, David Antonio Saba Cavalcante por todo conhecimento e auxílio em questões técnicas e a toda minha família: minha mãe Roseli Aparecida Franco e minha avó Tereza Viertel Franco, por todo apoio nesse período.

RESUMO

Com o cenário atual de cibersegurança, empresas são invadidas no dia a dia em questão de segundos, porém podendo demorar entre dias e até semanas para detectar essas invasões. Esse défice de detecções acontece mesmo entre empresas de grande porte onde existem grandes orçamentos exclusivos para equipes de segurança, ferramentas e criação de processos. No caso de empresas de pequeno e médio porte a utilização de equipes de segurança se torna inviável devido ao seu alto custo de implantação com pessoas especializadas, ferramentas e processos. O principal objetivo deste trabalho foi desenvolver um sistema utilizando ferramentas *open source* que possibilite a detecção de eventos de segurança de forma automatizada em tempo real. Juntamente com o monitoramento de eventos de segurança, um dos objetivos é a centralização de *logs*, possibilitando centralizar a utilização de uma interface de visualização permite uma melhor análise dos dados, assim contribuindo com uma melhora na investigação de possíveis incidentes. Primeiramente definiu-se as ferramentas que seriam utilizadas, definindo a utilização do pacote de software *Elk Stack*. Em seguida foi definido a linguagem de programação para o desenvolvimento de um script para consulta e análise dos dados. Durante o desenvolvimento do script foi definido a criação de regras utilizadas para analisar as informações em busca de eventos específicos. Por fim a definição do fluxo de envio de alertas informativos contendo informações sobre os eventos detectados. A geração de *logs* utilizando o *framework* de auditoria do Linux foi possível coletar 2.071.123 de eventos sobre um *host*, consumindo cerca de 1.4 Gigabytes em espaço de disco. Diariamente foram coletados em média 49.312 eventos. Com a execução do script de monitoração a cada 1 hora, foi possível identificar 124 eventos do uso de comandos com elevação de privilégios administrativos. Com a centralização dos dados, foi implantado uma interface de visualização de dados permitindo a realização de consultas de simples até complexas facilitando a análise dos dados, além da possibilidade de extração de relatórios com uma infinidade de filtros. O desenvolvimento deste trabalho foi possível implementar uma monitoração de ativos baseada em eventos pré-definidos, criando e enviado alertas informativos. Outra contribuição é a melhora na forma de análise de grandes volumes de dados, extração de relatórios e identificação de padrões.

Palavras-chave: cibersegurança ; detecção; incidentes; elasticsearch; siem.

ABSTRACT

With the current cybersecurity scenario, companies are invaded on a daily basis in a matter of seconds, but it can take between days and even weeks to detect these invasions. This detection deficit happens even among large enterprises where there are large budgets dedicated to security teams, tools and process creation. In the case of small and medium-sized companies, the use of security teams becomes unfeasible due to their high cost of implementation with specialized people, tools and processes. The main objective of this work was to develop a system using open source tools that allows the detection of security events in an automated way in real time. Along with the monitoring of security events, one of the objectives is the centralization of logs, making it possible to centralize the use of a visualization interface that allows a better analysis of the data, thus contributing to an improvement in the investigation of possible incidents. First, the tools that would be used were defined, defining the use of the Elk Stack software package. Next, the programming language was defined for the development of a script for querying and analyzing the data. During the development of the script, the creation of rules used to analyze the information in search of specific events was defined. Finally, the definition of the flow of sending informative alerts containing information about the detected events. Logging using the Linux auditing framework was able to collect 2,071,123 events on a host, consuming about 1.4 Gigabytes of disk space. An average of 49,312 events were collected daily. By running the monitoring script every hour, it was possible to identify 124 events involving the use of commands with elevation of administrative privileges. With the centralization of data, a data visualization interface was implemented, allowing queries from simple to complex, facilitating data analysis, in addition to the possibility of extracting reports with an infinity of filters. The development of this work made it possible to implement asset monitoring based on predefined events, creating and sending informative alerts. Another contribution is the improvement in the way of analyzing large volumes of data, extracting reports and identifying patterns.

Keywords: cybersecurity; detection; incidents; elasticsearch; siem.

LISTA DE FIGURAS

Figura 1 – Funcionamento de um SOC (ZIMMERMAN, 2014). **18**
Figura 2 – Esquema de detecção considerando as duas abordagens (ZIMMERMAN, 2014). **22**
Figura 3 – Arquitetura geral do sistema. **33**
Figura 4 – Tela inicial do sistema **35**
Figura 5 – Tela para consulta de *logs*. **36**
Figura 6 – *Dashboard* geral referente a tecnologia Auditbeat. **37**
Figura 7 – Tela de gerenciamento de usuários. **38**
Figura 8 – Configuração da fase morna de armazenamento. **43**
Figura 9 – Monitoramento da infraestrutura usando Metricbeat. **43**
Figura 10 – Alerta recebido via e-mail. **46**
Figura 11 – Fluxograma da execução do *script* **47**
Figura 12 – Os 15 comandos mais executados com elevação de privilégios administrativos. **48**
Figura 13 – Pico de indexação obtido em período de promoções **50**

LISTA DE TABELAS

Tabela 1 – Exemplo de avaliação de risco (MUNIZ GARY MCINTYRE, 2016). **21**

Tabela 2 – Categorização de incidentes (MUNIZ GARY MCINTYRE, 2016). **23**

Tabela 3 – Volumetria de dados indexados por varejista brasileira **49**

LISTAGEM DE CÓDIGOS FONTE

Listagem 1 – Arquivo de configuração `elasticsearch.yml` 40

Listagem 2 – Parte do arquivo de configuração `Auditbeat.yml`. 41

Listagem 3 – Arquivo de configuração `Auditbeat.yml`. 42

Listagem 4 – Função para conexão com o Elasticsearch 44

Listagem 5 – Função para busca de eventos pré definidos 45

Listagem 6 – Configuração do agendador de tarefas (*crontab*) 47

LISTA DE ABREVIATURAS E SIGLAS

Siglas

BYOD	Bring Your Own Device
CA	Certificate Authority
CEO	Chief Executive Officer
CIO	Chief Information Officer
CISO	Chief Information Security Officer
DACOM	Departamento Acadêmico de Computação
ELK	Elasticsearch, Logstash, Kibana
HIDS	Host Intrusion Detection System
HTTPS	Hyper Text Transfer Protocol Secure
IA	Inteligência Artificial
IDS	Intrusion Detection System
IoC	Indicator of Compromise
IP	Internet Protocol
IPS	Intrusion Prevention System
JSON	JavaScript Object Notation
KQL	Kibana Query Language
NIDS	Network Intrusion Detection System
NIST	National Institute of Standards and Technology
OODA	Observe, Orient, Decide, Act
RMF	Risk Management Framework
SIEM	Security Information and Event Management
SMTP	Simple Mail Transfer Protocol

SO	Sistema Operacional
SOC	Security Operation Center
SSL	Transport Layer Security
TI	Tecnologia da Informação
UTFPR	Universidade Tecnológica Federal do Paraná
VM	Virtual Machine

SUMÁRIO

1 INTRODUÇÃO 13
1.1 Problema de Pesquisa 13
1.2 Objetivos 13
1.3 Contribuições 14
1.4 Organização do Texto 14
2 CONCEITOS 15
2.1 Cibersegurança 15
2.2 Security Operation Center 16
2.2.1 Fundamentos de SOC 17
2.2.2 Avaliação de risco 19
2.2.3 Detecção 20
2.2.4 Triagem de incidentes 22
2.2.5 Encerramento do incidente 24
2.2.6 Modelos organizacionais 24
2.2.7 Autoridade 25
2.3 Pacote ELK 26
2.3.1 Elasticsearch 26
2.3.2 Logstash 27
2.3.3 Kibana 27
2.3.4 Beats 28
2.3.5 Utilização do pacote Elasticsearch, Kibana, Logstash 28
2.4 Trabalhos Relacionados 28
2.4.1 Gerenciamento centralizado de *logs* usando Elasticsearch, Logstash e Kibana 28
2.4.2 GT-BIS 29
2.5 Considerações finais 29
2.6 Considerações 30
3 MATERIAIS E MÉTODOS 31
3.1 Materiais 31
3.1.1 Elasticsearch 31
3.1.2 Kibana 31

3.1.3 Auditbeat . . . 32
3.1.4 Python . . . 32
3.1.5 Servidor para envio de e-mails . . . 32
3.1.6 Descrição do sistema . . . 32
3.2 Métodos . . . 33
3.2.1 Geração de eventos . . . 34
3.2.2 Coleta, estruturação e armazenamento . . . 34
3.2.3 Processamento de dados . . . 34
3.2.4 Notificação de alertas . . . 34
3.3 Escopo do sistema . . . 35
3.4 Apresentação do sistema . . . 35
3.4.1 Análise e visualização de dados . . . 36
3.4.2 *Dashboards* . . . 36
3.4.3 Monitoramento de infraestrutura e gerenciamento de usuários . . . 37
3.5 Implementação do sistema . . . 37
3.5.1 Instalação e configuração do mecanismo de pesquisa (Elasticsearch) . . . 37
3.5.2 Instalação e configuração do software de visualização (Kibana) . . . 39
3.5.3 Instalação e configuração do agente (AuditBeat) . . . 40
3.5.4 Criação de índice e configuração do ciclo de vida . . . 42
3.5.5 Monitoramento da infraestrutura . . . 43
3.5.6 Desenvolvimento do *script* de revisão de *logs* . . . 43
3.5.7 Revisão e análise de *logs* . . . 44
3.5.8 Função de conexão (*conect_elastic*) . . . 44
3.5.9 Busca (*search_logs*) . . . 44
3.5.10 Criação do alerta e envio via e-mail (*create_alerts* e *send_email*) . . . 45
3.5.11 Agendador de tarefas (`Crontab`) . . . 46
3.6 Considerações finais . . . 47
4 EXPERIMENTOS E RESULTADOS . . . 48
4.0.1 Considerações finais . . . 49
5 CONCLUSÕES . . . 51
5.1 Trabalhos futuros . . . 52
5.2 Considerações finais . . . 52

REFERÊNCIAS 53

1 INTRODUÇÃO

Mesmo com os avanços tecnológicos na área da cibersegurança, a constante evolução na forma dos ataques vem chamando atenção de profissionais da área para o défice na detecção de incidentes.

Dados mostram que 60% das empresas são invadidas em questão de minutos e as invasões são descobertas apenas dias e até meses após o ocorrido (ENTERPRISE, 2015). Em abril de 2018 o Banco Central do Brasil publicou uma resolução informando que bancos e instituições financeiras devem utilizar políticas de segurança cibernética (BRASIL, 2018). Essas políticas têm o objetivo de contemplar detecção, prevenção e redução de vulnerabilidades que possam levar a ocorrência de incidentes de segurança cibernética. Outro problema recorrente é o elevado número de brechas em sistemas e redes que possam ser explorados, por exemplo, o uso de Bring Your Own Device (BYOD) (SANTOS, 2018). Por mais que redes possam ser projetadas com foco em segurança, quando dispositivos de uso pessoal são conectados na rede, esses trazem consigo *malwares* gerando vulnerabilidades que podem ser explorar em futuras invasões.

Considerando o cenário descrito anteriormente, uma alternativa para minimizar os riscos da utilização de BYOD é a utilização de uma equipe de Security Operation Center (SOC). O papel do SOC é monitorar, detectar e realizar contenções de atividades suspeitas que possam evoluir para casos de incidentes. A missão do SOC é assegurar a integridade, disponibilidade e confidencialidade de dados. As principais funções são: monitorar, detectar, analisar, correlacionar e implantar soluções contra incidentes. Porém, o custo de implantação e manutenção dos SOC é alto devido à licença de software, pessoas especializadas, criação de processos. Assim dificultando a utilização de equipes de SOC para empresas de pequeno e médio porte.

1.1 Problema de Pesquisa

SOCs são equipes responsáveis por realizar operações de segurança, referente à detecção, prevenção, monitoramento de vulnerabilidades e respostas a incidentes de segurança. Empresas de grande porte possuem equipes de SOC com o intuito de garantir a integridade, confidencialidade e disponibilidade dos dados, além de prevenir que não ocorra vazamentos e roubos de informações. A implantação de SOC por sua vez é um processo custoso devido as ferramentas, processos e pessoas.

1.2 Objetivos

O presente trabalho tem como objetivo a elaboração de um sistema baseado nos conceitos de SOC que possa ser implantado em empresas de pequeno ou médio porte. Na tentativa

de desenvolver um sistema com baixo custo de implantação, será utilizado ferramentas *Open Source* que consigam atender as necessidades identificadas. Um dos objetivos é desenvolver uma monitoração de ativos automatizada de fácil manutenção e que seja possível implementar futuras melhorias.

1.3 Contribuições

As contribuições do presente trabalho são:

- Viabilização da utilização de conceitos SOC para empresas de pequeno e médio porte;
- Centralização de fontes de dados (*logs*);
- Implantação de interface para visualização de dados;
- Análise dos dados com mais facilidade e agilidade;
- Detecção de possíveis incidentes em tempo real e notificação sobre eventos encontrados.

1.4 Organização do Texto

O trabalho está organizado com a seguinte estrutura: o Capítulo 1 apresentou uma introdução do trabalho, definindo o problema de pesquisa, objetivos e contribuições. No Capítulo 2 é apresentada a fundamentação teórica para a compreensão do trabalho, contendo conceitos de cibersegurança, SOC e sobre as principais ferramentas utilizadas. Seguido pelos trabalhos relacionados detalhados no Capítulo 2.4. No Capítulo 3, é apresentada a metodologia utilizada, detalhando arquitetura e estudo de caso. No Capítulo 4 são discutidos os experimentos e resultados. Finalizando no Capítulo 5 serão feitas as considerações finais e apresentação de trabalhos futuros.

2 CONCEITOS

No contexto da cibersegurança existem ocorrências de incidentes nos quais pessoas mál-intencionadas realizam tentativas de invasões, seja para roubo de dados ou simplesmente na tentativa de afetar a disponibilidade de serviços, entre outras formas de ataques. Considerando este contexto, neste capítulo são abordados conceitos de cibersegurança, principais conceitos de SOC, como: modelo organizacional, como é realizado a triagem de incidentes de segurança, avaliação de riscos, detecção de incidentes e trabalhos relacionados.

2.1 Cibersegurança

No contexto da segurança existe uma vertente que diz respeito a segurança de computadores e/ou dados digitais denominada cibersegurança. O termo foi utilizado pela primeira vez em 1987 com a remoção do primeiro vírus de computador documentado por Bernd Fix Reasearch (2012). A cibersegurança não se limita apenas a assuntos relacionados com Internet, mas sim, em todo o escopo que envolva sistema digitais, redes de computadores e informações. A empresa Kaspersky classifica a cibersegurança em algumas categorias comuns (KASPERSKY, 2019):

- **Segurança de rede**: é a proteção de redes de computadores contra vulnerabilidades que possam ocasionar incidentes devido a intrusos ou *malwares*;
- **Segurança de aplicativos**: é toda a segurança envolvida na elaboração do projeto de aplicativo ou softwares, quais vulnerabilidades podem ocorrer e já sendo resolvidas na fase de projeto;
- **Segurança da informação**: é a tentativa de proteger os dados mantendo sua disponibilidade, confidencialidade e integridade;
- **Segurança operacional**: diz respeito a decisões como nível de hierarquia para acesso de dados, a forma em que certos dados são armazenados baseados em sua importância e confidencialidade;
- **Recuperação de incidentes**: são políticas de recuperação após a ocorrência de incidentes em que possam ocorrer vazamentos ou comprometimento de dados. Essas políticas determinam como será a resposta ao incidente, realizando a contenção e como será a recuperação do ambiente.
- **Conscientização do usuário final**: é considerado o fator mais imprevisível em relação a segurança, diz respeito a necessidade de orientação do usuário final para não causar vulnerabilidades dentro dos sistemas das empresas, por exemplo ao utilizar e-

mails não ser enganado em técnicas comumente utilizadas, por exemplo, *phishing*, que consiste na utilização de páginas falsas, com o objetivo de roubar dados.

Em cibersegurança, Systems (2015) traz a definição de incidente como sendo: um evento avaliado que tenha potencial ou comprometa diretamente a integridade ou disponibilidade de algum sistema ou a informação que o sistema processa, transmite ou armazena, podendo ser violações ou ameaças iminentes de violações em políticas de segurança ou procedimentos de segurança.

Na cibersegurança existem vários problemas que podem ir desde uma simples atualização de software que não foi aplicada até mesmo vulnerabilidades que correm o risco de vazar informações de milhares de usuários. O SOC tenta solucionar grande parte desses problemas.

2.2 Security Operation Center

SOC tentam solucionar problemas de segurança principalmente com detecção e monitoramento de sistemas, possuindo a missão de defender a empresa no qual se encontra, gerenciando recursos para coleta, análise, prevenção e resposta a incidentes. O SOC é constituído por um grupo de analistas de diferentes níveis de experiências, podendo ser desde analistas juniores até pessoas com conhecimentos mais aprofundados. Os objetivos do SOC possuem alguns elementos sendo:

1. Prevenção de ataques de forma proativa:
    - Realizar e analisar varreduras em busca de vulnerabilidades em *hosts* e rede;
    - Implementar medidas contra ataques;
    - Realizar consultorias buscando melhorar a infraestrutura da empresa.
2. Monitorar em tempo real, detectar possíveis intrusões, analisar grandes fluxos de dados na busca de tendências, utilizar dados providos de outros SOC e/ou fontes externas como bancos de assinaturas de *malware*, relatórios públicos de incidentes vindo de fontes externas como repositórios de segurança;
3. Defesa contra incidentes já confirmados, coordenando recursos e estratégias de defesa, buscando o plano mais efetivo para implementar respostas mais apropriadas contra incidentes de segurança;
4. Fornecer dados e relatórios a respeito do atual estado de segurança e tendências encontradas sobre ataques à empresa;
5. Engenharia para implementação de defesas como Intrusion Detection System (IDS) e outras técnicas que possam ajudar a mitigar ameaças dentro da infraestrutura da empresa.

De todas as responsabilidades do SOC, a que leva mais tempo é a análise de dados fornecidos por sensores implantados em pontos estratégicos da rede, como por exemplo Network Intrusion Detection System (NIDS) ou Host Intrusion Detection System (HIDS). Um sistema IDS é constituído por softwares implantados em *hosts* ou redes detectando possíveis intrusões e/ou atividades maliciosas e gerando alertas, sendo que estes provem dados para a análise dentro do SOC. Tipicamente, o SOC não trabalha somente com dezenas de alertas gerados por IDS, mas com análise de *logs* e eventos, base de dados, podendo chegar a centenas de milhares de dados.

O presente trabalho mantém o seu foco de implementação nos pontos de monitoramento em tempo real, envio de alertas informativos e disponibilidade de dados em uma plataforma unificada com fácil visualização e manipulação dos dados. Desta forma possibilitando a criação de relatórios, identificação de padrões, entre outras.

Um evento é definido por qualquer ocorrência em sistemas e/ou redes. Esses eventos podem conter indícios de possíveis incidentes. Um evento não é nada além de um *log* com diversas informações, nas quais podem ser relevantes ou não. O processo de análise pode ser feito com interação humana ou de forma automatizada quando existem padrões bem definidos. O SOC designará um conjunto de pessoas apenas para triagem de alertas, entre outras tarefas mais cotidianas, como ligações telefônicas (ZIMMERMAN, 2014). Esse grupo de analistas é denominado como nível 1.

O grupo de analistas de nível 1 realiza a triagem inicial dos alertas, categorizando sua complexidade e quando possível realizando o tratamento adequado. O analista nível 1 é responsável de realizar a triagem inicial e contenções de complexidade baixa, quando necessário o evento é escalonado para o nível 2. O limite de tempo estipulado para cada análise é baseado no potencial da atividade maliciosa e sua complexidade.

O nível 2 realiza uma investigação um pouco mais aprofundada. Esta análise pode levar muito mais tempo se comparado com o nível 1, podendo demorar de dias a semanas. O nível 2 não realiza triagem de alertas em tempo real, assim tendo o tempo necessário para realizar uma avaliação completa baseada nas informações contidas nos alertas e/ou obtidas através da correlação de dados. A Figura 2.1 ilustra como é um fluxo de funcionamento de um SOC baseados nos conceitos apresentados por Zimmerman (2014).

O SOC desenvolvido no presente trabalho irá realizar de forma automatizada o papel do analista nível 1, buscando eventos específicos determinados em sua implementação. A triagem será feita de forma automatizada e ao invés de realizar um escalonamento ou contenção, irá enviar alertas informacionais comunicando os eventos detectados.

2.2.1 Fundamentos de SOC

A metodologia adotada para o SOC se baseia nos conceitos idealizados pelo coronel da Força Aérea dos Estados Unidos, John Boyd. De acordo com Rule (2013), a metodologia se

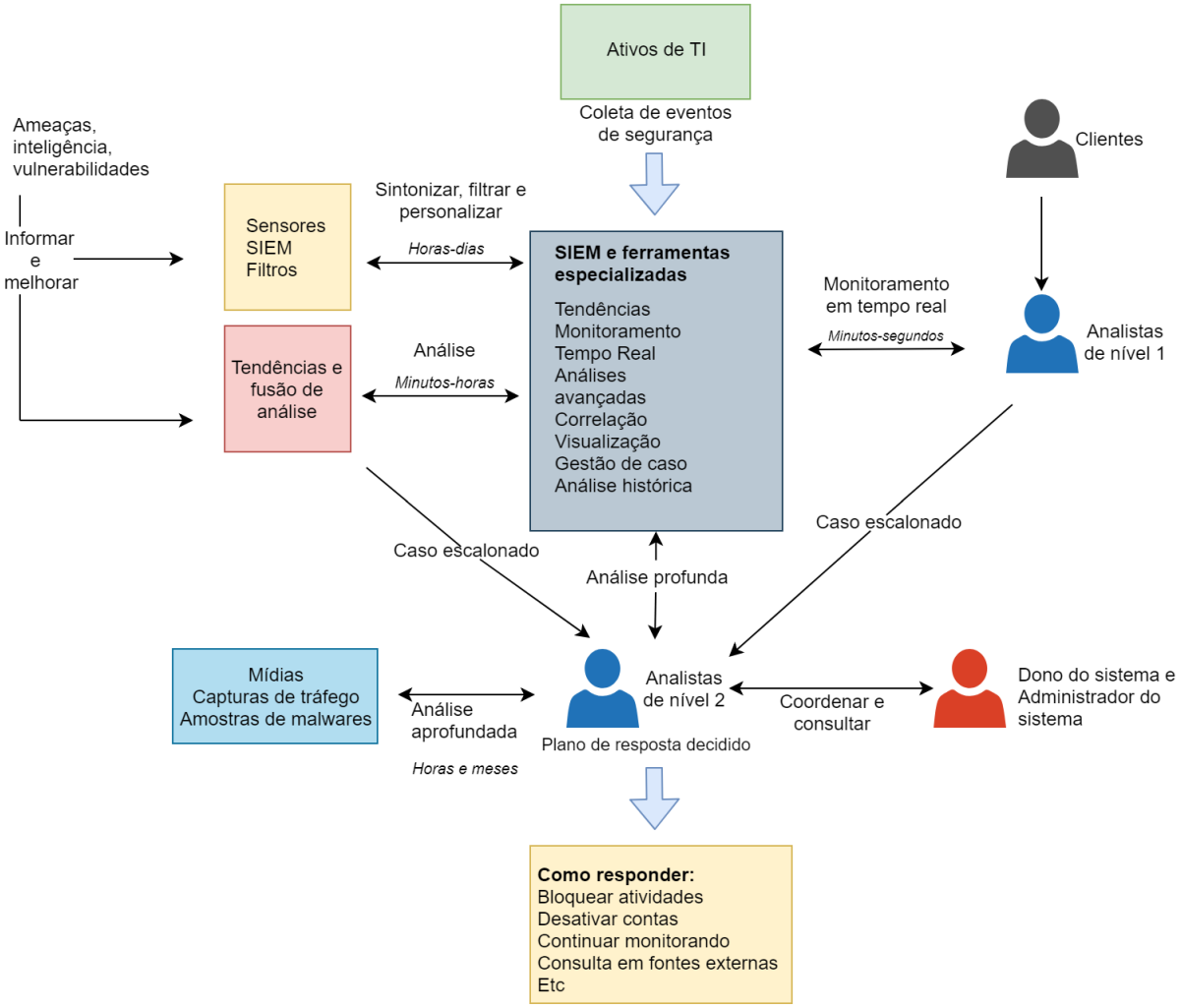

Figura 1 – Funcionamento de um SOC (ZIMMERMAN, 2014).

baseia em quatro passos fundamentais para a resolução de qualquer problema sendo: observar, orientar, decidir e agir. A tendência é melhorar a cada ciclo completo pois, cada ciclo gera mais conhecimento que pode ser utilizado em novos ciclos. No contexto da cibersegurança esses passos são divididos em:

1. Observar: Coletar, armazenar e monitorar dados de todos os sensores implantados no sistema;

2. Orientar: Analisar todos os tipos de dados coletados no passo 1, com o uso de ferramentas de análise de grandes fluxos de dados;

3. Decidir: Determinar quais ações tomar utilizando toda análise gerada nas interações anteriores, e se baseando na análise de histórico e tendências que foram gerados em outros incidentes;

4. Agir: Implantar as ações determinadas no passo anterior, logo após o incidente ser tratado devidamente e são gerados relatórios documentando todo processo.

Mesmo sendo criado para utilização em ataques militares, o ciclo de Observe, Orient, Decide, Act (OODA) é o princípio fundamental dos SOC. Baseado no ponto de vista de invasores a equipe de respostas a incidentes da Lockheed Martin (HUTCHINS; CLOPPERT; AMIN, 2011) desenvolveram 7 passos utilizados para planejar e executar ataques em alvos. Este modelo foi criado para auxiliar profissionais da cibersegurança para identificar e negar ações maliciosas. Os 7 passos são:

1. Reconhecimento: Nessa fase o invasor identifica o alvo, seja em pesquisas na Internet, listas de discussão, informações tecnológicas específicas, relações com outros invasores;

2. Armamento: Implantação de *trojans* de acesso remoto com exploração em algum dado com carga útil podendo ser um e-mail por exemplo, utilizando ferramentas automatizadas;

3. Entrega: Transmite o *trojan* para o alvo selecionado;

4. Exploração: Busca explorar alguma vulnerabilidade de software ou sistema operacional, mas também podendo explorar o usuário ou aproveitar algum recurso do sistema que possa executar o código automaticamente;

5. Instalação: Instala o *trojan* de acesso remoto ou um *backdoor* no alvo, permitindo que o invasor tenha acesso ao sistema da vítima;

6. Comando e Controle: Estabelece um canal de comando direto entre o sistema da vítima e o invasor;

7. Ações e objetivos: Quando o invasor completa seus objetivos, como por exemplo tornar indisponível algum recurso da vítima, roubar dados, extrair informações comerciais ou apenas obter acesso ao sistema da vítima para utilizar como entrada para outros sistemas.

O ciclo de OODA tenta identificar essas ações tomadas pelo invasor podendo acontecer uma interação completa de ciclo para cada passo em que o invasor possa estar, seja o passo de exploração de vulnerabilidade ou até mesmo a entrega do *trojan*. O objetivo do ciclo de OODA é quebrar a cadeia de eventos no qual um invasor realiza um ataque. O SOC utiliza esses conceitos com variação do ciclo com foco em três elementos: pessoas, processos e tecnologia.

2.2.2 Avaliação de risco

Para entender melhor a avaliação de risco é necessário entender o que são os riscos. A definição comum de risco é dada pela probabilidade de ameaças que exploram ou criam

vulnerabilidades. Uma definição semelhante é listada no guia National Institute of Standards and Technology (NIST) para conduzir avaliações de risco.

A avaliação de risco é o processo usado para atribuir algum valor ao risco associado a empresa. Todo conteúdo gerado em uma avaliação de risco é utilizada em análises e decisões para determinar contramedidas a serem tomadas. A determinação de contramedidas podem possuir um ou mais procedimentos conhecidos como suavizar os riscos, transferir riscos, aceitar ou evitar os mesmos. O gerenciamento de riscos é formado pela combinação de conteúdo útil para segurança provido da avaliação de riscos com a decisão sobre qual a melhor forma de tratar os riscos.

Um exemplo de *framework* que tenta lidar com o gerenciamento de risco é o Risk Management Framework (RMF). Muniz Gary McIntyre (2016) apresenta um algoritmo utilizado para avaliação de risco, suas etapas são:

1. Estabelecer o contexto do risco;
2. Identificar o risco;
3. Analisar as informações necessárias e possíveis sobre o risco;
4. Avaliar o risco;
5. Tratar o risco relatado.

O processo de avaliar e gerenciar riscos deve aceitar fontes de dados externas em tempo real ou não. Por exemplo, a má formação de um pacote IP desencadeou uma falha em algum sistema da empresa, todo o processo de falha deve ser investigado para identificar o risco e esse risco deve ser associado ao sistema, e muito possivelmente a outros sistemas semelhantes, como, por exemplo servidores. As alterações que podem ocorrer no ambiente também podem afetar o risco associado ao sistema, por exemplo, a renúncia de cargo de um administrador de sistema que possuía acesso privilegiado a informações sigilosas. As ações que serão tomadas pela empresa, diante desta situação, devem ser baseadas em processos já predefinidos para lidar com os novos valores de risco no ambiente em questão.

Para exemplificar a avaliação de risco, a Tabela 1 ilustra a avaliação de de risco para uma vulnerabilidade publicada sobre softwares utilizados em servidores.

2.2.3 Detecção

De acordo com Zimmerman (2014), existem duas abordagens de detecção de intrusão sendo:

- Utilização indevida ou detecção baseada em assinatura, em sistemas que se enquadram nesta abordagem existe um conhecimento prévio do tipo da atividade. Assim

Tabela 1 – Exemplo de avaliação de risco (MUNIZ GARY MCINTYRE, 2016).

Componente do risco	Descrição
Vulnerabilidade	Uma nova vulnerabilidade que afeta servidores foi anunciada. Análises mostram que os servidores estão vulneráveis.
Descrição da ameaça	Os servidores vulneráveis são considerados críticos. O ataque pode ser facilmente executado caso o invasor tiver acesso a Internet.
Contramedidas existentes	Os servidores não estão conectados na Internet. Os servidores possuem um *firewall* que aceita apenas acessos internos; Os servidores possuem um Intrusion Prevention System (IPS), entretanto o fornecedor do IPS não disponibilizou nenhuma assinatura contra a vulnerabilidade recém anunciada.
Probabilidade	Improvável. Os servidores só podem ser explorados por usuários com acesso a rede interna da empresa.
Impacto	Critico. Explorando a vulnerabilidade o invasor pode obter acesso administrativo total ao sistema.

podendo caracterizar uma assinatura e gerar um alerta que indique a equipe que está acontecendo uma atividade maliciosa.

- Detecção de anomalias: o sistema tenta caracterizar comportamentos benignos e tenta indicar qualquer comportamento que fuja do escopo de benigno ou normal.

as duas abordagens possuem vantagens e desvantagens no sistema. Ferramentas baseadas em assinaturas utilizam a detecção de padrões previamente estabelecidos, porém ataques ” *zero day*“ são novos métodos de invasão, desta forma não possuindo análises prévias onde seus padrões são identificadas. Já na abordagem de anomalias, existe a desvantagem que qualquer ato que fuja dos padrões constitua uma ação maliciosa, ocasionando muitos alertas e assim não sendo detectado pelo sistema.

Zimmerman (2014) mostra que os vários softwares de detecção como IDS/IPS cogitados para o presente trabalho possuem algumas características similares na forma de detecção de intrusões, sendo:

- Quando existe algum conhecimento do ambiente e sobre a ameaça, é possível criar políticas de detecção que defina os comportamentos;
- Os mecanismos de detecção utilizam variáveis virtuais (eventos, por exemplo) para determinar se eventos suspeitos caracterizam assinaturas e assim gerar alertas após compará-los às políticas de detecção determinadas pela equipe do SOC;
- Os alertas gerados sobre eventos são armazenados para análises;
- Após analisar e determinar a forma de combater o incidente, ajustes são feitos nas políticas de detecção;

- Os eventos gerados podem ser filtrados para uma melhor visualização.

Mesmo considerando que o IDS e o Security Information and Event Management (SIEM) são de camadas de abstração diferentes, todos se encaixam no mesmo modelo, pois IDS caracteriza intrusões e gera alertas que serão tratados e correlacionados. A Figura 2 ilustra a abstração de ferramentas utilizadas em SOCs para a detecção.

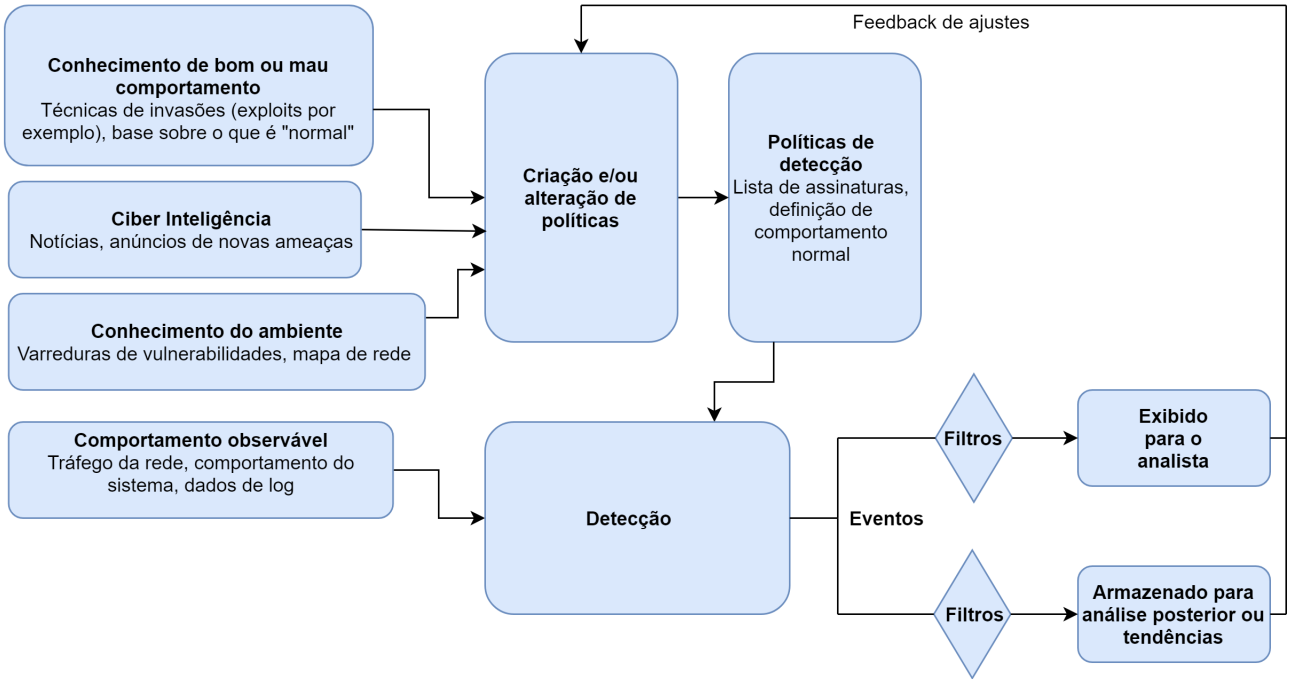

Figura 2 – Esquema de detecção considerando as duas abordagens (ZIMMERMAN, 2014).

2.2.4 Triagem de incidentes

Após um incidente ser identificado por relato de usuários da empresa e/ou analistas por meio de alertas, o incidente deve ser devidamente classificado e analisado buscando um melhor entendimento sobre qual tipo de ameaça se trata. A classificação diz respeito as primeiras ações tomadas pela equipe ao formular um plano de resposta.

A classificação de incidentes é a base das próximas etapas de elaboração de contra-medida e pode ser divida em três fases menores, sendo: verificação, classificação inicial e atribuição. A fase de verificação é quando o analista detecta eventos e analisa a fim de descobrir se existe necessidade de uma investigação mais detalhada sobre aquele evento. Já na classificação é atribuído o nível de ameaça daquele incidente. Atribuição é o ato no qual o analista atribui a responsabilidade sobre como a equipe do SOC vai tratar tal incidente. Algumas são levantadas por parte da equipe quando o incidente é detectado, por exemplo, o incidente em questão é reincidente ou é uma nova ocorrência. A categoria deve ser incluído de acordo com a Tabela 2. Caso seja uma ocorrência já conhecida, existe o cronograma de comportamento do

Tabela 2 – Categorização de incidentes (MUNIZ GARY MCINTYRE, 2016).

Categoria	Tipo	Descrição
0	Teste	É utilizado quando ocorrem testes aprovados como de penetração por, exemplo.
1	Acesso não autorizado	Representa que o invasor obteve acesso físico ou lógico de algum sistema e/ou rede, dados ou recursos do cliente sem as devidas permissões
2	Negação de serviço	Acontece quando um ataque prejudica a disponibilidade ou o funcionamento normal de algum serviço, sistema ou recurso.
3	Código malicioso	Utilizado quando ocorre a identificação da instalação bem-sucedida de algum software malicioso como vírus, *worm*, *trojan horse*, infectando algum Sistema Operacional (SO) ou recurso.
4	Varreduras/ tentativas de acesso	Está incluído nesta categoria qualquer tipo de atividade suspeita buscando preparar o ambiente para um suposto ataque no futuro, seja por instalação de algum software, portas abertas, etc.
5	Investigação	Inclui incidentes que não foram confirmados, mas potencialmente pode se tornar um incidente e devem ser melhor investigados posteriormente.

incidente. O incidente pode fazer parte de várias categorias ou até mesmo mudar de categoria com o decorrer de investigações e novas descobertas.

Além da triagem em categorias, o incidente também necessita da atribuição de nível de severidade. Os níveis são baseados no impacto esperado do incidente. Com o nível de severidade definido é possível a priorização de incidentes mais graves, a alocação de recursos para a resolução, assim como na categorização um incidente pode ser escalado e ter vários níveis de severidade com o decorrer da investigação. Os níveis de incidentes mais comumente utilizados são:

- Baixo: incidentes com impacto mínimo, mas com potencial significativo ou severo nos serviços, operações, recursos.
- Médio: incidentes com impacto significativo, ou com potencial para um impacto mais severo nas operações.
- Alto: incidentes com grande impacto nas operações.

Após a devida classificação de categoria e severidade, segue a fase de resposta ao incidente, com isso as contramedidas definidas com o plano de resposta baseado em dados obtidos

através das análises, fontes externas de segurança, assinaturas com o perfil do incidente, entre outras. Não existe sequência de passos exatos ao responder incidentes, alguns casos são feitas mais análises antes da contenção do invasor, como casos nos quais a contenção é efetuada imediatamente. Na fase de contenção é possível incluir alguns sub-passos como: desconectar o sistema infectado da rede; mover o sistema para quarentena; visando o comportamento do invasor e obter mais dados a respeito do mesmo; parar algum serviço; adicionar novas regras no *firewall* da rede; inserir novas assinaturas nos sistemas de IPS para detecção e bloqueio do invasor.

2.2.5 Encerramento do incidente

Após a implantação do plano de resposta contra incidentes, a equipe do SOC deve notificar o responsável pela aplicação solicitando a correção de todas as vulnerabilidades exploradas pelo invasor. Após a correção, testes devem ser efetuados para se ter a certeza que não haverá chances de repetição do mesmo incidente. Caso o incidente viole alguma lei a equipe do SOC deve entrar em contato com as autoridades. Alguns exemplos de ações a serem tomadas após incidentes podem ser: reconfiguração de sistemas, atualizações de software, exclusão de contas de usuários e/ou arquivos. Com as devidas correções implantadas e testadas a equipe do SOC irá documentar tudo que foi feito, todos os resultados de análises, categorização do incidente, forma de resposta ao incidente, comportamento adotado por parte da equipe. Com todas as ações documentadas são realizadas novas análises buscando extrair alguma informação nova, padrão de comportamento por parte do invasor, buscando gerar mais conhecimento que possa ser utilizado na tentativa de prevenção a futuros ataques.

2.2.6 Modelos organizacionais

O presente trabalho realizará a categorização de SOC descrita por Killcrece Klaus-Peter Kossakowski (2003), os modelos foram divididos em:

1. Equipe de segurança: Não tem a capacidade de detectar ou até mesmo responder a incidentes. Em casos que um incidente foi confirmado os recursos são reunidos dentro do grupo buscando identificar o motivo do incidente e como lidar com o mesmo, logo que possível desativando o sistema infectado. Em SOCs que utilizam este modelo não existem formas de monitoramento centralizado, os processos para respostas a incidentes geralmente são mal definidos, assim afetando o resultado final. As empresas que geralmente se enquadram neste modelo normalmente não passam de 1.000 usuários e/ou endereços Internet Protocol (IP);

2. SOC distribuído interno: As equipes de SOCs que se encaixam neste modelo não trabalham somente com segurança e sim com assuntos relacionados a Tecnologia da

Informação (TI). O SOC possui uma equipe pequena de pessoas envolvidas na missão. Em casos de incidentes, as funções que necessitam de equipes mais especializadas são terceirizadas. As empresas que melhor se encaixam neste modelo possuem entre 500 e 5.000 usuários e/ou endereços IPs;

3. SOC centralizado interno: A equipe que trabalha com SOCs deste modelo já podem ser mais especializadas e possuem o cargo diretamente voltado a defesa da rede. Todos os recursos são providos pela equipe e a operação possui orçamento próprio e sua declaração formalizada dentro da empresa. A equipe possui um gerente de SOC no qual monitora toda a defesa da rede da empresa. A maior parte dos SOC existentes se encaixam neste modelo, atendendo empresas com um número variado de usuários e endereços IPs, variando entre 5.000 e 100.000.

4. SOC distribuído e centralizado: É modelo hibrido que utiliza os pontos positivos dos modelos apresentados anteriormente. É possível ter uma equipe centralizada, mas contando com recursos de fora, por exemplo de outros departamentos de TI que não lidam necessariamente com segurança. Este modelo geralmente é indicado em casos nos quais a empresa necessita de defesas em filiais geograficamente separadas e/ou possuem um ambiente computacional extremamente heterogêneo. A variância entre usuários e endereços IPs deste modelo é entre 25.000 até 500.000;

5. SOC coordenado: Funciona exatamente como um sistema distribuído, com um SOC central que coordena outros SOC menores, provendo serviços de consultoria disponibilizando a troca de informações entre os SOC subordinados. Este modelo abrange SOC a nível nacional. Utilizando a Universidade Tecnológica Federal do Paraná (UTFPR) como exemplo, seria como se cada câmpus possuísse um SOC local e em Curitiba existisse um SOC central, o qual disponibiliza informações a respeitos dos incidentes e/ou eventos em análise.

2.2.7 Autoridade

Diferentes implantações de SOC podem possuir níveis de autoridades de acordo com a necessidade. Existem alguns pontos a serem considerados sobre o tipo de autoridade que o SOC irá possuir em caso de incidentes. De acordo com Killcrece Klaus-Peter Kossakowski (2003), os possiveis níveis de autoridade são:

- Autoridade nula: Neste caso o SOC não pode definir o plano de resposta contra o incidente e nem implantá-lo. O papel do SOC nessas ocasiões é somente a respeito de como comportar-se diante do incidente. Cabe aos superiores decidirem se devem seguir ou não tais recomendações.

- Autoridade compartilhada: o SOC continua com o papel de prover recomendações a altos cargos da empresa, como Chief Executive Officer (CEO), Chief Information Security Officer (CISO), Chief Information Officer (CIO) que possuem a autoridade de ordenar a implantação do plano de resposta recomendado. Tais recomendações possuem uma importância, nesses casos o SOC no momento da decisão possui uma voz ativa, mas nunca a palavra final.
- Autoridade total: o SOC pode prover recomendações da mesma forma que outros casos com menor autoridade, mas não precisa de aprovação de superiores para a implantação da resposta, a autoridade do SOC define como será o comportamento diante do incidente.

Existem casos em que a autoridade do SOC é simplificada em apenas duas opções, sendo: proativa e reativa. Reativa, sendo medidas temporárias e mais táticas, afetam apenas as pessoas que trabalham diretamente com o sistema envolvido no incidente em questão. Um exemplo seria isolar um *host* infectado de forma lógica e física. Já na autoridade proativa, as medidas são tomadas buscando a prevenção de ameaças antes mesmo que sejam confirmadas, as medidas tomadas são em sua maioria de longo prazo e de natureza estratégica. Exemplos dessa autoridade proativa seria envio de *patch* de correção de vulnerabilidades, adição de regras em *firewall*, redefinições de senhas.

O SOC proposto neste trabalho adota o modelo organizacional distribuído interno. Foi adotada a metodologia virtual para minimizar os gastos. O SOC possui autoridade nula. A utilização dos conceitos de SOC poderão sofrer alterações em trabalhos futuros com o amadurecimento do sistema.

2.3 Pacote ELK

Considerando o objetivo de centralização de *logs* relacionado a centralizando *logs* de forma que seja possível de maneira fácil e rápida realizar análises simples ou complexas, consultas, relatórios, o pacote de software Elasticsearch, Logstash, Kibana (ELK) possui as características que vão de encontro com as necessidades.

2.3.1 Elasticsearch

O ElasticSearch é uma ferramenta de busca distribuída e altamente escalável desenvolvida utilizando Apache Lucene (N.V., 2015b). Seu desenvolvimento teve inicio em 2010 por Shay Banon (N.V., 2015b) e se trata de uma ferramenta *open source* com uma grande comunidade ativa e colaborativa. Com a utilização do ElasticSearch é possível realizar o armazenamento, pesquisa e análise de um grande volume de dados. Esses dados podendo ser previamente

estruturados ou não. Exemplo de fontes de dados não estruturados: *Logs* de diferentes tecnologias; Documentos variados; *Tweets*; Fontes de dados em forma de texto plano.

Sua arquitetura foi desenvolvida sob o conceito da utilização de *clusters* possibilitando a distribuição do processamento dos dados em diferentes nós, assim aumentando sua escalabilidade e tolerância a falhas. Seu modelo de consultas aceita todo tipo de consultas sendo complexa e/ou flexível. Dessa forma disponibilizando várias formas de triagem dos dados baseado nos critérios que melhor atenderem as necessidades do usuário.

2.3.2 Logstash

Juntamente com o lançamento da empresa ElasticSearch Inc. em 2012, outro projeto estava em desenvolvimento para complementar o funcionamento do mecanismo de buscas. O projeto se chamava Logstash (N.V., 2015b), cujo objetivo é facilitar a coleta, normalização e/ou estruturação dos dados e o envio de dados para um determinado destino como o mecanismo de busca Elasticsearch ou qualquer outra tecnologia utilizada pelo usuário. O Logstash funciona como uma *pipeline* de ingestão de dados no qual permite o usuário estruturar qualquer tipo de dado, normalizando esses dados, filtrar informações irrelevantes e envie diretamente para um sistema de armazenamento e/ou análise. Seu processo de ingestão de dados pode ser divido em três etapas distintas sendo elas: Entrada; Filtragem;Saída.

Na etapa de entrada os *logs* são consumidos de inúmeras fontes e/ou uma única fonte, por exemplo: banco de dados, filas de mensagem, sistemas diferenciados, entre outras. Após a etapa de ingestão dos *logs* se inicia a etapa de filtragem, o processo permite o usuário filtrar dados que possam ser considerados irrelevantes, estruturar a informação da melhor forma, enriquecer os dados, aplicar enriquecimento dos dados, etc. Na última etapa após realizar toda estruturação do dado, o Logstash irá enviar os dados ao destino desejado, seja para armazenamento, análises ou qualquer outro tipo de finalidade.

2.3.3 Kibana

Juntamente ao desenvolvimento de uma ferramenta que facilitaria o envio dos *logs*, existia a necessidade de facilitar a visualização desses dados. Esse desafio ficou sob a responsabilidade de Rashid Khan (N.V., 2015b) que estava trabalhando com o desenvolvimento de uma interface amigável chamada Kibana.

O Kibana funciona como uma ferramenta que facilita de forma amigável e intuitiva a visualização de dados, possibilitando a criação de diversas visualizações em um mesmo espaço chamado de *dashboards*. As visualizações customizadas permitem a execução de consultas previamente definidas em tempo real, possibilitando diversas formas de monitorações.

2.3.4 Beats

Desenvolvido pela Elastic, se trata de diversos agentes de dados leves e escalonáveis (N.V., 2015b). Cada agente foi projetado para realizar a coleta de dados de diversas fontes com a possibilidade de enviar diretamente ao Elasticsearch ou para um Logstash realizar o armazenamento. Cada agente da família Beat é especializado em uma única função específica de coleta de dados, por exemplo, Auditbeat é realizado para coleta de eventos de segurança diretamente nas trilhas de auditoria., Filebeat é usado para coleta de *logs* referente a arquivos, Heartbeat utilizado para monitoramento de disponibilidade de serviços.

2.3.5 Utilização do pacote Elasticsearch, Kibana, Logstash

O Elasticsearch irá armazenar e consultar os eventos enviados com o uso do Auditbeat, por fim possibilitando a visualização e análise utilizando o Kibana. Como o destino dos *logs* será o Elasticsearch, a utilização do Auditbeat se mostrou mais simples, porém atendendo as necessidades.

2.4 Trabalhos Relacionados

Neste capítulo são apresentados trabalhos relacionados ao tema da presente monografia, justificando suas contribuições e/ou diferenças em relação ao trabalho apresentado.

2.4.1 Gerenciamento centralizado de *logs* usando Elasticsearch, Logstash e Kibana

O trabalho desenvolvido por Ahmed Farrukh e Jahangir (2020) consiste na utilização do pacote de aplicativos disponibilizado pela empresa Elastic para coleta, consumo e análise de dados. Os aplicativos definidos foram divididos em 4 módulos, sendo:

1. Módulo de visualização: módulo que realiza consultas diretamente no Elasticsearch e utiliza as análises para criação de relatórios e *dashboards*;
2. Módulo de banco de dados: módulo pertencente ao Elasticsearch que realiza a conexão entre os coletores e a interface de visualização;
3. Módulo de ingestão: seu funcionamento é simples e se resume na estruturação dos dados, higienização das informações. Essa análise é realizada utilizando Logstash;
4. Módulo de coleta: a coleta de dados utilizando Filebeat realizando a estruturação e enriquecimento dos dados no Logstash antes de enviar os dados para armazenamento no Elasticsearch.

A utilização do pacote ELK possibilitou os autores a ingestão de mais de 2 milhões de eventos. Com essa volumetria de dados o desenvolvimento de métricas mais assertivas sobre a qualidade dos produtos. Para facilitar a visualização, mais de 12 *dashboards* foram desenvolvidas, dentre todas as visualizações criadas as principais foram: painel de visão geral da execução dos testes, painel de visão de projeto, painel de relatórios personalizados. Anteriormente a utilização do pacote ELK essas informações eram de difícil extração e visualização. O presente trabalho difere do método de análise definidos pelo autor, no processo definido no trabalho detalhado acima ainda existe a necessidade da interação humana dentro da plataforma, a análise desenvolvida e detalhada no 3 é de forma automatizada descartando qualquer interação humana obrigatória.

2.4.2 GT-BIS

O trabalho de Batista (2017) consiste em um protótipo de análise para *big data* com a função de detectar incidentes de segurança em ambientes computacionais. Os dados analisados pelo protótipo são provenientes de sistemas de IDS e sistemas de *logs*. Já no processo de análise são utilizados processos envolvendo Inteligência Artificial (IA) e Aprendizagem de Máquina correlacionando os eventos de segurança. O intuito de utilização desses métodos de IA é detectar e evidenciar casos de incidentes que passaram como falso negativo em outros sistemas da mesma natureza. A arquitetura desenvolvida e detalhada no Capítulo 3 utilizou como referência a arquitetura proposta no trabalho de Batista (2017), com o foco para a tecnologia utilizada no protótipo.

A utilização de técnicas de IA torna o objetivo de viabilização do SOC proposto neste trabalho mais complexa, assim será descartado a utilização de tais técnicas. A pesquisa de Batista (2017) se assemelha ao presente trabalho em relação a detecção de incidentes baseado no consumo de dados coletados a partir de sensores.

2.5 Considerações finais

Com todos os conceitos de SOC juntamente com a explicação referente as ferramentas que serão utilizadas, foi possível estabelecer uma base sólida de conhecimento para realizar o desenvolvimento do presente trabalho.

2.6 Considerações

Com os conceitos apresentados nos Capítulos 2 e 2.4 é possível compreender o funcionamento do SOC, metodologias e processos utilizados. A metodologia e desenvolvimento será detalhada no Capitulo 3.

3 MATERIAIS E MÉTODOS

Neste capítulo são detalhado os métodos, ferramentas, decisões de projetos para a implementação do sistema proposto. O capítulo é divido em materiais, métodos e desenvolvimento para atingir os seguintes objetivos:

1. Viabilização da utilização de SOC para empresas de pequeno e médio porte;
2. Implantação de centralização de fontes de dados (*logs*);
3. Implantação de interface de visualização;
4. Análise dos dados com mais facilidade e rapidez;
5. Detecção de possíveis incidentes em tempo real e notificação sobre eventos encontrados.

3.1 Materiais

Visando facilitar a implementação e futuras manutenções se deu a necessidade da utilização de toda infraestrutura disponibilizada gratuitamente pela empresa Elastic. Dentre os softwares disponibilizado os utilizados são detalhados a seguir.

3.1.1 Elasticsearch

O Elasticsearch foi definido como principal mecanismo de busca, devido a sua facilidade de integração com os demais aplicativos disponibilizados em seu pacote. A versão utilizada se limita na utilização de apenas um nó em seu plano gratuito de assinatura. A ferramenta possui uma arquitetura distribuída que possibilita a utilização de vários nós, distribuindo o processamento de consulta de dados. Com a utilização de diversos nós a sua escalabilidade aumenta, porém o processo de manutenção se torna mais complexo juntamente com o maior consumo de recursos.

3.1.2 Kibana

Com a utilização do Elasticsearch como mecanismo de busca, a necessidade da visualização dos dados foi atendida com a utilização do Kibana. Possibilitando a criação e importação de várias *dashboards* disponibilizadas em forma de *plugins*.

3.1.3 Auditbeat

Como agente coletor de dados foi definido o Auditbeat (N.V., 2015a), devido as avançadas formas no qual é possível realizar a coleta, estruturação e monitoramento de dados do sistema operacional. Sua desvantagem é a sua limitação de funcionamento apenas em sistemas Linux. Juntamente com a utilização do agente a própria Elastic disponibiliza uma série de *dashboards* previamente prontas que tratam e analisam os dados coletados pelo agente de diferentes formas enfatizando diversos tipos diferentes de informações.

3.1.4 Python

A linguagem de programação utilizada foi Python (FOUNDATION, 2001), devido a sua codificação fácil obtida através de seu paradigma de programação interpretado. Juntamente com a linguagem utilizou-se os seguintes recursos:

- Elasticsearch: responsável por realizar a integração com o *cluster* (ELASTICSEARCH B.V, 2023);
- Json: responsável por codificar e decodificar os dados utilizando o protocolo JavaScript Object Notation (JSON) (PYTHON SOFTWARE FOUNDATION, 2023a);
- Smtplib: responsável por criar um objeto de sessão utilizado para envio e recebimento de e-mails utilizando o protocolo Simple Mail Transfer Protocol (SMTP) (PYTHON SOFTWARE FOUNDATION, 2023b).

3.1.5 Servidor para envio de e-mails

Buscando uma menor complexidade e uma manutenção fácil, foi definido a utilização do envio de alertas via e-mail utilizando o protocolo SMTP, o servidor utilizado foi o Outlook (MICROSOFT, 2023). O servidor SMTP será responsável por enviar alertas contendo as informações do evento encontrado relacionado com os critérios de busca pré estabelecidos. As informações enviadas são: descrição do evento contendo o usuário que executou a ação, comando executado, resultado da operação, endereço do *host* onde se originou a ação, orientação como validação do evento e orientação sobre contenção do evento.

3.1.6 Descrição do sistema

O sistema usa cada material citado anteriormente definindo uma responsabilidade para cada, sendo:

- O Auditbeat é responsável por coletar os eventos gerados com a utilização do *framework* de auditoria do Linux, após a coleta dos eventos é realizado o processamento e estruturação do dados, por exemplo, remoção de dados considerados irrelevantes para o projeto e/ou dados duplicados, alteração no nome dos campos;
- Após a coleta e estruturação o dado será enviado para o Elasticsearch que possui a responsabilidade de armazenar e posteriormente consultar os dados armazenados;
- Após a coleta e armazenamento dos dados, é necessário uma alternativa que possibilite o usuário visualizar e consumir esses dados, essa responsabilidade será relacionada com o Kibana, possibilitando análises para geração de relatórios, busca por padrões, etc;
- Com toda estrutura referente aos dados implementada o sistema possui um *script* desenvolvido em Python que processa os dados diretamente no Elasticsearch buscando padrões definidos na implementação do *script*;
- Quando esses padrões são encontrados, será criada e transmitida uma estrutura de alerta contendo as informações relevantes que será responsável pelo envio do alerta para o destinatário definido no *script* Python.

3.2 Métodos

Durante a modelagem do sistema alguns processos foram definidos com o objetivo de facilitar a implementação. O primeiro processo realizava a divisão do sistema em 4 módulos distintos, sendo: geração de eventos; coleta, estruturação e armazenamento; consumo de dados; notificação de alertas.

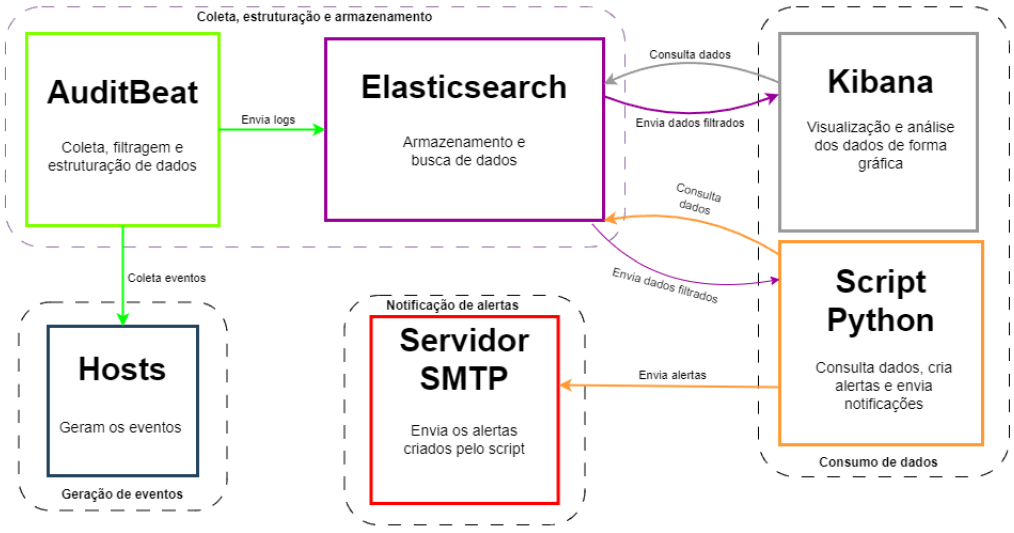

Figura 3 – Arquitetura geral do sistema.

3.2.1 Geração de eventos

No grupo de geração de eventos ficou definido a utilização do *framework* de auditoria do Linux (CANONICAL LTD, 2019a), que gera dados sobre:

- *Host*: tempo do sistema ativo, endereços IP, hardware, etc;
- Autenticações: entrada e saída de usuários, inicializações do sistema;
- Processos: processos iniciados e parados;
- *Sockets*: abertura e fechamento de conexões utilizando *sockets*;
- Usuários: informações sobre os usuários;
- Integridade dos principais diretórios do sistema.

3.2.2 Coleta, estruturação e armazenamento

O método de coleta constitui-se a leitura do arquivo padrão de auditoria chamado `audit.log`, com a leitura e coleta dos dados o agente do Auditbeat realizou a etapa de estruturação de dados. Na estruturação foi definido não alterar nenhuma informação contida no evento, desta forma garantindo a integridade da informação. Com toda coleta e estruturação pronta foi realizado o armazenamento com o Elasticsearch, os dados foram enviados e armazenado utilizando a estrutura padrão do componente, na forma de documentos JSON.

3.2.3 Processamento de dados

O processo de consumo de dados para visualização e análise visual foi a utilização da integração direta entre Kibana e Elasticsearch, e o consumo de dados via *script* foi realizado utilizando a biblioteca desenvolvida pelo fabricante do Elasticsearch para conexões com *scripts* Python (ELASTICSEARCH B.V, 2023), assim simplificando o processo de consumo e futuras manutenções.

3.2.4 Notificação de alertas

A notificação de alertas utilizou o envio de e-mails com o intuito de facilitar a leitura e visualização dos alertas enviados. A utilização de um servidor SMTP gratuito facilitou o processo dos envios, pois disponibiliza autenticações simplificadas e conexão segura para transmissão das informações.

3.3 Escopo do sistema

O sistema de monitoramento de eventos deve coletar e armazenar todos os eventos gerados a partir de *hosts*, após a coleta e armazenamento uma análise será executada de forma automática buscando eventos que se encaixem nos critérios de buscas detalhados na Subseção 3.5.9. A visualização dos dados coletados é feita com a utilização de uma interface web disponibilizada pela ferramenta de visualização. Os dados são centralizados todos na mesma plataforma.

O sistema tem o seu foco na detecção de eventos e criação de alertas, inicialmente o sistema não possui funcionalidades para realização de contenções, sendo possível adição de novas funcionalidades em futuras melhorias.

3.4 Apresentação do sistema

O sistema funciona de forma automática em relação a coleta, processamento, armazenamento e análise dos dados. Para realizar outras análises e configurações o usuário deverá se conectar na rede interna do DACOM e se conectar pelo endereço 172.16.2.159:8080. Após acessar deverá efetuar autenticação informando usuário e senha de um usuário válido. Na Figura 4, é possível identificar o menu no lado superior direito onde é disponibilizado com todos os módulos que o usuário poderá utilizar. Para análise e visualização dos *logs* armazenados, deverá acessar o modulo *Analytic* na opção*discover*. Caso deseje acessar as *dashboards* a opção acessada deverá ser a segunda opção (*Dashboard*).

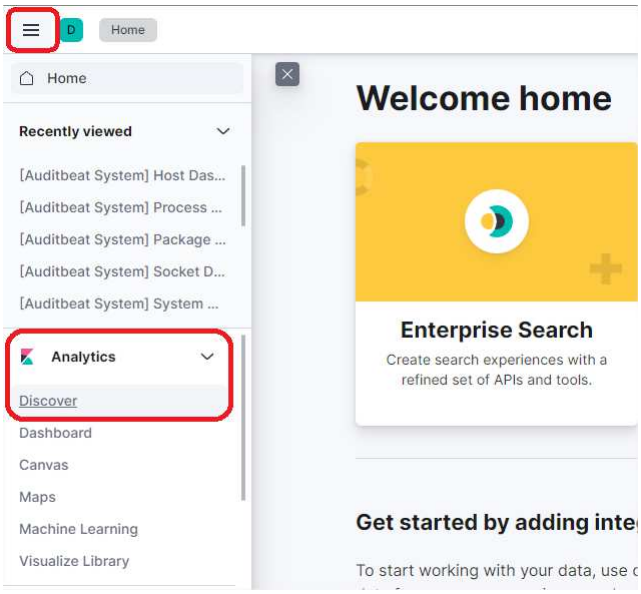

Figura 4 – Tela inicial do sistema

3.4.1 Análise e visualização de dados

A Figura 5 apresenta a tela de visualização de dados. A demarcação 1 indica onde deverá ser selecionado o índice referente a tecnologia desejada. Na demarcação 2 apresenta o campo onde deve ser informado a consulta que será utilizada para filtragem dos dados utilizando Kibana Query Language (KQL). Após realizada a consulta a demarcação 3 indica o quadro onde é possível realizar uma segunda filtragem referente aos campos dos *logs* encontrados como resultado da busca. Por fim, a demarcação 4 demonstra onde serão apresentados todos os *logs* resultantes da consulta e filtragem realizada.

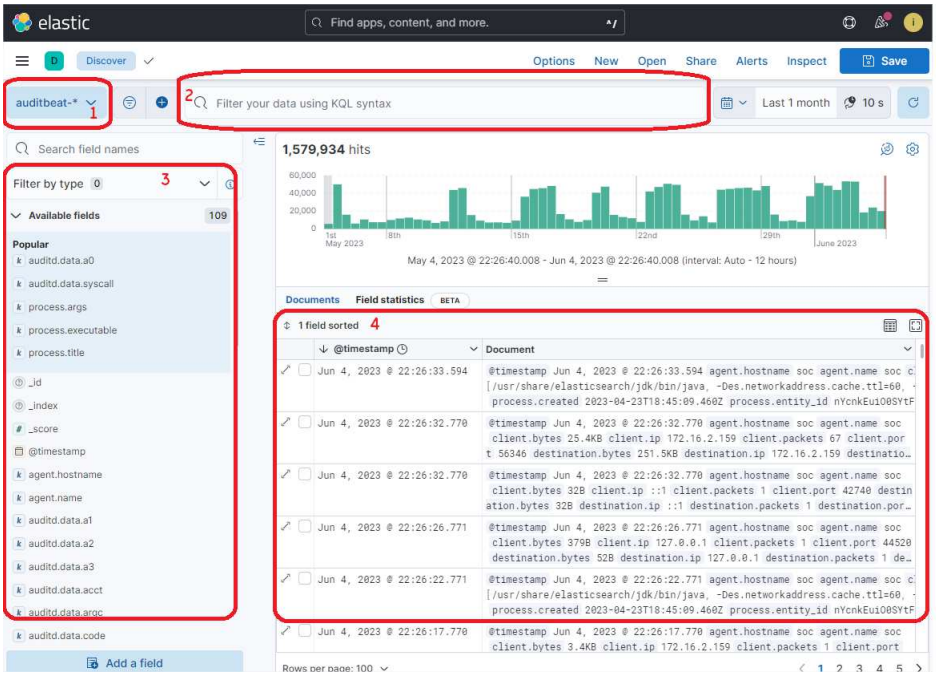

Figura 5 – Tela para consulta de *logs*.

3.4.2 *Dashboards*

A utilização de componentes disponibilizados por parte da fabricante acaba trazendo vantagem com a implantação de *dashboards* previamente desenvolvidas e configuradas referente a tecnologia do agente instalado, facilitando a visualização de dados possivelmente relevantes. Na Figura 6 é possível visualizar análises gerais sobre os sistemas onde foram instalados agente coletores e se encontram ativos, possibilitando uma visualização geral da monitoração. Porém, quando necessário é possível aprofundar a visualização apresentada. Baseado nas informações apresentadas na Figura 6 seria possível extrair informações sobre quantos *hosts* estão com monitoramento ativo e enviado dados para o Elasticsearch, o número total de autenticações realizadas durante o intervalo de tempo selecionado, a distribuição de sistemas operacionais, alterações em pacotes do sistema, número de processos em execução, suspensos e iniciados, *sockets* de comunicação abertos e fechados.

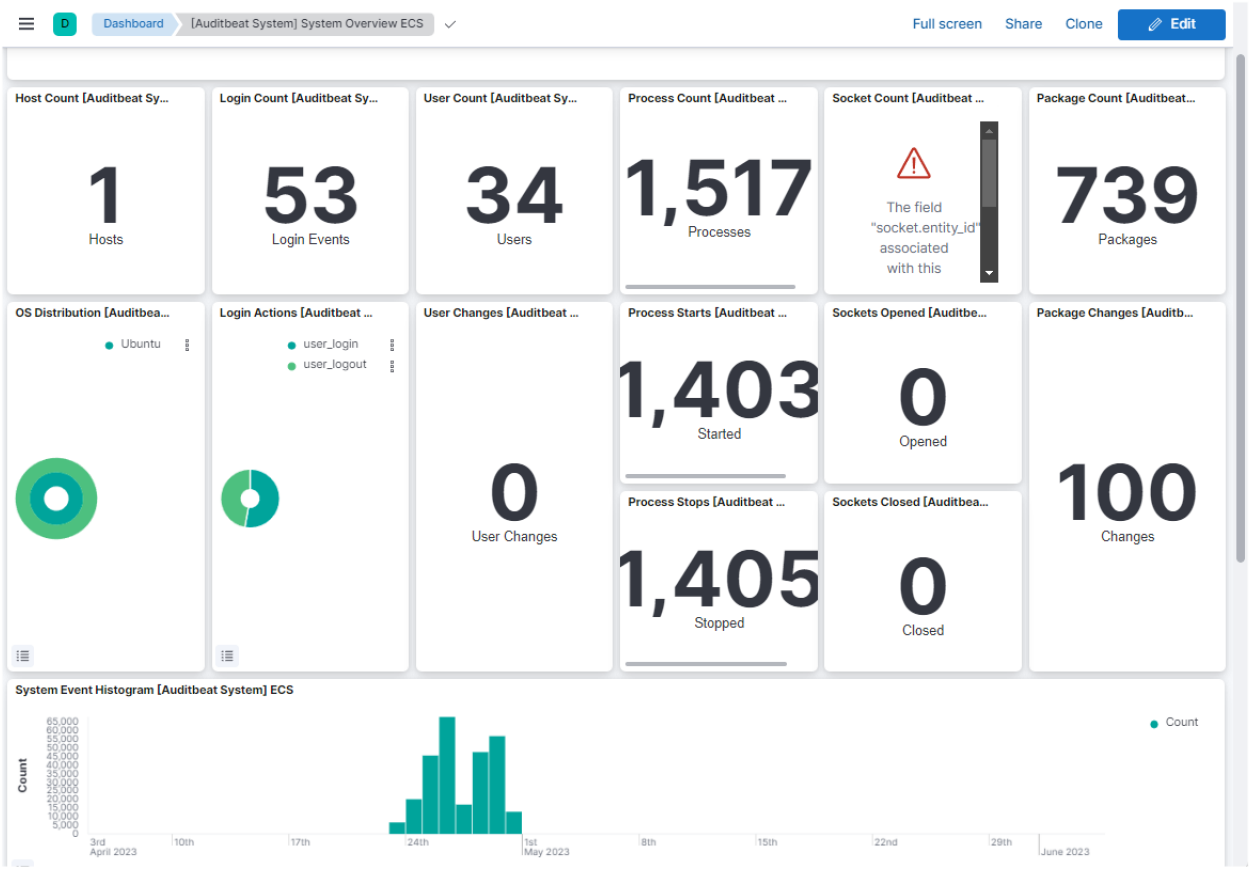

Figura 6 – *Dashboard* geral referente a tecnologia Auditbeat.

3.4.3 Monitoramento de infraestrutura e gerenciamento de usuários

Para visualizar o monitoramento referente a infraestrutura do sistema, o usuário deverá acessar o menu esquerdo lateral e selecionar *Stack Monitoring* dentro da seção *Management*. Ainda na opção de *Management* é possível encontrar o gerenciamento de usuários, opção onde o administrador da plataforma poderá criar, alterar ou remover usuários como demonstrado na Figura 7.

3.5 Implementação do sistema

Após a apresentação do escopo e do sistema detalhado anteriormente, nesta seção são abordadas as questões técnicas relacionadas à implementação do sistema. Serão discutidas as documentações, os passos e as configurações nos componentes.

3.5.1 Instalação e configuração do mecanismo de pesquisa (Elasticsearch)

Inicialmente foi realizado o provisionamento do ambiente no qual todo o sistema foi instalado. Foi disponibilizada uma Virtual Machine (VM) com a seguinte configuração:

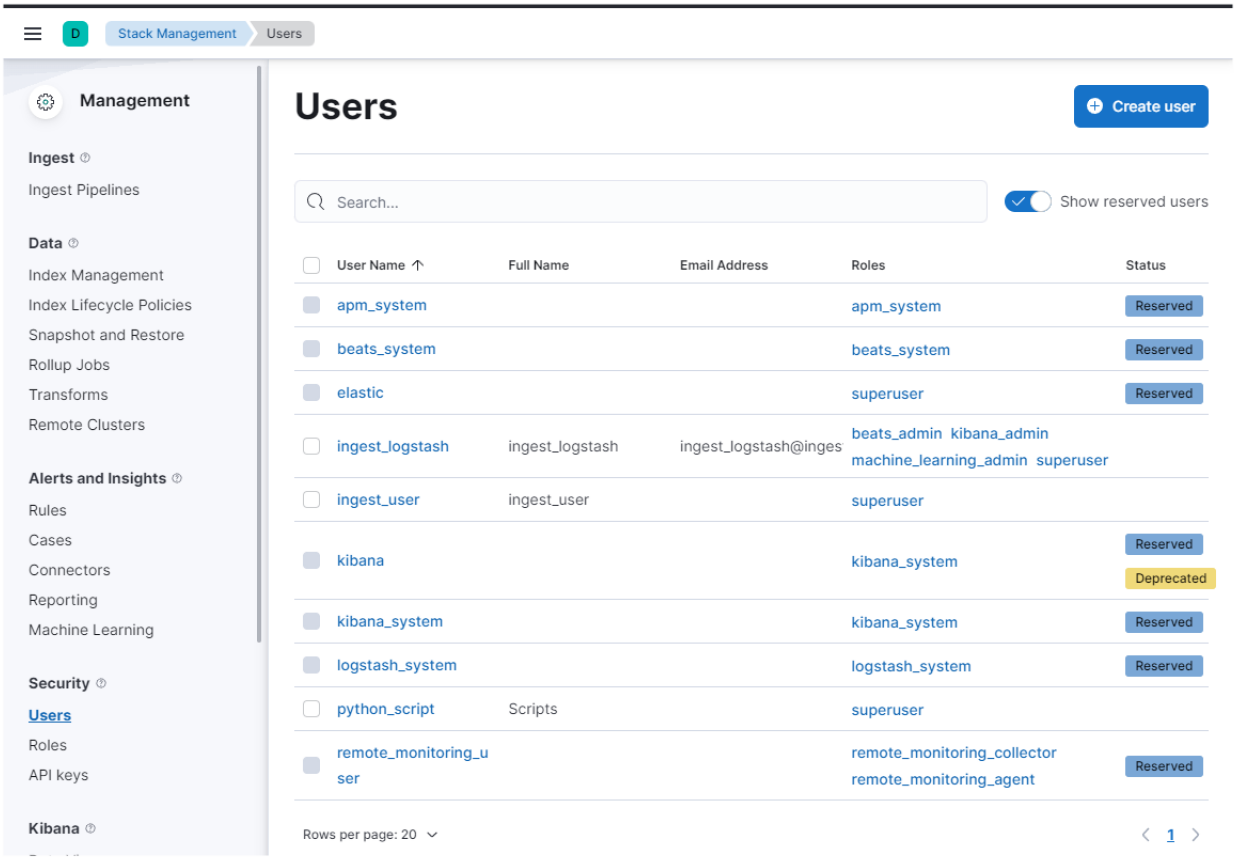

Figura 7 – Tela de gerenciamento de usuários.

- i7-7700K CPU @ 4.20GHz;
- 108 GB de armazenamento;
- 4 GB de memória RAM;
- Ubuntu server 18.04.6 LTS (Bionic Beaver).

Após o provisionamento do ambiente, realizou-se a instalação e configuração do mecanismo de busca (Elasticsearch), versão 8.6.

Todos os passos da instalação e configuração foram baseados na documentação oficial disponibilizada no *site* da fabricante (ELASTICSEARCH B.V., 2023b). Primeiramente foi efetuado o download da chave de assinatura digital disponibilizada pela empresa Elastic, essa chave foi utilizada na verificação dos pacotes obtidos e instalados no sistema. Após o download da chave foi realizado a instalação do pacote *apt-transport-https* para permitir o download de pacotes utilizando o protocolo Hyper Text Transfer Protocol Secure (HTTPS). Logo após foi realizada a adição do repositório oficial do Elasticsearch na versão 8.7 e realizado o download e instalação com a utilização do gerenciador de pacotes APT (CANONICAL LTD, 2019c). Após a instalação e configuração inicial algumas informações extremamente relevantes foram disponibilizados no arquivo de *log* próprio do sistema. Dentre as informações disponibilizadas estavam:

- Orientações para conectar o nó diretamente com um *cluster*;

- Comandos para geração de *tokens* para outros nós ou Kibana;
- Redefinição de senha do super usuário;
- Senha para o super usuário criado como padrão.

Após a configuração inicial foi realizado a ativação de criação automática de índices pelo próprio Elasticsearch. Foi necessário realizar a inicialização do serviço manualmente na primeira vez, após a inicialização foi realizado a vinculação de inicialização do serviço com o SO, desta forma o serviço será inicializado automaticamente com a inicialização do sistema. Em sua primeira configuração, o serviço realizou a criação de uma Certificate Authority (CA) que foi utilizado para a criação de certificados digitais utilizados para realizar a transferência de dados totalmente criptografados utilizando o protocolo Transport Layer Security (SSL). Portanto foram criados 3 arquivos:

1. `http_ca.crt`: Autoridade Certificadora utilizada para criação de certificados digitais e assinaturas para o *cluster*;
2. `http.p12`: *Keystore* contendo a chave privada e o certificado que será utilizado para a transmissão de dados deste nó especifico;
3. `transport.p12`: *Keystore* contendo o certificado e a chave privada que foi utilizada em todo transporte de dados entre todos os nós existentes no *cluster*.

Na Listagem 1 é visualizado o arquivo de configuração por completo. Nas linhas 1 e 2 foi realizado a configuração ativando as opções padrões de segurança para o *cluster* (nó principal) e as demais integrações que foram implantadas. Nas linhas 3, 4 e 5 foi definido o protocolo para as conexões com Kibana e AuditBeat. Definido o local do certificado que deve ser utilizado para criptografar toda a transmissão de dados. Entre as linhas 6 e 10 estão definidas comunicações para eventuais futuros nós, permitindo a comunicação utilizando o protocolo HTTPS. Por fim, a linha 11 define o nome do *cluster* como soc e finaliza com a linha 12 definindo o endereço do *host*.

3.5.2 Instalação e configuração do software de visualização (Kibana)

O procedimento de instalação realizado com o Kibana é o mesmo realizado com o Elasticsearch detalhado anteriormente e também é disponibilizado no *site* da fabricante (ELASTICSEARCH B.V., 2023c). Houve uma tentativa de instalação do Kibana em sua versão mais atual 8.7, porém aconteceram erros de compatibilidade devido a versão instalada do Elasticsearch ser a versão 8.6. Em consequência a este problema foi definido a utilização 8.6 em todos os softwares. A instalação do pacote foi executada utilizando o gerenciador de pacotes APT existente em sistemas baseado em Debian. Após a instalação foi realizado a

Listagem 1 – Arquivo de configuração `elasticsearch.yml`

```
xpack.security.enabled: true
xpack.security.enrollment.enabled: true
xpack.security.http.ssl:
  enabled: true
  keystore.path: certs/http.p12
xpack.security.transport.ssl:
  enabled: true
  verification_mode: certificate
  keystore.path: certs/transport.p12
  truststore.path: certs/transport.p12
cluster.initial_master_nodes: ["soc"]
http.host: localhost
```

Fonte: Autoria própria (2023).

execução do *script* disponibilizado pela Elastic referente a criação de um *token* de vinculação entre Elasticsearch e Kibana. O nome do *script* executado para geração do *token* é `elasticsearch-create-enrollment-token`. Após a geração na primeira conexão utilizando o navegador Web, foi realizada a vinculação entre Elasticsearch e Kibana. Com sua instalação, configuração e vinculação completa foi possível acessar o sistema via navegador web para a criação dos usuários que serão utilizados para as demais integrações, os usuários criados foram:

1. ingest_Logstash;
2. ingest_user;
3. python_script.

É recomentado a utilização do mesmo usuário em todos os coletores, caso aconteça algum comprometimento com o sistema, deve-se revogar o acesso do usuário todo processamento de dados será suspenso.

3.5.3 Instalação e configuração do agente (AuditBeat)

Houve a necessidade de definição de qual agente seria utilizado como sensor, as opções foram AuditBeat desenvolvido pela Elastic ou a utilização do AuditD disponibilizado pelo próprio SO, porém a sua utilização acarretaria na necessidade de instalação e configuração de um coletor de *logs*, neste caso Logstash. Com o intuito de simplificar todo processo de instalação e manutenção, ficou definido a utilização do Auditbeat, pois o agente realiza a coleta, processamento e transmissão dos *logs* diretamente para o Elasticsearch. O Auditbeat possibilita a estruturação dos dados semelhante ao Logstash, porém sua desvantagem é referente ao enriquecimento dos dados, pois o agente é especificamente para coleta de dados de apenas uma tecnologia.

O processo de *download* e instalação é semelhante ao processo utilizado na Subseção 3.5.1 e foi realizado seguindo a documentação disponibilizada no *site* da fabricante (ELASTICSEARCH B.V., 2023a). Na Listagem 2 é demonstrado as configurações referentes aos módulos ativos, arquivos customizados com regras de auditoria, caminho de diretórios que foram selecionados para a monitoração de integridade, ou seja, toda e qualquer alteração gerou um *log* de evento. Logo após foi detalhado os *datasets* de dados coletados e utilizados para a utilização via *dashboards* que foram instaladas juntamente com o agente.

Listagem 2 – Parte do arquivo de configuração `Auditbeat.yml`.

```
1  auditbeat.modules:
2  - module: auditd
3    audit_rule_files: [ '${path.config}/audit.rules.d/*.conf' ]
4    audit_rules: |
5  - module: file_integrity
6    paths:
7    - /bin
8    - /usr/bin
9    - /sbin
10   - /usr/sbin
11   - /etc
12   datasets:
13     - host
14     - login
15     - process
16     - socket
17     - user
```

Fonte: Autoria própria (2023).

É possível observar na Listagem 3 é definido na linha 1 foi ativado o monitoramento de alteração de senha de usuários. Entre as linhas 4 e 10 se trata de configurações de destino para o envio de *logs*, em casos em que são utilizados mais nós, é necessário definir seus endereços IP, configuração de autenticação, configuração referente a criptografia dos dados. Nas linhas finais entre 11 e 17 são definidos a estruturação dos dados, ou seja, é possível enriquecer, transformar e descartar as informações originadas nos *logs*. Para uma melhor higienização foram retirados 44 campos considerados irrelevantes. É possível alterar ou até remover o processamento dos dados, bastaria remover os campos definidos como descarte. Todos os eventos coletados via Auditbeat foram centralizados em apenas um índice específico no Elasticsearch, cujo o nome é `auditbeat-*`, o processo de configuração é detalhado na Subseção 3.5.4. Com a instalação do agente, houve a instalação de 7 *dashboards* desenvolvidas pela Elastic refente ao consumo das informações enviadas pelo Auditbeat.

Listagem 3 – Arquivo de configuração `Auditbeat.yml`.

```
user.detect_password_changes: true
setup.template.settings:
  index.number_of_shards: 1
output.elasticsearch:
  hosts: ["https://localhost:9200"]
  username: "XXXXXXXXXXXX"
  password: "xxxxxxxxxxxxxx"
  ssl:
    enabled: true
    ca_trusted_fingerprint: "xxxxxxxxxxxxxxxxxxxx"
processors:
  - add_host_metadata: ~
  - add_cloud_metadata: ~
  - add_docker_metadata: ~
  - drop_fields:
      fields: [#Local onde deve ser adicionado os campos de filtragem]
      ignore_missing: false
```

Fonte: Autoria própria (2023).

3.5.4 Criação de índice e configuração do ciclo de vida

Durante a instalação do agente AuditBeat, de forma automática o próprio Elasticsearch identifica a necessidade da criação de um índice específico para a tecnologia. Por padrão, definido pela fabricante o índice é criado com o nome da tecnologia do agente que irá popular com dados. Cada índice possui uma configuração própria referente ao ciclo de vida dos dados, para isso existem três possíveis armazenamento de dados sendo: *Hot phase*, *Warm phase*, *Cold phase*. Denominada como *Hot phase* ou fase quente é referente aos dados recebidos em tempo real, isto é, serão ativamente utilizados em consultas e/ou visualizações. A fase quente possui uma melhor indexação, com isso aumentando a desempenho das consultas que foram executadas, porém consumindo mais recursos de hardware. A segunda fase denominada *Warm phase* ou fase morna foi definida como armazenamento para dados com mais de 7 dias mas que existe a possibilidade de serem consultados. Nesta fase os dados possuem uma otimização de indexação para consultas porém é inferior ao desempenho oferecido na primeira fase. Na terceira fase chamada de fase fria ou *Cold phase* é referente a um armazenamento de dados que raramente serão utilizados, não possui otimização de indexação e necessita da configuração de um repositório onde os *logs* serão armazenados, essa fase possui o menor custo entre as três fases. Infelizmente a fase fria é uma funcionalidade exclusiva da versão *enterprise* e não foi utilizada no presente trabalho. Na Figura 8 é possível observar a definição de movimentação de dados da fase quente para a morna utilizando a idade de cada *log*.

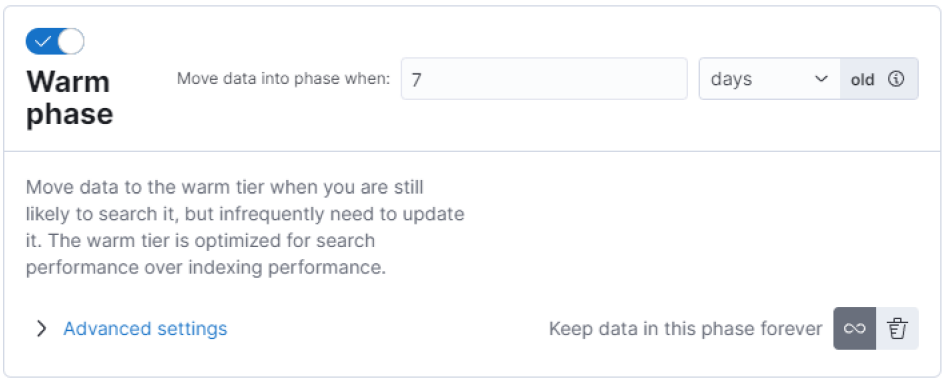

Figura 8 – Configuração da fase morna de armazenamento.

3.5.5 Monitoramento da infraestrutura

Com toda infraestrutura configurada a última etapa foi realizar a implantação do agente Metricbeat com o intuito de coletar e enviar *logs* com informações referente a infraestrutura. A coleta é realizada em todos os nós e *clusters* existentes. É possível observar na Figura 9 informações referentes a uso de disco, versão, quantidade de *logs* indexados, tempo de resposta a requisições feitas no Kibana, etc. Sua visualização é fácil, pois é segregada em informações referente a Elasticsearch, Kibana, instâncias, nós e índices.

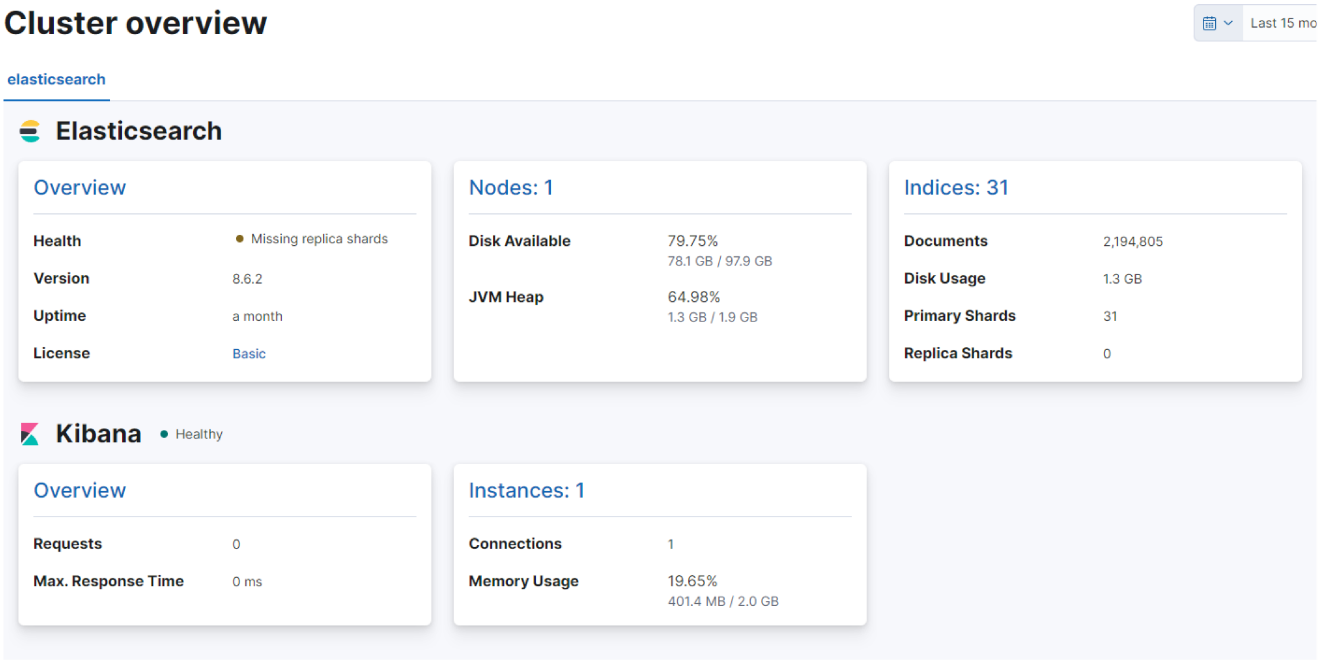

Figura 9 – Monitoramento da infraestrutura usando Metricbeat.

3.5.6 Desenvolvimento do *script* de revisão de *logs*

O intuito da utilização do pacote de softwares Elasticsearch, Kibana e AuditBeat é a ingestão de uma grande volumetria de dados que foram analisados buscando eventos de segu-

rança. Porém esse grande volume dados acaba se tornando humanamente impossível de ser analisa diariamente, devido a essa necessidade, o *script* foi desenvolvido para automatizar essa análise e triagem de dados.

3.5.7 Revisão e análise de *logs*

Para automatizar a revisão diária de *logs*, um *script* em Python foi desenvolvido realizando toda análise e caso encontre eventos definidos em código, dispara um e-mail informativo com o evento em questão. O código está divido em 2 arquivos: `alertas.py` que possui o código fonte que será executado e `config.py` onde é configurado todas as credenciais utilizadas no código.

O código fonte foi implementado em 4 funções principais sendo: *conect_elastic*, *send_email*, *create_alerts* e *search_logs*. As funções implementadas são detalhadas nas subseções a seguir.

3.5.8 Função de conexão (*conect_elastic*)

Na Listagem 4 é detalhado a função de conexão com o Elasticsearch, retornando um objeto com uma conexão ativa utilizando as credenciais informadas, neste caso o usuário *ingest_logstash*. A conexão é o primeiro passo do algoritmo.

Listagem 4 – Função para conexão com o Elasticsearch

```
1 def conect_elastic():
2     client = Elasticsearch(
3         "https://localhost:9200",
4         ca_certs="http_ca.crt",
5         http_auth=(config.elastic, config.elastic_password)
6         )
7     return(client)
```

Fonte: Autoria própria (2023).

3.5.9 Busca (*search_logs*)

Logo após a conexão ser realizada com sucesso, é realizado uma chamada para a função *search_logs* demonstrada na Listagem 5. Essa função tem o objetivo de utilizar os critérios de busca definido na variável *query_body*, os critérios utilizados foram:

1. O campo `auditd.data.syscall` deve ser `execve`. Nas trilhas de auditoria do Linux, o registro `execve` é utilizado para o rastreamento de execuções de aplicações.

Com a utilização desse registro é armazenado as informações sobre caminho do arquivo executável, argumentos de linha de comando, váriaveis de ambiente;

2. O campo `process.name` deve ser `sudo`. O comando `sudo` é utilizado para execução de processos com privilégios administrativos, ou seja, irá filtrar apenas os comandos onde houve uma tentativa de execução com privilégios administrativos;

3. O campo `event.outcome` deve ser `success`. Esse campo irá filtrar apenas os comandos onde houveram sucesso na execução, filtrando possíveis tentativas com falha devido ao permissionamento;

4. O campo `@timestamp` deve ser 1 hora anterior a hora atual. Esse campo registra o momento exato no qual o evento foi gerado.

Dessa forma é possível buscar todos os eventos onde o usuário executou qualquer comando com privilégios de administrador no *host*.

Listagem 5 – Função para busca de eventos pré definidos

```
def search_logs(client):
    query_body = {
      "query": {
        "bool": {
          "must": [{"match": {"auditd.data.syscall": "execve"}},
            {"match": {"process.name":"sudo"}},
            {"match": {"event.outcome":"success"}},
            {"range": {"@timestamp": {"gte": "now-1h","lt": "now"}}}]
        }
      }
    }
    print("Iniciando a busca por eventos \n")
    result = client.search(index="auditbeat-*", body=query_body)
    all_hits = result['hits']['hits']
    if(len(all_hits) > 0):
        print("Criando alerta....")
        criar_alerta(all_hits)
    else:
        print("Não foram encontrados eventos na última hora.")
```

Fonte: Autoria própria (2023).

3.5.10 Criação do alerta e envio via e-mail (*create_alerts* e *send_email*)

Com a triagem de dados realizada na etapa de busca, a função *create_alerts* executa criando uma variável no qual será armazenado apenas as informações relevantes que será enviado no alerta via e-mail. Com os dados relevantes todos estruturados a função *send_email*

inicia sua execução, foi desenvolvido um texto utilizando *tags* html para estruturar o corpo do e-mail, adicionando as informações do evento, assim facilitando a leitura e compreensão do alerta. O envio é realizado via protocolo SMTP utilizando o Outlook. No envio foi configurado: assunto, destinatário, remetente o corpo do e-mail. Na Figura 10 é possível visualizar um exemplo de alerta enviado via e-mail. Os alertas foram enviados utilizando a conta "notificacaoseguran-cautfprcm@outlook.com".

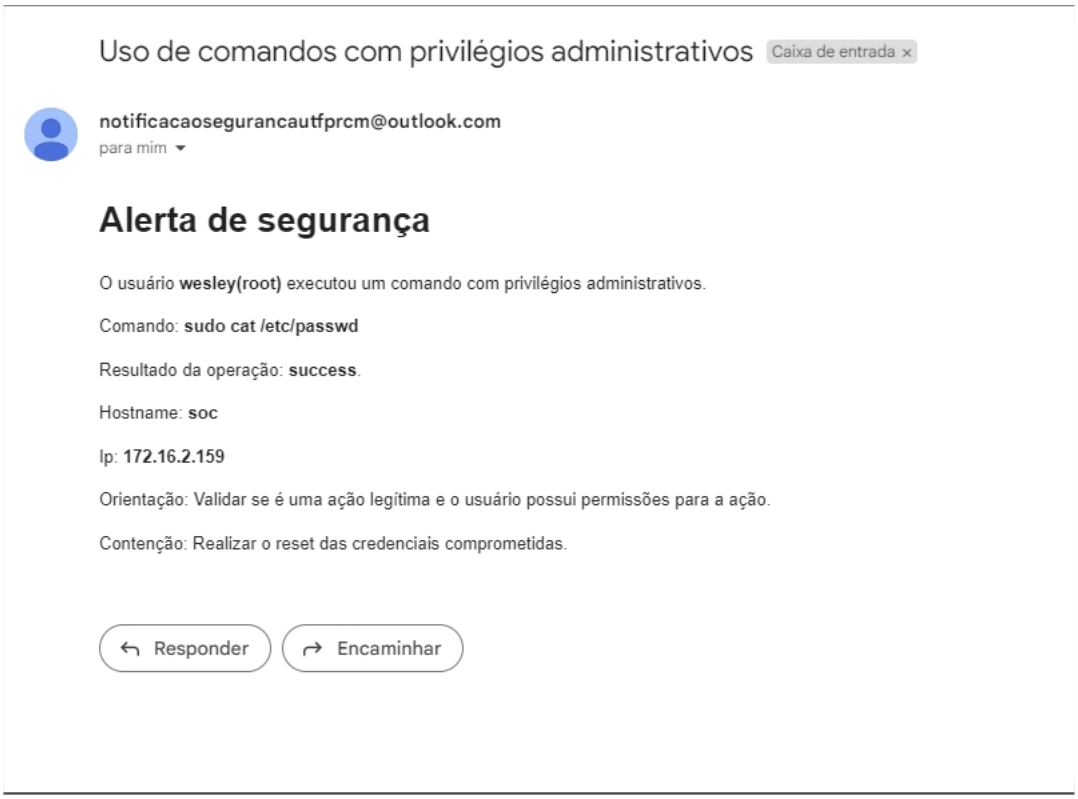

Figura 10 – Alerta recebido via e-mail.

3.5.11 Agendador de tarefas (`Crontab`)

Com o desenvolvimento do *script* a necessidade de triagem e análise dos dados de forma automática foi resolvida, porém outra necessidade apareceu. O problema agora foi uma forma de controlar que um mesmo alerta não seja enviado duas vezes, para isso foi definido o fluxo de execução do *script* a cada uma hora. Dessa forma, analisando apenas os eventos gerados dentro da última hora. A execução foi realizada com a utilização do agendador de tarefas `crontab` que possibilita o agendamento de execução de tarefas no SO (CANONICAL LTD, 2019b). Na Listagem 6 é possível analisar a configuração do agendador de tarefas, a configuração utilizada define a execução do *script* a cada 60 minutos, na implementação do *script* foi configurado para buscar e analisar os dados gerados nos últimos 60 minutos referente ao momento de execução.

Listagem 6 – Configuração do agendador de tarefas (*crontab*)

```
1 0 * * * * /usr/bin/python3.6 /root/scripts/alertas.py
```

Fonte: Autoria própria (2023).

3.6 Considerações finais

Após a finalização da implementação do sistema, a Figura 11 detalha o fluxo de execução do *script*, possibilitando uma visualização geral de todas as possíveis execuções. Os experimentos e resultados são apresentados no capítulo a seguir.

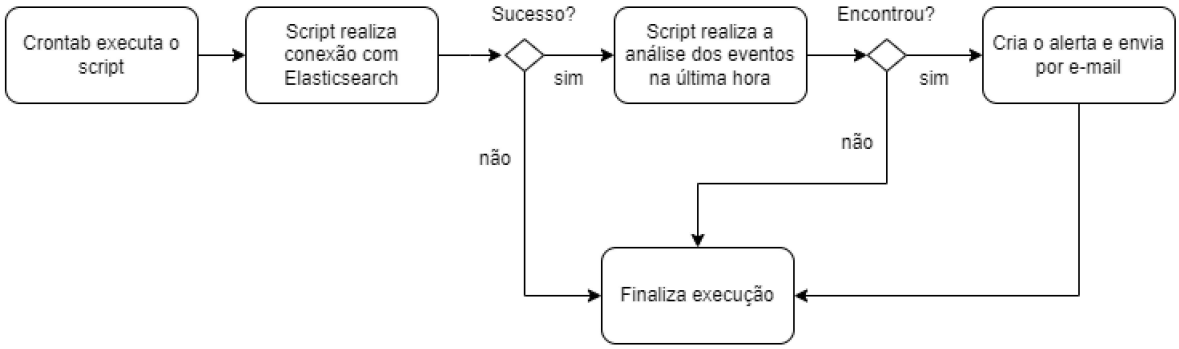

Figura 11 – Fluxograma da execução do *script*

4 EXPERIMENTOS E RESULTADOS

Iniciaremos o capítulo com a apresentação dos resultados juntamente com uma breve análise quantitativa, logo a seguir é apresentado e explicado o gráfico extraído diretamente na interface de visualização dos dados.

Baseado nos conceitos apresentados no Capítulo 2 podemos classificar o sistema desenvolvido com o modelo organizacional de SOC distribuído interno, pois seu escopo é especificamente em detectar eventos pré-definidos. Sua autoridade é nula, seu objetivo no desenvolvimento deste trabalho é realizar apenas a detecção, assim não possuindo nenhuma forma de contenção nos alertas detectados. A duração dos dados que será analisada e discutida neste capitulo se iniciou no dia 23 de Abril de 2023 às 17:15:52 até 4 de Junho de 2023 às 23:06:14. A coleta de eventos efetuou a centralização total de 2.071.123 eventos. O número total de eventos utilizou 1.4 GigaBytes de armazenamento em disco. Do número total de eventos 1.619.425 estão armazenados na fase morna e 451.738 na fase quente. Em média 49.312 eventos por dia, em um cenário onde uma pessoa que levaria cerca de 30 segundos para analisar cada evento, seria necessário 17,12 dias para analisar a volumetria coletada de apenas 1 dia.

Inicialmente o *script* foi executado em sua primeira vez no dia 30 de abril de 2023 às 20:46 analisando todos os dados coletados até esse dia, após essa primeira execução o *script* foi configurado para executar a cada hora buscando a utilização de comandos com privilégios administrativos (`sudo`). No total foram encontrados 124 eventos onde houveram a utilização da elevação de privilégio. A Figura 12 extraída diretamente do Kibana apresenta a categorização dos 15 comandos mais utilizados. O comando mais utilizado foi a execução do próprio *script* durante seu período de desenvolvimento .

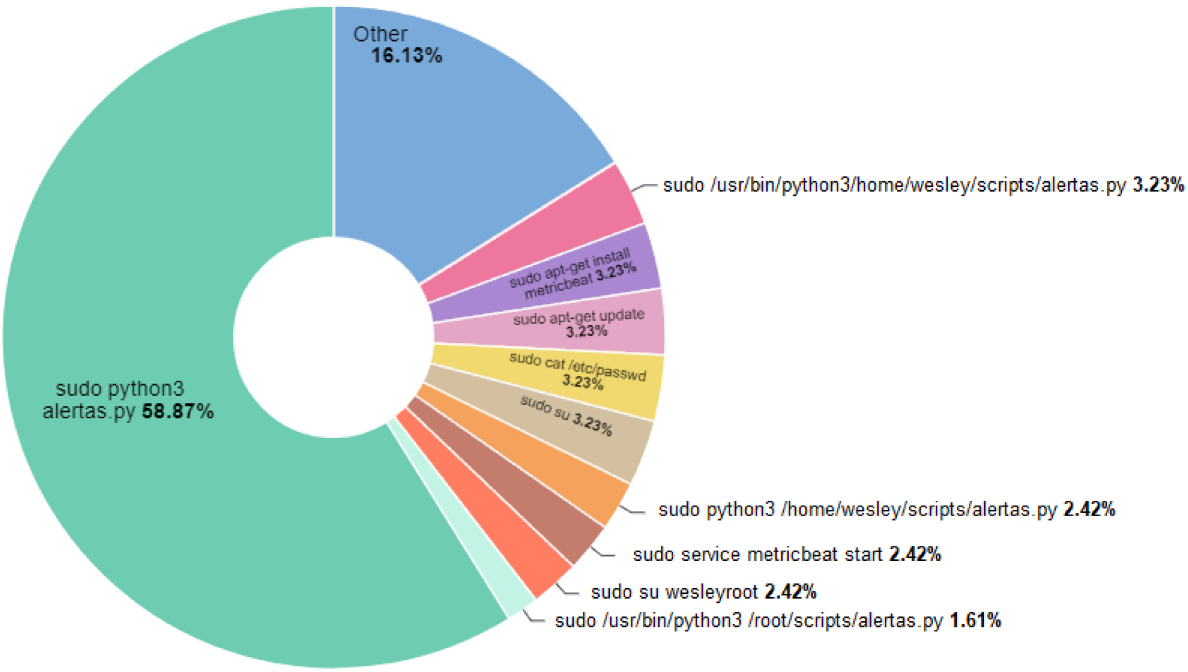

Figura 12 – Os 15 comandos mais executados com elevação de privilégios administrativos.

O experimento realizado definiu a execução do *script* a cada 1 hora, porém sendo possível realizar a verificação em um tempo menor, por exemplo, a cada 1 minuto. Desta forma a detecção de um incidente de segurança seria em um tempo muito menor possibilitando o planejamento e execução de planos de ações mais rapidamente. O recomendado é criar uma regra e configurá-la baseada em sua criticidade, regras com criticidade alta com menores tempos e criticidades baixas com um maior intervalo de tempo.

Como o objetivo do trabalho é focado na viabilização de monitoramentos de SOC para empresas com baixo orçamento, não foi possível validar sua escalabilidade, porém com informações obtidas com uma das equipes de segurança, mais especificamente a equipe de SOC de uma grande empresa de varejo brasileira, que anteriormente utilizou o mesmo conjunto de software como centralizador de eventos, é possível ter uma ideia do seu poder de escalabilidade. Os dados apresentados a seguir foram obtidos no ano de 2021 no período de promoções conhecido como *Black Friday* ocorrido entre os dias 22 de novembro de 2021 até 28 de novembro de 2021. A Tabela 3 demonstra a volumetria de dados indexados durante a semana de promoções, ou seja, é relacionado com a quantidade de dados armazenados durante 24 horas. O pico de indexação aconteceu no dia 25/11 às 22:50 e chegou a 295.693 indexações por segundo, a Figura 13 demonstra a evidência do pico de indexações extraído do próprio Kibana utilizado na época. Na Figura 13 é possível identificar a taxa de indexação, isto é, a quantidade de dados que foram armazenados no intervalo de 1 segundo. O pico máximo ocorreu próximo as 22:30 horário no qual acontecem as maiores promoções devido a virada de quinta-feira para sexta-feira. A empresa utilizava a versão paga do pacote ELK, porém a versão paga difere apenas com relação as funcionalidades disponíveis, em relação ao armazenamento e processamento de dados não possui nenhuma diferença da versão gratuita.

Tabela 3 – Volumetria de dados indexados por varejista brasileira

Dia	**Volumetria**
22/11	1,81 TB
23/11	2,27 TB
24/11	3,10 TB
25/11	4,19 TB
26/11	4,61 TB
27/11	3,97 TB
28/11	3,64 TB
Total	23,59 TB

4.0.1 Considerações finais

Neste capítulo, foram apresentados os resultados obtidos referente ao uso do sistema em um determinado intervalo de tempo. Os resultados demonstraram as contribuições que podem ser obtidas com a utilização do sistema. No Capítulo 5 serão discutidas as conclusões sobre o trabalho, com vantagens, desvantagens e trabalhos futuros.

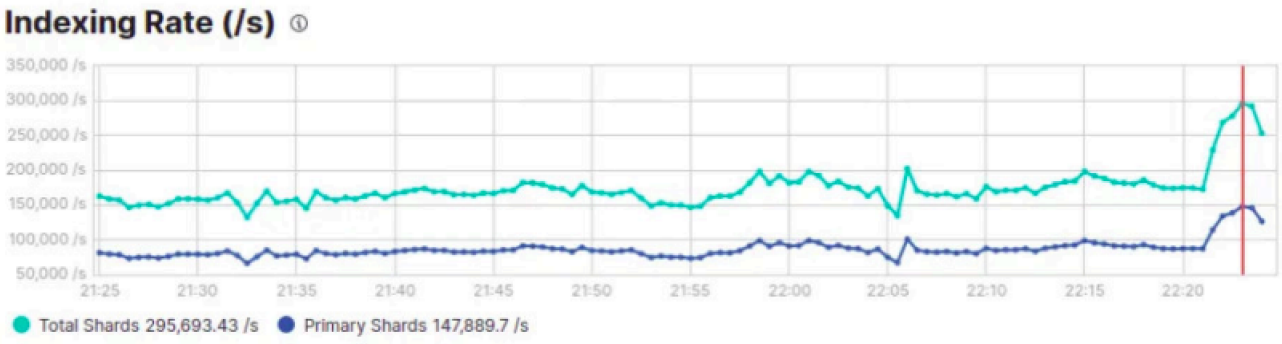

Figura 13 – Pico de indexação obtido em período de promoções

5 CONCLUSÕES

Atualmente, o mundo ainda enfrenta desafios na detecção de eventos maliciosas de cibersegurança, mesmo sendo capaz de rastrear diversas ações em empresas em seu ambiente digital por meio da geração e coleta de *logs*. No entanto, a análise de dados de forma rápida ainda é acompanhada de um custo extremamente alto, referente a ferramentas de cibersegurança e pessoas especializadas. Devido a isso empresas de orçamentos limitados acabam sendo impactadas.

Objetivando solucionar esse problema e aplicando os conceitos de SOC descritos no Capítulo 2, foi explorado a possibilidade de centralizar todos esses dados em uma ferramenta de uso fácil e escalável de forma gratuita. Essa abordagem permite que empresas de pequeno e médio porte possam ter acesso a uma solução acessível que permita realizar análises de formas automatizadas e/ou manuais.

A utilização do pacote de software ELK para centralização de *logs* contribuiu com a triagem e análise dos dados, possibilitando o desenvolvimento de consultas com o intuito de buscar indícios de possíveis incidentes em seu ambiente. Ao adicionar uma análise automatizada consumindo esses *logs*, possibilita a realização de monitoramento em tempo real, mesmo em ambientes heterogêneos é possível identificar padrões e comportamentos que possam ser encontrados e monitorados com o objetivo de gerar alertas. Devido a sua arquitetura, o sistema permite adição de novas regras para criação de alerta, integração com fontes de Indicator of Compromise (IoC) externas, indexação de vários tecnologias diferentes, etc. A utilização do Elasticsearch como centralizador de *logs* não se limita apenas a cibersegurança, possibilitando seu uso para diversos tipos de análises, por exemplo, grande utilização na área de ciência de dados. Outra vantagem é o envio de alertas informativos para diversas plataformas de comunicação.

Sua principal desvantagem é referente a sua escalabilidade, pois infraestrutura que será utilizada deve ser escalada da mesma forma. Referente ao *script*, mesmo que a linguagem de programação Python seja de fácil utilização, ainda requer um mínimo conhecimento em programação para o desenvolvimento de novas regras. A utilização de uma inscrição gratuita a traz diversos benefícios porém para cada bônus temos em contrapartida um ônus relacionado. Na versão gratuita, todo gerenciamento da infraestrutura é responsabilidade do administrador do ambiente. A plataforma possui integração com inteligência artificial, porém infelizmente é disponível apenas em sua inscrição paga. O envio de alertas por e-mail utilizando um servidor SMTP neste caso sendo o Outlook traz algumas desvantagens devido a limitação diária de envios.

5.1 Trabalhos futuros

O presente trabalho possui inúmeras melhorias que possam ser desenvolvidas no futuro, algumas possibilidades sendo:

- Indexação de novas tecnologias;
- Criação de mais regras para diferentes cenários;
- Aumento da autoridade do sistema, isto é, implementação de contenção de eventos de forma automatizada;
- Integração de regras baseadas nas táticas de ataques categorizadas na diretriz MITRE ATT&CK (THE MITRE CORPORATION, 2015);
- Adição de mais nós de processamento;
- Implantação de níveis de criticidade para cada regra, por exemplo, baixo, médio, alto, critico e informativo.
- Colocar o sistema desenvolvido para utilização em produção. Devido a sua complexidade o sistema não foi colocado em produção, para isso deverá ser realizado alguns levantamentos sobre: quantidade de *hosts*, tecnologias que serão armazenadas, comportamentos a serem monitorados.

5.2 Considerações finais

Os principais objetivos do trabalho foram atingidos, a utilização do pacote ELK trouxe a possibilidade de implantação de uma rotina de revisão diária de *logs* automatizada, contribuiu na identificação de eventos de segurança que anteriormente não seriam analisados ou reportados, centralizou diversos *logs* possibilitando desde análises simples à complexas quando necessário. O desenvolvimento deste trabalho proporcionou uma base sólida com grandes conhecimentos sobre cibersegurança, SOC, geração de alertas, identificação de comportamentos possivelmente maliciosos, estruturação de dados, instalação e configuração de ferramentas de cibersegurança e manipulação de dados.

REFERÊNCIAS

AHMED FARRUKH E JAHANGIR, U. e. R. Gerenciamento centralizado de logs usando elasticsearch, logstash e kibana. **2020 Conferência Internacional sobre Ciência da Informação e Tecnologia da Comunicação (ICISCT)**, p. 1–7, 2020.

BATISTA, D. M. **Mecanismos para Análise de Big Data em Segurança da Informação**. [*S.l.*], 2017. Disponível em: http://wrnp.rnp.br/sites/wrnp2018/files/wrnp2018_gtbis_gcc_v3.pdf. Acesso em: 03 jun. 2023.

BRASIL, B. C. do. **What is Cyber-Security?** 2018. Disponível em: https://www.bcb.gov.br/estabilidadefinanceira/exibenormativo?tipo=Resolu\%C3\%A7\%C3\%A3o&numero=4658. Acesso em: 09 jun. 2023.

CANONICAL LTD. **auditctl - a utility to assist controlling the kernel's audit system**. [*S.l.*], 2019.

CANONICAL LTD. **crontab - tables for driving systemd-cron**. [*S.l.*], 2019.

CANONICAL LTD. **Ubuntu Manpage: apt - command-line interface**. [*S.l.*], 2019.

ELASTICSEARCH B.V. **Auditbeat quick start: installation and configuration**. [*S.l.*], 2023.

ELASTICSEARCH B.V. **Install Elasticsearch with Debian Package**. [*S.l.*], 2023.

ELASTICSEARCH B.V. **Install Kibana with Debian package**. [*S.l.*], 2023.

ELASTICSEARCH B.V. **Python Elasticsearch Client**. [*S.l.*], 2023.

ENTERPRISE, V. **2015 data breach investigations report**. 2015. Disponível em: https://www.researchgate.net/publication/289254638_2015_Verizon_Data_Breach_Investigations_Report. Acesso em: 09 mai. 2023.

FOUNDATION, P. S. **Python**. 2001. Disponível em: https://www.python.org/about/. Acesso em: 06 jun. 2023.

HUTCHINS, E.; CLOPPERT, M.; AMIN, R. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. **Leading Issues in Information Warfare & Security Research**, v. 1, 01 2011.

KASPERSKY. **What is Cyber-Security?** 2019. Disponível em: https://www.kaspersky.com/resource-center/definitions/what-is-cyber-security. Acesso em: 15 may. 2023.

KILLCRECE KLAUS-PETER KOSSAKOWSKI, R. R. G. **Organizational Models for Computer Security Incident ResponseTeams (CSIRTs)**. [*S.l.*]: Software Engineering Institute, 2003.

MICROSOFT. **Configurações POP, IMAP e SMTP para Outlook.com**. [*S.l.*], 2023.

MUNIZ GARY MCINTYRE, N. A. J. **Security Operations Center Building, Operating, and Maintaning Your SOC**. [*S.l.*]: Cisco Press, 2016. 41 p.

N.V., E. **Auditbeat: Agente Lightweight Para Dados de Auditoria**. 2015. Disponível em: https://www.elastic.co/pt/beats/auditbeat. Acesso em: 06 jun. 2023.

N.V., E. **Nossa história**. 2015. Disponível em: https://www.elastic.co/pt/about/history-of-elasticsearch. Acesso em: 06 jun. 2023.

PYTHON SOFTWARE FOUNDATION. **JSON encoder and decoder**. [*S.l.*], 2023.

PYTHON SOFTWARE FOUNDATION. **SMTP protocol client**. [*S.l.*], 2023.

REASEARCH, I. **Virus Timeline**. 2012. Disponível em: https://web.archive.org/web/20121027045532/http://www.research.ibm.com/antivirus/timeline.htm. Acesso em: 25 abr. 2019.

RULE, U. S. M. C. L. C. J. N. A symbiotic relationship: The ooda loop, intuition, and strategic thought. 03 2013.

SANTOS, E. L. Byod : Riscos e desafios para sua implementação nas organizações. *In*: . [*S.l.*: *s.n.*], 2018.

SYSTEMS, C. on N. S. **Committee on National Security Systems(CNSS) Glossary**. 2015. Disponível em: https://www.serdp-estcp.org/Tools-and-Training/Installation-Energy-and-Water/Cybersecurity/Resources-Tools-and-Publications/Resources-and-Tools-Files/CNSSI-4009-Committee-on-National-Security-Systems-CNSS-Glossary. Acesso em: 20 out. 2019.

THE MITRE CORPORATION. **Enterprise Matrix**. [*S.l.*], 2015.

ZIMMERMAN, C. **Ten Strategies of a World-Class Cybersecurity Operations Center**. [*S.l.*]: The MITRE Corporation, 2014. ISBN 978-0-692-24310-7.