Hochschule Worms

Fachbereich Informatik

Studiengang Angewandte Informatik B.Sc.

Implementierung von einem OpenSource Log-Analyse-Tool zur Erkennung von Cyberangriffe

Abschlussarbeit zur Erlangung des akademischen Grades

Bachelor of Science

Bruno Macedo da Silva
676839
inf3645@hs-worms.de
Bebelstraße 22 Z10
67549 Worms

Betreuer	Prof. Dr. Zdravko Bozakov
Bearbeitungszeitraum:	Sommersemester 2023
Abgabedatum:	xx. xxx 2023
Sperrvermerk:	Ja/Nein

Inhaltsverzeichnis

Abstract

The aim of this thesis is to develop a reliable, cost-effective solution for monitoring security events by utilizing an Open Source, Security Information and Event Management (SIEM)-like tool. Since many existing SIEM solutions are either proprietary or offer limited free features, we chose to use Grafana and its integrated tools - Promtail, Loki, and Alerting - to create our monitoring system. Grafana is primarily used to generate customizable graphics based on user input, and in our study, we used Secure Shell Protocol (SSH) log files as input. Promtail extracted the files from Endpoints and sent them to Loki, which used defined rules to aggregate and filter the content in order to identify possible cyberattacks against an SSH server. Once the information was extracted, Grafana was used to provide a visual overview of the SSH connections. Additionally, we employed the Alerting tool to send notifications about potential attacks identified by our rules. The ruleset we used to recognize potential attacks and the descriptions of these attacks were based on the Mitre ATT&CK Matrix. We found that the combined use of these tools was reliable, affordable, and useful for detecting static-based attacks. The main challenges in using these tools as a replacement for a SIEM solution are properly defining the ruleset used to read and extract information about cyberattacks from log files and adapting those rules to scenarios where attacks have more dynamic flows.

Keywords: Monitoring Tool, Grafana Loki Cyberattacks,SIEM

Abbildungsverzeichnis

Tabellenverzeichnis

Glossar

Abfragesprache oder *Query Language* funktioniert wie ein Filter für die Suche nach spezifischen Daten in einer Datenbank (at, 2022). 16, 19, 20, 23, 63

Application Programming Interface(API) bezieht sich auf Code und Regeln, die die Kommunikation zwischen verschiedenen Anwendungen ermöglichen. In diesem Fall kann eine Anwendung eine Anfrage an eine andere Anwendung senden, um Daten zu holen oder zu senden (IBM, 2020). 36, 37

Frontend bezieht sich auf die Benutzeroberfläche und die Elementen, mit denen die Benutzer direkt interagieren (at, 2022). 18, 20, 23, 53, 54

Backend bezieht sich auf Elementen, mit denen die Benutzer keine direkte Kontakt haben, wie Server und Database(at, 2022). 23

Brute-Force Angriffe systematische Versuche, Zugangsdaten oder andere sensible Daten zu erraten, indem verschiedene Buchstaben, Ziffern und Symbole kombiniert werde (Sowmya et al., 2012). 10, 28, 39, 55, 71

Container funktionieren ähnlich wie virtuelle Maschinen (VMs), jedoch sind Container Anwendungen mit den notwendigen Ressourcen, um eingepackte Anwendungen auszuführen. Container werden oft für einzelne verwendet und teilen Ressourcen wie den Kernel des Host-Betriebssystems. Jeder Container ist in einer isolierten Umgebung mit den notwendigen Ressourcen für den Betrieb der ausgewählten Anwendung. Docker ist eine der bekanntesten Plattformen zur Verwaltung von Containern (Douglis and Nieh, 2019). 25, 29, 34, 36, 54

Cortex ist eine Open-Source-Plattform zur Verwaltung und Weiterverarbeitung von Sicherheitsvorfällen. Es fungiert als Analyse-Engine, indem es Informationen sammelt und je nach Fall Antworten oder Aktionen durchführt. Cortex kann eigenständig oder in Kombination mit anderen Tools verwendet werden (Project, 2021). 48

Cyberangriff Angriffe über Cyberspace. Solche Angriffe zielen darauf ab, Unternehmen und ihre Infrastrukturen zu zerstören, zu lähmen, zu kontrollieren oder die Integrität ihrer Daten zu stehlen oder zu manipulieren (NIST, 2020b). 1, 2, 6, 24–26, 63, 64

Confidentiality, Integrity and Availability (CIA) beschreiben die drei wichtigsten Schutzziele der IT-Sicherheit: Vertraulichkeit, Integrität und Verfügbarkeit (Wendzel, 2018). 8

Taktiken, Techniken, Prozeduren (TTP) beschreiben in der Mittre ATT&CK Matrix Verhalten, Methode un Mustern bei Cyberangriffe (Maymi et al., 2017). 1, 2, 24, 27, 28, 61, 63

Cyber Kill Chain (CKC) auch *Cyberattack Lifecycle* genannt, bezieht sich auf ein Sicherheitsmodell für die Identifizierung, Analyse und Unterbrechung von fortgeschrittenen Cyberangriffen. Dieses Modell hat sieben festgelegte Phasen: *Reconnaissance*, *Weaponization*, *Delivery*, *Exploitation*, *Installation*, *Command & Control (C2)* und *Actions on Objectives* (Martin, 2018). 10

Cybersicherheit - Diese Domäne umfasst Kenntnisse und Methoden für den Schutz, die Prävention und Wiederherstellung von elektronischen Kommunikationsmitteln und deren Inhalten. Dabei konzentriert sie sich auf deren Verfügbarkeit, Integrität, Authentizität, Vertraulichkeit und Nichtabstreitbarkeit. (NIST, 2020b). 27

Endpoint bezieht sich auf Geräte oder Systeme, die mit dem Netzwerk verbunden sind. Diese können z.B. Handys, Servers, Computers, Sensoren sein (Microsoft Security, 2022). iii, 5–7, 9, 11, 13, 17, 19, 20, 36–38, 40, 45, 48, 51

Hashwerte sind Zeichenfolgen, die durch Anwenden einer mathematischen Funktion (Hashfunktion) auf einen Text oder eine Datei erzeugt werden. Die Rückführung auf das ursprüngliche Objekt aus dem Hashwert sollte jedoch unmöglich sein (Wendzel, 2018). 28

Hypertext Transfer Protocol (HTTP) ist die Grundlage des Internets. Dieses Protokoll definiert die Regeln für die Übertragung von Texten und Dateien im Internet. Das Protokoll verwendet acht Methoden, um die Kommunikation zwischen Clients und Servern herzustellen: *GET, POST, HEAD, DELETE, CONNECT, OPTIONS, PUT* und *TRACE* (Chai and Ferguson, 2021) and (tutorialspoint, 2009). 21

Zweites Gesetz zur Erhöhung der Sicherheit informationstechnischer Systeme ist das zweite Gesetz zur Erhöhung der Sicherheit informationstechnischer Systeme wurde im Jahr 2021 als verabschiedetes Bundesgesetz zur Erhöhung der Sicherheit von informationstechnischen Systemen besonders von den kritischen Infrastrukturen (Harmes, 2023). 8

Falsch Positiv ist eine Warnmeldung einer nicht vorhandenen Verwundbarkeit (NIST, 2020c). 11

grafische Benutzeroberfläche (GUI) - Es handelt sich dabei um eine visuelle Schnittstelle, die es dem Benutzer ermöglicht, mit Anwendungen mittels Symbolen und grafischen Elementen zu interagieren. Im Gegensatz dazu verwendet die textbasierte Benutzeroberfläche (CLI) Befehlszeilen und Texteingabe zur Steuerung von Anwendungen (Fu, 2018). 8, 29

Hydra ist eine Open Source Tool für Brute-Force Angriffe (Kali, 2022a). 30, 31

Health Insurance Portability and Accountability Act (HIPAA) ist ein US-Bundesgesetz über den Schutz von sensitiven personenbezogenen Gesundheitsdaten (U.S. Depart-

ment of Health & Human Services, 2016). 8

JavaScript Object Notation (JSON) ist eine Standard text-basierte Dateiformat, die von Menschen leicht zu verstehen ist und von Maschine einfach zu analysieren und strukturieren (parsen). Es ist eine Untermenge der JavaScript Programmiersprache (Ecma, 2017). 39

Kali ist eine Open-Source-Linux-Distribution, die speziell auf den Einsatz von Sicherheitstools für Angriffe und Sicherheitstests ausgelegt ist (Kali, 2022b). 29

Künstliche Intelligenz (KI) bezeichnet die Fähigkeit, Maschinen menschenähnliche kognitive Fähigkeiten wie Verständnis, Entscheidungsfindung, Lernen und Problemlösung zu entwickeln (Collins et al., 2021). 64

LogQL ist eine für Grafana Loki entwickelte Abfragesprache. Sie wird verwendet, um Logdateien zu zusammenzustellen (Grafana Labs, 2021c). 20, 23, 40–45, 49, 61–63

Multi-Faktor-Authentisierung (MFA) bezeichnet ein Authentifizierungsverfahren, bei dem mindestens zwei unabhängige Komponenten zur Identitätsprüfung verwendet werden, um eine höhere Sicherheit zu gewährleisten. Zum Beispiel kann ein Benutzer aufgefordert werden, sich mit einem Passwort und einem Fingerabdruck oder einem Token und/oder einer Gesichtserkennung zu authentifizieren (Ibrokhimov et al., 2019). 39

Machine Learning (ML) bezieht sich auf die Fähigkeit von Systemen, automatisch und menschenähnlich Probleme zu lösen und spezifische Aufgabe zu erledigen (Janiesch et al., 2021). 9, 15

Mitre ATT&CK Abkürzung für *Adversarial Tactics, Techniques and Common Knowledge*. Es bezieht sich auf eine weltweit zugängliche Wissensbasis mit detaillierter Beschreibung, Klassifizierung und Bekämpfung von verschiedenen Angriffstechniken (MITRE ATT&CK, 2018a). iii, 1, 2, 10, 24–28, 61–63

Grafana Ecosystem beinhalt die Tools Loki, Grafana, Tempo, Mimir und Phlare (Grafana Labs, 2022b). 36–38, 64

Mimir ist ein in Grafana integriertes Tool, das ähnlich wie Grafana Loki funktioniert. Es ermgölicht skalierbare Dateispeicherung, Bearbeitung und Abfrage mit der Abfragesprache LogQl (Grafana Labs, 2022d). 36, 48

Tempo ist ein Tool von Grafana Ecosystem, das sich für die Unterscheidung und Erkennung von Prozessen beschäftigt, dieses Verfahren heißt verteilte Rückverfolgung (Grafana Labs, 2020a) und (DevInsider, 2021). 36

Phlare ist auch ein Tool vom Grafana Ecosystem, das sich mit der Sammlung und der

Analyse von Daten über die Leistung von Anwendung beschäftigt (Grafana Labs, 2022e) und (Salinger, 2021). 36

National Institute of Standards and Technology (NIST) ist eine US-Behörde, die für die Regelungen, Vereinheitlichung und Weiterentwicklung von Standards im Bereich Informationstechnologie zuständig ist (NIST, 2020a). 1

Open Source beschreibt Software, die folgende Voraussetzungen erfüllen: freie Verteilung, Kopierung, Modifizierung und Nutzung und keine Diskriminierung gegenüber Personen und/oder Gruppe (Open Source Initiative, 2007). iii, 1, 2, 5, 8, 10, 12, 16, 22, 24

Query Domain Specific Language (DSL) Abfragesprache von Elasticsearch für die Kommunikation mit der Databank. Diese Abfragesprache hat zwei verschiedene Typen: „Leaf“ für die Suche nach spezifischen Muster und „Compound“ für die logische Kombination von mehrere Abfrage (elastic, 2015). 17, 23

Password Spraying ist ein Angriff gegen Anmeldedaten, indem mögliche Passwörter gegen verschiedenen viele Benutzernamen verwendet werden. Das Ziel dieses Angriffes ist eine Kontosperrung zu vermeiden, indem wenige Versuche pro Nutzer stattfindet (Swathi, 2022). 28, 29, 32, 33

Password Stuffing ist ein Angriff gegen Passwörtern, indem bekannte Anmeldedaten von vorherigen Angriffen verwendet werden. Dieser Angriff basiert sich auf die Idee, dass Nutzer dasselbe Passwort für verschiedenen Systemen verwenden (Ba et al., 2021). 28, 30–32

Payment Card Industry Data Security Standard (PCDI DSS) sind Sicherheitsstandards, die von Unternehmen, die Kreditkarten akzeptieren, verarbeiten, speichern oder übertragen, eingehalten werden müssen (Centers for Disease Control and Prevention, 2016). 8

Network Operations Center (NOC) ist ein zentralisierter Bereich eines Unternehmens, der für die Überwachung und Verwaltung von Netzwerkaktivitäten verantwortlich ist. (Mohammed et al., 2021). 16

OpenTelemetry ist eine Sammlung von Tools zu Generierung, Sammlung und Exportierung von Messdaten, auch telemetrische Daten genannt. Das Tool besteht aus *Agents* und *Collectors*. Der Agent wird auf jedem Endpunkt installiert, um Daten zu sammeln. Der Collector empfängt die Daten und leitet sie weiter (Grafana Labs, 2022c), (OpenTelemetry, 2023) und (Höfling, 2022). 36–38

Log- und Messdaten und Ablaufverfolgung sind drei große Datenquelle für die Überwachung eines Systems und spiele eine wesentliche Rolle in der Beobachtbarkeit. Logdateien speichern Ereignisse oder Aktionen, die in einem System stattfinden.

Messdaten zeigen quantifizierte Daten, wie Aufnahmerate oder Anzahl von verwendeten Resource. Ablaufverfolgung beschäftigt sich mit dem Informationsfluss bei der Ausführung einer Anwendung (Tozzi, 2022). 64

Plugin sind optionale Software-Komponenten, die weitere Funktionalitäten zu einer Anwendung hinzufügen (IT-Service.Network, 2020). 9, 16, 19, 24, 48

Portnummer ist eine numerische Identifikation eines Dienstes oder einer Verbindung. Es handelt sich um eine logische Adressierung, die zur Identifikation eines oder mehrerer Prozesse verwendet wird (Tanenbaum and Wetherall, 2011). 43

Prometheus ist ein Open-Source-Tool der Firma SoundCloud. Es dient der Überwachung und Erstellung von Warnmeldungen, die auf der Grundlage von vordefinierten Regeln konfiguriert werden (Prometheus, 2016). 19, 36, 37, 40, 48

Proprietär bezieht sich auf Software, die einer Firma oder Person gehört. Für die Nutzung ist in der Regel der Kauf einer Lizenz erforderlich. In diesem Fall haben Kunden nur begrenzten oder keinen Zugriff auf den Quellcode (Nexcess, 2022). 2, 8, 22

Reguläre Ausdrücke (RegExp) ,*regular expressions* im Original, sind Methode, um Muster in Zeichenketten zu beschreiben. Im Informatikbereich werden solche Ausdrücke verwendet, um spezifische Texte oder Einträge in Textdateien zu finden (Qusef and Hassan, 2018). 39, 52

Rockyou ist eine Textdatei mit über 8 Milliarden Passwörtern im Klartext. Diese Datei stammt aus einem Angriff gegen Yahoo im Jahr 2009 und wird seitdem ständig aktualisier (Mikalauskas, 2023). 30–32

Security Operations Center (SOC) ist ein zentralisierter Bereich eines Unternehmens, der für die Überwachung, Identifizierung, Bewertung und Reaktion auf Sicherheitsvorfälle verantwortlich ist. (Vielberth, 2021). 1, 5

Secure Shell Protocol(SSH) ist ein Netzwerkprotokoll, das eine verschlüsselte Verbindung zwischen Endpunkten bietet. SSH wird meistens für die Fernadministration von Computern verwendet. Dieses Protokoll ermöglicht die Erstellung einer sicheren Verbindung in einer unsicheren Umgebung (Wendzel, 2018). iii, 30

Ubuntu ist eine Linux-Distribution, die oft für Server, Clients und Internet of Things (IoT) verwendet wird (Ubuntu, 2023b). 29

Use Cases sind narrative Beschreibungen der Interaktionen zwischen Systemen und Benutzern. Sie dienen der Anforderungserhebung für ein System (Savic et al., 2012). 2, 10, 11, 63

Virtuelle Maschine (VM) ist eine Kopie der Hardware-Struktur mit einer eigenen Aufteilung von Ressourcen und einem eigenen Betriebssystem. Auf einer physischen Maschine, auch Host genannt, können mehrere solcher VMs ausgeführt werden. Sie emulieren ein echtes und unabhängiges System (Tanenbaum, 2009). 25, 29

Abkürzungsverzeichnis

API Application Programming Interface (API).

BSI Bundesamt für Sicherheit in der Informationstechnik.

CIA Confidentiality, Integrity and Availability.

TTP Taktiken, Techniken, Prozeduren.

CKC Cyber Kill Chain.

HTTP Hypertext Transfer Protocol.

IDS Intrusion Detection System.

GUI grafische Benutzeroberfläche.

IPS Intrusion Prevention System.

FPO Fachspezifische Prüfungsordnung.

HIPAA Health Insurance Portability and Accountability Act.

JSON JavaScript Object Notation .

KI Künstliche Intelligenz.

MFA Multi-Faktor-Authentisierung.

ML Machine Learning.

NIST National Institute of Standards and Technology.

OTX Open Threat Exchange.

LML Log Monitoring Lackey.

OSSIM Open Source Security Information Management.

DSL Query Domain Specific Language.

PCDI DSS Payment Card Industry Data Security Standard.

NOC Network Operations Center.

OWASP Open Web Application Security Project.

RegExp Reguläre Ausdrücke.

SIEM Security Information and Event Management.

SEM Security Event Management.

SIM Security Information Management.

SOC Security Operations Center.

SSH Secure Shell Protocol.

USM Unified Security Management.

VM virtuelle Maschine.

1. Einleitung

Der heutige Netzwerkverkehr ist fast tausendfach größer als vor 20 Jahre (Roser et al., 2015). Das Internet wird heutzutage für fast all unsere Tätigkeiten verwendet: Soziale Netzwerke, Video und Audio-Streaming, Einkauf, behördliche Angelegenheiten und viele andere. So viel Verkehr generiert eine unermessliche Menge von Daten, die alle möglichen Inhalte beinhalten, von unschuldigen Anfragen nach einem eigenen Kontostand bis zur Ausführung von beabsichtigten Anfragen, um Systeme lahmzulegen. Um ersteres vom letzterem zu unterscheiden, verwenden viele Firmen das sogenannte Security Information and Event Management (SIEM) oder Log-Analyse-Tools.

Das National Institute of Standards and Technology (NIST) definiert SIEM als Software für die Sammlung, Anpassung, Analyse, Überwachung und Bedrohungserkennung von Sicherheitsdaten aus verschiedenen Quellen (NIST, 2020d). Die Bewertung dieser Daten spielt eine wesentliche Rolle bei solchen Anwendungen, um zu entscheiden, ob es sich um legitime Anfrage oder um einen Cyberangriffe handelt. Mit den Daten von SIEM kann das Security Operations Center (SOC) Team Maßnahmen ergreifen. Log Analysis und Log Management beziehen sich auf die Sammlung, Bearbeitung, Speicherung und/oder Löschen, Weiterleitung und Überwachung von Loginformationen. In dieser Arbeit benutzen wir den Begriff „Log-Analyse-Tools“, um diese Systeme zu referenzieren.

In diesem Projekt recherchieren und vergleichen wir existierende SIEM und Log-Analyse-Tools. Danach entscheiden wir uns für eine Open Source Lösung, um eine kostengünstige Verbreitung und Implementierung zu ermöglichen. Mit dem ausgewählten Tool analysieren und bewerten wir spezifische Logdateien, damit wir demnächst potenzielle Angriffe erkennen können. Die Regelsätzen für die Angriffserkennung sollen mithilfe der Taktiken, Techniken, Prozeduren (TTP) von Mitre ATT&CK aufgebaut werden.

Unser Ziel ist es, eine umfangreiche Open Source Lösung zu finden bzw. zu gesltaten, die uns ermöglicht, Cyberangriffe nach vordefinierten Regelsätzen zu detektieren. Proprietäre Lösungen gibt es viele auf dem Markt. Sie sind meistens kostenpflichtig und verlangen spezielle Wartung. Da sich solche Lösungen eher an große Konzerne richten, beschäftigen wir uns mit dem Aufbau und Strukturierung einer eigenen Lösung mithilfe von Open Source Tools.

Diese Arbeit wird in folgende Teile geteilt:

- Definition von SIEMs und Log-Analyse-Tools
- Beschreibung von existierenden Proprietären und Open Source Lösungen
- Entscheidung für die Implementation von einer Open Source Lösung
- Generierung und Extrahierung von Logdateien nach der Ausführung von einem ausgewählten Cyberangriff
- Installation, Konfiguration und Generierung von Warnmeldungen mit den ausgewählten Anwendungen
- Definition der Use Cases und Implementierung der Regelsätze für die Erkennung der vorherigen Angriffen anhand der Taktiken, Techniken, Prozeduren (TTP) der Mitre ATT&CK Matrix
- Auswertung der implementierten Tools mit der Verwendung von spezifischen Logdateien der Hochschule in der ausgewählten Lösung

1.1. Problemstellung

Während der Entwicklung dieser Arbeit beschäftigen wir uns mit folgenden Fragenstellung:

- Wie können wir ein Log-Analyse-Tool konfigurieren, dass es vordefinierte Angriffe nach der Mitre ATT&CK Matrix automatisch erkennen kann?
- Wie können wir allgemeine Regelsätze definieren, sodass wir sie später für die verschiedene TTP der Mitre ATT&CK Matrix nanpassen können?

Das folgende Diagramm, 1, stellt den Aufbau und Entwicklung dieser Arbeit dar, wie oben beschrieben:

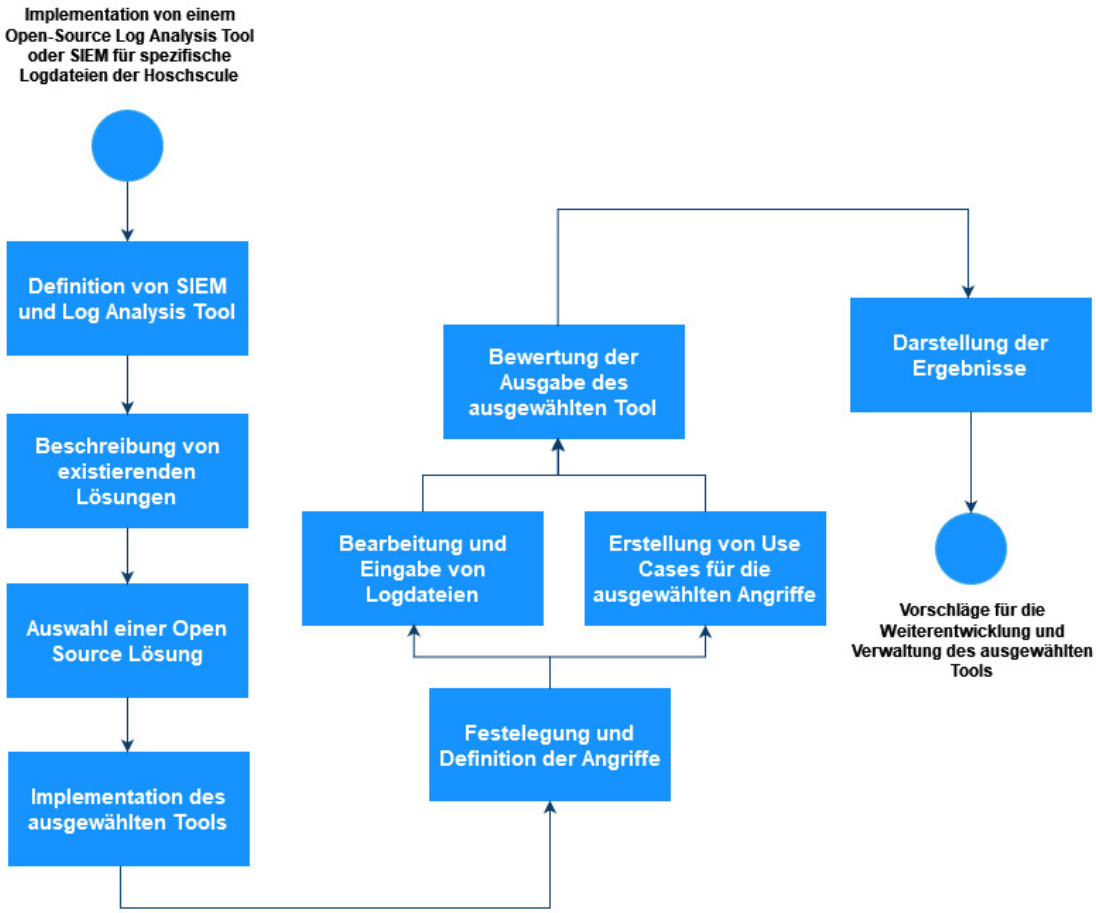


Abbildung 1: Aufbau dieser wissenschaftlichen Recherche
Quelle: Eigene Darstellung

2. Definition von SIEMs und Log-Analyse-Tools

Sowohl in der wissenschaftlichen als auch in der kommerziellen Literatur gibt es verschiedene Definitionen von SIEM. Diese widersprechen sich nicht, aber zeigen unterschiedliche Perspektiven. Eine von dieser Definitionen behaupt, dass SIEM das Ergebnis einer Kombination zwischen dem Security Event Management (SEM) und Security Information Management (SIM) ist (Dorigo, 2012). Das Erste bezieht sich auf die Identifizierung, Bewertung, Beobachtung und den Bericht von Sicherheitsvorfällen mithilfe von verschiedenen Log Dateien (techopedia, 2015). Das Zweite ist eine Software, die bei der automatischen Sammlung von Loginformationen aus vielen Quellen, wie Firewall und Servern unterstützt (techopedia, 2022). Da die meisten SIEM-Lösungen kostenpflichtig sind, existieren auch viele Open Source Log-Analyse-Tools, die eine ähnliche Aufgabe erledigen, ohne die Kernelemente von SIEM zu besitzen.

Log-Analyse-Tools sind in der Regel Anwendungen die Logdateien empfangen, speichern, bearbeiten und nach spezifisichen Regeln bewerten. Diese Tools unterstützen Programmierer und Systemadministratoren bei der Überwachung des Zustands von Systems oder einer Software. Ein solches Tools kann Logdateien von verschiedenen Endpoints und in verschiedenen Formaten empfangen, so dass es schließlich einen Bericht oder eine Grafik erzeugt (Łukasz Korzeniowski and Goczyła, 2022). Ihre Nutzung beschränkt sich nicht auf den Sicherheitsbereich ein, sondern kann für gesamte IT-Bereich nützlich sein.

In dem Universum des SOC mischen sich verschiedene Begriffe, die manchmal zur Verwirrung führen, weil sie ähnliche Bedeutungen haben und Verantwortungen abdecken. Intrusion Detection System (IDS), Intrusion Prevention System (IPS), Security Information and Event Management (SIEM) und Log-Analyse-Tools werden von Laien und sogar von Spezialisten oft verwechselt, da ihre Aufgaben mehr Gemeinsamkeiten als Unterschiede haben. Diese Tools sind feste Bestandteil eines SOC, jedoch konzentrieren wir uns auf Log-Analyse. Um den Fokus der Arbeit zu vergrenzen, erläutern wir demnächst Abschnitt die erwähnten Begriffe.

Intrusion Detection System (IDS), Intrusion Prevention System (IPS) und Security Information and Event Management (SIEM) sind Sicherheitstool als Software und/oder Hardware, die zusammenarbeiten können, um einen umfangreiche Netzwerksicherheit anzubieten. Intrusion Detection System (IDS) identifizieren und berichten über Cyberangriffe, indem er Netzwerkverkehr überwacht. Nach der Erkennung eines verdächtigen Verkehrs, muss das SOC Team zur Handlung kommen. Intrusion Prevention System (IPS) überwacht den Netzwerkverkehr und kann die Verbindung automatisch unterbrechen, falls es verdächtig ist (Wendzel, 2018). Ein IPS kann konfiguriert werden, um automatisch nach festgelegten Muster zu handeln. Beide Tools können Logdateien generieren, die von einer SIEM-Lösung oder Log-Analyse-Tools gesammelt werden können. Die Abbildung 2 stellt eine allgemeine Struktur von SIEM-Lösungen dar:

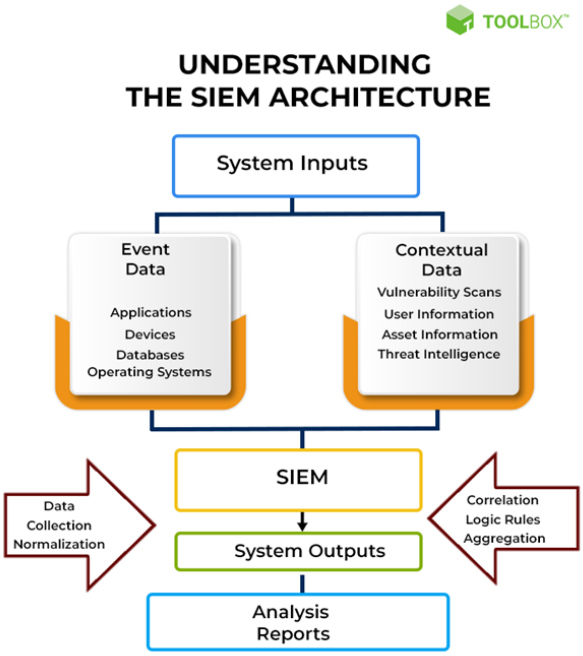


Abbildung 2: Allgemeine Struktur eines SIEM
Quelle: (Mohanan, 2022)

Obem auf dem Bild, sehen wir, dass es zwei wichtige Datenquelle gibt, auf der linken Seite sind die Logdateien der Endpoints und auf der rechten Seite die Informationen,

um Anomalie zu erkennen. Nur der linken Seite stellt ein Log-Analyse-Tool dar. Mit der Nutzung der Elementen der rechten Seite, werden die Daten verarbeitet, um Muster zu erkennen und Information herauszuholen. Diese Zusammenarbeit repräsentiert eine SIEM-Lösung, die als Ergebnis ein oder mehrere Berichten und/oder Graphik ausgeben kann. Granadillo et al. (2021) teilt ein SIEM in unabhängige Blöcke auf, wo jeder Block eine spezifische Funktion hat. Diese Blöcke und die Richtung der Information werden werden in der Abbildung 3 dargestellt:

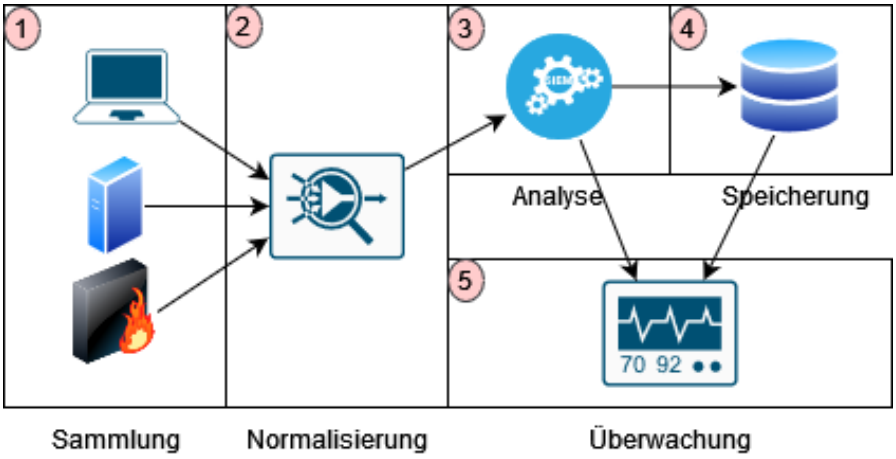


Abbildung 3: Allgemeine Informationsfluss eines SIEM nach Granadillo et al. (2021)

In der Abbildung 3 sehen wir den Informationsfluss, wo die Logdateien erstellt werden, bis ihr Inhalt bearbeitet und verarbeitet wird. Die Logdateien der Endpoints werden von einem sogenannten *collector* gesammelt (1). Diese werden dann angepasst, damit sie eine einheitliche Formatierung bekommen (2), da sie verschiedene Format und Inhalte beinhalten. Danach werden die normalisierten Daten analysiert und nach Angriffsmuster verarbeitet (3). Der Inhalt wird dann in einer Datenbank gespeichert (4) und sowohl das Ergebnis der Analyse als auch der Inhalt können von einer Überwachungstool in Graphik- und/oder Textformat aufgerufen werden (5) Granadillo et al. (2021).

Aus bisheriger Abbildung stellen wir fest, dass SIEM das Ergebnis der Integration von zwei wichtigen Komponenten ist, Datensammlung und Verarbeitung. Das Ziel dieser Soft-

ware ist es die automatische Analyse zu ermöglichen, indem Daten kombiniert und bewertet werden können. In vielen Bereichen, wie Finanzen (Payment Card Industry Data Security Standard (PCDI DSS)), Gesundheitswesen (Health Insurance Portability and Accountability Act (HIPAA)), sind SIEMs eine gesetzliche vorgegeben (Jog, 2020). In Deutschland verpflichtet das Zweites Gesetz zur Erhöhung der Sicherheit informationstechnischer Systeme die Anwendungen solcher Lösungen, um Schädigung der Confidentiality, Integrity and Availability (CIA) zu verhindern (BSI, 2021). Log-Analyse-Tools sind seinerseits Tools zu der Speicherung, Anpassung, Bewertung und Darstellung von Logdateien, ohne dass sie sich auf die Sicherheitsebenen fokussieren.

2.1. Existierende SIEMs Lösungen und Log-Analyse-Tools

Die existierenden SIEMs und Log-Analyse-Tools werden in *Proprietär* und *Open Source* getrennt. In folgenden Abschnitten präsentieren wir das *Proprietär*e Tool Splunk, um einen Maßstab für unsere Auswahl über Funktionalität zu definieren und analysieren zusätlich folgende Tools:

- AlienVault Open Source Security Information Management (OSSIM)
- Prelude
- FortiSIEM
- Elastic Stack
- Grafana integriert mit Loki

2.1.1. Splunk

Splunk, von dem gleichnamigen Unternehmen, wurde 2003 in den USA auf dem Markt gebracht (Splunk, 2022b). Splunk bietet einfache Wartung, benutzerfreundliche GUI und Skalierbarkeit (Kazarov et al., 2018). Er gehört zu der meistverwendeten SIEM und zu ihren Kunden gehören große Konzerne wie Airbus, Coca-Cola, Intel und die Deutsche Bahn. Splunk bietet laut seiner Webseite folgende Funktionalitäten an (Splunk, 2015a):

- Skalierbare Datenplattform
- Risk-based Warnmeldung
- Bedrohungserkennung mithilfe von Machine Learning (ML)
- Automatische Aktualisierung von der Bedrohungs- und Schwachstelle-Database
- Unkomplizierte Installation und Anwendung

Die Architektur und der Informationsfluss von Splunk unterscheidet sich nicht von der oben dargestellten Struktur in der Abbildung 2 und Abbildung 3. Da es sich hier um eine proprietäre Lösung handelt, lässt sich Splunk mit anderen Funktionalitäten verwalten und erweitern.

Die nächste Abbildung, 4, zeigt ein zusammenfassendes Diagramm über den Umfang des Informationsflusses von Splunk laut Splunk (2015b):

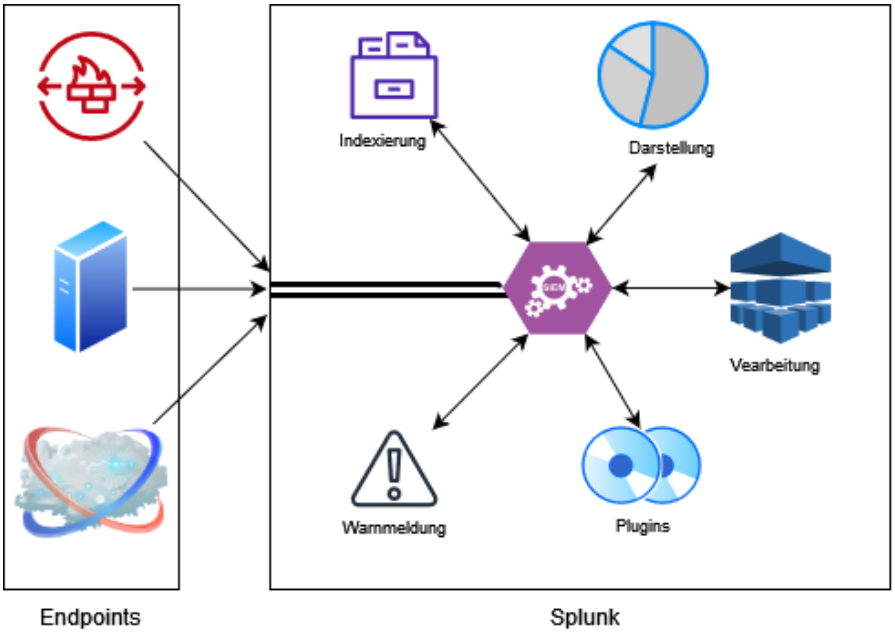


Abbildung 4: Allgemeine Informationsfluss von Splunk

Auf der Abbildung 4 haben wir links die Endpoints, dessen Logdateien von Splunk und seine Plugins und Funktionalitäten (rechts) verarbeitet und analysiert werden. Der Kon-

zept von Splunk lässt sich von der Idee formulieren, dass verschiedenen und unabhängige Funktalitäten zusammenarbeiten (Splunk, 2015b).

Wie in anderen Tools, funktioniert die Bedrohungserkennung mithilfe von Regelsätzen, die aus Uses Cases entstehen. Laut der Dokumentation existieren sie in folgenden Szenarien: Überwachung, Untersuchung und Erkennung. Die Software ist sowohl mit Mitre ATT&CK Matrix als auch mit Cyber Kill Chain (CKC) für die Gestaltung ihrer Erkennungsregel integriert (Splunk, 2022a).

In der wissenschaftlichen Literatur haben wir Su et al. (2016), wo Angriffe auf einem System simuliert und schließlich mit Splunk analysiert wurden, um Gefahren zu identifizieren und diese im Voraus zu sehen. In Selvaganesh et al. (2022) wurde beschrieben, wie eine Splunk-Instanz installiert und konfiguriert wird, um spezifische Brute-Force Angriffe zu erkennen.

2.1.2. AlienVault OSSIM

AlienVault OSSIM ist eine im Jahr 2007 entwickelte Open Source SIEM Lösung. Im Jahr 2018 wurden sie von der Firma AT&T Communication gekauft (CBNINSIGHTS, 2020). In der Beschreibung des Anbieters steht, dass er sie auch dabei unterstützt, Daten zu sammeln, zu normalisieren und zu bewerten. Er behauptet auch, dass sein Tool in der Lage ist, Schwachstellen und Angriffe zu erkennen, das Verhältnis zu beobachten und Datenzusammenhänge zu erschließen (AT&T Cybersecurity, 2022).

AlienVault hat eine kostenpflichtige Version, die Alien Vault Unified Security Management (USM) heißt. Auf der Webseite von AT&T steht, dass es keine dedizierte Dokumentation für die Open Source Version AlienVault OSSIM gibt, da viele Funktionalitäten von der kostenpflichtigen Version stammen (AT&T Cybersecurity, 2022).

Die nächste Abbildung 5 stellt das von dem Anbieter freigelegte Architekturdiagramm von der USM Version dar (AT&T Cybersecurity, 2022):

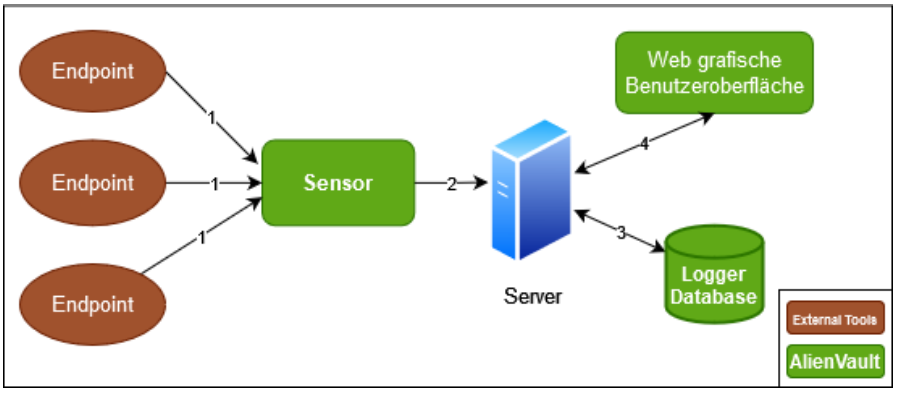


Abbildung 5: Architekturdiagramm von AlienVault USM

Links auf der Abbildung 5 sehen wir die mit einem Sensor verbundenen Endpoints (1). Der Sensor analysiert die Daten und leitet diese zu dem Server (2) weiter (Vault, 2019). Die Daten werden auf Logger (3) gespeichert und über die GUI (4) dargestellt.

Laut der Website Comparitech steht AlienVault auf dem 13ten Platz von den bestbewertetsten SIEM-Lösungen. Die Seite beschreibt auch, dass zu dem Tool einen IDS, ein Verhaltensüberwachungssystem und einen Schwachstellen-Scanner integriert sind. Die Anwendung ist auch mit der Plattform Open Threat Exchange(OTX) verbunden - diese ermöglicht eine Teilung von Informationen über die Schwachstelle. Comparitech highlighted, dass die Anwendung wegen ihrer niedrigen Kosten besser für kleine oder mittelständige Unternehmen geeignet ist (comparitech, 2023).

Die Anwendung soll konsistenten Daten Zusammenhang anbieten und das Auftauchen von Falsch Positiv vermeiden. AlienVault kommt mit vordefinierten Uses Cases, die dabei unterstützen, gewöhnliche Angriffsszenarien zu erkennen. Die Installation, die Einstellung und die Integration mit anderen Tools ist auch benutzerfreundlich (Gómez et al., 2022). Nabil et al. (2017) behauptet, dass für viele Quellen eine manuelle Normalisierung der Logdatein notwendig ist.

Die meisten Publikationen über AlienVault OSSIM stammen aus kommerziellen Quellen und diese konzentrierten sich auf eine kostenpflichtige SIEM-Lösung von AT&T.

2.1.3. Prelude

Das im Jahr 2002 in Frankreich von Yoann Vandoorselaere released Tool Prelude zählt zu einer europäischen Open Source SIEM Lösung. Laut dem Anbieter verfügt Prelude unter anderem folgende Funktionalitäten (Prelude SIEM, 2018):

- Informationszentralisierung
- Datenaggregation und -Zusammenhang mit vordefinierten und von dem Nutzer angepassten Regeln
- Einbruchserkennungsmechanismen
- Datennormalisierung

Die Anwendung besteht aus verschiedenen unabhängigen Modulen. Unter denen nennen wir Warnmeldung, Archivierung, Analyse und Verwaltung. Erstens gehört zu der zentralen Aufgabe dieser Lösung - es kann folgendens: Daten empfangen, normalisieren, Zusammenhänge erschließen und Meldungen generieren. Das zweite Modul, Archivierung, konzentriert sich auf die Speicherung und Verfügbarkeit der Daten. Der Analyse-Modul stelt Daten in verschiedenen Formaten dar. Das letzte Modul, Verwaltung, dient dazu, die Anwendung zu steuern, Nutzer zu erstellen und deren Rechte zu konfigurieren (European Comission, 2015).

Die Abbildung 6 zeigt die Integration verschiedener Module von Prelude und wie sie mit einander kommunizieren, um Analyse, Meldung und Speicherung zu generieren (Prelude Team, 2007):

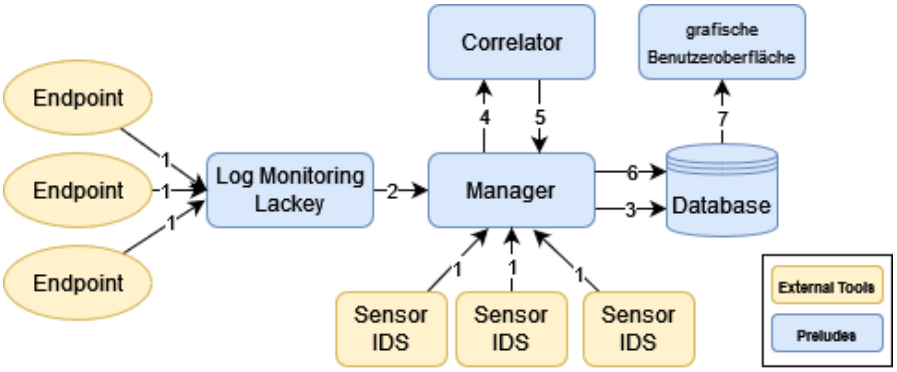


Abbildung 6: Integration zwischen den Modulen von Prelude

Aus der Abbildung und der Dokumentation können wir folgenden Informationsfluss erkennen - die Daten werden von Endanwendung generiert und zum Loganalyzer (Prelude Log Monitoring Lackey (LML)) (1) geschickt, wo sie normalisiert und bewertet werden Für die Logs, wo es verdächtige Werte nach dem vordefinierten Regelsätze in Log Monitoring Lackey (LML) gibt, werden Warnmeldungen generiert. Diese Meldungen werden zum Manager Module (2) weitergeleitet. Der Manager kann auch Daten von IDS mithilfe von Sensoren empfangen (1), die an den Endpoints installiert sind, um Events zu analysieren und zum Manager zu schicken. Die Daten werden in der Database gespeichert (3) und auch zum Correlator weitergeleitet (4), wo dieser nach einem Zusammenhang zwischen anderen Daten sucht. Das Ergebnis von Correlator wird wieder zum Manager (5) geschickt und danach zu der Datenbank (6). Schließlich stehen die Berichte in dem GUI (7) zur Verfügung (Prelude SIEM, 2020).

Die Architektur der Anwendung ermöglicht sowohl einen zentralisierten als auch einen dezentralisierten Aufbau, wie auf der Abbildung 7 gezeigt wird. In der ersten funktioniert Prelude als zentral und empfängt Daten von verschiedenen Datenquelle und in den zweiten gibt es mehrere sogenannten „Branches“ von Preludes, dessen Manager sich miteinander verbunden sind, damit das Ergebnisse sich in einem GUI darstellen lassen (Prelude Team, 2007).

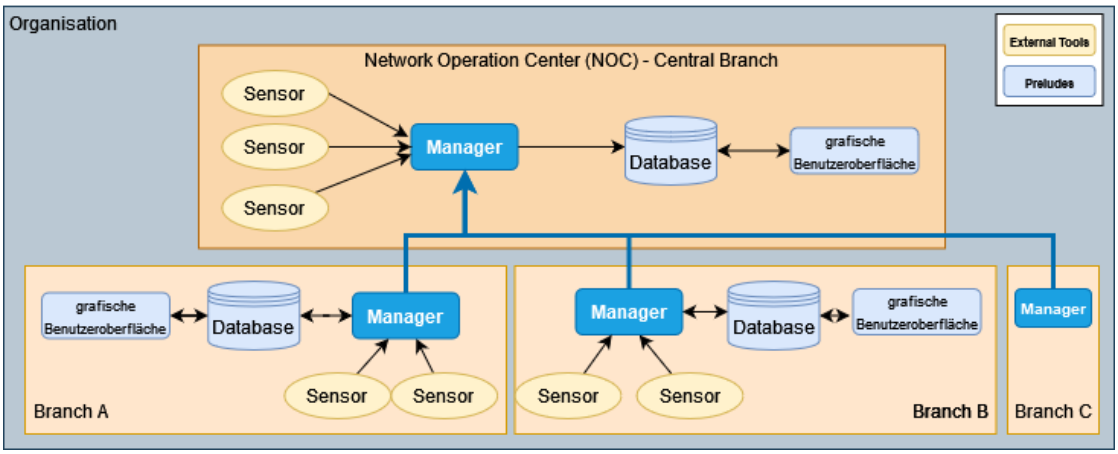


Abbildung 7: Erweiterte Architektur von Prelude mit dezentralisierten Datenquellen Datenverarbeitung

Radoglou-Grammatikis et al. (2021) hat Preludes, AlienVault und Cyberoam iView anhand technischer und nutzerfreundlicher Kriterien zu verglichen. Von diesn Kriterien nennen wir folgende (Radoglou-Grammatikis et al., 2021):

- **technische Kriterien**
  - Echtzeige Leistung
  - Umfang und Flexibilität der Meldungen
  - Zusammenhang von Warnmeldung
- **nutzerfreundliche Kriterien**
  - Vollständige Dokumentation
  - Schwierigkeitsgrad der Installation
  - Schwierigkeitsgrad der Einstellung

In den technischen Kriterien lag Prelude auf dem dritten Platz und bei Benutzerfreundlichkeit bekam Prelude den ersten.

2.1.4. FortiSIEM

FortiSIEM ist eine SIEM-Lösung von der US-amerikanische Firma Fortinet. Fortinet kaufte im Jahr 2016 das Unternehmen AccelOps und dessen SIEM-Lösung und benannte es zum FortSIEM (Fortinet, 2016).

Laut dem Anbieter hat FortiSIEM eine robuste Integration mit anderen Tools und lässt sich leicht und einwandfrei skalieren. Andere Versionen des Tools sind mit Machine Learning (ML) integriert, sodass die Anwendung auch Verhältnisanalysen durchführen kann (Fortinet, 2022). Das Tool bietet auch eine umfangreiche und ausführliche Dokumentation an. Die nächste Abbildung, 8, zeigt die skalierbare Architektur von FortiSIEM:

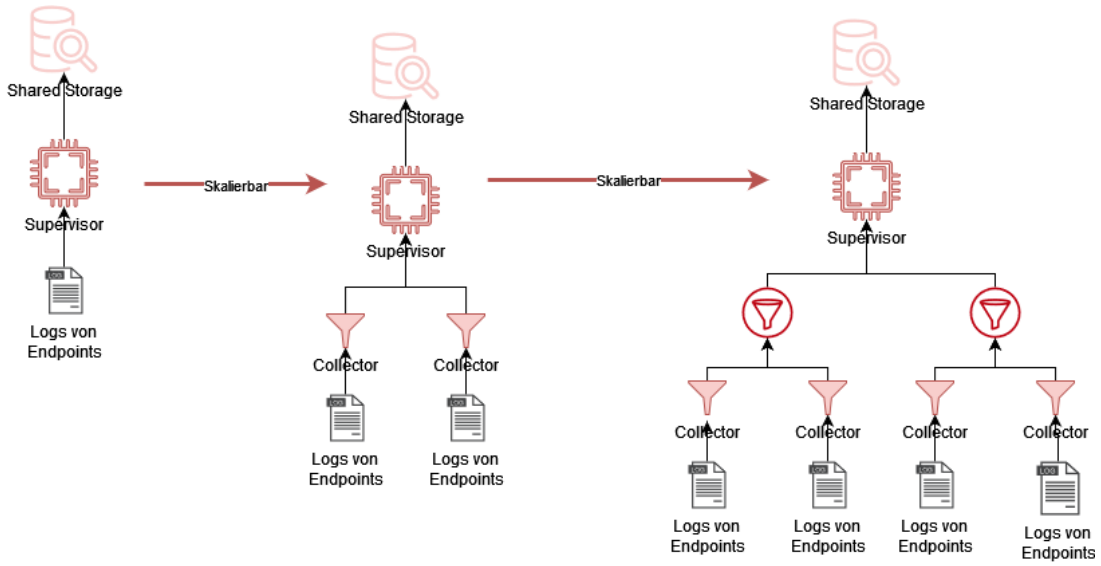


Abbildung 8: Skalierbare Architektur von FortiSIEM laut Fortinet (2020)

Auf der linken Seite der Abbildung 8 haben wir eine einfache Struktur, wo die Logdatei zu dem „Supervisor“ geschickt ist. Diese ist für die Analyse und Darstellung der Daten zuständig. Wenn die Architektur skaliert (mitte), kommen die „Collectors“. Diese tragen dazu bei, die Skalierbarkeit zu unterstützen, da sie Logdateien aus verschiedenen Quellen empfangen und normalisieren. Auf weitere Erweiterung des Tools (rechts) kommen dann die „Workers“, die die Datananaylse durchführen und für die Database zuständig ist

(Fortinet, 2018).

In der wissenschaftliche Literatur sagt Ramírez Tomás (2018), dass FortiSIEM eine schnelle Erkennung von Angriffen bietet und über Network Operations Center (NOC) Funktionalität verfügt, wie Netzmanagement. Wie andere SIEMs Lösungen, biete FortiSIEM die folgenden Funktionalitäten: Datensammlung und Normalisierung, Daten Zusammenhang, Generierung von Berichten, Warnmeldungen, Datenauswertung.

2.1.5. Elastic Stack

Elastic Stack stammt aus der Verbindung von drei Tools: Elasticsearch, Logstash und Kibana. Erstes ist eine Such-und Analyse-Maschine. Das Zweite ist eine serverseitige Anwendung zur Datenverarbeitung, -Weiterleitung und Sammlung von Logdateien. Schließlich Kibana hat eine eigene Abfragesprache, die dafür zuständig ist, Daten zu filtern und visuelle Darstellungen in einem Grafik-Format auszugeben (packt, 2019). Von diesen drei Tools Logstash ist das einzige Open Source (elastic, 2021). Obwohl die anderen zwei kostenlos verwendet werden können, gehören sie nicht zu der Open Source Kategorie (Open Source Initiative, 2007). Dieses Tool besitzt viele Eigenschaften einer SIEM-Lösung und wird von vielen SOC verwendet, ist aber für viele Experten, kein SIEM für sich, da es über keine Warnmeldungssystem, Daten Zusammenhang und Vorfälleverwaltung verfügt (Miller, 2021). Diese und anderen Funktionalitäten lassen sich aber durch Plugins integrieren.

Die nächste Abbildung, 9, stellt die Architektur von Elastic Stack mit ihren integrierten Elementen dar:

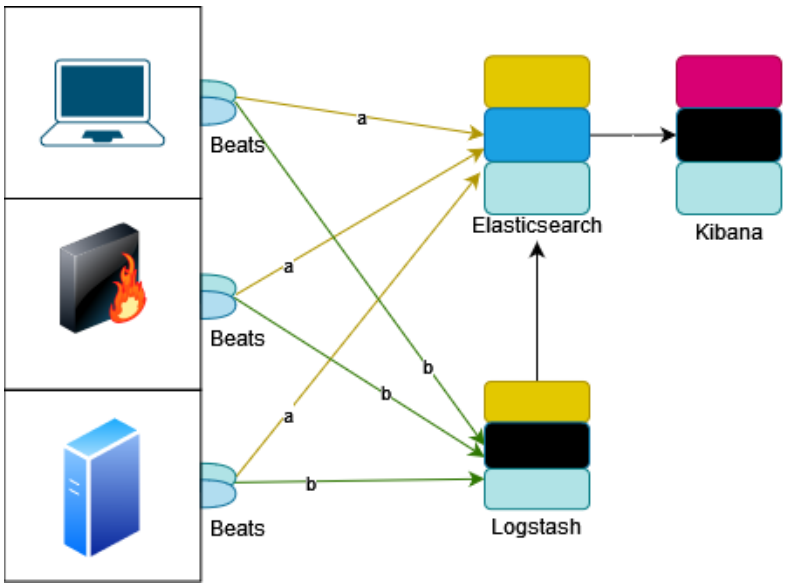


Abbildung 9: Integration zwischen Elasticsearch, Logstash und Kibana laut packt (2019)

Die Beats auf der Abbildung sind an der Endpoints installiert und leiten Daten entweder zu Elasticsearch (a) oder zu Logstash (b) weiter. In Elasticsearch werden die Daten nach Muster mithilfe der Abfragesprache Query Domain Specific Language (Query DSL) s gesucht. In Logstash werden die Daten verarbeitet, gespeichert und weitergeleitet. Schließlich stellt Kibana die Daten grapisch über die GUI dar(Jain, 2018).

Advani et al. (2020) recherchiert über die Log Analyse-Funktionalitäten von Elastic Stack und die Unterstützung bei Normalisierung und Indexierung von Daten für eine lesbare Ausgabe. Die Skalierbarkeit wurde in der Studie von Wang et al. (2019) erwähnt, wo Elastic Stack für Wi-Fi Logging eingesetzt wurde.

Die offizielle Dokumentation von Elastic Stack beschreibt, dass die Anwendung folgende Funktionalitäten besitzt (elastic, 2022):

- Datensuche, -Normalisierung, -Analyse und

- Speicherung
- visuelle Ausgabe

Die nächste Abbildung, 10 zeigt laut der offiziellen Dokumentation die Aufteilung der Funktionalitäten pro Element von Elastic Stack:

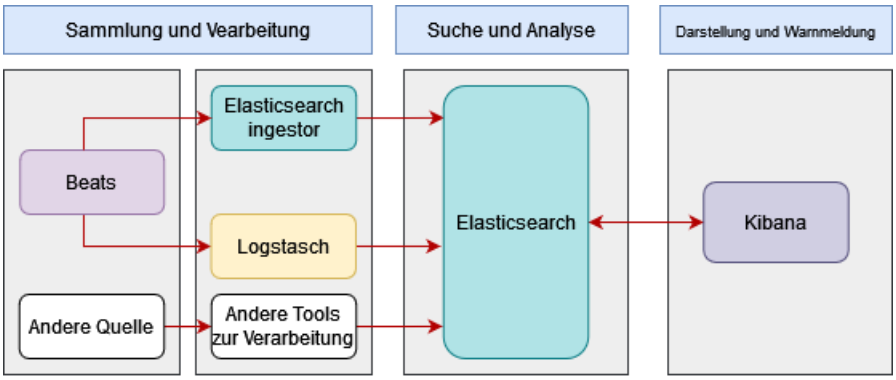


Abbildung 10: Aufteilung der Funktionalitäten zwischen den Komponenten

Die aufgeteilte Struktur auf der Abbildung 10 hat die folgende Komponenten: die „Ingestor“, die für das Hinzufügen, die Sammlug und die Anpassung des Inhalts der Logdateien zuständig ist; die Speicherung, wo die Daten analysiert werden und schließlich der „Verbrauchen“, wo die Darstellung und die Warnmeldungen stattfinden (elastic, 2022).

Die wissenschaftliche Publikation über Elastic Stack ist vielfältiger als bei anderen recherchierten Tools. Die Mehrheit von denen beschäftigen sich eher mit dem Logging als mit den SIEM-Eigenschaften der Anwendung.

2.1.6. Grafana

Von allen recherchierten Lösungen ist Grafana die Einzige, die weder SIEM noch Log-Analyse-Tools ist. Grafana wird als Frontend Plattform für Visualisierung von Daten beschrieben. Mit dem Tool ist es möglich Graphiken zu erstellen und Warnmeldungen zu generieren. Das Ziel der Anwendung ist, Information in einer einfachen und verständlichen Art und Weise zur Verfügung zu stellen (redhat, 2022).

Im Jahr 2014 wurde Grafana von der Firma Grafana Labs released. Ursprünglich sollte Grafana ein einfacheres Bearbeitungstool für Grafiken sein und ermöglichen, Datenanfragen unkomplizierter zu machen. Das Tool sollte auch mithilfe von Plugins erweitern werden (Ödegaard, 2019).

In der Webseite betont der Anbieter, dass Grafana die Zentralisierung und Zugang von Daten vereinfachen. Alle Art von Daten lassen sich analysieren und darstellen, von der Leistung von Anwendungen bis Verkaufsdaten und Krankheitsfällen. Die Anwendung soll auch den Zusammenhang von Daten ermöglichen, um wichtige Informationen herauszunehmen (Grafana Labs, 2016a). Grafana ermöglicht die Aufrufe der Daten mithilfe von verschiedenen Abfragesprache, jenachdem welche Datenquelle verwendet wird. Das Tool bietet auch eine eigene Funktionalität, um Warnmeldung zu generieren oder lässt sich mit externen Tools, wie Alertmanager von Prometheus, integrieren.

Grafana ist auch mit dem Logging Tool Loki und Promptail integriert. Promtail ist für Sammlungen der Logdateien und Weiterleitung an Loki zuständig während Loki sich um die Speicherung, Indexierung und Abfrage kümmert.

Die Architektur von Loki besteht aus folgenden Komponenten: „distributor“, „ingester“ und „querier“, wie auf der Abbildung 11 dargestellt:

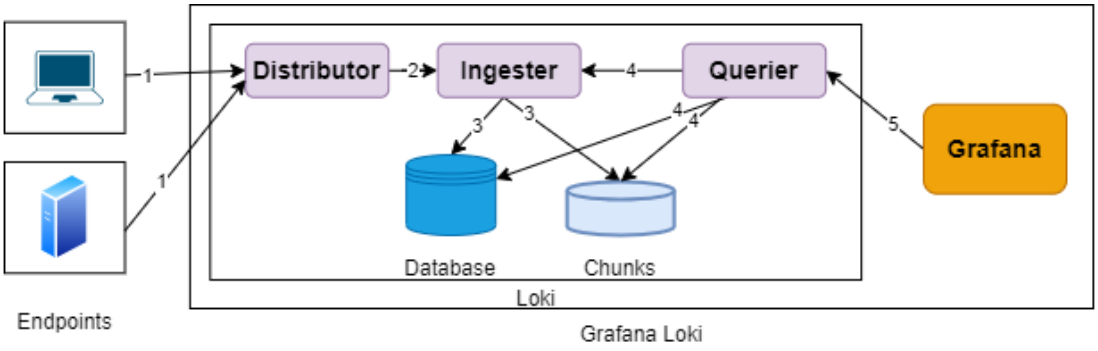


Abbildung 11: Architektur von Loki Veeramachaneni (2018)

Auf der Abbildung 11 empfängt „distributor“ die Streams von den (1) Endpoints und

überprüft ihre Richtigkeit im Bezug auf den Zeitstempel und Länge der Logzeile. In dem „distributor“ wird auch die Aufnahmenrate konfiguriert. Der „distributor“ leitet die Streams zum (2) „ingester“ weiter, wo er die Daten in kleinen Datenblöcken (3) speichert. Diese Daten werden wiederum in der Database (3) je nach „label“ aufgenommen werden. Dieses Verfahren soll Skalierbarkeit und Fehlertoleranz ermöglichen. Schließlich beschäftigt sich der „querier“ mit der Abfragen (4) in der Abfragesprache LogQL. Der „querier“ holt Daten sowohl von „ingester“ als auch von der Database. Sobald dieses Verfahren fertig ist, kann Grafana Frontend nach diesen Daten fragen (Grafana Labs, 2021b) und (Veeramachaneni, 2018).

Promptail wird an jeden Endpoint installiert und schickt den Inhalt der Logdateien als Stream zu Loki. Diese Streams werden dann in Loki mit Labels (Schlüssel-Wert-Paar) identifiziert. Jeder Stream ist ein Teil des Inhalts der Logdateien. Laut Grafana Labs (2016b) bleibt der Stream ohne Indexierung, um Effizienz bei der Verwaltung des Inhalts zu gewinnen. Im Vergleich zu anderen Tools, wie Elastic Stack, wird der gesammte Inhalt indexiert (Anand, 2023). Bei Loki werden nur die Metadata, wie Zeitstempel oder andere kundenspezifische Labels, dann indexiert.

Für eine optimale Nutzung von Grafana wird es empfohlen, wenig „labels“ zu benutzen, um eine Eskalation des Speicherbedarfs zu vermeiden Welch (2020). Die Abbildung 12 zeigt den Eskalationseffekt, der durch die Nutzung von „labels“ entsteht:

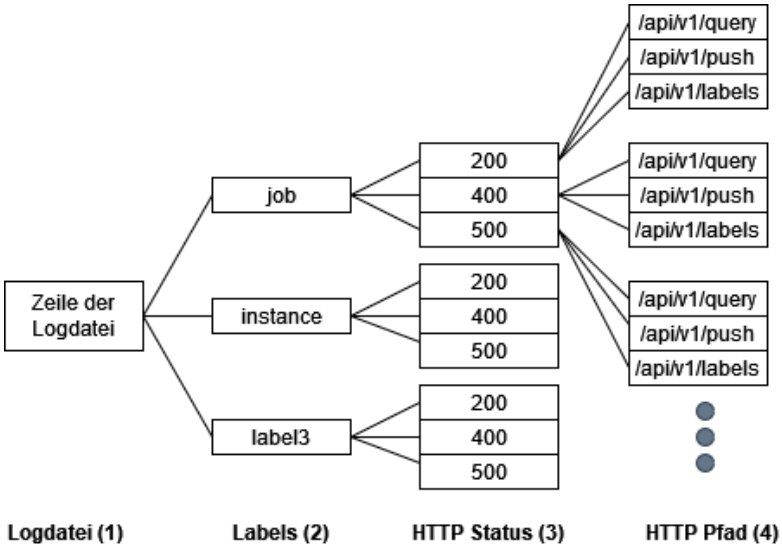


Abbildung 12: Eskalation bei Verwendung von „labels“ laut Welch (2020)

Auf Abbildung 12 haben wir vier Ebenen. Die erste ist die Zeile der Logdatei (1), die zweite sind die vom Benutzer definierten „Labels“ (2), und die dritte und vierte beziehen sich auf die Antwort (3) und die verwendete HTTP-Anfrage (4), um nach Inhalten zu suchen, sie hinzuzufügen und zu indizieren. Aus einer Zeile der Logdatei haben wir drei kundendefinierte „Labels“ definiert, die wiederum drei potenzielle Zustände haben. Aus diesen drei Zuständen haben wir drei mögliche HTTP-Pfade. Aus drei „Labels“ ergibt sich insgesamt 27 Streams, um die Zeile zu lesen, zu indizieren und zu speichern. Dies kann laut Welch (2020) zu einer Beeinträchtigung der Leistung führen. Die folgende mathematische Formel zeigt das Ergebnis unserer Beschreibung:

```
Labels = 3
Anfrage = 3
HTTP-Antworte = 3
Total Streams = Labels x Anfrage x HTTP-Antworte
Total Streams = 3 x 3 x 3
Total Streams = 27
```

Promtail kann aber Logdateien nur zur Grafana Loki oder zu einem anderen Promtail Instanz schicken. In der Konfigurationsdatei von Promtail kann die Häufigkeit, in der

nach neuem Inhalt in der Logdateien gesucht wird, konfiguriert werden. Auf Abbildung 13 wird die Verbindung zwischen Promtail und Grafana Loki dargestellt (Grafana Labs, 2022a):

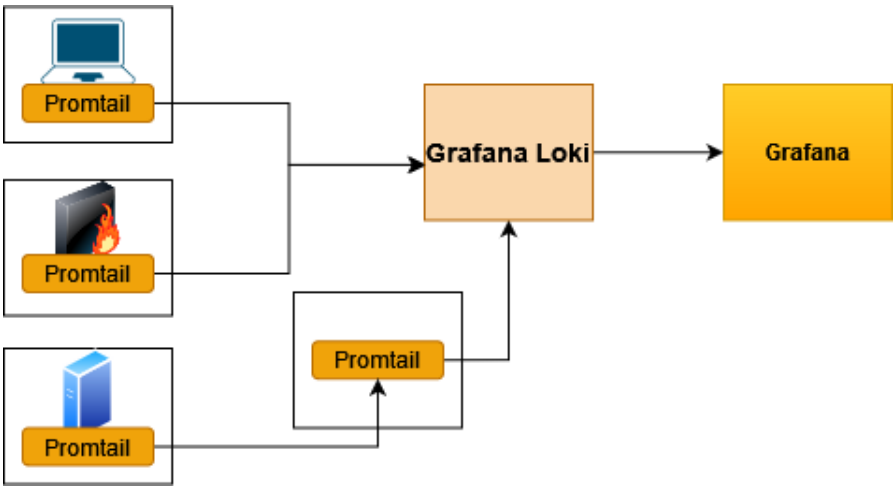


Abbildung 13: Integration von Log-Quellen mit Promptail (links), Loki (mitte) und Grafana (rechts)

Die wissenschaftlichen Literatur über Grafana konzentriert sich eher auf die Anwendung des Tools für die grafische Darstellung von Daten als für ihre Nutzung in dem Sicherheitsbereich. Eine Recherche von Manases and Zinca (2022) wollte das Ergebnis von der Überwachung von Cloud-basierten Systemen, von Netzwerkaktivitäten und von Netzwerkverkehr mithilfe von Grafana darstellen. Die wissenschaftliche Recherche über die Implementierung und Integration von Grafana mit anderen Tools zum Sicherheitszweck ist neu und bietet deshalb viele Perspektive für die Weiterentwicklung an.

2.2. Auswahlkriterien

Der Erwerb einer SIEM Lösung würde wahrscheinlich die Anforderungen dieser wissenschaftlichen Arbeit decken. Da solche Lösungen meistens (oder alle) Proprietär und kostenpflichtig sind, legten wir als Auswahlkriterium fest, dass die Anwendungen für unsere Arbeit Open Source sein müssen. Von allen analysierten Tools sind die Kombinationen

Grafana, Loki, Promtail und Kibana, Elasticsearch, Logstash am geeignetsten für unsere Auswahl. In der Tabelle 1 vergleichen wir beiden dieser Konbinationstool (Anand, 2023):

Grafana	**Elastic Stack**	**Gemeisame Rolle**
Grafana	Kibana	Frontend, graphische Darstellung
Loki	Elasticsearch	Backend, Verarbeitung des Inhalts der Logdateien
Promtail	Logstash	Backend, Sammlung von Logdateien

Tabelle 1: Gemeinsamkeiten zwischen den Kombinationen Grafana, Loki, Promtail und Kibana, Elasticsearch, Logstash

Die nächste Tabelle, 2, fasst die Unterschieden zwischen den Tools zusammen:

Eigenschaft	**Grafana, Loki und Promtail**	**Kibana, Elasticsearch und Logstash**
Opensource	ja	nicht im Elasticsearch
Dokumentation von Promtail und Logstash (Kray, 2022)	einfach	Umfangreich
Entwickelt spezifisch für Verarbeitung von Logdateien (Yigal, 2013)	nein	ja
Komplexität bei der Installation und Einstellung (Better Stack Team, 2023)	niedrig	hoch
Graphische Darstellung	ja	ja
Dashboards und Graphik	ja	ja
Eigene Abfragesprache	abhängig von dier Datenquelle (Grafana), LogQL (Loki)	Query Domain Specific Language (Query DSL)
Indexierung	nur Metadata (Loki)	vollständig (Elasticsearch)
Dekomprimierung von Datei	ja (Promtail)	ja (Logstash)
Integration mit anderen Database Tools	ja	nur mit Elasticsearch
Weiterleitung des Loginhalts	nur an Promtail und an Loki (Promtail)	umfangreiche Integration mit anderen Tools (Logstash)

Eigenschaft	Grafana, Loki und Promtail	Kibana, Elasticsearch und Logstash
Integrietes Tool für Generierung von Warnmeldungen (Yigal, 2013)	ja (Alerting)	von Plugins abhängig

Tabelle 2: Unterschiede zwischen den Kombinationen Grafana, Loki, Promtail und Kibana, Elasticsearch, Logstash

Grafana erfüllt die folgende Voraussetzungen für unsere Auswahl: 100% Open Source, integriertes Alerting Tool und breiter Kompatibilität (außer Promtail). Da Grafana nicht spezifisch für Logdateien konzipert wurde (Yigal, 2013), verlangt die Einstellung spezifische Anpassungen, um unseren Bedürfnisse vollständig zu erfüllen.

In den nächsten Kapiteln beschäftigen wir mit der Integration dieser Tools, um eine ähnliche SIEM Lösung zu präsentieren. Wir beschreiben auch, wie Cyberangriffe anhand der Taktiken, Techniken, Prozeduren (TTP) der Mitre ATT&CK Matrix erkennt werden können. Für die Generierung der Logdateien mit Angriffsmustern in ihren Inhalten simmulieren wir zwei Cyberangriffe. Danach beschäftigen wir uns mit der Installation, Einstellungen und Sammlungen von Logdateien in Promtail, Grafana und Loki. Nachdem die Grundfunktionalitäten eingerichtet sind und einwandfrei funktionieren, untersuchen wir Regelsätze für die Erkennung und Warnmeldung von potenziellen Angriffe. Schließlich überprüfen wir unseren Aufbau anhand spezifischen Logdateien aus der Hochschule Worms. Unser Ziel ist Grafana, Loki und Promtail so einzustellen, dass es in der Lage ist, die Muster dieser Cyberangriffe zu erkennen und darüber zu berichten.

3. Implementierung

In diesem Kapitel geht es um die Implementierung und den Aufbau von Grafana, um Cyberangriff mithilfe der Mitre ATT&CK Matrix zu erkennen. Das Labor wird mit einem Container und virtuellen Maschine (VM) aufgebaut, wie im Diagramm in der Abbildung 14 dargestellt.

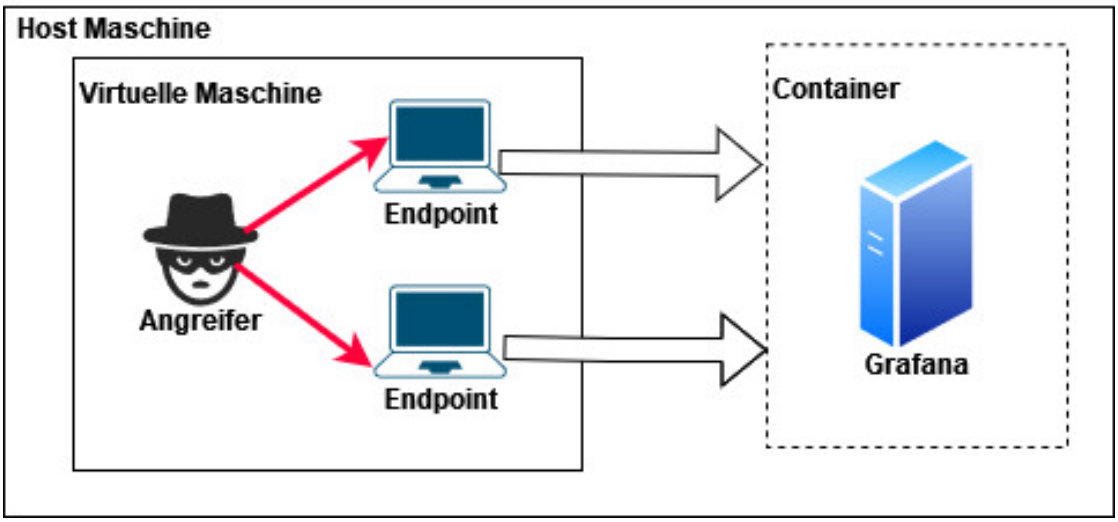


Abbildung 14: Aufbau unseres Arbeitslabors

Unser Aufbau verfolg folgende Ziele: die Aufnahme und Anpassung von Logdateien für Grafana, die Mustererkennung für ausgewählte Cyberangriffe und schließlich die Erstellung von Warnmeldungen für die Endnutzer, damit sie geeignete Sicherheitsmaßnahmen ergreifen können.

Der gezielte Ablauf unserer Arbeit ist in der Abbildung 15 dargestellt:

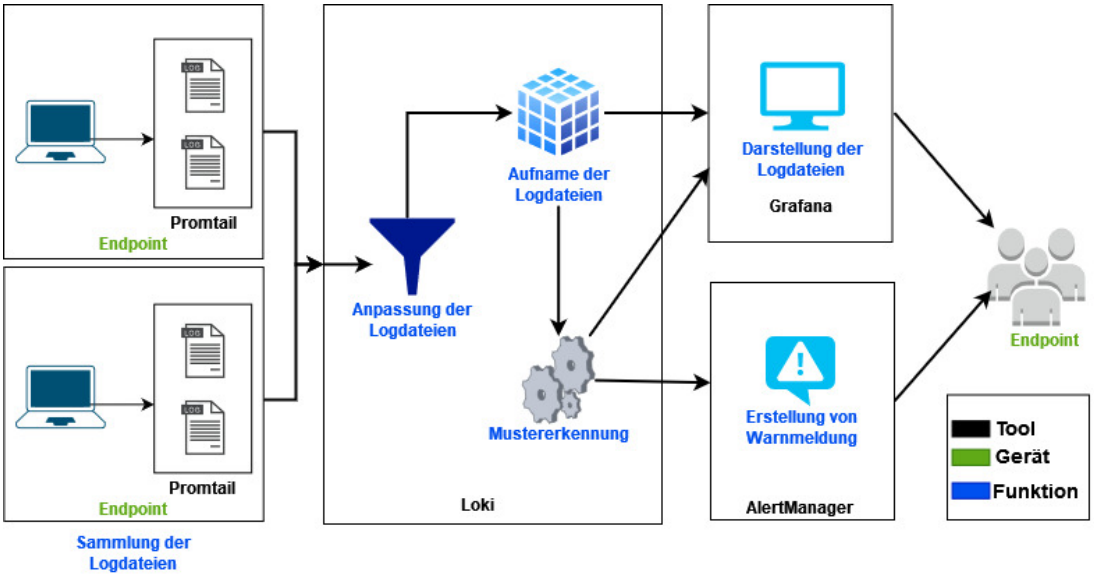


Abbildung 15: Erwarteter Ablauf der Sammlung der Logdateien bis zur Warnmeldung

3.1. Angriffserkennung anhand der Mitre ATT&CK Matrix

Es gibt verschiedene Methoden und Frameworks, die von SOC-Teams verwendet wird, um Cyberangriffe zu vermeiden, zu erkennen und zu unterbrechen. Da sich die Richtlinien und Schwerpunkte dieser Frameworks und Methoden unterscheiden können und somit unterschiedliche Anforderungen an den Aufbau unserer Struktur stellen könnten, entschieden wir uns für die Mitre ATT&CK Matrix, insbesondere da dieses Framework auch in Splunk integriert ist.

Die Mitre ATT&CK Matrix hat folgenden Zwecke (MITRE ATT&CK, 2018b):

- Erkennung und Analyse von Angriffstechnik
- strukturierte Datensammlung über Bedrohungen
- Emulieren von Cyberangriffe für die Anwendung an Angriffsübungen
- Systemhärtung und Verbesserung der Verteidigungsmaßnahmen

Die Matrix ermöglichen Unternehmen und SOC-Teams umfassende Möglichkeiten, um ihre Ressourcen zu schützen und ihr Fachwissen im Bereich der Cybersicherheit zu erweitern (Hazel, 2021). In dieser Arbeit konzentrieren wir uns auf die Entwicklung und Implementierung einer Methode zur automatischen Erkennung und Analyse von Angriffstechniken in Grafana.

Die Mitre ATT&CK Matrix ist auf Taktiken, Techniken, Prozeduren (TTP) basiert. Angriffe, Gegenmaßnahmen und Erkennung werden nach TTP definiert. Die Matrix besteht aus 14 Taktiken, zu denen jeweils Techniken gehören, die wiederum in Sub-Techniken unterteilt sind. Jede Sub-Technik wird mit Beispielen, Härtungsmaßnahmen und Erkennungsregeln beschrieben. Die nächste Abbildung, 16, zeigt, wie die TTP aufgebaut werden:

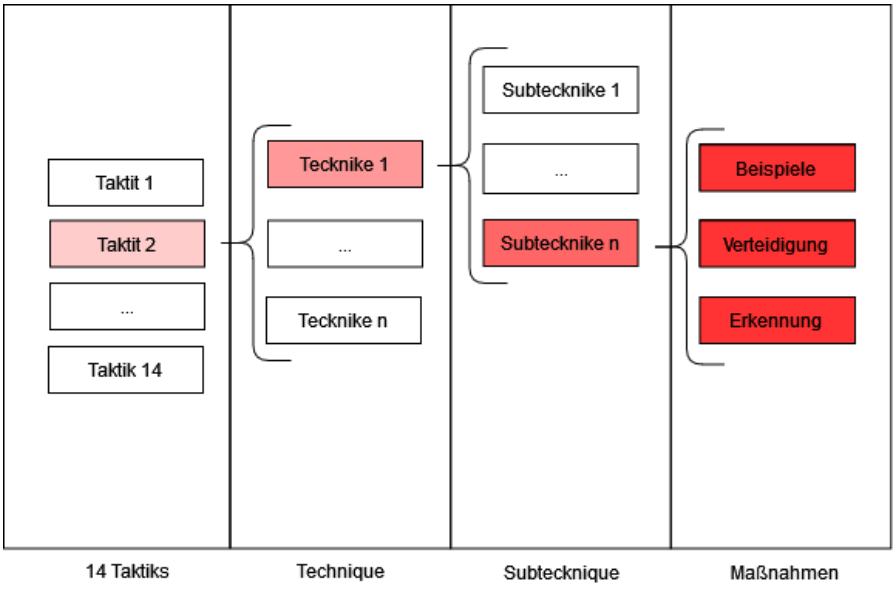


Abbildung 16: Struktur der Mitre ATT&CK Matrix laut MITRE ATT&CK (2018b)

3.2. Auswahl des Angriffes

In dieser Arbeit beschäftigen wir uns mit der Taktik „Zugang zu Anmeldedaten“ und deren Technik Brute-Force Angriffe. Diese Technik ist in vier Untertechniken aufteilt:

In dieser Arbeit beschäftigen wir uns mit der Taktik „Zugang zu Anmeldedaten“ und ihrer Technik „Brute-Force Angriffe“. Diese Technik ist in vier Untertechniken unterteilt:

- Brute-Force Angriffe
- Entschlüsselung von Hashwerte
- *Password Stuffing*
- *Password Spraying*

Da unser Ziel hier ist, Grafana zu verwenden, um Angriffe zu erkennen, haben wir uns für einen einfachen und reproduzierbaren Angriff entschieden, der wenige Ressourcen erfordert. In diesem Fall kann ein Brute-Force Angriffe mit zwei VMs problemlos durchgeführt werden. Für diesen Angriff verwenden wir die Sub-Technik Ërraten von Anmeldedaten und *Password Stuffing*, da sie ähnliche Erkennungsmethoden aufweisen. Da unser Fokus bei dieser wissenschaftlichen Arbeit auf der Angriffserkennung schließen andere Maßnahmen wir hierbei aus.

Die nächste Abbildung, 17, zeigt den Umfang unseres Implementationsversuchs mithifel von Mitre ATT&CK:

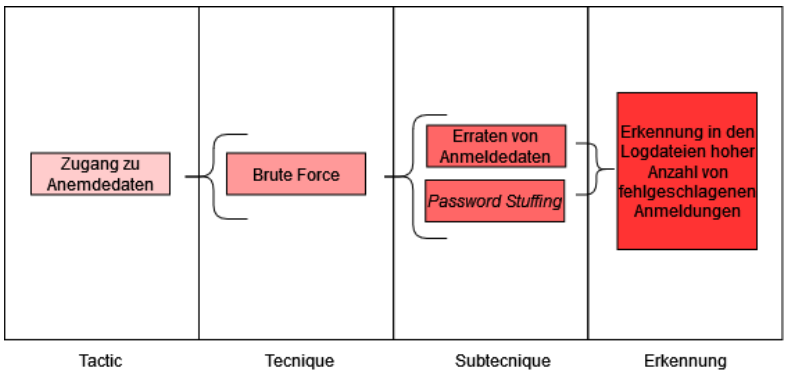


Abbildung 17: Taktiken, Techniken, Prozeduren (TTP) für unseren Angriff laut MITRE ATT&CK (2020)

3.3. Installation und Generierung von Logdateien

In diesem Abschnitt konzentrieren wir uns auf die folgenden Punkte:

1. Einrichtung von VMs für das Opfersystem und den Angreifer
2. Simulation des Angriffe zur Erzeugung von Logdateien
3. Installation und Konfiguration von Grafana Loki und Promtail mit Container
4. Weiterleitung der Logdateien an Grafana

Die Installation und Verwendung können entweder über eine grafische Benutzeroberfläche (GUI) des Betriebssystems oder über die Kommandozeile durchgeführt werden. In dieser Arbeit verwenden wir die Kommandozeile.

3.3.1. Einrichtung der VMs für Opfersystem und Angreifen

Die beiden VMs sind eine „Kali virtuellen Maschine (VM)“ und „Ubuntu Server 22.04.2“ mit standardmäßigen Einstellungen. Beide Maschinen wurden entsprechend ihrer jeweiligen Dokumentation installiert (Kali, 2019) und (Ubuntu, 2023a).

Für das Opfersystem haben wir uns für die Passwörter „qwertz“ und „password“ entschieden. Laut einer Umfrage gehören diese Passwörter zu den zehn am häufigsten verwendeten Passwörtern in Deutschland (silicon.de, 2022). Für die Durchführung des Password Spraying haben wir folgende Benutzername-Passwort Kombinationen erstellt:

Opfersystem 1	Opfersystem 2
admin:123456	bob:hallo
user1:passwort	master:alice
user2:abc123	hans:daniel
user3:qwertyuiop	bruno:super123

3.3.2. Generierung von Logdateien mit der Angrifsssimulation

Für den Angriff verwenden wir folgende Tools:

- Secure Shell Protocol (SSH)
- Hydra

In diesem Szenario sendet Hydra gleichzeitig mehrere Authentifizierungsversuche an das Opfersystem, um eine SSH-Verbindung herzustellen. Das Tool verwendet ein sogenanntes Wörterbuch mit verschiedenen Einträgen, die als Passwörter dienen. Für unseren Test benutzen wir die bekannte Rockyou-Wörterbuch.

Die Abbildung 18 zeigt, wie das Password Stuffing abläuft:

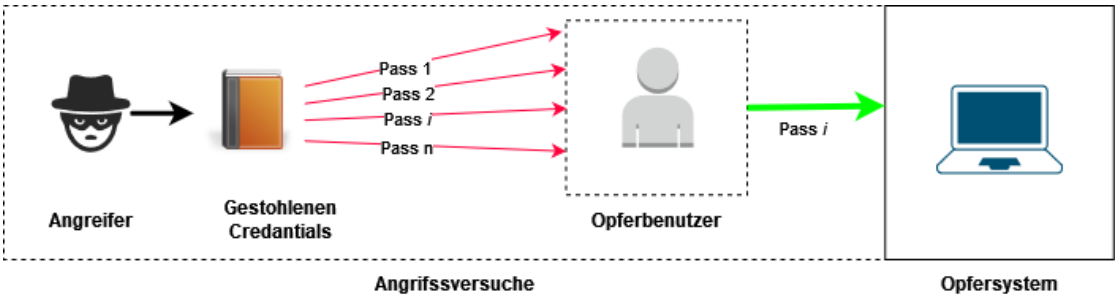


Abbildung 18: Darstellung von *Password Stuffing* nach Ba et al. (2021)

In diesem Angriff versucht der Angreifer sich mit einem Konto anzumelden, indem er mit vielen Passwörter aus dem Wörterbuch probiert, bis eins richtige gefunden ist. Es können mehrere Anmeldungsversuche geschickt werden, bis eine von denen funktioniert.

Password Stuffing wurde mit folgendem Kommando durchgeführt (Kali, 2022a):

```
hydra -l [Benutzername] -P rockyou.txt [Opfersystem] ssh -V -t 4

# Erklärung
-l: Spezifikation des Benutzernamens, den wir angreifen
-P: Auswahl der Datei mit bekannten Passwörtern
ssh: Auswahl der Anwendung, die wir angreifen
-V: Ausführliche Ausgabe über Versuche, Fehler und Erfolg
-t 4: Anzahl von gleichzeitigen Verbindungen
```

Die Abbildungen 19 und 20 zeigen ein Teil der Ausgabe von Hydra während der Ausführung von Password Stuffing gegen das Opfersystem1 und Opfersystem2:

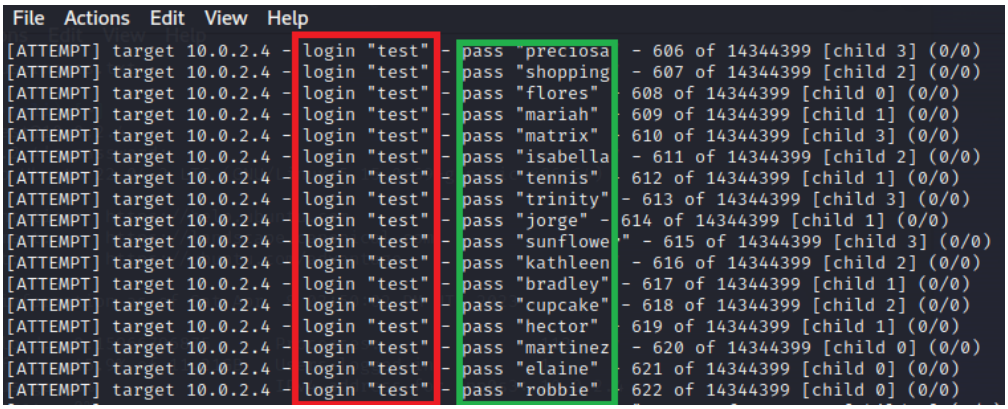


Abbildung 19: Ausgabe von *Password Stuffing* gegen Opfersystem1

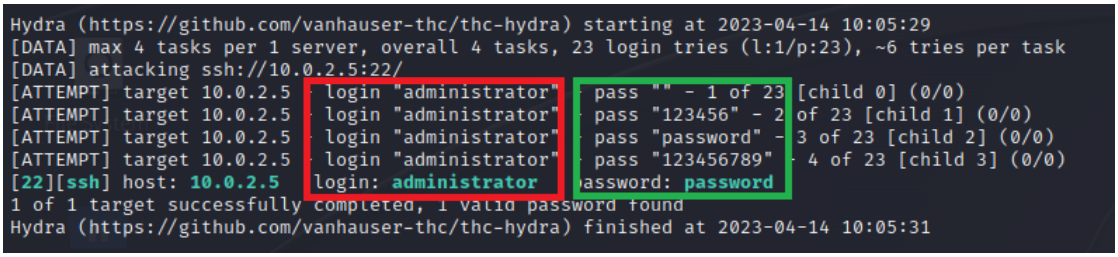


Abbildung 20: Ausgabe von *Password Stuffing* gegen Opfersystem2

Auf den Abbildungen 19 und 20 sehen wir in rot markiert, dass der Angriff den Benutzernamen „test“ im Opfersystem1 und „Administrator“ Opfersystem2 zielt. In grün werden die verschiedenen Passwörter aus Rockyou-Wörterbuch verwendet. Auf der Abbildung 20 wird das gefundene Passwört grün geschrieben.

Unser nächster Angriff, Password Spraying, ist in der Abbildung 21 dargestellt:

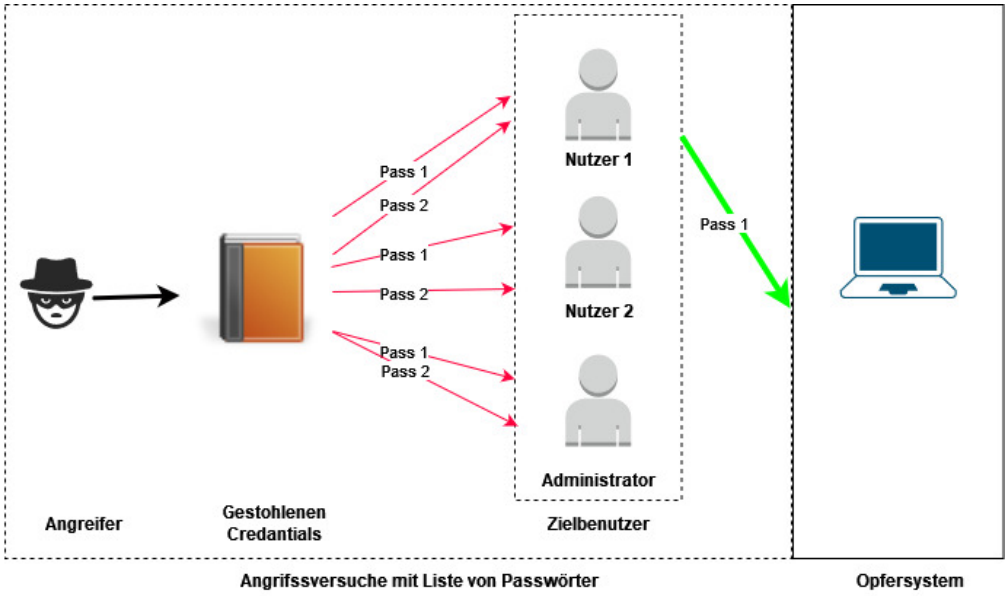


Abbildung 21: Darstellung von *Password Spraying* laut Swathi (2022)

Aus der Abbildung 21 sehen wir, dass bei Password Spraying weniger Passwörter im Vergleich zum Password Stuffing verwendet werden. In diesem Fall werden gegen mögliche viele existierenden Benutzername versucht. Hier will der Angreifer Kontosperrungen vermeiden und gegenüber Sicherheitsmaßnamen Unauffälig bleiben.

Für diesen Angriff benutzen wir folgendes Kommando:

```
hydra -L username2.txt -P passwoerter.txt [Opfersystem2] ssh -V -t 4
-L: Auswahl der Datei mit gefunden Benutzernamen
```

In diesem Fall gehen wir davon aus, dass der Angreifer einige oder alle Benutzernamen bereits kennt. Da bei diesem Angriff weniger Anmeldeversuche pro Nutzer durchgeführt werden, verwenden wir eine selbst erstellte Datei mit weniger Passwörtern als die Rocky-ou-Datei. Unsere Datei enthält die am häufigsten verwendeten Passwörter in Deutschland (silicon.de, 2022).

Die Abbildungen 22 und 23 zeigen die Ausgabe von Password Spraying:

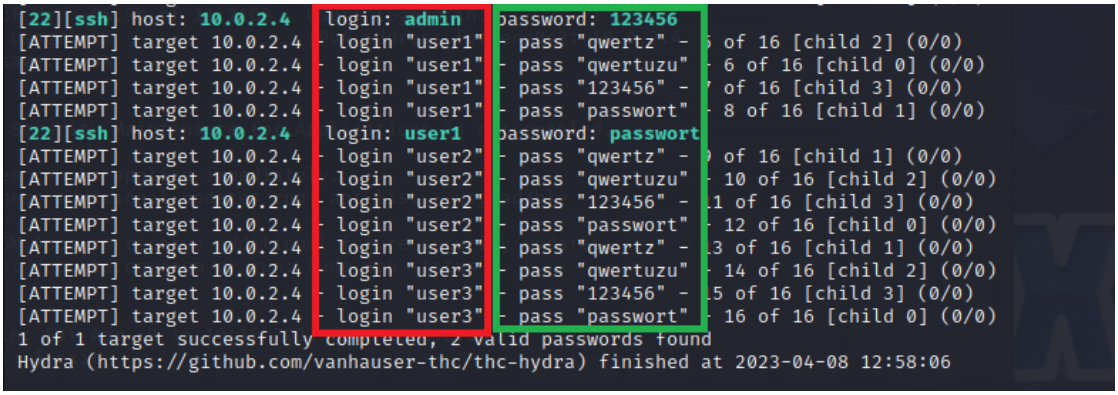


Abbildung 22: Ausgabe von *Password Spraying* in Kali Linux gegen Opfersystem1

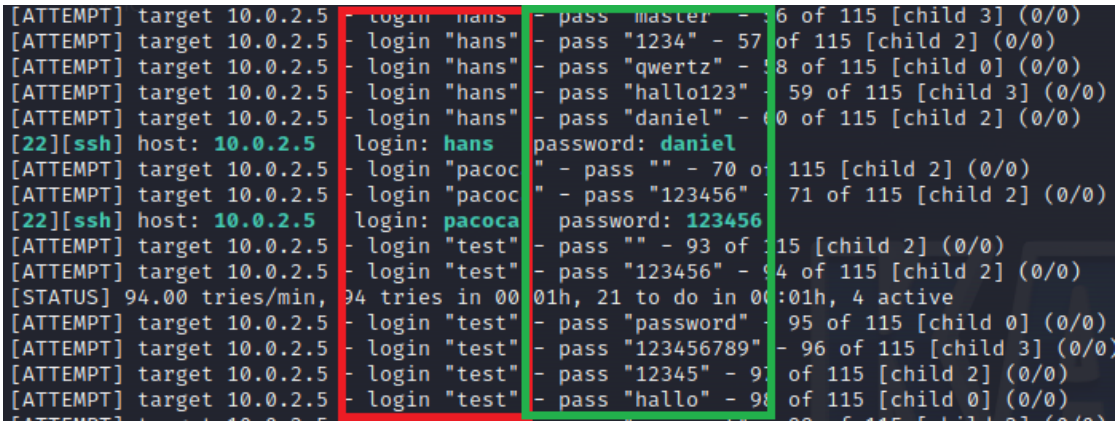


Abbildung 23: Ausgabe von *Password Spraying* in Kali Linux gegen Opfersystem2

Die Abbildungen 22 und 23 zeigen die verschiedenen Benutzername (rot markiert) aber wenige Passwörter pro Nutzer (grün markiert). Auf beiden Abbildungen werden die gefundenen Passwört grün geschrieben.

Nach den Durchführungen dieser Simulationen generierten wir drei Logdateien mit insgesamt 40 KB Größe.

3.3.3. Installation und Einrichtung von Grafana, Loki und Promtail

Für die Einrichtung haben wir sowohl offizielle Dokumentation von Grafana als auch auf externe Quellen zurückgegriffen, um die Einstellungen an unsere Umgebung anzupassen (Polinowski, 2019). In den Anhänge A und B befinden sich die heruntergeladenen originalen Konfigurationsdatei und die an unsere Umgebung angepasst. Diese Dateien benutzten wir für die Installation, Einstellung und Verwendung der Applikationen in Containers.

Für diesen ersten Test wurden die Logdateien des Opfersystems manuell auf den Container übertragen, da wir hier nur eine Instanz von Promtail verwendeten. Für unsere Arbeit nahmen wir die folgenden Versionen von den Anwendungen:

Anwendung	**Angewandte Version**
Grafana	9.5.2
Loki	2.8.2
Promtail	2.8.2

Tabelle 3: Verwendete Versionen der Anwendungen laut Grafana Labs (2023b) und Grafana Labs (2023a)

Nach der Ausführung des Kommandos ist die Anwendung benutzbar, wie in der Abbildung 24 dargestellt:

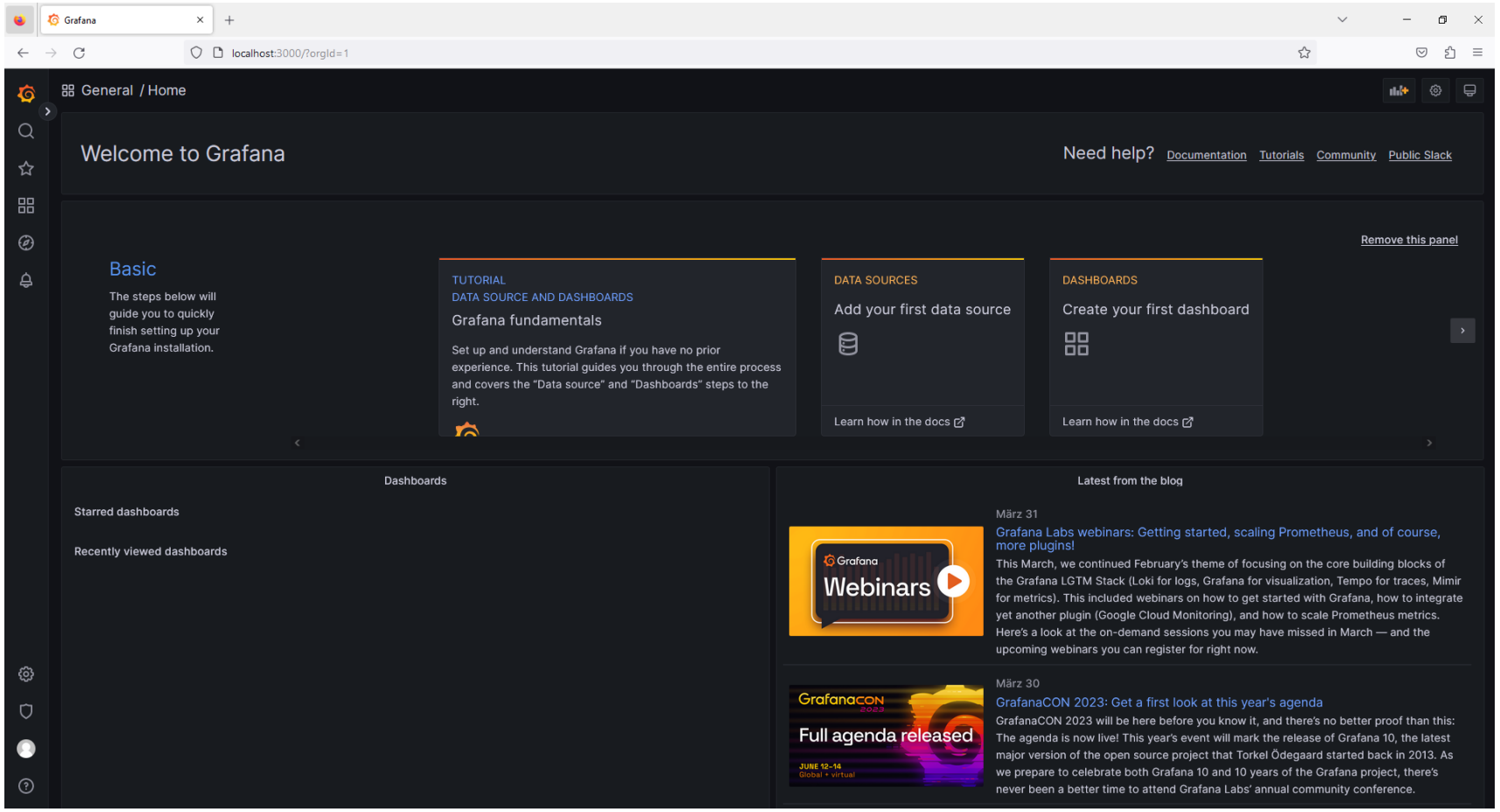


Abbildung 24: Screenshot der Willkommensseite von Grafana Loki

3.3.4. Weiterleitung der Logdateien zu Grafana

Grafana Loki bietet interne und externe Möglichkeiten Logdateien zu empfangen. Die internen beziehen sich auf Grafana Tools, während die externen unabhängige Methode von Grafana benutzen:

1. **interne Methode**:
    a) Promtail
    b) Grafana Agents
2. **externe Methode**
    a) Application Programming Interface (API)
    b) OpenTelemetry

In unserer Arbeit verwenden wir **Promtail**, der in einem Container läuft. Diese Instanz sendet die von uns ausgewählten Logdateien an Grafana und verarbeitet alle Dateien innerhalb eines sogenannten „jobs“. Promtail kann Logdatein nur zu Grafana Loki oder zu anderen Promtail-Instanz schicken (Grafana Labs, 2020e). Die Abbildung 13 auf der Seite 21 zeigt diese beschriebene Struktur.

In einer produktiven Umgebung ist die Installation von **Grafana Agents** auf jedem Endpoint eine andere Lösung, um Grafana Loki mit Logdateien zu füllen. Während Promtail Logdatei nur zu Loki schickt, kann Grafana Agents Logdateien zu Prometheus, OpenTelemetry und zu Tools von *Grafana Ecosystem*, wie Mimir, Tempo, Phlare, Loki und Grafana (Grafana Labs, 2022b).

Die nächste Abbildung, 25, zeigt den Kommunikationfluss zwischen Grafana Agents und die integrieten Tools:

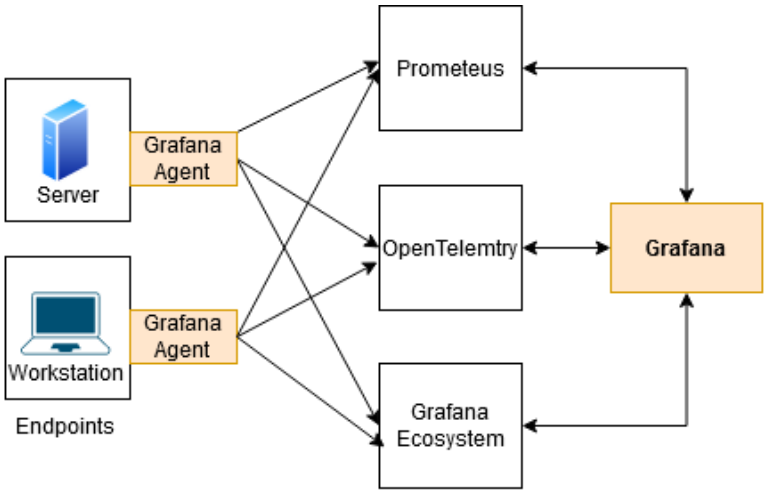


Abbildung 25: Kommunikation zwischen Grafana Agents, Prometheus, OpenTelemetry und *Grafana Ecosystem* laut Grafana Labs (2022b)

Der Kommunikationsfluss bei Grafana Agents funktioniert ähnlich, wie bei Promtail. Die Endpoints (links), wo die Agents installiert sind, schicken die Logdateien zu den kompatiblen Tools (mitte), die sich wiederum mit Grafana (rechts) kommunizieren.

Die Sendung des Inhalts der Logdateien findet auch mithilfe von Grafana Loki HTTP **API** statt. In diesem Fall werden die Zeile der Logdateien und nicht der Datei zum Endpoint von Loki mit HTTP POST-Anfrage geschickt.

Grafana Loki bietet auch eine Integration mit dem Open-Source-Tool OpenTelemetry an, um Logdateien zu empfangen (Grafana Labs, 2022c). Die Integration mit Grafana Loki erfolgt über die Nutzung von APIs. Der *Collector* läuft in derselben Umgebung wie Grafana Loki, damit er die Logdateien empfangen und verarbeiten kann. Die *Agents* laufen auf jedem Endpunkt und kommunizieren mit dem *Collector*.

Die folgende Abbildung stellt das Kommunikationsverfahren zwischen OpenTelemetry und die Tools von *Grafana Ecosystem*:

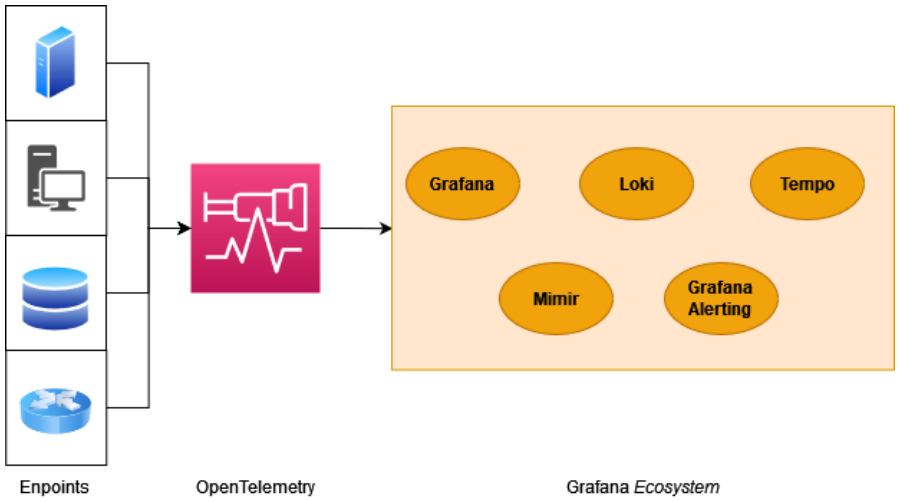


Abbildung 26: Datenfluss zwischen OpenTelemetry und *Grafana Ecosystem* laut Grafana Labs (2021d)

Auf der linken Seite der Abbildung 26 haben wir die verschiedenen Endpoints, auf denen jeweils ein *Collector* läuft. In der Mitte ist der OpenTelemetry Endpoints, der die Datei sammelt und dessen Inhalt verarbeitet. Diese werden schließlich an die Tools von *Grafana Ecosystem* weiterleitet.

3.4. Aufbau der Erkennungsregel für den ausgewählten Angriff

Ein Brute-Force Angriffe lässt sich durch eine hohe Anzahl der fehlgeschlagenen Anmeldeversuche erkennen (Selvaganesh et al., 2022). Wir betrachten eine Situation, in der keine Gegenmaßnahmen wie Kontosperre nach n beliebigen Versuchen oder MFA, implementiert sind. Das folgende Abbildung, 27, stellt Diagramm mit einem allgemeinen Ablauf eines Anmeldungsverfahrens dar:

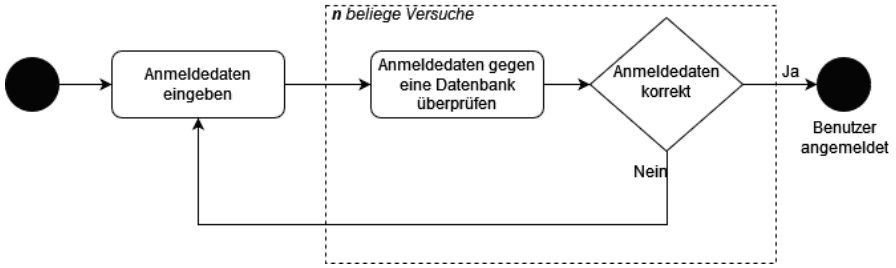


Abbildung 27: Allgemeiner Ablauf eines Anmeldungsverfahrens laut Selvaganesh et al. (2022)

Grafana bietet ein Konfigurationsmuster für die Eingabe und Darstellung von SSH Eventds an. In dieser Konfiguration sind bereits Regesätze für die Verarbeirtung der Log-Einträge in Loki und Quellcode für die Generierung von Grafik in Grafana. Diese Konfigurationsdatei ermöglicht eine umfassende Analyse dieser Daten (VoidQuark, 2022). Die gesentet Logdateien werden mithilfe der folgenden Elemente gelesen und verarbeitet:

Element	Beschreibung
JavaScript Object Notation (JSON)	Lesbare Dateiformat, deren Daten nach dem Regel Schlüssel-Wert-Paar gespeichert sind
Muster	Lesen und Extraktion der Information der Logdateien
Reguläre Ausdrücke (RegExp)	Mustererkennung aus der Logdatei
Logfmt	Extraktion von Schlüssel:Wert Paar der Logdateien

Tabelle 4: Elementen eines Regelsätzes in Grafana Loki laut VoidQuark (2022) und (Setter, 2015)

Für jedes Angriffszenario benutzen wir spezifische Regeln, die mit LogQL aufgebaut sind. Die Filterung findet mithilfe von zweil Labels „instance“ und „job“ statt. In Promtail wird jeder Endpoint als „Instance“ bezeichnet. Eine oder mehrere „instances“ werden einem „job“ zugewiesen. „jobs“ beziehen sich auf die Bearbeitung der Logdateien nach dem spezifizieren Regeln, in unserem Fall, Überprüfung von SSH-Logdateien. Diese Struktur stammt aus dem Tool Prometheus. Alle unsere „instance“ werden in einem „job“ eingepackt, wo sie nach den gleichen Regeln verarbeitet. Zusätliche Labels können auch definiert werde (Prometheus, 2015). Das folgende Diagramm, 28, stellt die Beziehung zwischen dieser beiden Labels dar:

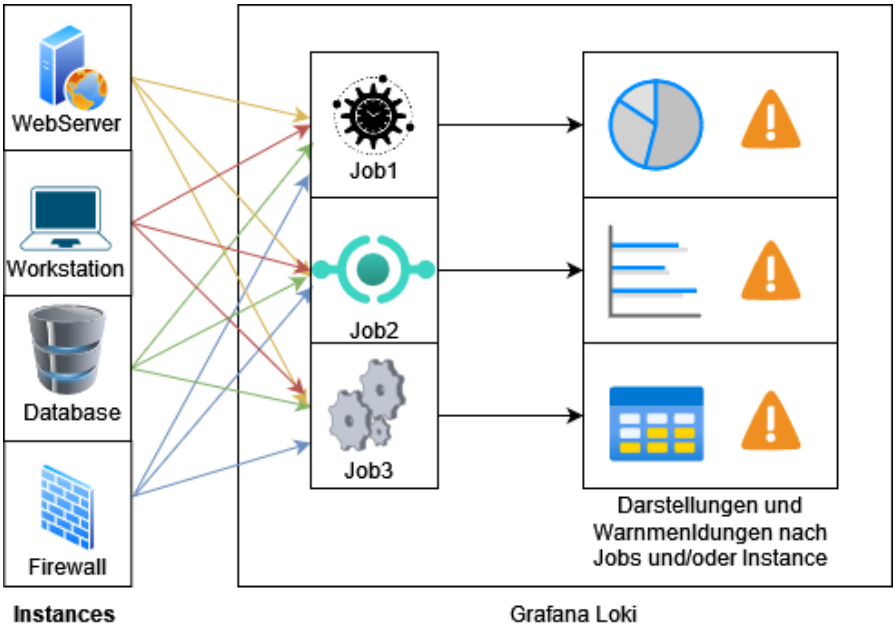


Abbildung 28: Beziehung zwischen „instance“ und „job“

Der Inhalt Logdateien kann dann in Grafana nach dem definierten Labels aufgerufen werden, wie auf der folgenden Abbildung 29 dargestellt:

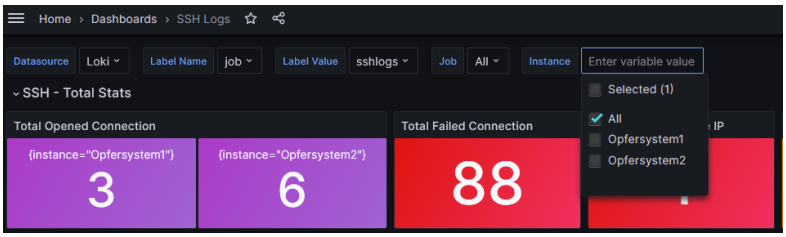


Abbildung 29: Aufrufe des Inhalts der Logdateien nach bestimmten Labels

Mit LogQL können auch Filterung verwendet, um nach bestimmten „instance“ und/oder „jobs“ die Daten aufzurufen, wie auf der Abbildung 30 dargestellt:

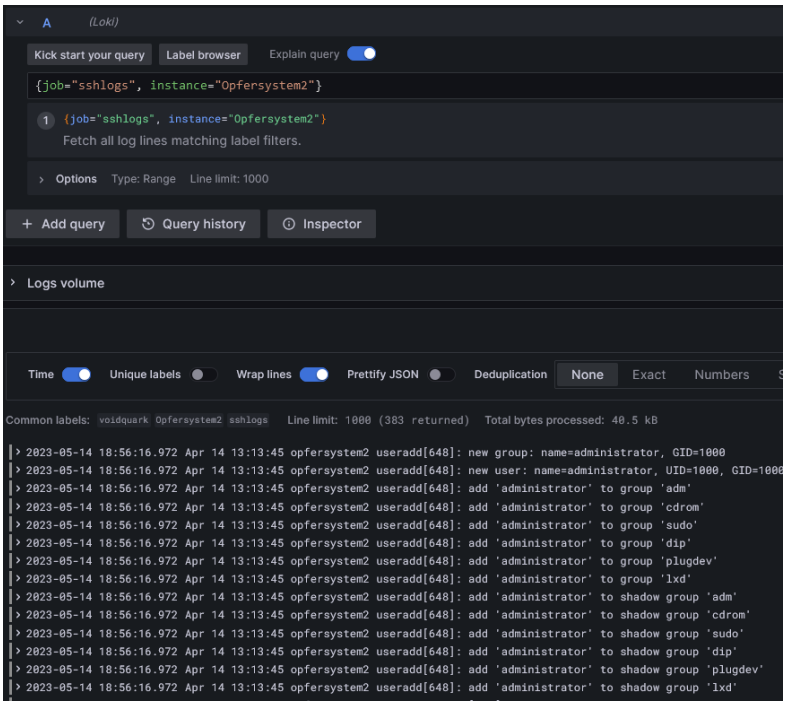


Abbildung 30: Aufrufe des Inhalts der Logdateien LogQL

In dem nächsten Abschnitt beschreiben wir, wie diese Regel in LogQL geschrieben werden.

3.4.1. Regelsätze in LogQL

In diesem Abschnitt fassen wir zusammen, wie eine Abfrage in LogQL für eine Logdatei mit SSH Einträgen aussieht. Für ausführliche Informationen über den Aufbau der Abfrage verweisen wir die offizielle Dokumentation, auf die diese Erklärung basiert ist (Grafana Labs, 2021c). Unsere Logdatei enthält unter anderem folgende Zeile:

```
14 14:05:30 opfersystem2 sshd[1698]: Failed password for administrator
from 10.0.2.15 port 58036 ssh2
```

Um fehlgeschlagene Anmeldeversuche zu erkennen, extrahieren wir folgende Felder aus den SSH-Logdateien. Diese Information verwenden wir um gleiche Events zu erkennen und deren Anzhal festzustellen.

```
14 14:05:30 opfersystem2 sshd[1698]: Failed password for administrator
from 10.0.2.15 port 58036 ssh2
```

Wir teilen die Abfrage unten mit, um ihre Bestandteile besser zu verstehen:

LogQL-Codeschnipsel	**Beschreibung**
sum by(add) (rate({job="*JOBNAME*" instance= "$instance"}	Hiermit wird die Aufsummierung der Benutzernamen definiert, die wir mit „Patterns“ in LogQL definiert haben. „Patterns“ ermöglichen die einfache Extrahierung von Informationen aus einer Zeile. Wir holen alle Log-Einträge, die sich auf den von uns definierten Job beziehen. Wir können auch nach spezifischen Endpoint filtern, indem wir das Schlüsselwort „instance“ benutzen.
|	„|“ funktioniert in LogQL wie eine Pipeline für die Verkettung von mehreren Suchmustern.
|= ‘**sshd**[’ |= ‘: **Failed**’	Suche nach Zeilen mit den in den rot markierten Einträgen.

Element	Beschreibung
!~ 'invalid user' !~ 'Legitimer_Nutzer' !~ 'Legitime_Adresse'	Suche nach Zeilen **ohne** diese Einträge. Wir können beispielsweise Einträge ausschließen, die auf legitimen Nutzer oder IP-Adresse beziehen, um falsche Positive zu vermeiden
| pattern '<_> for <Benutzername> from <Quelladresse> port <_>' [$ __range]))	Die Definition der Wörter „Benutzername", „Quelladresse" und als „Patterns" dienen dazu, einen Benutzernamen und eine Quelle IP-Adresse aus der Logdatei zu extrahieren. Die Platzhalter „<_>" sind unbenannte Elemente, die in diesem Fall auf die Einträge „password" und Portnummer in der Zeile verweisen.

Tabelle 5: Elementen eines Regelsätzes in Grafana Loki laut VoidQuark (2022) und Setter (2015)

Schließlich sieht der Regelsatz so aus:

```
sum by(add) (rate(job="JOBNAME", instance=~"$instance"
|= 'sshd['
|= ':Failed' !~ 'invalid user' !~ 'Legitimer_Nutzer' !~ 'Legitime_Adresse'
| pattern '<_> for <Benutzername> from <Quelladresse> port <_>' [$__range]))
```

Eine allgemeine Erkennungsregel in LogQL würde so aussehen:

```
Ergebnis = Operation (GesuchterWert) (Operation(label1="LabelWert",
label2="Label2Wert") |= 'Gesuchte Inhalt im Logdatei' | !'Inhalt im
Logdatei ausschliessen' | Regulärer Ausdrucke | pattern '<_>
WortImLogDatei <GesuchterInhalt> WortImLogDatei <_>' [\$__range]))
```

3.5. Hinzufügen der Regelsätze Grafana Loki

Die Regelsätze in Grafana Loki können sowohl **manuell** im Menü „Code“ als auch über die **GUI** im Menü „Builder“ geschrieben werden. Letzteres bietet eine benutzerfreundlichere Umgebung, um die Regeln zu schreiben. Die folgenden Abbildungen, 31 und 32, zeigen diese beiden Optionen:

```
sum by (username) (count_over_time({job=~"varlogs", job=~".*",
instance=~".*"} |="sshd[" |~": Invalid|: Connection closed by
authenticating user|: Failed .* user" | pattern `<_> user <username> <_>
port` | __error__="" [2m]))
```

Abbildung 31: Feld in Grafana Loki für manuelle die Eingabe des LogQL-Codes

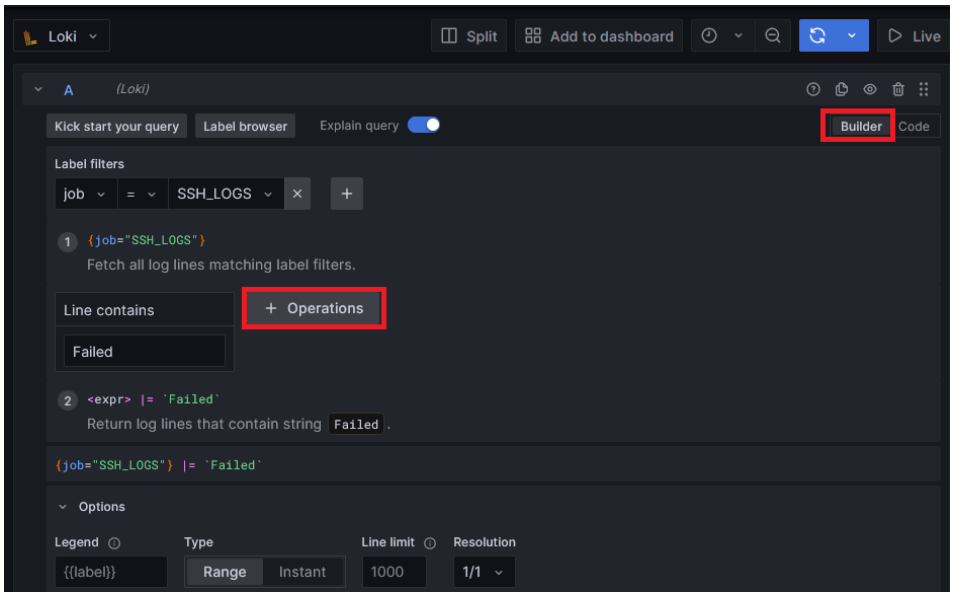


Abbildung 32: „Builder“ in Grafana Loki für nutzerfreundlichere Eingabe des LogQL-Codes

Beide Optionen bieten die Möglichkeit, eine Erklärung zur Abfrage anzuzeigen, wie auf der Abbildung 33 gezeigt wird:

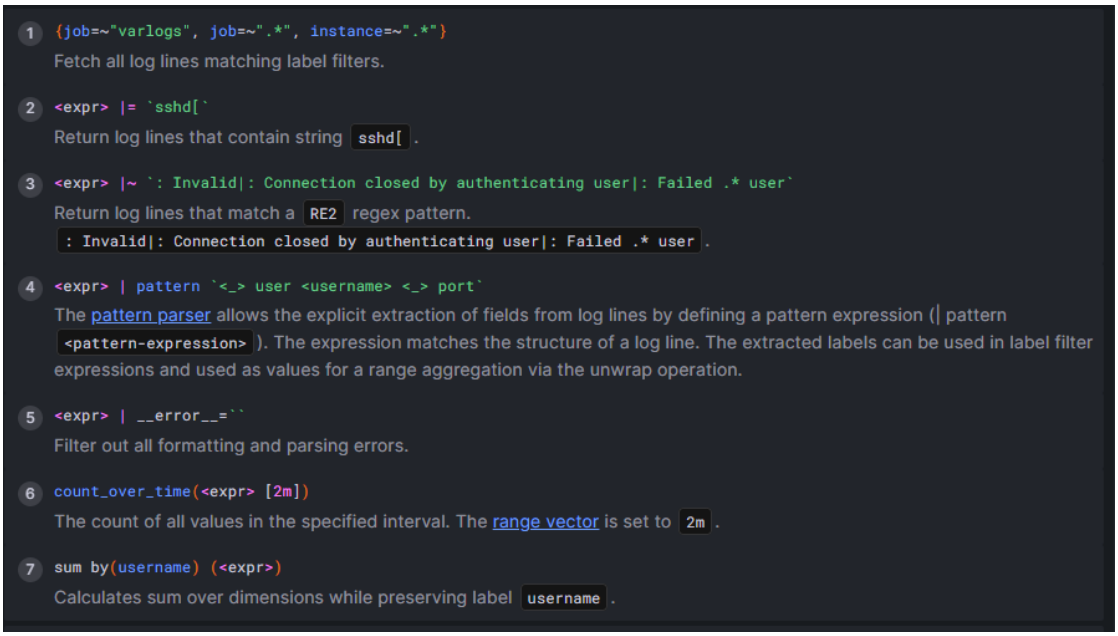


Abbildung 33: Ausführliche Information über die Abfrage

Mit der Nutzung von *API* Endpoint von Loki ist es möglich nach dem Inhalt der Logdateien abzufragen, indem die Regelsätze in LogQL geschrieben werden. In diesem Fall bekommen wir die gefilterte Ergebnis als Antwort (Grafana Labs, 2020c).

```
# Muster für die Anfrage
curl -G -s  "http://LokiInstance/Endpoint" --data-urlencode 'Logql Abfrage'
| jq

# Beispiel
curl -G -s  "http://LokiInstance/loki/api/v1/query" --data-urlencode
'sum by(add) (rate({job="JOBNAME", instance=~"$instance"} |= 'sshd[' |= ':
Failed' !~ 'invalid user' !~ 'Legitimer_Nutzer' !~ 'Legitime_Adresse' |
pattern '<_> for <Benutzername> from  <Quelladresse> port <_>' [$__range]))'
| jq
```

Nachdem die SSH-Logdateien gelesen und verarbeitet wurden, bekommen wir von Grafana Loki die zusammenfassende Ergebnissen, wie unter auf der Abbildung 35 dargestellt:

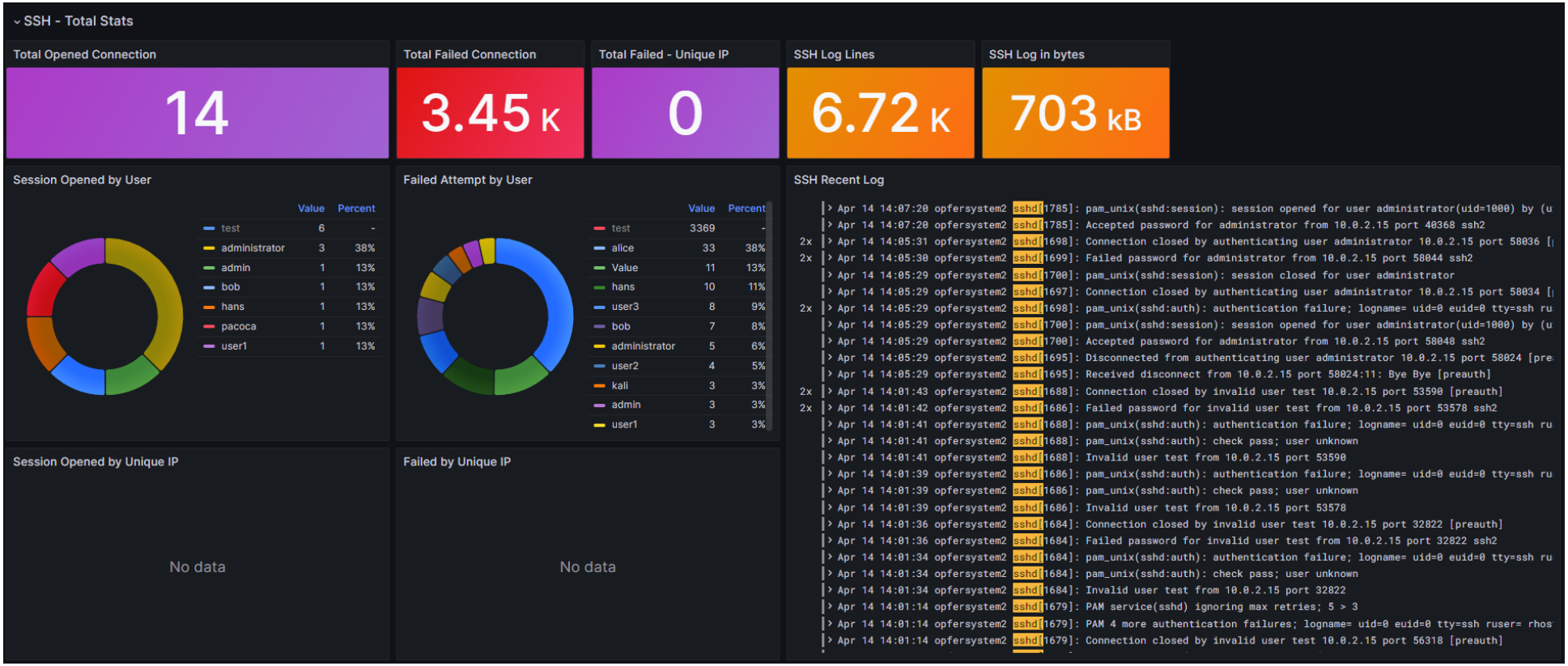
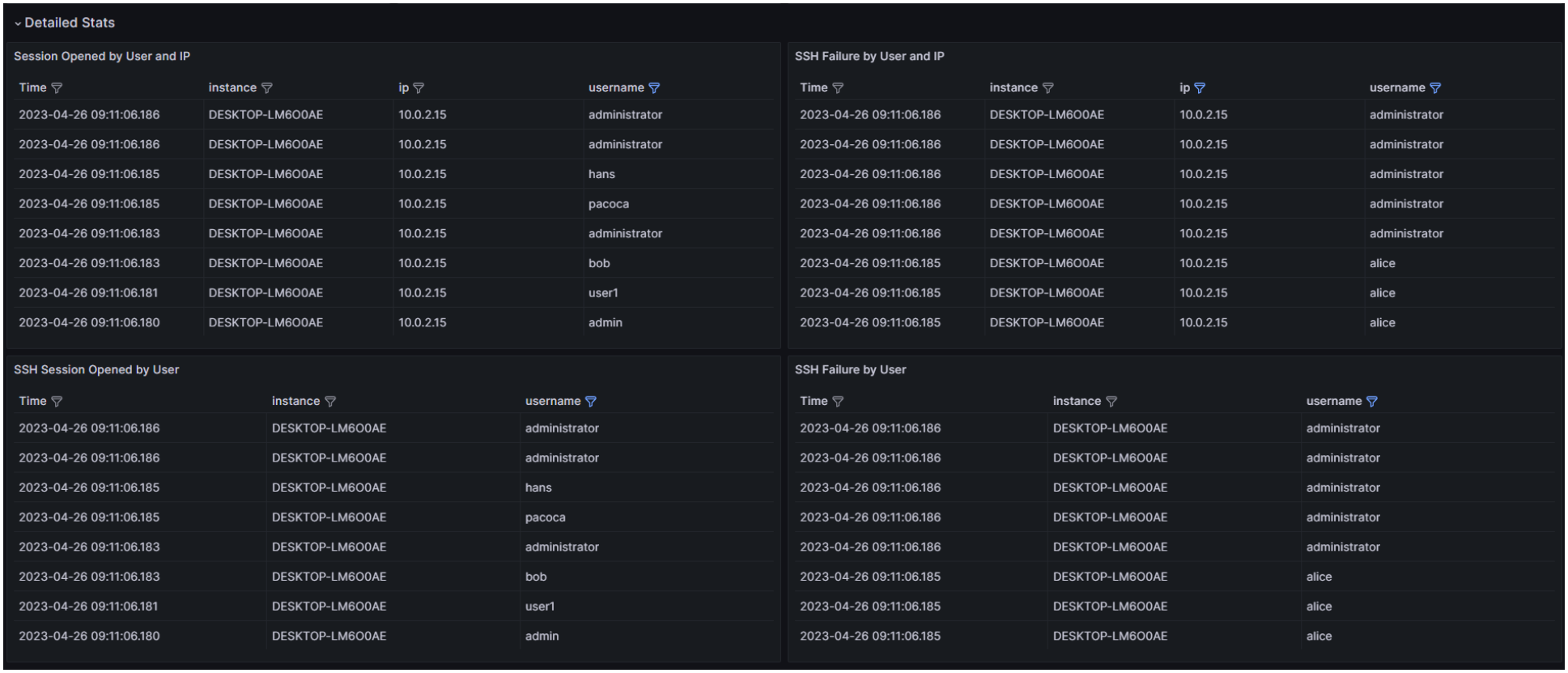


Abbildung 34: Ausgabe der Verarbeitung der SSH Logdateien von Grafana Loki

Das nächste Abbidung, 35, gibt ausführliche Informationen der Logdateien:



Detailed Stats

Session Opened by User and IP

Time	instance	ip	username
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	10.0.2.15	administrator
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	10.0.2.15	administrator
2023-04-26 09:11:06.185	DESKTOP-LM6O0AE	10.0.2.15	hans
2023-04-26 09:11:06.185	DESKTOP-LM6O0AE	10.0.2.15	pacoca
2023-04-26 09:11:06.183	DESKTOP-LM6O0AE	10.0.2.15	administrator
2023-04-26 09:11:06.183	DESKTOP-LM6O0AE	10.0.2.15	bob
2023-04-26 09:11:06.181	DESKTOP-LM6O0AE	10.0.2.15	user1
2023-04-26 09:11:06.180	DESKTOP-LM6O0AE	10.0.2.15	admin

SSH Failure by User and IP

Time	instance	ip	username
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	10.0.2.15	administrator
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	10.0.2.15	administrator
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	10.0.2.15	administrator
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	10.0.2.15	administrator
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	10.0.2.15	administrator
2023-04-26 09:11:06.185	DESKTOP-LM6O0AE	10.0.2.15	alice
2023-04-26 09:11:06.185	DESKTOP-LM6O0AE	10.0.2.15	alice
2023-04-26 09:11:06.185	DESKTOP-LM6O0AE	10.0.2.15	alice

SSH Session Opened by User

Time	instance	username
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	administrator
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	administrator
2023-04-26 09:11:06.185	DESKTOP-LM6O0AE	hans
2023-04-26 09:11:06.185	DESKTOP-LM6O0AE	pacoca
2023-04-26 09:11:06.183	DESKTOP-LM6O0AE	administrator
2023-04-26 09:11:06.183	DESKTOP-LM6O0AE	bob
2023-04-26 09:11:06.181	DESKTOP-LM6O0AE	user1
2023-04-26 09:11:06.180	DESKTOP-LM6O0AE	admin

SSH Failure by User

Time	instance	username
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	administrator
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	administrator
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	administrator
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	administrator
2023-04-26 09:11:06.186	DESKTOP-LM6O0AE	administrator
2023-04-26 09:11:06.185	DESKTOP-LM6O0AE	alice
2023-04-26 09:11:06.185	DESKTOP-LM6O0AE	alice
2023-04-26 09:11:06.185	DESKTOP-LM6O0AE	alice

Abbildung 35: Ausführliche Darstellung der SSH Logdateien von Grafana Loki

3.6. Einrichtung der Warnmeldungen in Grafana

In den vorherigen Teilen dieser Arbeit haben wir uns damit auseinandergesetzt, Grafana so einzurichten, dass wir schließlich eine Lösung ähnlich einer SIEM erhalten. Von unseren ursprünglichen Ziele haben wir bereits Folgendes erreicht:

1. Sammlung der Logdateien von den Endpoints mit Promtail
2. Anpassung der Logdateien für die Weiterleitung an Grafana Loki
3. Nutzung von Regelsätzen in Loki für die Analysierung der SSH Logdateien
4. Graphische Darstellung der Ergebnissen in Grafana mit den in Loki verwendeten Regelsätzen

Unser letztes Ziel besteht darin, Warnmeldungen für potenzielle Angriffe mithilfe der Ergebnisse von Loki zu generieren. Grafana kann sowohl intern mit der Funktionalität „Alerting“ als auch extern mit Plugins, wie **Alertmanager**, Warnmeldungen generieren. Der zweite kann Daten von Prometheus, Cortex und Mimir als Datenquelle verwenden (Grafana Labs, 2021a) und kann Daten von beliebigen Endpoints empfangen. Die Regelsätze des Alertmanagers haben folgendes Muster:

```
# Warnmeldungen können in beliebigen Gruppen kategorisiert werden. Diese
können von den Nutzern entsprechend ihrer Anforderungen und Bedürfnisse
definiert werden.
groups:
      # Ab diesem Punkt beginnen wir mit der Definition der Regelsätze
      für die Erkennung von Warnmeldungen. Diese umfassen:
   - name: example
     rules:
    - alert: HighRequestLatency

      # LogQL-Regelsätze für die Erkennung der Warnmeldung, welche die
      in den vorherigen Schritten definierten Abfragen verwenden.
      expr: job:request_latency_seconds:mean5m{job="SSH_LOGS"} > 0.5
      for: 10m
       labels:
         severity: page
       annotations:
         summary: High request latency
```

Grafana hat auch ein eigenes internes Tool, um Warnmeldungen zu konfigurieren: **Alerting**. In dieser Arbeit versuchen wir unser Warnmeldungs-System mithilfe dieses Tools aufzubauen.

Die Warnmeldungen können direkt in der GUI von Grafana konfiguriert werden. Dazu folgt man den folgenden Schritten (Grafana Labs, 2019):

1. Name der Regel
2. Regelsätze in LogQL
3. Definition von Gruppen für jede Art von Warnmeldung. Gruppen können später verschiedenen Einstellungen zugewiesen werden, wie z.B. Benachrichtigungen und Inhalte.
4. Informationen über die Warnmeldung, wie eine eindeutige ID und eine Beschreibung. Der Nutzer kann diese Felder so definieren, wie es notwendig ist.
5. Benachrichtigung der Zielgruppe, die diesen Fall später bearbeiten wird.
6. Labels zur besseren Organisation der Warnmeldungen.
7. Konfiguration von E-Mail in Grafana für die Weiterleitung der Warnmeldungen.

Für unseren ersten Test erstellen wir Warnmeldungen über die **GUI** von Grafana für fehlgeschlagene Anmeldeversuche. Wir haben die oben genannten Elemente definiert und die folgenden Regelsätze verwendet (VoidQuark, 2022):

```
# (A) Anzahl von fehlgeschlagenen Anmeldeversuche für existierenden
Benutzernamen:
sum by (username) (count_over_time({job=~"sshlogs"}
|="sshd["
|~": Invalid
|:Connection closed by authenticating user
|: Failed .* user"
|pattern '<_> user <username> <_> port'
| __error__=""[$__interval]))

# (B) Anzahl von Fehlgeschlagenen Anmeldeversuche für nicht
existierenden Benutzernamen:
sum by (username) (count_over_time({job=~"sshlogs"}
|="sshd["
|=": Failed" !~"invalid user"
| pattern '<_> for <username> from <_> port'
| __error__=""[$__interval]))

# Wenn die Anzahl von (A) oder von (B) größer als fünf ist, dann wird
die Warnmeldung als E-Mail an dem Ziel geschickt.
```

Im Anhang C befindet sich die Konfigurationsdatei für unsere Warnmeldung.

Nachdem alles korrekt konfiguriert wurde, haben wir die folgende E-Mail erhalten:

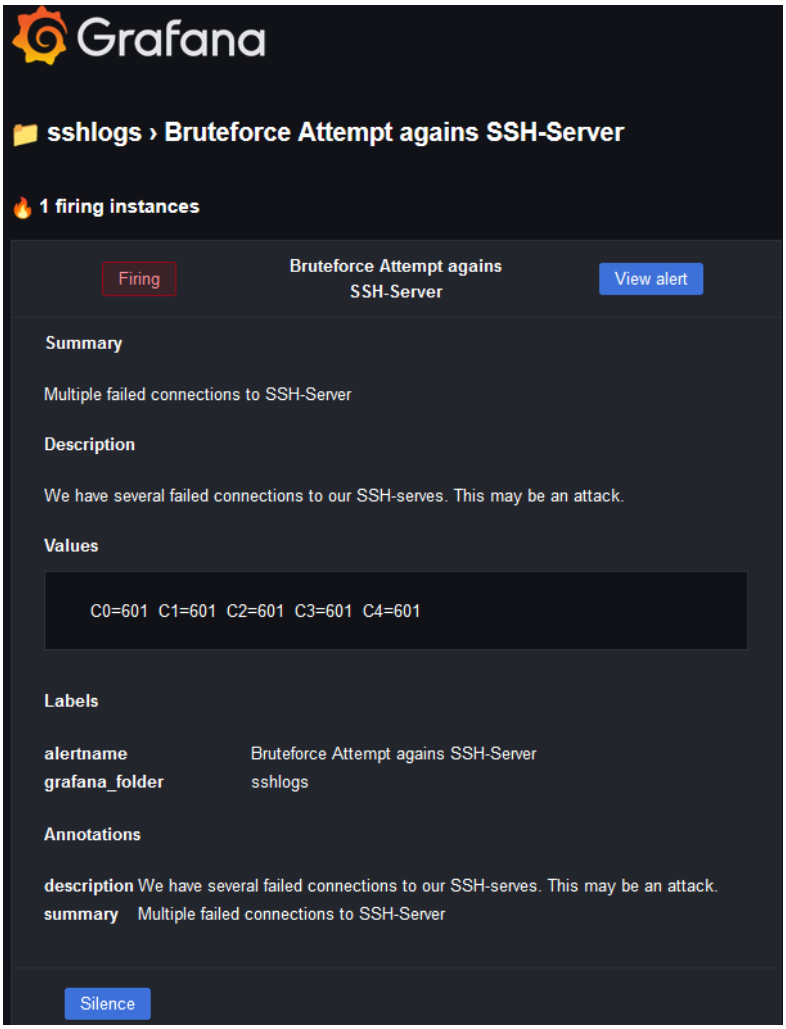


Abbildung 36: E-Mail Warnmeldung von Grafana

4. Evaluation der Implementation mit echten Logdateien

In diesem Abschnitt präsentieren wir unsere Implementierung für die Analyse von SSH-Logdateien der Hochschule. Hier hat unsere Logdateien ungefähr vier Megabyte

4.1. Einstellungen von Promtail und Loki

Die Extrahierung des Inhalts der Logdateien erfolgt im Promtail mit folgenden Konfigurationen aus der offiziellen Dokumentation (Grafana Labs, 2020f):

Konfigurationsfeld	**Beschreibung**
scrape_configs	Steht für die Funktionalität von Promtail, automatisch nach Logdateien zu suchen.
- job_name: sshlogs	Definition des Names unseres „job“
decompression enabled: true initial_sleep: 10s format: gz	Promtail kann verschiedene Komprimierungsformate verarbeiten, darunter auch .gz, welches wir in unserer Arbeit verwenden. Das Feld *initial_sleep* beschreibt das Intervall, bevor die Dekomprimierung beginnt. Dieses Feld kann nützlich sein, wenn komprimierte Dateien vorhanden sind, deren Komprimierungsvorgang jedoch noch nicht abgeschlossen ist. Das Feld *format* gibt das Komprimierungsformat an (Grafana Labs, 2020e).
static_configs: - targets: - loki labels: job: sshlogs instance: Endpoint-Name __path__: /var/log/**.gz	Das Feld *targets* bezieht sich auf die Kommunikation mit der Loki-Instanz. Das Feld *labels* zeigt an, unter welcher Bezeichnung der Inhalt dieser Datei in Loki aufgerufen werden kann. *__path__* gibt den Pfad zur Datei im System an.
pipeline_stages: - match: selector: '{job="sshlogs"}' action: keep	Hier können wir den Inhalt der Logzeile definieren, bevor wir es zu Loki schicken. Nur Logzeilen mit diesem „label“ werden modifiziert und dessen Inhalt wird beibehalten. Alternativ gibt es „drop“, um diesen Inhalt zu löschen.

Konfigurationsfeld	Beschreibung
stages: - regex: (Reguläre Ausdrücke (RegExp) am Ende dieser Tabelle) - timestamp: source: time format: “Jan _2 15:04:05” location: „Europe/Berlin“	Promtail bietet verschiedene Typen von „stages“ zur Bearbeitung von Logzeilen an. Diese „stages“ werden nacheinander verarbeitet. In unserem Fall verwenden wir die „stages“ RegExp, „labels“ und „Timestamp“. Die erste „stage“, RegExp, liest den Zeitstempel und die IP-Adresse aus der Logzeile. Sie ermöglicht es uns, bestimmte Muster in den Logzeilen zu erkennen und die relevanten Informationen zu extrahieren. Die zweite „stage“, „labels“, nutzt die zuvor gefundene IP-Adresse aus der ersten „stages“ und erstellt ein neues Label. Dadurch können wir die Logzeilen basierend auf der IP-Adresse weiter kategorisieren und filtern. Die letzte „stage“, „Timestamp“, nimmt den Zeitstempel aus der Logzeile und speichert ihn in Loki. Dies sorgt dafür, dass das korrekte Datum der Logzeile in Grafana Loki angezeigt wird, anstatt das Datum des Hochladens in Grafana Loki. Durch die Verwendung dieser „stages“ ermöglicht uns Promtail eine flexible und effiziente Bearbeitung der Logzeilen, um sie besser zu analysieren und visualisieren zu können

Tabelle 6: Konfigurationsausschnitt von Promtail

```
’^(?P<time>[A-Za-z]{3}\s{1,2}\d{1,2}\s\d{2}:\d{2}:\d{2}).*from.(?P<source
IP>(?:25[0-5]|(?:2[0-4]|1\d|[1-9]|)\d)\.(?:25[0-5]|(?:2[0-4]|1\d|[1-9]|)\
d)\.(?:25[0-5]|(?:2[0-4]|1\d|[1-9]|)\d)\.(?:25[0-5]|(?:2[0-4]|1\d|[1-9]|)
\d))’
```

Unsere gesamte Einstellung für Promtail befindet sich im Anhang B auf der Seite 76.

In der nächsten Tabelle, 7, zeigen wir einen Konfigurationsausschnitt von Loki, die wir anpassen mussten, damit unsere Logdateien verarbeitet werden könnten. Diese Konfiguration wurde mithilfe der offizielen Dokumentation (Grafana Labs, 2020d) und des offizielen Forumsbeitrags von Grafana Loki (itsnotv, 2022) gestaltet.

Konfigurationsfeld	Beschreibung
query_range: parallelise_shardable_queries: true	Bezieht sich auf Abfrage und Ergebnis von Inhalt des Logdateien in spezifischen Zeitspanne. Der Abfrage-Prozess findet parallel statt.
frontend: max_outstanding_per_tenant: 10000	Dieser Block bezieht sich auf Abfrage in Frontend-Ebene. Anzahl von erlaubten ausstehenden Abfrage. Um Leistung zu gewinnen, sagten wir, dass ein einzelner Nutzer, diese Anzahl von ausstehenden Anfrage hat. In einer produktiven Umgebung ist dieser Wert von der Rechenkapazität abhängig.
querier: max_concurrent: 2048	Festelegung der Verarbeitung von Abfrage Anzahl der gleichzeitigen Abfragen, die verarbeitet werden.
limits_config: reject_old_samples: false split_queries_by_interval: 15m max_query_parallelism: 32	Festlegung der Aufnahmerate und Nutzung von Ressourcen. Ermöglicht die Aufnahme von alten Logdateien, was in unserem Fall notwendig ist, da unsere Datei von April 2022 ist. Trennung von Abfragen nach einem definierten Intervall. Jeder Intervall wird gleichzeitig ausgeführt Maximale Anzahl von parallelen Abfragen.

Tabelle 7: Konfigurationsausschnitt von Loki

4.2. Generierung von Graphiken

Nachdem die Konfiguration fertig ist, können wir die Containers starten und mithilfe des Musters von VoidQuark (2022) Grafiken mit Informationen über SSH-Verbindungen generieren. Mit der Frontend-Anwendung Grafana können wir Daten in verschiedenen Zeitspannen anzeigen und nach „labels“ filtern.

Für unsere Graphiken wollen wir die Anzahl von fehlgeschlagenen Anmeldeversuchen pro Benutzername ableiten. Dafür benutzen wir folgende Abfrage:

```
sum by (username) (count_over_time({ job=~"sshlogs"}
|="sshd["
|~": Invalid
|: Connection closed by authenticating user
|: Failed .* user"
| pattern `<_> user <username> <_> port`
| __error__="" [$__interval]))

sum by (username) (count_over_time({job=~"sshlogs"}
|="sshd["
|=": Failed" !~"invalid user"
| pattern `<_> for <username> from <_> port`
| __error__="" [$__interval]))
```

Die generierten Graphiken sind auf den Abbildungen 37 und 38 dargestellt:

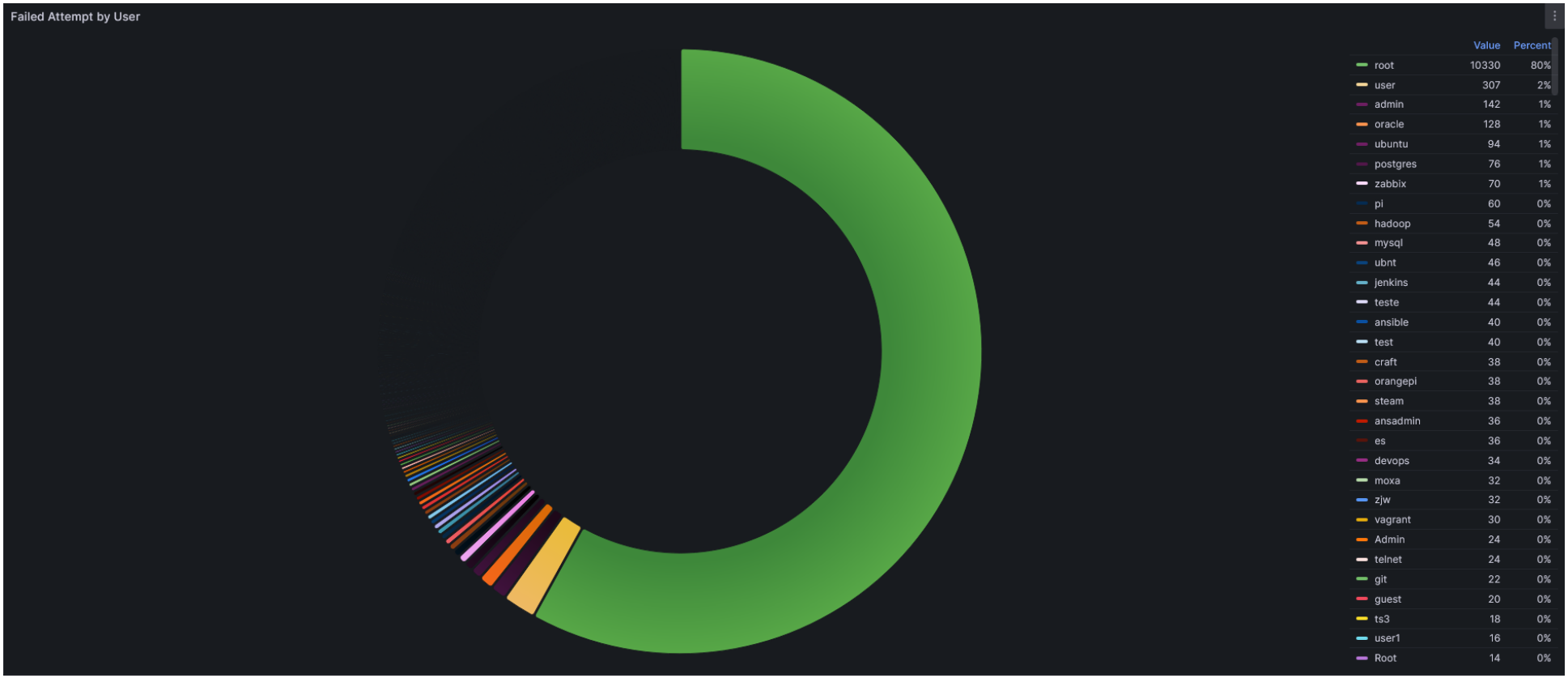


Abbildung 37: Kuchendiagramm von Anzahl fehlgeschlagenen Anmeldeversuche pro Benutzername

Aus der Abbildung 37 können wir ableiten, dass am „22.5.2022“ möglicherweise ein Brute-Force Angriffe stattgefunden hat. Dabei wurden gängige Benutzernamen verwendet, um Anmeldeversuche durchzuführen. Zum Beispiel gab es 10.330 Versuche mit dem Benutzernamen „root“, 307 Versuche mit „user“ und 142 Versuche mit „admin“.

Wir können weitere Filter anwenden, um beispielsweise zu zeigen, zu welchen Uhrzeiten die meisten Versuche stattgefunden haben, wie auf der Graphik 38:

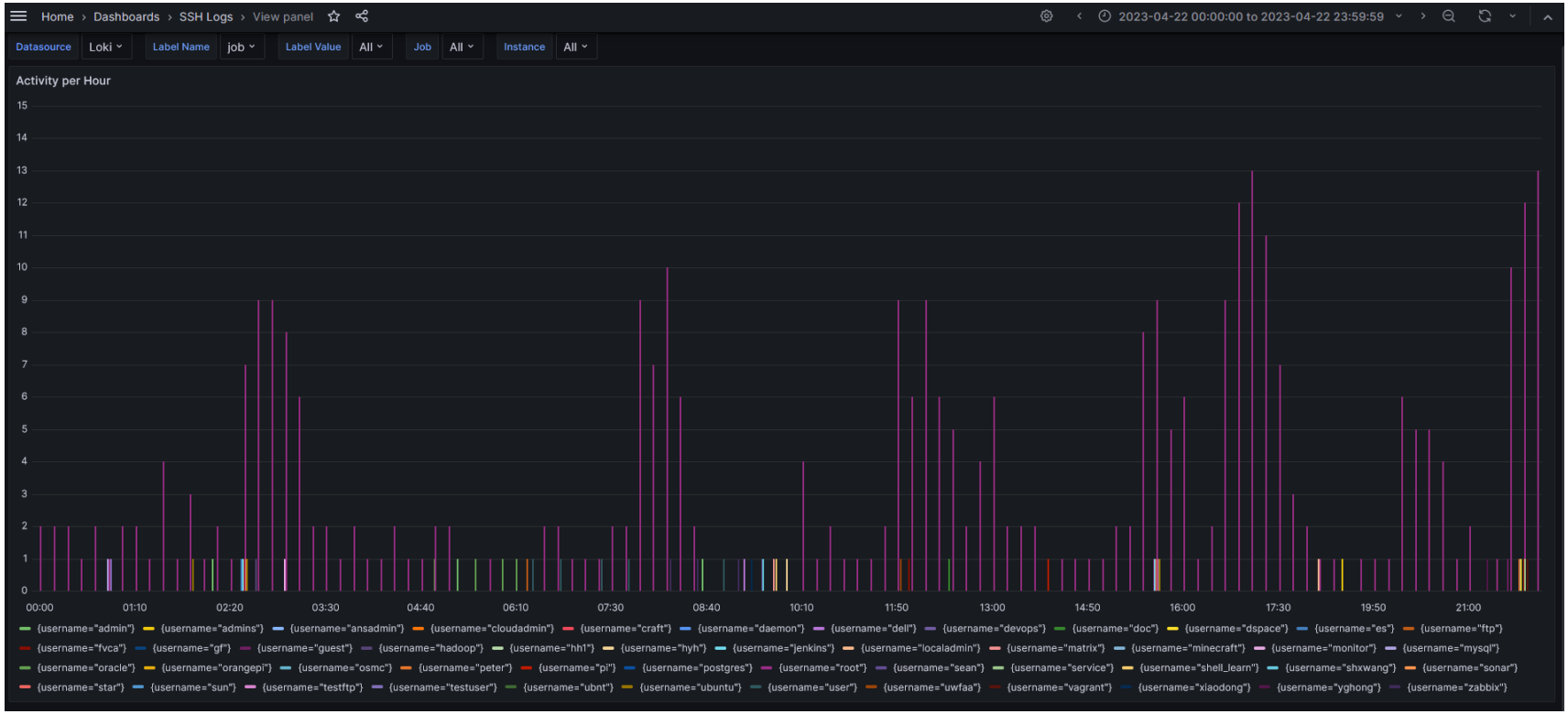


Abbildung 38: Balkendiagramm Darstellung der fehlgeschlagenen Anmeldeversuche in einem Zeitfenster von 24 Stunden am „22.5.2022“

Die Abbildung 38 zeigt, dass die meisten Versuche zwischen 16:00 und 17:30 Uhr stattgefunden haben.

Die nächste Abbildung, 39, soll die Anzahl von fehlgeschlagene Anmeldeversuche pro IP-Adresse zeigen. Die benutzten Abfragen lauten wie folgt:

```
count by (ip) (count_over_time({job=~"sshlogs"}
|="sshd["
|~": Invalid
|: Connection closed by authenticating user
|: Failed .* user"
| pattern `<_> from <ip> port`
| __error__="" [$__interval]))

count by (ip) (count_over_time({job=~"sshlogs"}
|="sshd["
|=": Failed" !~"invalid user"
| pattern `<_> from <ip> port`
| __error__="" [$__interval]))
```

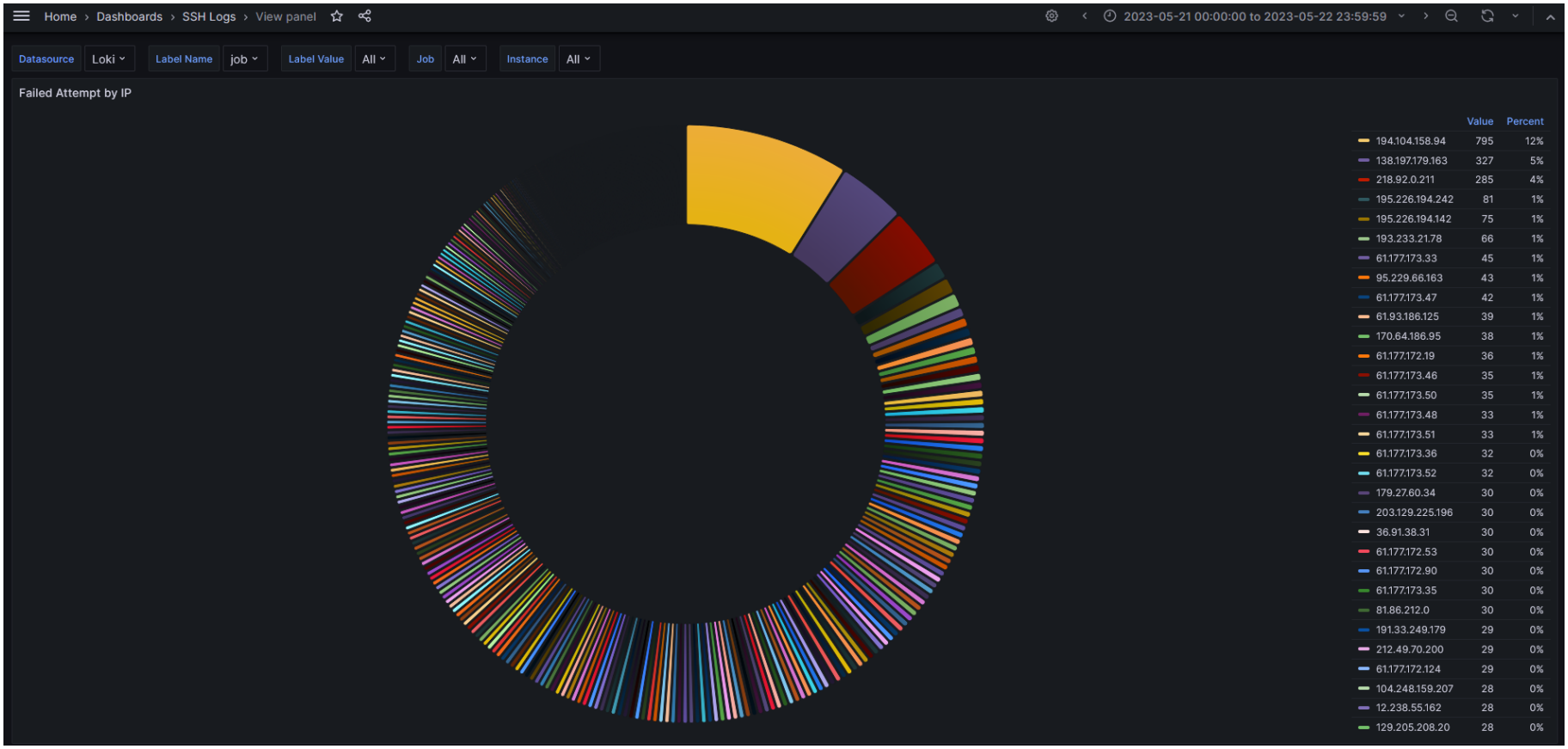


Abbildung 39: Kuchendiagramm von Anzahl fehlgeschlagenen Anmeldeversuche pro IP-Adresse

Aus Abbildung 39 können wir die IP-Adressen identifizieren, von denen die meisten fehlgeschlagenen Anmeldeversuche zwischen dem „21.5.2023“ und dem „22.5.2022“ stammen.

4.3. Generierung von Warnmeldungen

Mithilfe der oben gezeighten Abfragen generieren wir auch unsere Warnmeldung mit dem Alerting Tool von Grafana.

Warnmeldungen dienen hauptsächlich der Echtzeitanalyse. Da unsere Logdateien jedoch aus den Monaten März und April 2023 stammen, verschoben wir die Daten in die entsprechenden Monate, um „Echtzeit“-Daten für die Analyse zu simulieren. Dies ermöglicht uns, aktuelle Warnungen und Alarme basierend auf den verschobenen Daten zu generieren.

```
                    Mar ===> Apr | März ====> April
                    Apr ===> May | April ===> May
               26 Apr ===> 27 May | 26 April ===> 27 May
```

Für unsere Warnmeldung benutzten wir folgende Abfrage:

```
sum by (Source_IP) (count_over_time({job=~"sshlogs"}
|="sshd["
|~": Invalid
|:Connection closed by authenticating user
|: Failed .* user"
| pattern `<_> from <Source_IP> port`
| __error__="" [5m]))
```

Aus dieser Abfrage finden wir heraus, wie viele fehlgeschlagene Anmeldeversuche pro IP-Adresse es gibt. Mit einem Zeitfenster von einer Woche lösen wir immer dann eine Meldung aus, wenn dieser Wert größer als fünf ist. Fünf ist ein von uns ausgewählter Wert.

In der Abbildung 40 zeigen wir, wie eine Warnmendung assieht und wir können auch die Quelladresse des fehlgeschlagenen Anmeldeversuchs erkennen:

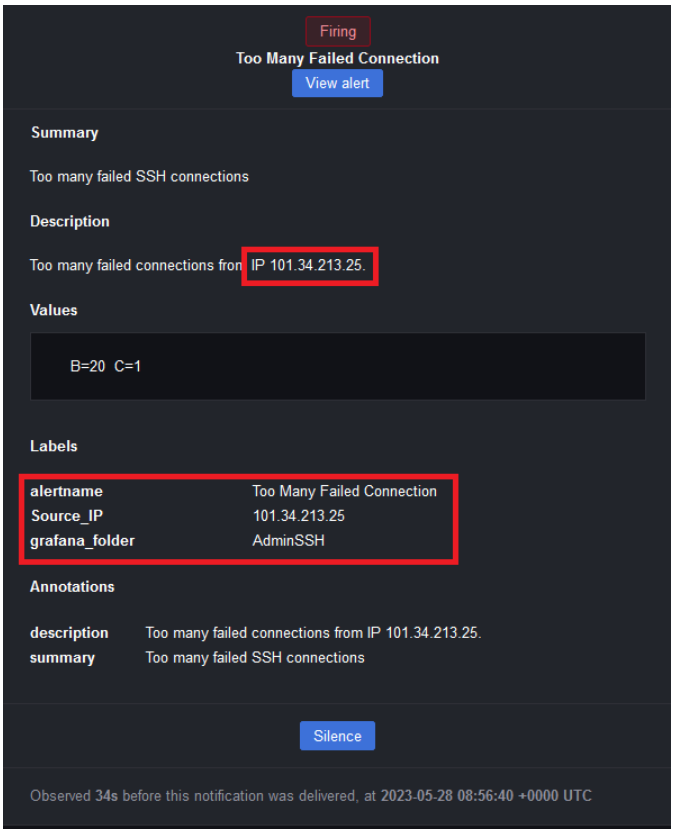


Abbildung 40: Warnmeldung von Grafana über fehlgeschlagenen SSH-Anmeldeversuch

4.4. Zusammenfassung

Grafana bietet zahlreiche Möglichkeiten, um Daten zu kombinieren, darzustellen und Warnmeldungen zu generieren. Die Nutzungsmöglichkeiten sind praktisch unbegrenzt und hängen von der Kreativität des Benutzers sowie den Anforderungen der jeweiligen Situation ab.

5. Fazit

5.1. Diskussion der Ergebnisse

In dieser Arbeit haben wir versucht, mithilfe von Grafana, Loki und Promtail eine SIEM-Lösung zu emulieren, um Überwachungsmechanismen anhand von Logdateien zu erstellen. In der folgenden Tabelle zeigen wir die Rolle jedes verwendeten Tools bei der Erreichung unseres Ziels:

Tool	**Funktionalität**
Promtail	Datensammlung
Loki	Normalisierung und Vearbeitung
Grafana	Berichts- und Grafikgenerierung
Grafana:Alerting	Generierung von Warnmeldungen

Tabelle 8: Verwendete Tools und ihre Hauptfunktionalitäten
Verwendete Tools und ihre Hauptfunktionalitäten

Wir stellen fest, dass die verwendeten Tools eine kosteneffektive Möglichkeit bieten, ein Überwachungssystem zu implementieren. Die Methoden zur Erkennung von Angriffen lassen sich anhand der Taktiken, Techniken, Prozeduren (TTP) der Mitre ATT&CK Matrix definieren. Nach der Auswahl eines Angriffs erstellen wir Regelwerke mit der Abfragesprache LogQL in Loki, um Muster zu identifizieren, die auf den ausgewählten Angriff hindeuten. Diese Regelwerke werden dann verwendet, um Warnmeldungen über den Angriff zu generieren und zu versenden.

Zu unseren initiallen Ziele:

- Wie können wir ein Log-Analyse-Tool konfigurieren, dass es vordefinierte Angriffe nach der Mitre ATT&CK Matrix automatisch erkennen kann?
- Wie können wir allgemeine Regelsätze definieren, sodass wir sie später für die verschiedene TTP der Mitre ATT&CK Matrix nanpassen können?

können wir sagen, dass die Mitre ATT&CK Matrix umfangreiche Informationen anbietet,

um präzise Regelästze zu generieren.

5.2. Herausforderungen

Zu unserem primären Ziel können wir sagen, dass die Mitre ATT&CK-Matrix umfangreiche Informationen bietet, um zielgerichtete Regelästze zu generieren. Die Erstellung dieser Regelästze kann jedoch eine der größten Herausforderungen bei der Implementierung darstellen, da die Verwendung der Abfragesprache LogQL viel Zeit in Anspruch nehmen kann. Sobald diese Hürde jedoch überwunden ist, ist es möglich, präzise Regelästze zu erstellen, um potenzielle Angriffe zu identifizieren. Die Lernkurve für den Aufbau der richtigen Regelästze kann eine große Herausforderung darstellen, wie auch in unserem Fall.

Da Logdateien aus produktiven Umgebungen eine große Menge an Informationen enthalten, müssen die Regelästze so definiert werden, dass sie die relevanten Informationen wie IP-Adresse, Portnummer, Zeitfenster und Zeitabstände zwischen Anfragen filtern und nach Angriffsmustern kategorisieren können.

Die zweite große Herausforderung bestand darin, die richtigen Einstellungen und Funktionen von Promtail, Loki und Grafana zu verwenden. Das Beherrschen dieser Elemente kann dazu beitragen, dass die Anwendungen reibungslos funktionieren und vertrauenswürdige Ergebnisse liefern.

Die korrekte Nutzung der Filteroptionen von Promtail, genannt „scrape_configs“, ermöglicht die Extrahierung spezifischer Informationen und die Generierung präziser Labels. Das Verständnis jeder Funktionalität von Grafana stellt sicher, dass die ausgegebenen Daten die notwendigen Informationen enthalten, um den Entscheidungsprozess zu erleichtern. Die richtigen Einstellungen gewährleisten eine fehlerfreie Nutzung der Anwendungen und ermöglichen ihre Skalierbarkeit. In diesem Fall können sowohl die offizielle Dokumentation als auch die offiziellen Forenbeiträge dazu beitragen, die Tools richtig zu konfigurieren.

Letztendlich sind „Labels“ auch wichtige Elemente bei Grafana, Loki und Promtail. Die korrekte und präzise Indizierung spielt eine entscheidende Rolle für die Leistung der Anwendung. Die Verwendung vieler „Labels“ erfordert eine hohe Rechenkapazität und kann auch zu fehlerhaften Ergebnissen führen. Die Rechenkapazität muss ebenfalls angepasst werden, um sicherzustellen, dass die steigende Anzahl von Anfragen (siehe Abbildung 12 auf Seite 20) nicht zu Abstürzen führt.

5.3. Zukünftige Forschung

Dieser Arbeit ermöglicht eine Weiterentwicklugn in verschiedenen Bereichen:

- **Abdeckung vielen möglichen Cyberangriffen mit neuen Regelsätze**:

Mit der Nutzung der Taktiken, Techniken, Prozeduren (TTP) der Mitre ATT&CK Matrix ist es mögliche Regelsätze in LogQL für viele anderen Cyberangriffe aufzubauen und dadurch Logdateien aus verschiedenen Systemen und Anwendungen zu verwenden. Die Nutzung von anderen Regelsätzen, die sich auf verschiedenen Angriffe beziehen, kann eine umfangreiche Sicherheit für produktive Umgebung anbieten, indem mehr Uses Cases gedeckt werden, um Angriffe zu erkennen.

- **Beherschung der Tools: Promtail, Loki und Grafana**:

Grafana, Loki und Promtail bieten in ihrer Konfiguration verschiedenen Möglichkeiten und Informationen von Logdateien zu extrahieren, zu filtern und zur Genierierung von Warnmeldungen. Eine tiefe Beherrschung von „scrape_configs“ von Promtail trägt dazu bei, Logdateien zu erkennen und wichtige Informtionen direkt zu filtern, ohne das weitere Abfrage notwendig ist, indem auch Leistung gespart wird. Eine Weiterarbeit mit der Abfragesprache LogQL hilft dabei, präzise und effizieren Abfrage aufzubauen, um bessere Grafiken und/oder Warnmeldung zu genieren. Zusätzlich kann die vielfältige Funktionalitäten von Grafana dabei unterstützen, zuverlässige Grafiken und Tabelle zu generieren, um nützliche Informationen aus dem Logdaien grafisch darzustellen. Die Beherrschung

dieses Tool stellt auch eine mögliche und vielversprechende weitere Recherche dar.

- **Umfangreiche Beobachtbarkeit mit den Tools der *Grafana Ecosystem***:

Die Tools um den *Grafana Ecosystem* bieten viele Möglichkeiten, um eine deutliche und präzise Beobachtbarkeit eines Systems durchzuführen. Wenn kombiniert, ermöglichen sie eine holistische und tiefe Analyse von Log- und Messdaten und Ablaufverfolgung. Die kombinierte Implementierung in einer produktiven Umbegung kann dazu beitragen, die Sicherheit eines Systems auch bei skalierbaren Umgebungen zu verbessern. Eine Recheche in dieser Richtung hat auch die Möglichkeit, positive Ergebnisse zu liefern.

- **Automatische Antworte auf mögliche Cyberangriffen**:

Eine umfangreiche SIEM-Lösung bietet laut Mohammed et al. (2021) die wichtigen Informatinen, um Angriffe zu erkennen. Die Sicherheitsanalyse stoppt jedoch nicht in der Erkennung, sondern verlangt Handlungen, um laufenden Angriffe zu stoppen oder potenziellen zu verhindern. Die Entwicklung oder die Integration von existierenden Tools, um automatisch gegen Cyberangriffe zu handeln, stellen auch eine mögliche Perspektive für zukünftige Recheche dar.

- **Nutzung von KI**:

Moderne Angriffe haben heutzutage einen dynamischen Aspekt, der sich an die Umgebung anpasst, insbesondere durch die fortschreitende Entwicklung von Künstliche Intelligenz (KI) (Guembe et al., 2022). KI kann zur Automatisierung von Aufgaben oder zur effizienten Datenanalyse eingesetzt werden. Für die zukünftige Weiterentwicklung dieser Arbeit kann dieses Werkzeug eine große Unterstützung bieten, indem es performantere und effizientere Regelsätze vorschlägt, um die Log-Analyse effizienter und zuverlässiger zu gestalten.

Diese Möglichkeiten zusammen oder getrennt könnten dazu beitragen, einen sicheren Netzwerkverkehr zu gewährleisten.

Literaturverzeichnis

Advani, S., Mridul, M., Vij, P. S. R., Agarwal, M., and A., L. P. (2020). Iot data analytics pipeline using elastic stack and kafka. *International Journal of Computer Sciences and Engineering*, 8:144–148.
https://www.ijarcce.com/upload/2016/april-16/IJARCCE%2013.pdf. Zugriff am 07.03.2023.

Anand, A. (2023). Loki vs elasticsearch - which tool to choose for log analytics?
https://signoz.io/blog/loki-vs-elasticsearch/. Zugriff am 18.05.2023.

at (2022). Abfragesprache.
https://www.alexanderthamm.com/de/data-science-glossar/abfragesprache/. Zugriff am 08.04.2023.

AT&T Cybersecurity (2022). Alienvault ossim.
https://cybersecurity.att.com/products/ossim. Zugriff am 05.03.2023.

Ba, M. H. N., Bennett, J., Gallagher, M., and Bhunia, S. (2021). A case study of credential stuffing attack: Canva data breach. In *2021 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 735–740.
https://doi.org/10.1109/CSCI54926.2021.00187. Zugriff am 26.03.2023.

Better Stack Team (2023). Grafana vs Kibana: How to Choose in 2023.
https://betterstack.com/community/comparisons/grafana-vs-kibana/. Zugriff am 18.05.2023.

BSI (2021). Zweites Gesetz zur Erhöhung der Sicherheit informationstechnischer Systeme (IT-Sicherheitsgesetz 2.0).
https://www.bsi.bund.de/DE/Das-BSI/Auftrag/Gesetze-und-Verordnungen/IT-SiG/2-0/it_sig-2-0_node.html. Zugriff am 04.03.2023.

CBNINSIGHTS (2020). Alienvault.
https://www.cbinsights.com/company/alienvault. Zugriff am 05.03.2023.

Centers for Disease Control and Prevention (2016). Health Insurance Portability and Accountability Act of 1996 (HIPAA).
https://www.pcicomplianceguide.org/faq/. Zugriff am 04.03.2023.

Chai, W. and Ferguson, K. (2021). What is HTTP?
https://www.techtarget.com/whatis/definition/HTTP-Hypertext-Transfer-Protocolk. Zugriff am 17.04.2023.

Collins, C., Dennehy, D., Conboy, K., and Mikalef, P. (2021). Artificial intelligence in information systems research: A systematic literature review and research agenda. *International Journal of Information Management*, 60:102383.
https://www.sciencedirect.com/science/article/pii/S0268401221000761. Zugriff am 21.02.2023.

comparitech (2023). The Best SIEM Tools for 2023 Vendors & Solutions Ranked.
https://www.comparitech.com/net-admin/siem-tools/. Zugriff am 05.03.2023.

DevInsider (2021). Was ist distributed tracing?
https://www.dev-insider.de/was-ist-distributed-tracing-a-17a5fcbe722ca868e1f393fd6c35bbbb/. Zugriff am 05.03.2023.

Dorigo, S. (2012). Security Information and Event Management. Master's thesis, Radboud University Nijmegen.
https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiNu-XkhsD9AhV4FzQIHdMkBWYQFnoECCYQAQ&url=https%3A%2F%2Fwww.ru.nl%2Fpublish%2Fpages%2F769526%2Fthesissanderdorigo.pdf&usg=AOvVaw3oPn4KBFwgJwexoXZ1Be40. Zugriff am 03.03.2023.

Douglis, F. and Nieh, J. (2019). Microservices and containers. *IEEE Internet Computing*, 23(6):5–6.
https://doi.org/10.1109/MIC.2019.2955784. Zugriff am 236.03.2023.

Ecma, E. (2017). ECMA-404 - The JSON Data Interchange Syntax.
https://www.ecma-international.org/publications-and-standards/standards/ecma-404/. Zugriff am 18.05.2023.

elastic (2015). Query DSL.
https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html#query-dsl. Zugriff am 18.05.2023.

elastic (2021). *FAQ on 2021 License Change*.
https://www.elastic.co/pricing/faq/licensing. Zugriff am 26.03.2023.

elastic (2022). *Elastic Docs*.
https://www.elastic.co/guide/en/welcome-to-elastic/current/new.html. Zugriff am 5.02.2023.

European Comission (2015). SIEM design and development.
https://cordis.europa.eu/project/id/644425. Zugriff am 05.03.2023.

Fortinet (2016). Fortinet Announces Acquisition of AccelOps .
https://www.fortinet.com/corporate/about-us/newsroom/press-releases/2016/fortinet-announces-acquisition-of-accelops. Zugriff am 06.03.2023.

Fortinet (2018). Key Concepts.
https://help.fortinet.com/fsiem/5-1-2/Online-Help/HTML5_Help/Key_concepts.htm. Zugriff am 18.05.2023.

Fortinet (2020). FortiSIEM Reference Architecture.
https://www.fortinet.com/content/dam/maindam/PUBLIC/02_MARKETING/02_Collateral/DeploymentGuide/dg-fortisiem-reference-architecture.pdf. Zugriff am 06.03.2023.

Fortinet (2022). FortiSIEM Solutions.
https://www.fortinet.com/products/siem/fortisiem. Zugriff am 06.03.2023.

Fu, F. (2018). Chapter six - design and analysis of complex structures. In *Design and Analysis of Tall and Complex Structures*, pages 177–211. Butterworth-Heinemann.
https://www.sciencedirect.com/science/article/pii/B978008101018100006X.

Zugriff am 06.03.2023.

Grafana Labs (2016a). Dashboard anything. Observe everything. https://grafana.com/grafana/. Zugriff am 12.03.2023.

Grafana Labs (2016b). Grafana Loki documentation. https://grafana.com/docs/loki/latest/fundamentals/overview/. Zugriff am 10.05.2023.

Grafana Labs (2019). Alerting. https://grafana.com/docs/grafana/latest/alerting/. Zugriff am 21.04.2023.

Grafana Labs (2020a). About Grafana Tempo. https://grafana.com/oss/tempo/. Zugriff am 11.05.2023.

Grafana Labs (2020b). Getting started. https://grafana.com/docs/loki/latest/getting-started/. Zugriff am 09.04.2023.

Grafana Labs (2020c). Grafana Loki HTTP API. https://grafana.com/docs/loki/latest/api/. Zugriff am 17.04.2023.

Grafana Labs (2020d). Loki - configuration. https://grafana.com/docs/loki/latest/configuration/. Zugriff am 23.05.2023.

Grafana Labs (2020e). Promtail. https://grafana.com/docs/loki/latest/clients/promtail/. Zugriff am 11.05.2023.

Grafana Labs (2020f). Promtail - configuration. https://grafana.com/docs/loki/latest/clients/promtail/configuration/. Zugriff am 23.05.2023.

Grafana Labs (2021a). Alertmanager. https://grafana.com/docs/grafana/latest/alerting/manage-notifications/alertmanager/. Zugriff am 21.04.2023.

Grafana Labs (2021b). Grafana loki documentation - fundamentals - architecture - components. https://grafana.com/docs/loki/latest/fundamentals/architecture/components//. Zugriff am 24.05.2023.

Grafana Labs (2021c). LogQL: Log query language. https://grafana.com/docs/loki/latest/logql/. Zugriff am 14.04.2023.

Grafana Labs (2021d). What is opentelemetry? https://grafana.com/oss/opentelemetry/. Zugriff am 17.04.2023.

Grafana Labs (2022a). Dashboard anything. Observe everything. https://grafana.com/logs/. Zugriff am 12.03.2023.

Grafana Labs (2022b). Grafana Agent. https://grafana.com/docs/agent/latest/. Zugriff am 17.04.2023.

Grafana Labs (2022c). How to send logs to grafana loki with the opentelemetry collector using fluent forward and filelog receivers.
https://grafana.com/blog/2022/06/23/how-to-send-logs-to-grafana-loki-with-the-opentelemetry-collector-using-fluent-forward-and-filelog-receivers/. Zugriff am 17.04.2023.

Grafana Labs (2022d). What is Grafana Mimir?
https://grafana.com/docs/loki/latest/logql/. Zugriff am 21.04.2023.

Grafana Labs (2022e). What is Grafana Phlare?
https://grafana.com/oss/phlare/. Zugriff am 11.05.2023.

Grafana Labs (2023a). Grafana Loki (Version 2.8.2).
https://github.com/grafana/loki. Zugriff am 18.05.2023.

Grafana Labs (2023b). Grafana (Version 9.5.2).
https://github.com/grafana/grafana. Zugriff am 18.05.2023.

Granadillo, G., González-Zarzosa, S., and Diaz, R. (2021). Security Information and Event Management (SIEM): Analysis, Trends, and Usage in Critical Infrastructures. *Sensors*, 21:4759.
https://www.mdpi.com/1424-8220/21/14/4759. Zugriff am 21.02.2023.

Guembe, B., Azeta, A., Misra, S., Osamor, V. C., Fernandez-Sanz, L., and Pospelova, V. (2022). The emerging threat of ai-driven cyber attacks: A review. *Applied Artificial Intelligence*, 36(1):2037254.
https://www.tandfonline.com/doi/full/10.1080/08839514.2022.2037254. Zugriff am 14.05.2023.

Gómez, E. C. F., Almeida, O. X. B., and Gamboa, L. M. A. (2022). Analysis of centralized computer security systems through the alienvault ossim tool. *Ecuadorian Science Journal*, 6(1):23–31.
https://journals.gdeon.org/index.php/esj/article/view/181. Zugriff am 03.03.2023.

Harmes, T. (2023). It-sicherheitsgesetz 2.0.
https://rz10.de/knowhow/it-sicherheitsgesetz-2-0/. Zugriff am 04.03.2023.

Hazel, T. (2021). How To Use the MITRE ATT&CK Framework.
https://www.chaossearch.io/blog/how-to-use-mitre-attck-framework. Zugriff am 26.03.2023.

Höfling, M. J. (2022). Was ist opentelemetry?
https://www.datacenter-insider.de/was-ist-opentelemetry-a-e6c095b313e36269b752d760b2438bb2/. Zugriff am 12.05.2023.

IBM (2020). What is an api (application programming interface)?
https://www.ibm.com/topics/api. Zugriff am 17.04.2023.

Ibrokhimov, S., Hui, K. L., Al-Absi, A. A., hoon jae lee, and Sain, M. (2019). Multi-Factor Authentication in Cyber Physical System: A State of Art Survey. In *2019 21st International Conference on Advanced Communication Technology (ICACT)*, pages

279–284. https://doi.org/10.23919/ICACT.2019.8701960, Zugriff am 26.03.2023.

IT-Service.Network (2020). Was ist ein plug-in? https://www.hhs.gov/hipaa/for-professionals/privacy/index.html. Zugriff am 04.03.2023.

itsnotv (2022). Grafana dashboard shows "too many outstanding requests" after upgrade to v2.4.2 #5123. GitHub forum. https://github.com/grafana/loki/issues/5123. Zugriff am 21.05.2023.

Jain, U. (2018). *Lateral Movement Detection Using ELK Stack*. PhD thesis, University of Houston. https://uh-ir.tdl.org/handle/10657/3109. Zugriff am 07.03.2023.

Janiesch, C., Zschech, P., and Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets*, 31(3):685–695. https://doi.org/10.1007/s12525-021-00475-2. Zugriff am 13.03.2023.

Jog, Y. (2020). Security Information and Event Management (SIEM). https://www.linkedin.com/pulse/security-information-event-management-siem-yatin-jog. Zugriff am 04.03.2023.

Kali (2019). Kali inside virtualbox (guest vm). https://www.kali.org/docs/virtualization/install-virtualbox-guest-vm/. Zugriff am 02.04.2023.

Kali (2022a). Hydra. https://www.kali.org/tools/hydra/. Zugriff am 02.04.2023.

Kali (2022b). What is kali linux & kališ features. https://www.kali.org/docs/introduction/. Zugriff am 02.04.2023.

Kazarov, A., Avolio, G., Chitan, A., and Mineev, M. (2018). Experience with splunk for archiving and visualisation of operational data in atlas tdaq system. *Journal of Physics: Conference Series*, 1085:32052. http://dx.doi.org/10.1088/1742-6596/1085/3/032052. Zugriff am 04.03.2023.

Kray, M. (2022). Top 5 Open-Source Log Shippers (alternatives to Logstash) in 2022 . https://dev.to/max_kray/top-5-open-source-log-shippers-alternatives-to-logstash-in-2022-5f24. Zugriff am 18.05.2023.

Manases, L. and Zinca, D. (2022). Automation of network traffic monitoring using docker images of snort3, grafana and a custom api. In *2022 21st RoEduNet Conference: Networking in Education and Research (RoEduNet)*, pages 1–4. https://doi.org/10.1109/RoEduNet57163.2022.9921063. Zugriff am 13.03.2023.

Martin, L. (2018). The cyber kill chain. https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html. Zugriff am 12.03.2023.

Maymi, F., Bixler, R., Jones, R., and Lathrop, S. (2017). Towards a definition of cyberspace tactics, techniques and procedures. In *2017 IEEE International Conference on*

Big Data (Big Data), pages 4674–4679.
http://dx.doi.org/10.1109/BigData.2017.8258514. Zugriff am 09.05.2023.

Microsoft Security (2022). Endpoints defined.
https://www.microsoft.com/en-us/security/business/security-101/what-is-an-endpoint. Zugriff am 12.03.2023.

Mikalauskas, E. (2023). Rockyou2021: largest password compilation of all time leaked online with 8.4 billion entries.
https://cybernews.com/security/rockyou2021-alltime-largest-password-compilation-leaked/. Zugriff am 02.04.2023.

Miller, J. (2021). Is Elastic STACK (ELK) the best SIEM option?
https://www.bitlyft.com/resources/is-elk-the-best-siem-option#:~:text=The%20ELK%20stack%20is%20a,system%20from%20a%20system%20provider. Zugriff am 07.03.2023.

MITRE ATT&CK (2018a). Frequently Asked Questions.
https://attack.mitre.org/resources/faq/. Zugriff am 12.03.2023.

MITRE ATT&CK (2018b). Getting Started.
https://attack.mitre.org/resources/getting-started/. Zugriff am 26.03.2023.

MITRE ATT&CK (2020). Brute Force.
https://attack.mitre.org/techniques/T1110/. Zugriff am 26.03.2023.

Mohammed, S. A., Mohammed, A. R., Côté, D., and Shirmohammadi, S. (2021). A machine-learning-based action recommender for network operation centers. *IEEE Transactions on Network and Service Management*, 18(3):2702–2713.
https://doi.org/10.1109/TNSM.2021.3095463. Zugriff am 20.02.2023.

Mohanan, R. (2022). What Is Security Information and Event Management (SIEM)? Definition, Architecture, Operational Process, and Best Practices.
https://www.spiceworks.com/it-security/vulnerability-management/articles/what-is-siem/. Zugriff am 26.02.2023.

Nabil, M., Soukainat, S., Lakbabi, A., and Ghizlane, O. (2017). SIEM selection criteria for an efficient contextual security. In *2017 International Symposium on Networks, Computers and Communications (ISNCC)*, pages 1–6.
https://doi.org/10.1109/ISNCC.2017.8072035. Zugriff am 26.02.2023.

Nexcess (2022). Open source vs. proprietary: Which is better?
https://www.nexcess.net/blog/open-source-vs-proprietary/. Zugriff am 26.02.2023.

NIST (2020a). About nist.
https://www.nist.gov/about-nist. Zugriff am 19.02.2023.

NIST (2020b). Cyber attacke.
https://csrc.nist.gov/glossary/term/Cyber_Attack. Zugriff am 19.02.2023.

NIST (2020c). False positive.

https://csrc.nist.gov/glossary/term/false_positive. Zugriff am 05.03.2023.

NIST (2020d). Glossary.
https://csrc.nist.gov/glossary/. Zugriff am 19.02.2023.

Open Source Initiative (2007). The Open Source Definition (Annotated).
https://opensource.org/definition/. Zugriff am 17.02.2023.

OpenTelemetry (2023). Opentelemetry.
https://opentelemetry.io/. Zugriff am 12.05.2023.

packt (2019). What is elk stack?
https://subscription.packtpub.com/book/big-data-and-business-intellige
nce/9781788831031/1/ch01lvl1sec10/what-is-elk-stack. Zugriff am 07.03.2023.

Polinowski, M. (2019). What is elk stack?
https://mpolinowski.github.io/docs/DevOps/Provisioning/2021-04-07--lok
i-prometheus-grafana/2021-04-07/. Zugriff am 09.04.2023.

Prelude SIEM (2018). Prelude SIEM: Smart Security.
https://www.prelude-siem.com/en/prelude-siem-en/. Zugriff am 05.03.2023.

Prelude SIEM (2020). *Prelude Documentation: version 5.2*.
https://www.prelude-siem.org/docs/5.2/en/. Zugriff am 06.03.2023.

Prelude Team (2007). *Manual User*.
https://www.prelude-siem.org/projects/prelude/wiki/. Zugriff am 06.03.2023.

Project, T. (2021). Thehive - a 4-in-1 security incident response platform.
https://thehive-project.org/. Zugriff am 21.04.2023.

Prometheus (2015). Jobs and instances.
https://prometheus.io/docs/concepts/jobs_instances/. Zugriff am 08.05.2023.

Prometheus (2016). Documentation.
https://prometheus.io/docs/introduction/overview/. Zugriff am 14.04.2023.

Qusef, A. and Hassan, M. (2018). Power of using regular expression patterns in software coding standards quality control. In *2018 International Arab Conference on Information Technology (ACIT)*, pages 1–7.
https://doi.org/10.1109/ACIT.2018.8672682. Zugriff am 09.04.2023.

Radoglou-Grammatikis, P., Sarigiannidis, P., Iturbe, E., Rios, E., Martinez, S., Sarigiannidis, A., Eftathopoulos, G., Spyridis, Y., Sesis, A., Vakakis, N., Tzovaras, D., Kafetzakis, E., Giannoulakis, I., Tzifas, M., Giannakoulias, A., Angelopoulos, M., and Ramos, F. (2021). Spear siem: A security information and event management system for the smart grid. *Computer Networks*, 193:108008.
https://doi.org/10.1016/j.comnet.2021.108008. Zugriff am 03.03.2023.

Ramírez Tomás, I. (2018). *Implementación de un sistema de gestión de eventos de seguridad en una empresa de tamaño medio*. PhD thesis, Universitat Politècnica de València.
https://riunet.upv.es/bitstream/handle/10251/109765/Ram%c3%adrez%20-%20

Implementaci%c3%b3n%20de%20un%20sistema%20de%20gesti%c3%b3n%20de%20eventos%20de%20seguridad%20en%20una%20empresa%20de%20tama%c3%b1....pdf?sequence=1&isAllowed=y. Zugriff am 06.03.2023.

redhat (2022). What is grafana?
https://www.redhat.com/en/topics/data-services/what-is-grafana. Zugriff am 13.03.2023.

Roser, M., Ritchie, H., and Ortiz-Ospina, E. (2015). Internet. *Our World in Data*.
https://ourworldindata.org/internet. Zugriff am 17.02.2023.

Salinger, N. (2021). Introduction to Continuous Profiling.
https://granulate.io/blog/introduction-to-continuous-profiling/. Zugriff am 11.05.2023.

Savic, D., da Silva, A. R., Vlajic, S., Lazarevic, S., Stanojevic, V., Antovic, I., and Milic, M. (2012). Use case specification at different levels of abstraction. In *2012 Eighth International Conference on the Quality of Information and Communications Technology*, pages 187–192.
https://doi.org/10.1109/QUATIC.2012.64. Zugriff am 12.03.2023.

Selvaganesh, M., Karthi, P., Kumar, V. A. N., and Moorthy, S. R. P. (2022). Efficient brute-force handling methodology using indexed-cluster architecture of splunk. In *2022 International Conference on Electronics and Renewable Systems (ICEARS)*, pages 697–701.
https://doi.org/10.1109/ICEARS53579.2022.9752323. Zugriff am 12.03.2023.

Setter, M. (2015). Logfmt: A Log Format That's Easy To Read and Write.
https://www.cloudbees.com/blog/logfmt-a-log-format-thats-easy-to-read-and-write. Zugriff am 10.04.2023.

silicon.de (2022). Das beliebteste deutsche Passwort 2022 lautet: 123456.
https://www.silicon.de/41703603/das-beliebteste-deutsche-passwort-2022-lautet-123456. Zugriff am 02.04.2023.

Sowmya, G. V., Jamuna, D., and Reddy, M. V. K. (2012). Blocking of Brute Force Attack. *International journal of engineering research and technology*, 1.

Splunk (2015a). Splunk Enterprise Security.
https://www.splunk.com/en_us/products/enterprise-security.html. Zugriff am 12.03.2023.

Splunk (2015b). The splunk platform enables end-to-end visibility from edge to cloud.
https://www.splunk.com/en_us/products/splunk-enterprise.html. Zugriff am 03.05.2023.

Splunk (2022a). Use Cases.
https://docs.splunk.com/Documentation/ES/7.1.0/Usecases/Overview. Zugriff am 12.03.2023.

Splunk (2022b). What Is Security Information and Event Management (SIEM)?
https://www.splunk.com/en_us/data-insider/what-is-siem.html. Zugriff am

12.03.2023.

Su, T.-J., Wang, S.-M., Chen, Y.-F., and Liu, C.-L. (2016). Attack detection of distributed denial of service based on splunk. In *2016 International Conference on Advanced Materials for Science and Engineering (ICAMSE)*, pages 397–400. https://doi.org/10.1109/ICAMSE.2016.7840355. Zugriff am 12.03.2023.

Swathi, K. (2022). Brute Force Attack on Real World Passwords. *International Journal of Research Publication and Reviews*, 3(11):552–558. https://www.ijrpr.com/archive.php?volume=3&issue=11. Zugriff am 26.02.2023.

Tanenbaum, A. S. (2009). *Moderne Betriebssysteme*. Pearson, München.

Tanenbaum, A. S. and Wetherall, D. (2011). *Computer Networks*. Prentice Hall, München, 5 edition.

techopedia (2015). Security Event Management. https://www.techopedia.com/definition/25763/security-event-management. Zugriff am 03.03.2023.

techopedia (2022). Security Information Management (SIM). https://www.techopedia.com/definition/25763/security-event-management. Zugriff am 03.03.2023.

Tozzi, C. (2022). The 3 pillars of observability: Logs, metrics and traces. https://www.techtarget.com/searchitoperations/tip/The-3-pillars-of-observability-Logs-metrics-and-traces. Zugriff am 24.05.2023.

tutorialspoint (2009). HTTP - Methods. https://www.tutorialspoint.com/http/http_methods.htm. Zugriff am 17.04.2023.

Ubuntu (2023a). Get Ubuntu Server. https://ubuntu.com/download/server. Zugriff am 31.03.2023.

Ubuntu (2023b). Ubuntu. https://ubuntu.com/. Zugriff am 31.03.2023.

U.S. Department of Health & Human Services (2016). The HIPAA Privacy Rule. https://www.hhs.gov/hipaa/for-professionals/privacy/index.html. Zugriff am 04.03.2023.

Vault, A. (2019). AlienVault USM Anywhere. https://www.unifiedthreatworks.com/datasheets/DS-USM-Anywhere.pdf. Zugriff am 10.05.2023.

Veeramachaneni, G. (2018). Loki: Prometheus-inspired, open source logging for cloud natives. https://grafana.com/blog/2018/12/12/loki-prometheus-inspired-open-source-logging-for-cloud-natives/. Zugriff am 24.05.2023.

Vielberth, M. (2021). *Encyclopedia of Cryptography, Security and Privacy*, chapter Security Operations Center (SOC), pages 1–3. Springer Berlin Heidelberg. http://dx.doi.org/10.1007/978-3-642-27739-9_1680-1. Zugriff am 04.03.2023.

VoidQuark (2022). Parsing SSH Logs with Grafana Loki. https://voidquark.com/parsing-ssh-logs-with-grafana-loki/. Zugriff am 10.04.2023.

Wang, Y.-T., Yang, C.-T., Kristiani, E., and Chan, Y.-W. (2019). The implementation of wi-fi log analysis system with elk stack. In *Frontier Computing*, pages 246–255, Singapore. Springer Singapore. https://link.springer.com/chapter/10.1007/978-981-13-3648-5_28. Zugriff am 07.03.2023.

Welch, E. (2020). The concise guide to labels in loki. https://grafana.com/blog/2020/08/27/the-concise-guide-to-labels-in-loki/. Zugriff am 19.05.2023.

Wendzel, S. (2018). *IT-Sicherheit für TCP/IP- und IoT-Netzwerke*. Springer Vieweg, Wiesbaden.

Yigal, A. (2013). Grafana vs. Kibana: The Key Differences to Know. https://logz.io/blog/grafana-vs-kibana/. Zugriff am 18.05.2023.

Ödegaard, T. (2019). The (Mostly) Complete History of Grafana UX. https://grafana.com/blog/2019/09/03/the-mostly-complete-history-of-grafana-ux/. Zugriff am 13.03.2023.

Łukasz Korzeniowski and Goczyła, K. (2022). Landscape of automated log analysis: A systematic literature review and mapping study. *IEEE Access*, 10:21892–21913. https://doi.org/10.1109/ACCESS.2022.3152549. Zugriff am 12.03.2023.

A. Originalle Einstellungsdateien

Unten befindet sich die originalle Konfigurationsdateien (Grafana Labs, 2020b):

- **Loki**: Speicherung, Verarbeitung und Abfrage des Inhalts der Logdateien

```
auth_enabled: false

server:
  http_listen_port: 3100
  grpc_listen_port: 9096

common:
  instance_addr: 127.0.0.1
  path_prefix: /tmp/loki
  storage:
    filesystem:
      chunks_directory: /tmp/loki/chunks
      rules_directory: /tmp/loki/rules
  replication_factor: 1
  ring:
    kvstore:
      store: inmemory

query_range:
  results_cache:
    cache:
      embedded_cache:
        enabled: true
        max_size_mb: 100

schema_config:
  configs:
    - from: 2020-10-24
      store: boltdb-shipper
      object_store: filesystem
      schema: v11
      index:
        prefix: index_
        period: 24h

ruler:
  alertmanager_url: http://localhost:9093
```

- **Promtail**: Sammlung, Hinzufügen von „labels“ und Weiterletung an Loki des Inhalts der Logdateien,

```
  server:
  http_listen_port: 9080
  grpc_listen_port: 0

positions:
  filename: /tmp/positions.yaml

clients:
  - url: http://loki:3100/loki/api/v1/push

scrape_configs:
- job_name: system
  static_configs:
  - targets:
      - localhost
    labels:
      job: varlogs
      __path__: /var/log/*log
```

B. Angepasste Einstellungsdateien von Grafana

Unten befindet sich die angepasste Konfigurationsdateien (Polinowski, 2019):

- **Loki**

```
auth_enabled: false

server:
  http_listen_port: 3100
  grpc_listen_port: 9096

ingester:
  wal:
    enabled: true
    dir: /tmp/wal
  config:
    max_block_size: 67108864
    max_uncompressed_block_size: 67108864
    chunk_size: 1048576
  lifecycler:
    address: 127.0.0.1
    ring:
      kvstore:
        store: inmemory
      replication_factor: 1
    final_sleep: 0s
  chunk_idle_period: 1h
  # Any chunk not receiving new logs in this time will be flushed
  max_chunk_age: 1h
  # All chunks will be flushed when they hit this age, default is 1h
  chunk_target_size: 1048576
  # Loki will attempt to build chunks up to 1.5MB, flushing first if
  chunk_idle_period or max_chunk_age is reached first
  chunk_retain_period: 30s
  # Must be greater than index read cache TTL if using an index cache
  (Default index read cache TTL is 5m)
  max_transfer_retries: 0
  # Chunk transfers disabled

schema_config:
  configs:
    - from: 2020-10-24
      store: boltdb-shipper
      object_store: filesystem
      schema: v11
      index:
        prefix: index_
        period: 24h

storage_config:
  boltdb_shipper:
    active_index_directory: /tmp/loki/boltdb-shipper-active
    cache_location: /tmp/loki/boltdb-shipper-cache
    cache_ttl: 24h
    # Can be increased for faster performance over longer query
    periods, uses more disk space
    shared_store: filesystem
  filesystem:
    directory: /tmp/loki/chunks

compactor:
  working_directory: /tmp/loki/boltdb-shipper-compactor
```

```
  shared_store: filesystem

limits_config:
  reject_old_samples: true
  reject_old_samples_max_age: 168h
  ingestion_rate_mb: 1024
  ingestion_burst_size_mb: 1024
  ingestion_rate_strategy: local
  per_stream_rate_limit: 12MB
  max_query_series: 100000
  max_query_parallelism: 32
  split_queries_by_interval: 24h
  max_query_length: 0h

querier:
  max_concurrent: 2048

frontend:
  compress_responses: true
  scheduler_worker_concurrency: 20

query_range:
  parallelise_shardable_queries: false

table_manager:
  retention_deletes_enabled: false
  retention_period: 360s

ruler:
  storage:
    type: local
    local:
      directory: /tmp/loki/rules
  rule_path: /loki/rules-temp
  alertmanager_url: http://localhost:9093
  ring:
    kvstore:
      store: inmemory
  enable_api: true
```

- **Promtail**

```
---
server:
  http_listen_port: 9080
  grpc_listen_port: 0

positions:
  filename: /tmp/positions.yaml

clients:
  - url: http://loki:3100/loki/api/v1/push
    tenant_id: tenant1

scrape_configs:
- job_name: sshlogs
  pipeline_stages:
  - match:
      selector: '{job="sshlogs"}'
      action: keep
      stages:
      - regex:
      expression: '^(?P<time>[A-Za-z]{3}\s{1,2}\d{1,2}\s\d{2}:\d{2}:\d
      {2}).*from.(?P<sourceIP>(?:25[0-5]|(?:2[0-4]|1\d|[1-9]|)\d)\.(?:
      25[0-5]|(?:2[0-4]|1\d|[1-9]|)\d)\.(?:25[0-5]|(?:2[0-4]|1\d|[1-9]
```

```
        |)\d)\.(?:25[0-5]|(?:2[0-4]|1\d|[1-9]|)\d))'
    - labels:
        sourceIP: ${sourceIP}
    - timestamp:
        format: "Jan _2 15:04:05"
        source: time
        location: "Europe/Berlin"
    decompression:
      enabled: true
      initial_delay: 15s
      format: gz
    static_configs:
    - targets:
        - loki
      labels:
        job: sshlogs
        #env: voidquart
        instance: Opfersystem1
        __path__: /opt/*.gz
```

- **Docker Compose Datei**

```
version: "3"

networks:
  loki:

services:
  loki:
    image: grafana/loki:2.4.1
    volumes:
      -  ${PWD}/loki-config.yaml:/etc/loki/loki-config.yaml
    ports:
      - "3100:3100"
    command: -config.file=/etc/loki/local-config.yaml
    networks:
      - loki

  promtail:
    image: grafana/promtail:2.8.2
    container_name: Opfersystem1
    volumes:
      - ${PWD}/promtail-local-config_opfer1.yaml:/etc/promtail/promtail
      -config.yaml
      - ${PWD}/temp/:/opt/
    command: -config.file=/etc/promtail/promtail-config.yaml
             -config.expand-env=true
             #-querier.max-outstanding-requests-per-tenant= 2048
    networks:
      - loki

  grafana:
    image: grafana/grafana:latest
    ports:
      - "3000:3000"
    networks:
      - loki
```

C. Angepasste Einstellungsdateien von Grafana

Unten befindet sich unser Regel für die Generierung von Warnmeldungen in Fälle eines Brute-Force Angriffes gegen SSH Server.

```
apiVersion: 1
groups:
    - orgId: 1
      name: sshTeam
      folder: sshlogs
      interval: 1m
      rules:
        - uid: lHYZTLPVz
          title: Bruteforce Attempt agains SSH-Server
          condition: C
          data:
            - refId: A
              queryType: range
              relativeTimeRange:
                from: 600
                to: 0
              datasourceUid: sx2e5YE4k
              model:
                datasource:
                    type: loki
                    uid: sx2e5YE4k
                editorMode: code
                expr: 'sum by(username) (count_over_time({job=~"varlogs",
                job=~".*", instance=~".*"} |= `sshd[` |~ `: Invalid|:
                Connection closed by authenticating user|: Failed .*
                user` != `test` | pattern `<_> user <username> <_> port`
                | __error__=`` [2400h]))'
                hide: false
                intervalMs: 1000
                maxDataPoints: 43200
                queryType: range
                refId: A
            - refId: B
              queryType: range
              relativeTimeRange:
                from: 600
                to: 0
              datasourceUid: sx2e5YE4k
              model:
                datasource:
                    type: loki
                    uid: sx2e5YE4k
                editorMode: code
                expr: 'sum by(username) (count_over_time({job=~"varlogs",
                job=~".*", instance=~".*"} |= `sshd[` |= `: Failed` !~
                `invalid user` != `test` | pattern `<_> for <username>
                from <_> port` | __error__=`` [2400h]))'
                hide: false
                intervalMs: 1000
                maxDataPoints: 43200
                queryType: range
                refId: B
            - refId: C
              datasourceUid: __expr__
              model:
                conditions:
                    - evaluator:
```

```
                  params:
                      - 5
                      - 0
                  type: gt
                operator:
                  type: and
                query:
                  params:
                      - A
                reducer:
                  params: []
                  type: count
                type: query
              - evaluator:
                  params:
                      - 5
                      - 0
                  type: gt
                operator:
                  type: or
                query:
                  params:
                      - B
                reducer:
                  params: []
                  type: count
                type: query
        datasource:
            name: Expression
            type: __expr__
            uid: __expr__
        expression: ""
        intervalMs: 1000
        maxDataPoints: 43200
        refId: C
        type: classic_conditions
    noDataState: NoData
    execErrState: Error
    for: 5m
    annotations:
      description: We have several failed connections to our
      SSH-serves. This may be an attack.
      summary: Multiple failed connections to SSH-Server
    isPaused: false
```