RÉPUBLIQUE TUNISIENNE

****

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ DE SFAX

FACULTÉ DES SCIENCES DE SFAX

****

DÉPARTEMENT D'INFORMATIQUE ET DES
COMMUNICATIONS

GRADUATION INTERNSHIP

presented to

The Faculty of Sciences of Sfax

in order to obtain

LICENSE DEGREE IN

COMPUTER SCIENCE

by

Isslem ZOUARI

Maryem ALOULOU

« Server Monitoring System using Graylog »

sustained in 10 June 2023, to the jury composed of :

Mr.	**Ali SALEM**	*President*
Mm.	**Sirine MARRAKCHI**	*Examiner*
Mr.	**Boulbaba BEN AMMAR**	*Academic supervisor*
Mr.	**Ghazi ABIDA**	*Industrial supervisor*

Internship conducted at « SIFAST »

Academic year : 2022 – 2023

Code : LSI -16

DEDICACE

Throughout my pathway, I was surrounded by people who contributed to my success and growth. Thus, I would like to take this opportunity to express my gratitude toward them.

To my father **Amin,** I extend my deepest appreciation for his constant guidance and invaluable insights. Your profound wisdom and continuous support have been a constant source of inspiration.

To my mother **Nawel,** thank you for your unconditional love, patience, and sacrifices. Your belief in my abilities has given me the strength to overcome challenges and strive for excellence. You have played a pivotal role in my personal and academic growth

To my brother **Makki,** and my sisters **Marwa & Menyar** with whom, I shared countless moments of laughter, support, and encouragement, I am truly blessed to have you as my siblings.

To the soul of my grandmother **Kalthoum,** although you may not be physically present, your spirit and memory will forever be etched in my heart. I strive each day to honor your legacy by living a life filled with love, compassion, and resilience, just as you did.

To my grandmother **Mounira,** your presence and support instilled in me a deep sense of confidence, strength, and determination to face challenges head-on and pursue my dreams.

To my grandfathers **Abdelaziz & Hassen,** Your stories and experiences have taught me valuable life lessons that I will carry with me forever.

To my aunt **Imen Ayedi,** who was and still is my idol and my friend, as well as my uncles, especially **Mohamed Ali,** who was by my side whenever I needed him.

To my dear friends I would like to extend my gratitude for your support, understanding, and encouragement. Our shared experiences have made this academic journey enjoyable and

memorable. Your friendship has been a constant source of strength, and I am grateful for the memories we have created together.

To all those who made a good impact throughout my entire life.

I am truly grateful for every one of you.

Maryem Aloulou

DEDICACE

I dedicate this modest work, the result of my efforts from the very first day of school to the present day

To my dearest mother **Hela,** who has showered me with love and affection throughout my life. Your unwavering support and selflessness have been the driving force behind my success. May God bless and protect you.

To my beloved father, **Samir**, I dedicate this work as a token of my gratitude for your constant support, sacrifices, and efforts in ensuring my education and training. Your guidance and belief in me have been instrumental in shaping my academic path.

To my dear sister, **Sinda**, and my dear brother, **Oussema,** I express my heartfelt appreciation for your affection, understanding, and patience. Your encouragement and presence in my life have been a source of strength and inspiration.

To my friends and colleagues, I am profoundly grateful for your unwavering support.This achievement would not have been possible without you by my side. Thank you for being my rock and my biggest cheerleaders.

To **o** my entire family and all those dear to me.

This work is dedicated.

Zouari Isslem

ACKNOWLEDGMENT

In the context of this work, we would like to begin by expressing our gratitude to

Mr. Boulbaba BEN AMMAR, Assistant Professor at FSS Sfax, for his supervision, support and diligence, which contributed greatly to the completion of this report.

Mr. Ghazi ABIDA, Data Team Lead at SiFAST, for his guidance and advice.

Mr. Mohamed JMAL, BI Developer at SiFAST for his support and the pleasant moments we shared.

The entire team at SIFAST , we sincerely thank you for welcoming us and encouraging us to successfully complete this work.

Our sincere gratitude goes to the entire professional staff of FSS for the knowledge and skills they have imparted to us through their training.

Finally, we would like to express our sincere gratitude to the members of the jury for the honor they have bestowed upon us by agreeing to judge this work.

Isslem and Maryem

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

Logs are time-based records of events that occur across various applications and infrastructures. They are typically stored in one or more log files and come from different sources, including applications, services, hosts, and networks.

Nowadays, we are facing an explosion in the amount of log data, meaning, an exponential growth of logs generated by various systems and applications due to the increasing use of cloud computing, microservices, containerization, and new technology services in general. This issue made log data interfere in the Big Data field since the volume, variety, and velocity of log data have increased significantly and it has become challenging for organizations to manage and analyze log data effectively.

In fact, log management is a systematic process that involves aggregating all log data into one accessible location, making it easier to organize, search, and analyze the data generated by various applications and systems within an organization. An effective log management solution helps organizations improve real-time monitoring, identify issues and potential security threats, as well as maintain digital performance and reliability, enabling them to make informed decisions, improve end-user experiences, and gain insights into their systems and applications.

However, log management faces challenges such as the explosion of data, cloud adoption, and advanced data analysis requirements. To overcome these challenges, organizations are turning to log management tools like Graylog or ELK (Elasticsearch, Logstash, and Kibana) stack, which offer powerful functionality to create clarity from massive amounts of logs. These tools enable security, and operations teams to seamlessly monitor business applications by collecting log data from various sources, normalizing it, storing it, visualizing it, and analyzing

it in real-time.

The internship had the purpose of diving into the Graylog solution and was divided into 4 parts :

1. The first sprint was dedicated to researching and exploring the project's technologies and their functionalities including Elasticsearch, Graylog, MongoDB, and Filebeat as well as understanding their importance and the reason behind choosing them.
2. The second sprint was dedicated to the installation and configuration process which begins from setting up the system architecture until the log in into the Graylog web interface.
3. The third sprint was dedicated to diving into data analysis and manipulation.
4. The fourth sprint was dedicated to trying to extend the Graylog platform's functionality by using and creating custom plugins.

The upcoming report chapters will discuss each period's goal to compare at the end the performance of Graylog and ELK stack.

Chapter

1 PROJECT SCOPE

Contents

1.1 INTRODUCTION

The first chapter of this report serves as an introductory foundation. In fact, it provides a comprehensive overview of the host organization and the project's primary context, delves into the various technical objectives and perspectives that shape the project's direction and goals. Additionally, it elucidates the Agile methodology employed throughout the project's lifecycle, emphasizing its iterative approach and adaptability in addressing complex real-life issues involving human interactions with technology.

By establishing a clear understanding of the project's scope, goals, and stakeholders, this introductory chapter sets the stage for subsequent chapters, which will explore the project's progress, challenges, and outcomes in greater details.

1.2 HOST ORGANIZATION PRESENTATION

1.2.1 About SiFAST

SiFAST is a Digital Services Company (DSC) created in 2010 by service and IT professionals, it is one of the leaders of the Francophone nearshore and delivers IT services that allow its customers to improve their efficiency and their profitability. Thanks to its mastery of new technologies, its experience in outsourcing, and its collaborative approach, SiFAST helps its clients to develop innovative solutions to meet their challenges, focused on IT developments and outsourcing based on the principles of competence, respect, and innovation focused on customer satisfaction.

Inspired by various market standards such as CMMi and ISO 9001 and the nearshore experience, SiFast defines its quality system while maintaining its flexibility and agility.

1.2.2 Mission

SiFAST offers its customers quality service and innovative solutions while respecting their constraints and specificities. It enables them to optimize their sourcing by guaranteeing them a quality/price ratio and maximum flexibility.

1.2.3 Staff

In order to carry out its mission perfectly, SiFAST has built its human resources policy around excellence, convinced that the strength of the company lies in the quality of the expertise of its employees. It is open, attentive to its employees, and concerned about their well-being. SiFAST members share their knowledge, skills, and experiences.

SiFAST teams are organized by technology (LAMP /PHP /CMS, Mobile, .NET, Java EE, BI). Team leaders have acquired mastery of complex nearshore management and can avail themselves of real expertise in Front/Back delivery models allowing them to be strong in the proposal.

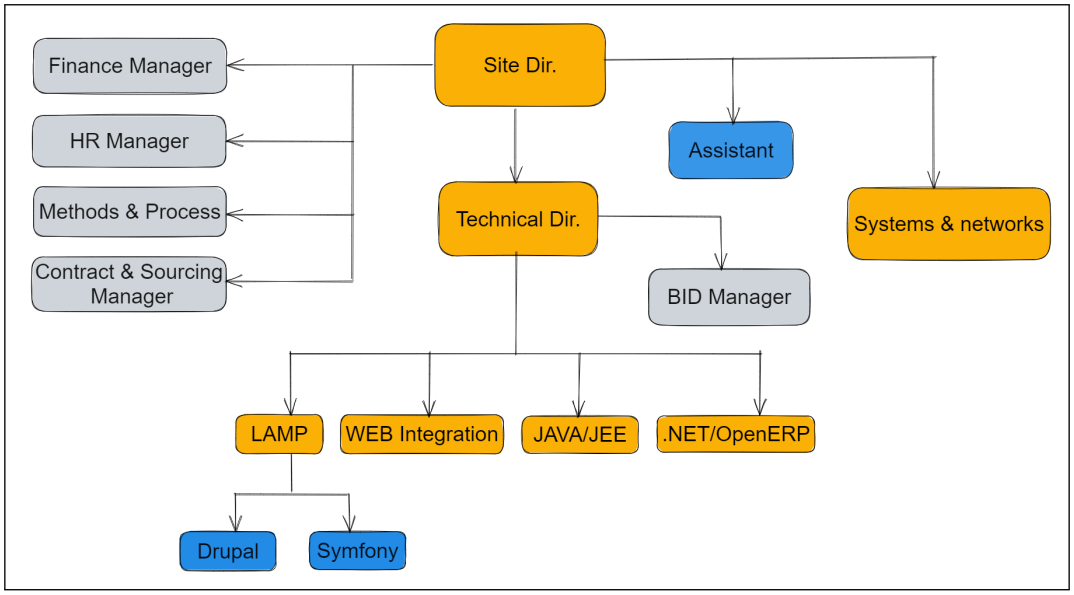

FIGURE 1.1 – SiFAST organization chart

SiFAST has about fifty employees with 80% engineers and is recognized as a training company which has enabled it to build a reliable recruitment network and to set up several partnerships with engineering schools in Tunisia.

1.2.4 Services :

- Web and Mobile application development.
- Business applications development.
- Existing applications extension.
- Open Source Software Integration.
- Technology migration and application redesign.

1.2.5 Vision

SiFAST as a recognized specialist in innovative solutions provides its clients with its highly specialized expertise so that their sites and applications are unique, captivating, and differentiating. This company tends to further improve its solutions while adopting new quality assurance (QA) strategies to ensure total customer satisfaction. All its employees apply the Agile approach using Scrum framework to better meet the needs of our customers and DevOps to automate and secure our development chain.

1.2.6 Engagements

- Quality Assurance Plan adapted for each project.
- Compliance with market standards.
- Compliance with costs and deadlines.
- Transparency of tariffs.
- Ongoing reporting.

1.2.7 Guarantees

SiFAST has put in place guarantees to collaborate with confidence and transparency :

- Software compliant with French and European rights.
- Software compliant with market norms and standards.
- Software that meets customer requirements and needs.
- Confidentiality and protection of intellectual property.
- Security system : Access, UTM, backup, UPS.

1.3 PROJECT'S CONTEXT

In this section, we will provide insights into the technical background and problem statement, as well as discuss the objectives and perspectives of the study.

1.3.1 Technical Background & Problem Statement

Our graduation project provides a solution for the company's need to monitor and analyze log files generated by its servers. In fact, this consists of a comparative study of Graylog and ELK stack utilities providing the intersections and the differences of each solution. For instance, the work is divided into 2 major parts : our report will cover the Graylog platform, and that of our colleagues will cover the ELK Stack. The cooperation between both teams will lead to a complete detailed study of these solutions through which we can conclude the ultimate choice of the best-fit solution according to the problem case.

The technical background of the project is a set of skills in the Big Data field, specifically in terms of setting up a system architecture.

The main idea is to demonstrate how Graylog can be used to optimize log management and isolate problems to their root causes, to establish a dashboard that visualizes the log data in real-time, and to alert users with specific events. Therefore, users will be up-to-date and notified

with logging details, even if they have no expertise in this field or they are out of the enterprise. Thus, they will be more comfortable and secure with the analyzes given by the system.

The need for a more complete, modern, and sophisticated version of the product requires the ultimate need of being open to plugin development in order to extend the Log Management System (LMS) functionalities.

1.3.2 Objectives

The 3 months graduation project is a set of objectives related to the needs of Sifast Company. The main context is a research work through which we try to reach a variety of goals :

- Learning about Graylog and its dependencies which include Elasticsearch, and Filebeat.
- Designing and setting up the system architecture to make the platform work adequately. This consists of installing and configuring Graylog and its dependencies.
- Collecting log data from various sources with Filebeat, a log shipper, to centralize them in Graylog.
- Analyzing log data through exploration using Graylog search and visualization tools.
- Manipulating logs to fit specific needs using pipelines and streams in order to get adequate data characteristics according to the need.
- Establishing an alerting system that will enable the user to keep track of what is going on in the server in real-time even while being out of the company.
- Extending Graylog functionalities. This can be done through using plugins which is a mission-related mainly to web development.
- A comparative study should be carried out between the Graylog platform and ELK stack in order to get the intersections and the differences of every built solution. The main goal to achieve, is to choose the most adequate solution regarding the use case and the company's needs. To crop it all, the focus is on the ratio between the quality of service (mainly development costs, time delay, dashboard quality, and analytics efficiency ...) and the given resources (hardware architecture of the user's device and its performances in terms of processing and memory).

Note : The Graylog platform is a complex system and not a trending software in the Big Data field. Reasonably, it's impossible to get a concrete result for just 3 months of work on this solution. Achieving all the previous goals and providing continuous work of ameliorating, validating, and testing to optimize and regulate the architectures, takes more than that. Therefore, our mission is to become familiar with using Graylog and start designing and building a simple and trustworthy model being able to defend itself and argue with the user concerning its choice and utility.

1.4 AGILE APPROACH AND SCRUM FRAMEWORK

As mentioned in a previous section, Sifast, when building projects, adopts the Agile approach using the Scrum framework. That is why, as interns, we followed this strategy to accomplish our mission. Throughout this section, we will detail Agile as well as Scrum concepts.

1.4.1 Agile Approach

To gain a thorough understanding of the Agile Methodology and its various aspects, it is essential to explore the foundational elements that form the basis of this approach to software development.

1.4.1.1 Agile Overview

The term "Agile" refers to the capacity to move quickly and easily, along with an emphasis on flexibility and willingness to change, adjust, as well as adapt.

1.4.1.2 Agile History

Although the Agile theory is not new, there is a difference between references in determining when it appeared. In fact, everything has been previously expressed, but due to some minor discrepencies, we find ourselves compelled to revisit and start anew.

Agile methodologies emerged during the 1990s when the software industry was booming, as a response to the traditional "waterfall" model of software development. In fact, Waterfall, as a project management methodology, follows a sequential and linear ordering of phases – we complete one phase at a time, not proceeding to the next until it is done, then we move down the line like a waterfall starting at the top of the mountain and traveling to the bottom. Whereas, Agile project management methodologies adopt an iterative approach, where project processes are repeatedly carried out throughout the project's life cycle – the team operates within a series of shorter timeframes known as iterations, with the possibility of repeating individual iterations based on feedback received. In very basic terms, we can think of iterations as a lot of mini waterfalls. A close look at the following graph (Figure 1.2) could simplify those approaches.

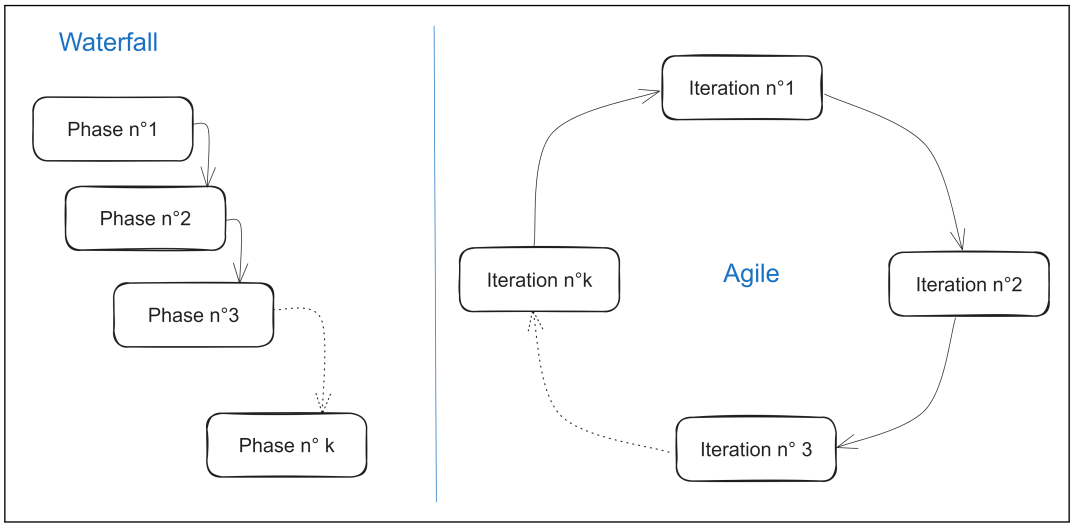

FIGURE 1.2 – Agile & Waterfall representational graph

In 2001, a group of 17 developers known as "organizational anarchists" discussed their shared frustrations with traditional development methods and exchange ideas about some approaches, referred to as "lightweight" frameworks, which aim to use few and simple rules that allow faster adaptation to changing environments. Among them, were proponents of Scrum, along with advocates of competing approaches like eXtreme Programming (XP), Crystal, Adaptive Software Development (ASD), Feature Driven Development (FDD), and the Dynamic Systems Development method (DSDM). Despite disagreements on several ideas, the group agreed on putting an end to the "lightweight" terminology and decided to adopt a new

name for their movement, which became known as "Agile". This term was suggested by one attendee who had come across a book called "Agile Competitors and Virtual Organizations : Strategies for Enriching the Customer".

After the naming convention, these thought leaders and creators came together to find common ground between their methods and experience and wrote the “Manifesto for Agile Software Development” in February 2001 which outlined four key values that everyone agreed upon. Subsequently, during the meeting and in the following months, they formulated a set of 12 guiding principles, referred to as the "Principles Behind the Agile Manifesto."

Consequently, any development frameworks that adhered to these values and principles would be recognized as agile techniques.

1.4.1.3 Agile Manifesto

The term MANIFESTO refers to a written statement declaring publicly the intentions, motives, or views of its issuer.

The Agile Manifesto outlined four key values along with twelve operating principles that could be grouped into four themes.

a) Agile Values

The Agile Manifesto is comprised of the components depicted in the figure 1.3

FIGURE 1.3 – Agile Manifesto

According to the Manifesto, it is important for Agile teams to consider both aspects of each statement throughout project execution. However, they should strive to prioritize and emphasize the elements on the left side while still acknowledging the ones on the right side.

In order to adopt an agile approach, teams should understand its four values which are listed below :

- **Value 1 :** Agile emphasizes the importance of effective communication, collaboration, and teamwork within the team rather than relying solely on processes and tools to achieve optimal results.
- **Value 2 :** Agile emphasizes that the primary focus should be on delivering the product that meets the customer's needs and satisfaction, rather than extensively documenting the process itself. Although this value is often associated with software projects, it can be applied to any project by replacing "working software" with the desired deliverable.
- **Value 3 :** Agile emphasizes customer satisfaction and the creation of a high-quality and valuable product over focusing on contracts. In fact, Agile promotes the freedom to engage and collaborate with customers early and frequently whether through showcasing early prototypes, seeking input through questions, or involving them in initial product testing. This allows teams to quickly respond and adapt to customer needs, rather than being hindered by lengthy contract negotiations for minor changes or resource requests.
- **Value 4 :** Agile emphasizes the need to acknowledge that change is inevitable and uncertainty is a natural part of the process. In fact, the ability to seamlessly integrate change is a key aspect of successful Agile projects. Above all, Agile grew out of a world that was changing so fast that organizations couldn't adapt and struggled to survive.

b) Agile Principles

The Agile Manifesto creators came up with twelve principles that we can group into four themes :

- **Value delivery :** This theme showcases how Agile teams deliver highly valuable products to their customers and crafts five principles :

"Our highest priority is to satisfy the customer through early and continuous delivery of valuable software."

Delivering value in a timely manner is a core principle of Agile, ensuring that customers, users, or organizations do not experience delays in receiving the benefits of the product, leading to trust, confidence, and early realization of business value through continuous feedback.

"Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale."

Delivering the product in small, regular increments is crucial as it provides opportunities for stakeholders, including customers, to share feedback on its progress. Thereby, the team avoids investing excessive time in the wrong direction.

"Working software is the primary measure of progress."

Unlike traditional or Waterfall projects, where progress is often measured based on the completion of project documents, in Agile project management, stating that the team is 80% complete with an activity lacks significance unless there is a tangible, demonstrable artifact that can be reviewed. Thus, Agile teams demonstrate their progress by presenting a working piece of the solution.

"Simplicity—the art of maximizing the amount of work not done—is essential."

Agile teams should avoid incorporating additional features – that were not explicitly requested by the user or product owner – into the solution.

"Continuous attention to technical excellence and good design enhances agility."

In Agile, speed should not come at the expense of design and quality. In fact, by delivering a well-constructed solution, the team can quickly respond to user feedback and new insights. Conversely, a low-quality product can create difficulties, complexity, and delays when implementing changes, affecting the overall team performance.

- **Business Collaboration :**

"Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage."

When working in Agile, recognizing and embracing the potential for plan modifications, even multiple times, ensures that you and your customers can achieve maximum success.

"Business people and developers must work together daily throughout the project."

Establishing open communication channels between developers and business-oriented individuals fosters trust and comprehension. This collaborative approach ensures that the builders of the solution, remain aligned with the users' needs and strengthens the connection between the technical and business aspects of the project.

- **Team Dynamics and Culture :**

"Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done."

Agile team members do not only trust one another to accomplish the work but are also trusted, empowered, and motivated by their sponsors and executives to fulfill their responsibilities as well as tackle challenging projects leading to the development of superior solutions.

"The most efficient and effective method of conveying information to and within a development is face-to-face conversation."

In Agile, face-to-face communication fosters effective teamwork, enabling the interpretation of body language, and facial expressions that may be missed in other forms of communication such as email or chats.

"Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely."

The Agile philosophy encourages achieving a balanced and steady pace of effort, avoiding overtime and burnout for optimal team performance.

"The best architectures, requirements, and designs emerge from self-organizing teams."

In Agile, team members have the autonomy to design their own work processes and practices, without a manager dictating how they operate.

- **Retrospectives and Continous Learning :**

"At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly."

In Agile, it is crucial to recognize that the process of learning from both successes and failures is ongoing. No team is immune to mistakes, challenges, trials, or triumphs. Therefore, it is important for teams to reflect upon all these aspects of their endeavors to make necessary adjustments and improvements.

1.4.1.4 VUCA

VUCA is a concept developed by the US Military War College that categorizes and analyzes the forces shaping our world, regardless of the industry. It provides a framework for understanding the conditions affecting organizations in a dynamic and complex environment.

VUCA is an acronym that stands for 4 conditions :

- **Volatility :** Refers to the speed at which change and instability occur in a business or situation.
- **Uncertainty :** Refers to the unpredictability or likelihood of encountering unexpected events or surprises.
- **Complexity :** refers to the high number of interrelated forces, issues, organizations, and factors that would influence the project.
- **Ambiguity :** Refers to the potential for misunderstanding or misinterpreting the conditions and the underlying causes of events or circumstances.

When starting a new project, it's important to evaluate the project's environment and conditions to choose the most suitable project Management approach. Hence, if the project exhibits significant VUCA parameters, it's a good sign we should consider an Agile approach.

1.4.2 Scrum Framework

1.4.2.1 Scrum Overview

Scrum is a widely used Agile framework that is often used interchangeably with Agile itself. Indeed, Scrum came before the Agile Manifesto and served as an inspiration for the entire Agile philosophy.

This project management framework is rooted in the concept of empiricism, where knowledge is derived from practical experience rather than assumptions or predictions.

Scrum is not an acronym. It refers to a formation in rugby where all team players lean forward, lock their heads together, and then work as one unit to try and gain precious yards toward the scoring line.

1.4.2.2 Scrum Pillars

The three fundamental pillars of empiricism, which also serve as the pillars of Scrum, are :

- **Transparency :** involves making the most important aspects of the work visible to all responsible parties, promoting effective communication and collaboration, as well as building trust among everyone involved to reduce errors. This includes maintaining transparency within the Scrum Team and extending it to stakeholders such as customers, sponsors, and management.
- **Inspection :** means regularly evaluating the progress and outcomes of a Sprint goal in order to identify and address any deviations or issues, allowing for continuous improvement and valuable opportunities to make necessary changes and enhancements.
- **Adaptation :** entails proactively seeking opportunities to adjust projects, products, or processes in order to address issues, and continuously improve the outcome.

1.4.2.3 Scrum Values

The presence of a shared value system within a team is highly beneficial as it establishes a mutual expectation for members to align their actions with these values. In the context of Scrum Teams, there are five core values that guide their work and behavior :

- **Commitment :** entails making a personal commitment to achieving the Scrum Team's goals. This can be seen in helping a team member in overcoming an obstacle that hinders their progress.
- **Courage :** means that Scrum team members are expected to exhibit courage as revealed by taking on challenging tasks, acquiring new skills, seeking help when needed, addressing negative behaviors, and responding effectively to difficult situations.
- **Focus :** highlights that everyone should direct their attention and efforts towards the essential work within the Sprint and the overall goals of the Scrum Team, enabling team members to concentrate on critical tasks, leverage their expertise, and ultimately accelerate progress.
- **Openness :** denotes that in Scrum, the team and stakeholders must be transparent and share all work-related challenges, which enables the collection of valuable data, facilitates effective problem-solving, and fosters resilience as well as collaboration.
- **Respect :** when established mutually, it fosters open communication and active listening, creating an environment where valuable feedback can be shared and considered, ultimately contributing to the product's or business's success.

1.4.2.4 Scrum Principles

Along with the Agile approach, the Scrum framework comes with 6 founding principles :

- **Built-in instability :** Scrum teams are empowered to incorporate changes and embrace challenges that play a vital role in the effective execution of strategic projects.
- **Self-organizing teams :** Scrum teams are intended to operate like their own start-up away from strict hierarchies

- **Overlapping development phases :** Members of a Scrum Team strive to synchronize their efforts and align their pace to meet project deadlines. As the process unfolds, the individual speeds of team members begin to overlap, resulting in the formation of a shared rhythm within the team.
- **Multi-learning :** Scrum, as a framework, emphasizes the importance of learning through trial and error as well as actively seeking to stay informed about evolving market conditions to enable swift and responsive actions.
- **Subtle control :** While Scrum Teams are self-organizing and autonomous, it doesn't imply a lack of structure.
- **Organizational transfer of learning :** In Scrum Teams, there is a culture of encouraging members to acquire new skills by supporting and assisting their teammates.

1.4.2.5 Scrum Events

a) Sprint

Sprints, the heartbeat of Scrum, are fixed-length events of one month or less. They serve as the engine for transforming ideas into value, encompassing essential activities like Sprint Planning, Daily Scrums, Sprint Review, and Sprint Retrospective. The product owner should ensure that the sprint goal remains intact and the quality is maintained, else the sprint can be canceled solely if the sprint goal becomes obsolete. During the sprint, we can refine the Product Backlog, allowing for scope adjustments, and promoting predictability, risk management, and learning cycles.

b) Sprint Planning

Sprint Planning, serving as the kickoff for the sprint, is a collaborative meeting in which the Scrum Team collectively determines the tasks to be accomplished. It encompasses evaluating the sprint's value, choosing Product Backlog items, and devising strategies for their completion.

The meeting is time-constrained, with a maximum duration of eight hours for a one-month Sprint. Furthermore, stakeholders could be invited to give feedback and provide advice.

c) Daily Scrum

The Daily Scrum is a brief, daily meeting of the Developers in the Scrum Team, consistently held at the same time and location during the Sprint. It has the purpose of assessing progress toward the Sprint Goal, making necessary adaptations to the Sprint Backlog, and creating a practical plan for the upcoming day. This gathering enhances communication, identifies obstacles, facilitates prompt decision-making, and reduces the need for additional meetings. However, developers have also the flexibility to make adjustments to their plan outside of the Daily Scrum by engaging in more detailed discussions throughout the day.

d) Sprint Review

The Sprint Review is a pivotal meeting where the Scrum Team shares the results of the Sprint with stakeholders. This involves engaging in discussions about progress toward the Product Goal, exploring potential adjustments which may involve modifying the Product Backlog, and collectively deciding on the next action. This session should be more than just a presentation and is the second-to-last event in the Sprint, with a maximum timebox of four hours for a one-month Sprint.

e) Sprint Retrospective

The Sprint Retrospective is conducted to strategize actions for enhancing quality and effectiveness. The Scrum Team evaluates the previous Sprint, regarding individuals, interactions, processes, tools, and the Definition of Done, pinpointing assumptions that led to issues and investigating their origins. Additionally, they discuss accomplishments, encountered challenges, and problem-solving approaches. Then, based on these discussions, the team identifies the most valuable improvements to enhance effectiveness, incorporating them into the next Sprint. The

Sprint Retrospective signifies the conclusion of the Sprint, with a timebox of up to three hours for a one-month Sprint.

1.4.2.6 Scrum Roles & Responsibilities

Every Scrum team member contributes significantly to the project's success as they collaborate to achieve shared objectives and fulfill the mission and vision.

a) Scrum Master

The Scrum Master plays a crucial role in ensuring the team understands and follows Scrum principles, helping them improve their practices within the Scrum framework and enabling their effectiveness. They also facilitate important meetings like the Daily Scrum, ensuring they stay within the designated timebox. The Scrum Master acts as a coach, supporting the team in achieving project goals, promoting self-management, cross-functionality, and the creation of high-value increments that meet the Definition of Done while removing impediments to progress.

b) Product Owner

The Product Owner's responsibility, as stated in the Scrum Guide, is to maximize the product's value by representing the customer's voice and ensuring the product fulfills their expectations. They achieve this by creating and communicating clear Product Backlog items, guaranteeing that every Scrum team member understands the why.

c) Development Team (or Developers)

According to the Scrum Guide, the Development Team consists of cross-functional specialists in various disciplines and is responsible for executing sprints by designing, building, and testing Product Backlog items, as well as adapting their plan toward the Sprint Goal.

1.4.2.7 Scrum Artifacts

Scrum artifacts are valuable sources of information used by stakeholders and the Scrum team to describe and track the progress of a product under development, maximizing transparency and providing a basis for adaptation.

According to the Scrum guide, every artifact carries a commitment to delivering information against which the progress is measured.

- For the Product Backlog it is the Product Goal.
- For the Sprint Backlog it is the Sprint Goal.
- For the Increment it is the Definition of Done.

a) The Product Backlog

The Product Backlog, being the central artifact in Scrum, is an ordered list of items, where all possible ideas, deliverables, features, or tasks are captured for the team to work on. This single authoritative source for project requirements serves as an evolving product roadmap.

The product owner is responsible for establishing, ordering, and clearly communicating the product backlog to the Scrum team, as well as making prioritization decisions that align with stakeholder interests and are influenced by the Scrum collaborators.

The Product Backlog Components

When working with Product Backlogs, it is important to consider several best practices and gather specific data to ensure its effective management. This artifact's commonly used components include :

- **Order :** Product Backlogs order items from highest to lowest priority : This is called a stacked rank. Both the estimate and value fields help the Product Owner figure out where to place an item in the Backlog's order of hierarchy.

- **Value :** This field in the Backlog represents the business value that an item brings to customers, the team, or the users, and it is a decision that should be made collaboratively by the Scrum Team.
- **Estimate :** This field in the Backlog indicates the estimated amount of work the Scrum Team believes is required to complete an item, and it is owned by the Development Team.
- **User Stories :** are concise and simple feature descriptions narrated from the user's viewpoint. They typically consist of three essential elements : the user, the action they intend to perform, and the benefit they expect to gain, expressed in various formats, but the most commonly used one is :

  As a <user role> I want <to perform an action> so that I can get this <value>

  To write impactful user stories, the team should envision a specific user who interacts with the product to achieve a desired outcome. This approach allows us to empathize with the users and their needs. Additionally, each user story should adhere to the six criteria represented by the acronym **I.N.V.E.S.T.**, ensuring its effectiveness and value.

  - Independent : Each user story should be self-contained, capable of being started and finished independently without reliance on another story to complete it.

  - Negotiable : The user story must allow for flexibility and open discussion.

  - Valuable : Completing the user story must result in delivering value.

  - Estimable : Having a clear Definition of Done is essential for the team to provide accurate estimates for each user story.

  - Small : To ensure that user stories can be accommodated within a planned Sprint, they should be appropriately sized. If a user story is too large, it can be decomposed into smaller stories. However, lower-priority stories in the Backlog can remain larger until they are prioritized for an upcoming Sprint.

  - Testable : A test can be created to verify whether the user story meets its acceptance criteria.

- **Epic :** represents a group or collection of interconnected user stories. They help organize the project by tracking larger, loosely-defined ideas, while user stories represent smaller units of work derived from end users or customers.

b) The Sprint Backlog

The sprint backlog serves as the team's to-do list, representing the part of the product backlog they will work on during the sprint. It is further divided into tasks for development, testing, and documentation. The product owner assists in creating the sprint backlog during the sprint meeting. The sprint backlog is a dynamic document that can be refined and modified by the scrum team, with regular discussions and updates during the sprint.

During our internship, we used Redmine, a flexible project management web application, to define our Sprint Backlog.

c) The Product Increment

The product increment is a crucial scrum artifact, representing all completed product backlog items within a sprint. It must align with the team's definition of done and meet the product owner's acceptance criteria.

The definition of done is a shared understanding within the scrum team, evolving as the project progresses, and ensuring the quality and completeness of each increment.

The product increment provides transparency to both the team and stakeholders, showcasing the cumulative value of completed increments from past sprints, reflecting the current state of the product.

1.5 CONCLUSION

In conclusion, following the Scrum methodology, the main focus of the report is working with Graylog, a powerful log management platform that offers a centralized log management

experience for IT, Network, and DevOps professionals, to accomplish a graduation project within Sifast company.

By the end of the report, readers will have a thorough understanding of Graylog, its features, and its advantages over the ELK Stack. This knowledge will be invaluable for organizations looking to improve their log management capabilities and enhance their overall system monitoring and management processes.

Chapter

2

1^{st} SPRINT - Research & Exploration

Contents

2.1 INTRODUCTION

Whether we are developing a new product, service, or process, research and exploration are critical components of any project or initiative, particularly when it comes to technology and innovation. In this chapter, we will delve into the technical and functional aspects of our Graylog project.

2.2 TECHNICAL ASPECT

In this section, we will explore the technical aspect of the project, which refers to the specific technological components, features, and specifications that are involved in its design, development, and functionality, discussing the various tools, and technologies employed to capture, store, and analyze log data.

2.2.1 Working Environment

To ensure optimal functioning and efficiency, the working environment is crucial to any project's success. In this subsection, we will examine its key components for our project.

2.2.1.1 Hardware Environment

To implement the project, we used two virtual machines :

a) 1st Virtual Machine : Server

- RAM Capacity : 8GB
- CPU : 4 with 4 cores each
- Hard disk : 100 GB
- Operating system : ubuntu 16.04

b) 2nd Virtual Machine : Client

- RAM Capacity : 4GB
- CPU : 2
- Hard disk : 100 GB
- Operating system : ubuntu 20.04

2.2.1.2 Software Environment

a) MobaXterm 

MobaXterm is a remote desktop solution, designed primarily for Windows, that provides users with terminal protocols such as SSH, VNC, RDP, or FTP and a complete Unix / Linux-like environment in a single portable exe file to access and manage jobs on remote servers and other remote computing resources.

MobaXterm is particularly useful for system administrators and IT professionals who need to manage multiple remote servers and computing resources from a single interface. It can also be helpful for developers and researchers who need to access and work with remote computing resources for their work.

In our project, we needed MobaXterm since the company provided us with the required resources allowing us to work remotely on their servers.

b) Web Browser

A web browser is a software application that allows users to locate, access, and display web pages on the World Wide Web (www), which is a system of interconnected web pages and other content accessible via the Internet.

The browser is responsible for interpreting the web page code, including HTML, CSS, and JavaScript, and rendering it as a visual representation for the user.

It serves as an intermediary between a client and a server to access and interact with web content and transfer data over the internet. In fact, when a user requests a web page through his browser, the browser sends a request to the web server, which then responds by sending the requested web pages back to the browser for display, facilitating HTTP(S) activity and other various protocols that are the foundation of internet use.

To access web pages, the browser uses URLs as traffic directions, utilizing IP addresses, port numbers, and other tools to establish connections.

In the context of our project, using a web browser, users can access the Graylog web interface and perform various tasks, such as :

- **Search and analysis :** The Graylog web interface allows users to search and analyze log data using a powerful search engine and advanced filtering options. Users can also create custom dashboards to visualize log data and gain insights into their systems' performance.
- **Monitoring :**Users can use Graylog to monitor logs and events from multiple sources in real-time using the web interface.
- **Administration :** Graylog's web interface provides users with a range of administrative functions, such as managing users and roles as well as configuring data sources.
- **Alerting** Users can set up alerts to notify them of specific events or trends.

In summary, a web browser is a crucial tool for using Graylog effectively. It allows users to access the Graylog web interface and perform various log management and analysis tasks efficiently and effectively.

2.2.2 Technical Choices

The technical choices made during the development process can have a significant impact on a project's functionality, usability, and scalability of a project, and in this subsection we will explore the various technical choices for our project and the reasons behind them.

a) Graylog 

Graylog is a highly powerful open-source log management system (LMS).
It is a versatile tool for collecting, analyzing, and alerting structured and unstructured log data from almost any source.

One of its standout features is its ability to efficiently parse petabytes of data, which involves converting data from one format to another. This feature simplifies the process of exploring data making it easier to manage and search through large and different amounts of log data and allowing users to visualize them in real-time dashboards through the web interface, to quickly and easily derive insights from their data and take prompt action.

Simply, whatever the use case, Graylog can assist businesses in analyzing their data more deeply and saving time and manpower, regardless of the specific use case. Therefore, it is a valuable tool for any business looking to streamline its data management processes and make data-driven decisions.

b) Elasticsearch 

Elasticsearch is a real-time distributed RESTful search engine, similar to Google Search but in the Big Data world. It excels in helping users locate desired content using keywords in millions of documents, making it the perfect tool for large-scale data searches.

It is also a NoSQL database that is designed for storing logs, developed in Java programming language. However, it is not a primary data store as there is no guarantee that your data will be correct.

Additionally, it is a real-time distributed data aggregation and analysis engine.
With Elasticsearch, users can easily aggregate data from multiple sources and analyze it in real-time, which helps businesses make data-driven decisions more effectively.

When talking about Big Data, Elasticsearch is designed to scale horizontally, allowing users to add servers (nodes) to a cluster to increase capacity and automatically distribute data as well as share the indexing and searching workload across all available nodes.

Therefore, Elasticsearch is a valuable tool for any business looking to streamline its data management processes and gain valuable insights from its data.

c) MongoDB

MongoDB is a powerful open-source NoSQL document database that stores data as JSON objects, making it perfect for handling both structured and unstructured data. It also has a flexible schema, which enables easy storage of data with varying structures.

Additionally, it is well-suited for big data applications as it can efficiently store and process large volumes of data. Moreover, MongoDB integrates with a wide range of big data tools and frameworks, including Hadoop, Spark, and Kafka, allowing data to be processed and analyzed using a variety of big data technologies. Besides, MongoDB is built on a horizontal scale-out architecture, ensuring high availability and performance, which is essential when working with Big Data.

Furthermore, MongoDB provides an excellent user experience, making it a popular choice for developers and organizations that need to manage and analyze large amounts of data.

In summary, MongoDB, as a database platform, provides numerous important features such as replication, sharding, ad-hoc queries, indexing, and real-time aggregation. It is designed to be scalable and flexible, with horizontal scaling and load-balancing capabilities.

d) Filebeat

Filebeat, an efficient log shipper tool developed by Elastic, can collect and forward log data from various sources to destinations like Elasticsearch, Logstash, and Graylog.

Due to its lightweight design, easy setup, and low resource consumption, Filebeat is a useful tool for logging and monitoring large volumes of data in real time.

The key features of Filebeat, include its ability to handle a wide range of log data types, its scalability and reliability, and its seamless integration with other tools.

2.3 Functional Aspect

In this section, we will examine the functional aspect of our project, which refers to the characteristics of the system that relates to its intended purpose or function, focusing on the Graylog architecture (see Figure 2.1) and how it supports the core functionalities of our logging and monitoring system.

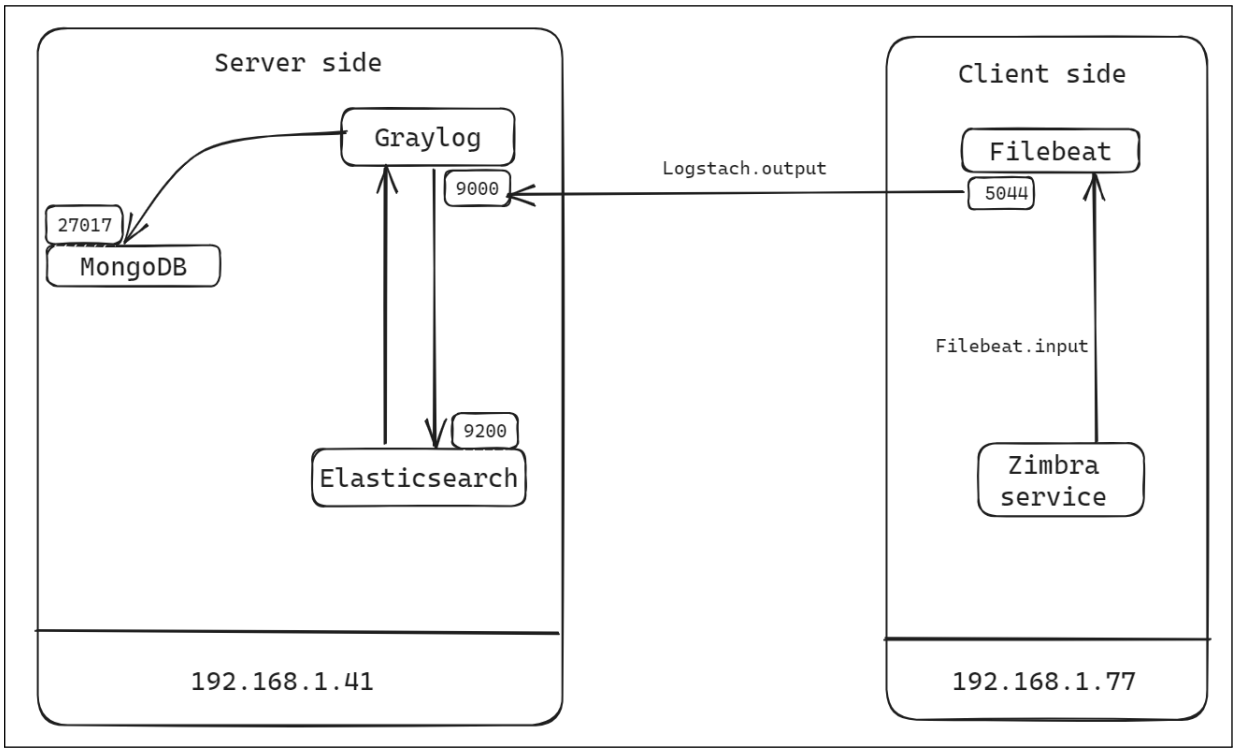

Figure 2.1 – Graylog Architecture

2.3.1 Elasticsearch utility with Graylog

When used as the primary data store for Graylog, Elasticsearch can provide benefits.

Thanks to this engine, Graylog can easily search logs using its robust search tools. It even enables advanced search functionalities like full-text search and fuzzy matching, simplifying finding relevant logs.

In addition to these benefits, Elasticsearch's ability to scale horizontally makes it an ideal solution for organizations dealing with high logging volumes. Moreover, it offers built-in support for data replication and backup, which ensures that log data are always accessible and recoverable in case of a system failure.

2.3.2 MongoDB Utility with Graylog

Graylog, as a log management platform, uses MongoDB to store its configuration data such as user information and the web interface entities, including stream configurations, alerts, users, settings, and cached stream counts. However, we should keep in mind that it is not used for storing log data.

2.3.3 Filebeat Utility with Graylog

Filebeat is utilized in conjunction with Graylog to streamline the process of forwarding and centralizing log data which means enabling log files to be sent to the Graylog server.

The most straightforward approach to set up Filebeat with Graylog is to create a Beats input on the Graylog server. Once the input is established, Filebeat can be installed and configured on any system from which logs need to be pushed to say the client machine.

2.4 NON-FUNCTIONAL ASPECTS

Non-functional aspects, often called "quality attributes" or "qualities of a system", are constraints or criteria that can be used to judge the operation of a system, rather than its functioning standards which it was designed for. This means they define the quality of the system behavior.

In our context, Graylog, the log management system, has several non-functional aspects which include :

- **Pricing :** Graylog comes with different pricing options, including an open-source version that is free and consists of the most features available, and the two paid products : "Graylog Operations" and "Graylog Security" which add some advanced features designed mainly for DevOps and security teams.
- **Scalability :** Graylog is able to handle a growing volume of logs and can scale horizontally across multiple nodes, one serves as the master with the rest as workers, ensuring that its performance remains consistent as the system expands.
- **Security :** Graylog provides robust security features, including authentication, authorization, and encryption, to protect log data and prevent unauthorized access.
- **Integration :** Graylog could integrate with other systems and tools, such as data shippers and storage systems.
- **extendability :** Graylog allows for plugins enabling users to customize and extend the platform to meet their specific needs.
- **Maintainability :** Graylog is designed with ease of maintenance in mind, allowing for straightforward updates, upgrades, and troubleshooting.

These non-functional aspects are essential for ensuring that Graylog can handle large amounts of data from multiple sources effectively and for guaranteeing its power to enterprises.

2.5 CONCLUSION

By examining the technical, functional, and non-functional aspects, we can gain a comprehensive understanding of the Graylog project and its impact on log management. This chapter drew on how Graylog, Elasticsearch, and MongoDB work together to store, search, and analyze logs coming from a log shipper such as Filebeat, and the results are displayed in the Graylog GUI. This brings us to the issue of their installation process and compatibility considerations that we will discuss in the next chapter.

Chapter

3

2^{nd} SPRINT - Installation & Configuration

Contents

3.1 INTRODUCTION

Graylog can be installed in different ways, allowing the user to choose. This chapter will provide a comprehensive guide to its installation and configuration according to our project. We will start by discussing the system requirements and prerequisites for the installation process, followed by a step-by-step guide to the installation and configuration of Graylog, and how to set up inputs to receive system logs. By the end of this chapter, it will be clear how to set up an environment for managing and analyzing log data using Graylog.

3.2 INSTALLATION PREREQUISITES

A support matrix depicted in the accompanying table 3.1, also known as a supportability matrix, is a tool used in the field of software development to identify which operating systems, databases, hardware, platforms, and other components are supported by a particular software product. The utility of a support matrix is to provide a clear understanding of what configurations are guaranteed to work with a software product and to help customers and support teams troubleshoot issues that arise. To choose the compatibility between versions of Graylog, JDK, Elasticsearch, and MongoDB, we referred to the support matrix provided by each of them and obtained ours. In this array, we bring together the compatible versions we worked with :

Component	Version	Support Status
Graylog	2.4.7	Supported
Elasticsearch	5.6.16	Supported
MongoDB	3.6.23	Supported
JDK	1.8.0	Supported
Ubuntu	16.04	Supported
Web Browsers	Google Chrome	Supported
Filebeat	5.6	Supported

TABLE 3.1 – Support Matrix of Graylog

3.3 GRAYLOG INSTALLATION AND CONFIGURATION

3.3.1 Server side

3.3.1.1 Step 1 : Graylog Installation

Graylog is a server-side application that receives, processes, and stores log messages. We need to install the Graylog server on a machine or virtual machine that meets the minimum requirements, such as a Linux server with at least 4GB of RAM and 2 CPU cores. Besides, we can install Graylog on a variety of platforms such as Ubuntu that we worked with in our project. Furthermore, Graylog requires MongoDB and Elasticsearch to be able to run. That's why Before installing Graylog, we should go through

a) Java installation

Java is critical for Graylog and Elasticsearch to run. Infact, Graylog and Elasticsearch are 2 apps written in Java ; which means they run on the Java virtual machine JVM.
Notice : The Graylog installer checks if Java is already installed on the system, else it automatically installs a compatible version of java to allow Graylog to work properly

b) Elasticsearch installation

Elasticsearch acts as a database that stores data from Graylog, and as a search engine just like Google Search in the Big Data world. For that, it is critical to go through the installation of Elasticsearch to enable Graylog to manage logs.

c) MongoDB installation

Before moving to Graylog Installation, we should install MongoDB as it stores Graylog's metadata.

Now, we can successfully install Graylog.

3.3.1.2 Step2 : Graylog Configuration

The Graylog Configuration is supposed to be done through the Graylog "server.conf" and Elasticsearch "elasticsearch.yml" files which are both configuration files that state how Graylog can work. users can use a text editor (such as nano) to open the file and modify entries that are of the form "propertyName=propertyValue" or "propertyName :propertyValue".

a) "server.conf" File :

Once the server is installed, users need to configure Graylog to meet their specific needs. First, we should edit the "server.conf" file to set basic server settings such as the server's IP address and port number. The "server.conf" file is typically located in the "/etc/graylog/server/" directory on Linux systems. The important parameters we need to configure include

- **root_username=**

  The default root user in Graylog is typically 'admin'. When configuring the Graylog login credentials, you would set this parameter to 'admin' .

- **password_secret=**

  It is a secret string used to encrypt user passwords. This should be a long (at least 64 characters), random string.

- **root_password_sha2=**

  SHA-2 which stands for Secure Hash Algorithm 2 is a family of cryptographic hash algorithms. Thus, the root_password_sha2 is a hashed version of the root password using the SHA-2 algorithm. Here, the root password is used to log in to the web interface.

- **http_bind_address=**

  It is a configuration parameter used to specify the IP address and port number on which

the Graylog web interface should listen for incoming HTTP requests. it can be set to :

$$@ - ip - machine : 9000$$

- **rest_listen_uri=**

  It is a configuration parameter used in Graylog to specify the URI of the REST API interface and it needs to be properly configured to ensure that the REST API interface is accessible from other nodes and clients. The REST API is a web service that provides access to the Graylog server's data and functionality.

- **web_listen_uri=**

  It is a configuration parameter used in Graylog used to specify the URI of the web interface and it needs to be properly configured to ensure that the web interface is accessible from other nodes and clients. The web interface needs to connect to the REST API using HTTP(S) to fetch the data it needs to display.

  Notice : The web interface is responsible for displaying the data in a user-friendly way, while the REST API is responsible for providing the data in a machine-readable format. The web interface and the REST API are two separate components of Graylog, and they communicate with each other using HTTP(S)

  Notice : You can choose between two ways to make its web interface accessible. The first option involves running both the REST API and the web interface on the same port, but using different paths. By default, the REST API is accessed at

  $$http : //@ - ip - machine : 9000/api/$$

  while the web interface is accessed at

  $$http : //@ - ip - machine : 9000/$$

The second option involves running the REST API and the web interface on different ports, such as :

$$http://@-ip-machine:12900/$$

for the REST API and

$$http://@-ip-machine:9000/$$

for the web interface.

- **elasticsearch_hosts=**

  It is a configuration parameter used in Graylog to specify the list of Elasticsearch hosts that Graylog should connect to. This parameter needs to be set as a comma-separated list of valid URIs for the HTTP ports of the Elasticsearch nodes and to be properly configured to ensure that Graylog can connect to the Elasticsearch cluster. If one or more of your elasticsearch hosts require authentication, include the credentials in each node URI that requires authentication :

$$http://node1:9200, http://user:password@node2:19200$$

b) "elasticsearch.yml" File :

The Elasticsearch cluster or nodes are used by Graylog to store and search log data, and it needs to be properly configured to ensure that Graylog can access them. That's why, users should edit, using a text editor, the elasticsearch.yml file which is typically located in the "/etc/elasticsearch/" directory on Linux systems.

- **network.hosts=**

  By default, Elasticsearch is only accessible on localhost. So, we should set a different address here to expose the node on the network.

- **http.port=**

  users can modify the port that Elasticsearch will listen on. However, it is recommended to use the default value which is 9200.

3.3.1.3 Step3 : Graylog web interface

Once Graylog is successfully running, its web interface can be accessed through the web browser. After entering the URL, you will be directed to the Graylog login page 3.1. Here, you can enter your login credentials as the username and the password pre-configured in "root_username" and "password_secret" parameters . Upon clicking the "sign in " button, if the provided credentials are correct, you will gain access to the Graylog web interface, enabling you to explore its functionalities and manage your logs effectively.

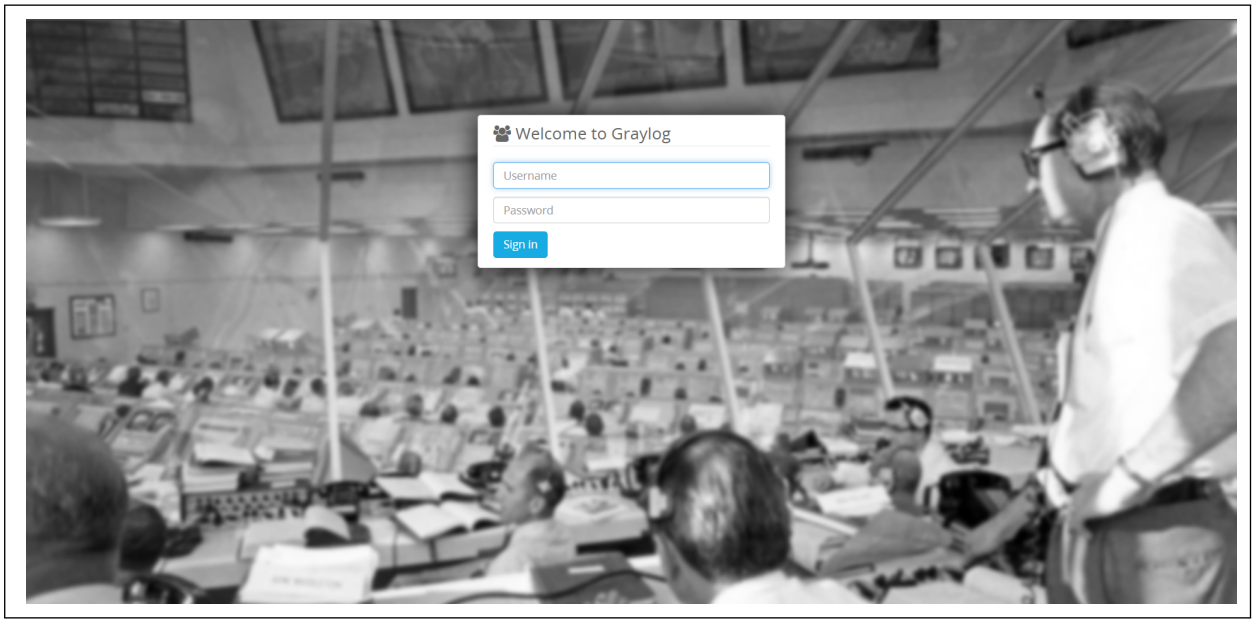

FIGURE 3.1 – Graylog Login Page

These are the general steps for installing Graylog. However, they may vary depending on your operating system and the version of Graylog you're installing. It's important to consult the official documentation and follow the instructions carefully.

3.3.2 Client side

When it comes to monitoring systems and applications, logging is an essential part of the process. Logs provide valuable information that can be used to troubleshoot issues, identify trends, and improve overall system performance. On the client side, we are going to log the Zimbra-server. So, we established the client server using a ubuntu Virtual, on which we installed

Zimbra-server and we need a log shipper like filebeat to read and collect its logs then send them to Graylog.

3.3.2.1 Step 1 : Filebeat Installation

Filebeat can be used to forward logs from Zimbra to graylog by configuring it to read. However, first, we should download and install it.

3.3.2.2 Step 2 : Filebeat Configuration

When configuring Filebeat to collect logs from the Zimbra server, you need to specify the location of the log files and define the type of logs you want to collect and the output. Here are the detailed steps :

a) "filebeat.yml" File

Under the Filebeat prospectors section we edit the following settings

- **Input_type :**

  It is a configuration parameter that specifies the type of input that Filebeat should use to collect log data. The input_type parameter can be set to log, stdin, syslog, filestream, tcp, udp, or beats.

- **Paths :**

  It is a Configuration parameter that specifies the paths (the location) of the log files that Filebeat should read. The paths parameter can be set to a single file or a list of files. According to our use case, it should be set to "/var/log/Zimbralog/zimbra.log"

Under the Logstash output section we edit the following setting :

- **Hosts :**

  This configuration parameter specifies the IP address and port number of the remote machine running Graylog so it can receive the logs :

$$[\text{"@} - ip - machine : 5044\text{"}]$$

3.3.2.3 Step 3 : Create Inputs

To create an input the user should log in to the graylog web interface.

Inputs are a crucial part of Graylog's logging infrastructure, as they define how logs are received from various sources, such as servers, applications, and devices. By creating an input, we can configure Graylog to receive logs from a specific source using a specific protocol, format, and port. The figure 3.2 details the steps of inputs creation.

First, we select the type of input and launch new input. Then, we need to configure it.

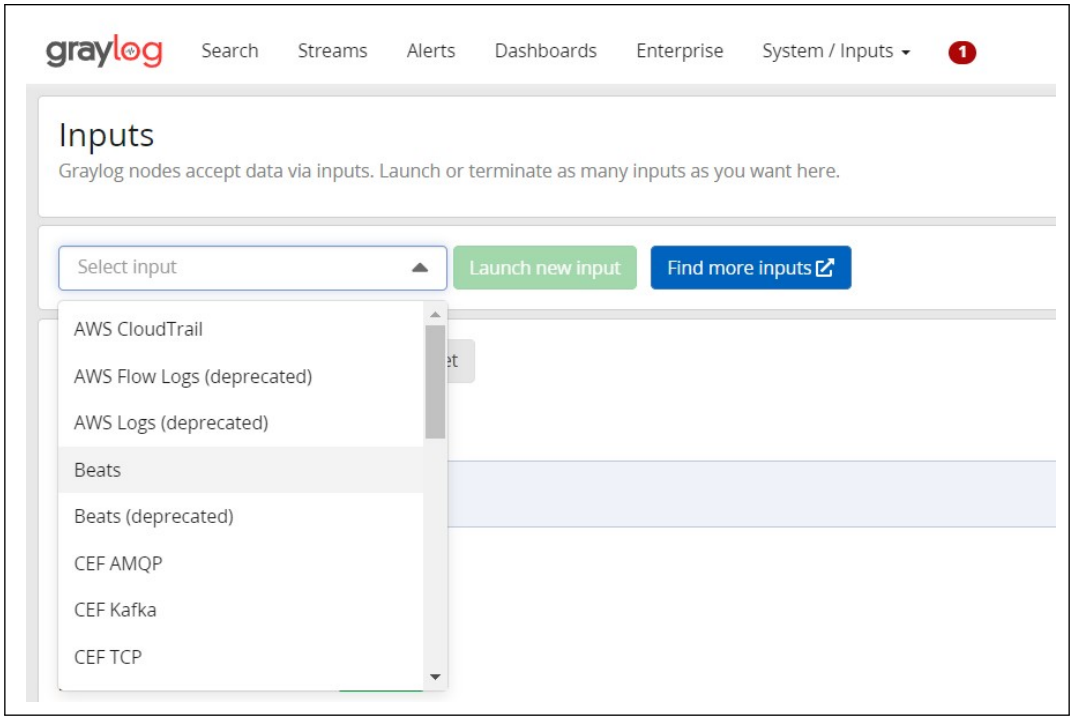

FIGURE 3.2 – Graylog input creation

3.3.2.4 Step 4 : Configure Inputs

To configure an input we should set some configuration parameters as shown in the figure 3.3

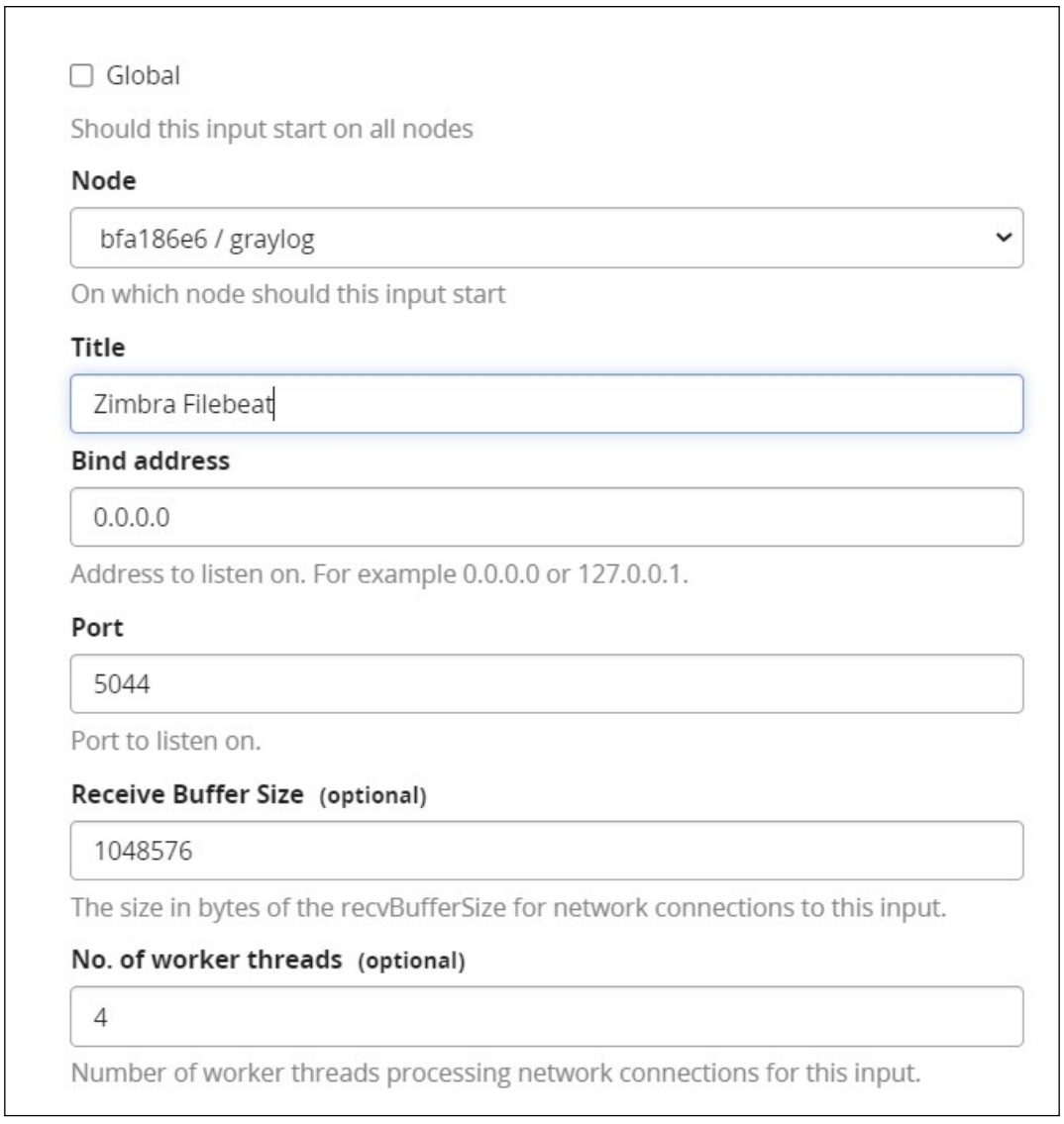

FIGURE 3.3 – Graylog input configuration

- **Port :** This consists of the port number on which the Filebeat messages will be received. The commonly used port number in log collection and management is 5044. It is used by Logstash/Graylog to receive logs from various sources using the Beats protocol.
- **Bind Address :**This consists of the IP address of the network interface to which the input is bound. The default bind address is 0.0.0.0 which means that the input is bound to all available network interfaces on the system.
- **Receive Buffer Size :** This consists of the size of the buffer used for receiving log messages. The default size is 1048576 bytes.
- **Number of Threads :** This consists of the number of threads used for processing incoming log messages. The default number of threads is 4.

Once we have created and configured the input, we can now start receiving logs from the source system that is configured to send logs to the input port using a specific protocol. To view the incoming logs, you can navigate to the "Search" menu and perform a search query based on the log messages' contents.

3.4 CONCLUSION

In conclusion, With the appropriate hardware and software prerequisites in place, Graylog can be a valuable asset for organizations seeking to optimize their log management process. However, After the installation and configuration of Graylog, everything can be done through Graylog web interface without the need of using the CLI.

Chapter

4

3^{rd} SPRINT - Log Analysis

Contents

4.1 INTRODUCTION

Log analysis is a critical part of any IT infrastructure, and Graylog provides a powerful platform for log data handling. In this chapter, we will explore Graylog's core manipulation, exploration, visualization, and alerting capabilities, which provide a solid foundation for effective log data handling.

4.1.1 Graylog Search Component

4.1.1.1 Search Concept

Searching in Graylog refers to the process of querying log data collected by the Graylog server. For this, Graylog provides a web-based search interface that allows users to search for log messages based on various criteria such as timestamp, source, message text, severity level, and other metadata fields. Here, sers can construct search queries using the Graylog Query Language (GQL), a flexible and powerful syntax that supports a wide range of search operators, wildcards, and regular expressions.

Deeper, for log data, Graylog can be viewed as the frontend, and Elasticsearch as the backend in performing search operations. In fact, when Graylog receives a log message, it first indexes this data in Elasticsearch meaning it breaks down the message into individual fields, such as timestamp, log level, and message content, and stores these fields in Elasticsearch as separate documents. Then, when a user queries Graylog, it will send the search request to Elasticsearch, which will perform the search operation based on the specified criteria and returns a set of matching documents. At this time, Graylog displays the search results to the user.

4.1.1.2 Search Syntax

To search for a specific message, users can enter search queries in the search field. They can also filter and sort search results.

By default, all message fields are included in the search if you don't specify a message field. However, the Graylog search query language allows users to search for specific log messages based on various criteria such as time range, message content, and source through a wide range of operators and functions.

The syntax is straightforward and consists of field names, operators, and search terms. Here are some examples of search syntax :

- **Simple Term Matching :**

$$field - name : search - term$$

  Matches logs where the specified field contains the search term.

- **Wildcard Matching :**

$$field - name : search - term*$$

  Matches logs where the specified field starts with the search term.

- **Logical Operators :**

$$field - name1 : term1 \quad AND \quad field - name2 : term2$$

  Matches logs where both the specified fields contain the corresponding terms.

$$field - name1 : term1 \quad OR \quad field - name2 : term2$$

  Matches logs where either of the specified fields contains the corresponding terms.

- **Range Queries :**

$$field - name : [value1 \quad TO \quad value2]$$

  Matches logs where the specified field falls within the specified range.

- **Field Existence :**

$$exists : field - name$$

Matches logs where the specified field exists.

- **Negation :**

$$NOT \quad field - name : search - term'$$

Matches logs where the specified field does not contain the search term

- **Regular Expressions :**

$$field - name : /regex - pattern/$$

Matches logs where the specified field matches the regular expression pattern

4.1.1.3 Search Examples

The figures presented below showcase the search section of Graylog, along with various examples of search queries.

a) Overview of the search section

The figure 4.1 shows the components of the search section, such as the histogram, the Zimbra server log messages- we used in our project- and the field filters in the side bar.

b) The first search example

The figure 4.2 shows how to search for messages that contain a specific string which represents here "mail system is down"

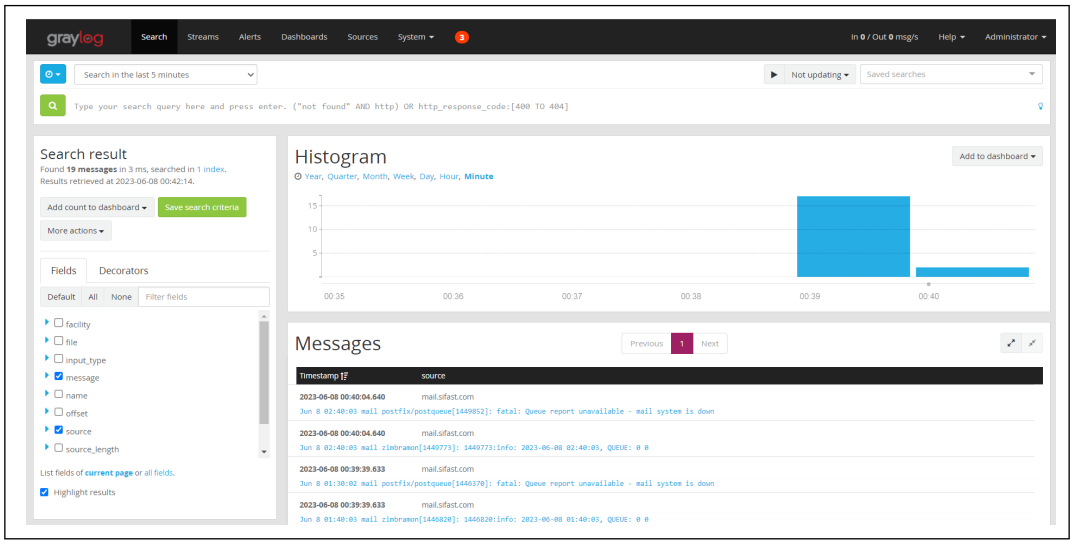

FIGURE 4.1 – Search section

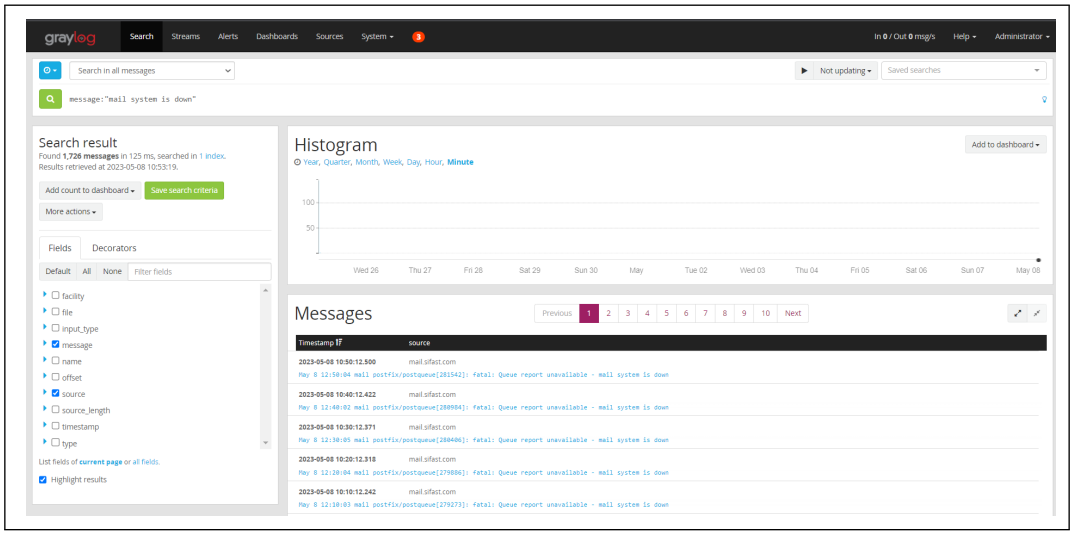

FIGURE 4.2 – String Search

c) The second example :

The figure 4.3 shows how to use regular expressions (regex) in Graylog search queries. The figure illustrates how to search for messages containing a specific pattern in the message field :

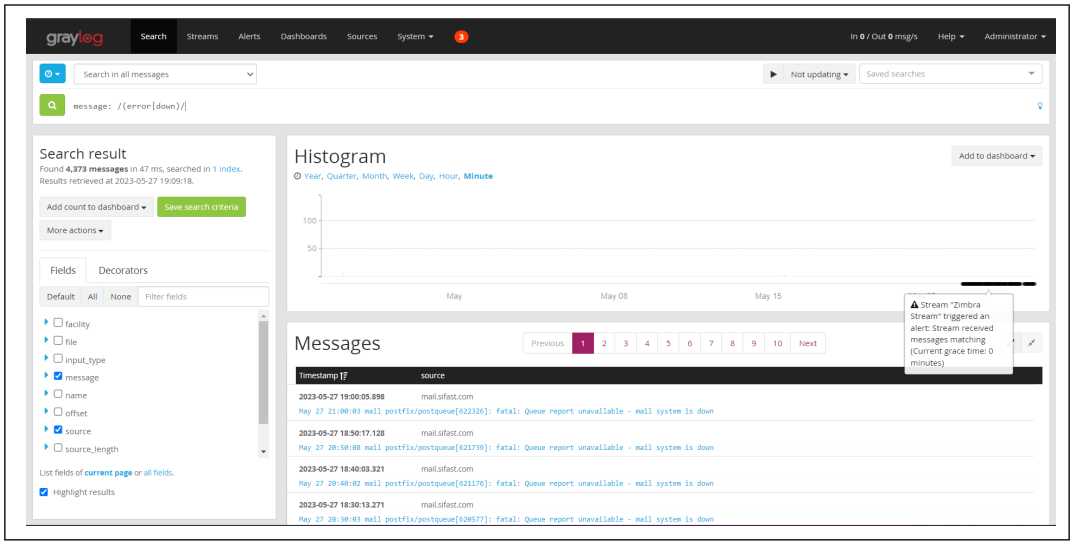

FIGURE 4.3 – Regex Search

4.1.1.4 Time frame selector

The time frame selector in Graylog offers three ways to define the search time range.

- **Relative :**

  The relative option allows searching from a selected time until the present. This option is useful in case of searching a recent events or monitor real-time data.

- **Absolute :**

  The absolute time frame selector enables specifying specific start and end times, and the keyword option uses natural language phrases for time frames. This option is helpful when you need to investigate events within a specific time period.

- **Keyword :**

  The keyword option uses natural language phrases for time frames. The web interface will display a preview of the actual timestamps that will be used for the search based on the chosed keyword.

Using the appropriate time frame selector and values in Graylog enables efficient searches, reduces system load, and provides flexibility in defining the time range for log analysis.

4.1.1.5 Saved Searches

Saved searches are features commonly used found in search interfaces that allow users to save their frequently used or preferred search queries for quick access in the future. This feature helps users avoid repetitive typing and simplifies the process of retrieving search results for specific topics of interest.

4.1.1.6 Histogram

The search page in Graylog includes a histogram -shown in figure 4.4 - that displays the number of messages received, grouped by an automatically adjusted time period. This visual representation helps users understand the distribution and frequency of search results.

Furthermore, the histogram also visualizes annotations for alerts triggered within a specific stream, allowing users to quickly identify and analyze time intervals associated with specific events or issues.

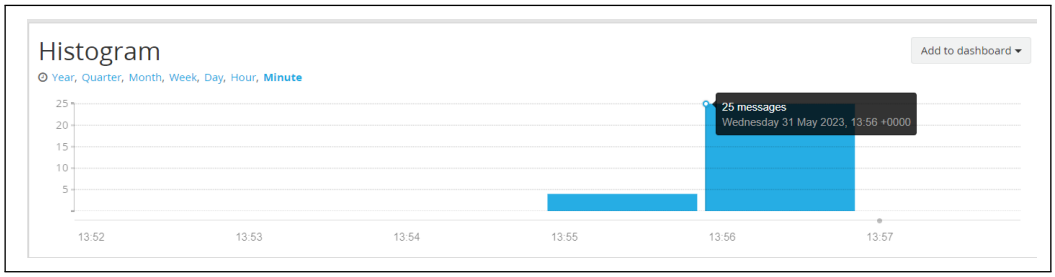

FIGURE 4.4 – Search Histogram

4.1.1.7 Search steps

Searching for data includes several steps :

1. Log in to your Graylog account and navigate to the "Search" tab.
2. In the search bar, enter the query you want to search for.
3. Use the drop-down menus to refine your search criteria.

4. Click the "Search" button to execute the search. Graylog will display the search results in the main window.
5. To view more details about a specific log entry, click on it in the search results. This will expand the entry and display additional information, such as the full log message and any associated metadata.
6. To save your search, click the "Save" button and give your search a name. You can then access your saved searches from the "Search" tab.
7. Finally, to export your search results, click the "Export" button and select the format you want to export to, such as CSV or JSON. Graylog will generate a file containing the search results that you can download and save for future analysis.

4.1.2 Widgets

The Graylog provides widgets as visual tools to analyze log data. These widgets allow you to perform various analyses on your search result and save them into dashboards for convenient access. By expanding a field in the search sidebar, you can utilize the available widgets to generate charts, visualize quick values, display statistics, and plot data on a world map.

- **Generate Chart :** Enables you to create visual charts and graphs that depict your log data in a clear manner. With various chart types available, you can easily identify trends, compare data, and visualize distributions within your logs.

  By expanding the source length and hitting the button “Generate chart “ we obtain a surface graph shown in Figure 4.5, presented with linear interpolation shows the sum of logs messages received by Graylog weekly and containing the fields’ source-length. The horizontal axis represents the weeks, and the vertical axis is the sum of recovered messages.

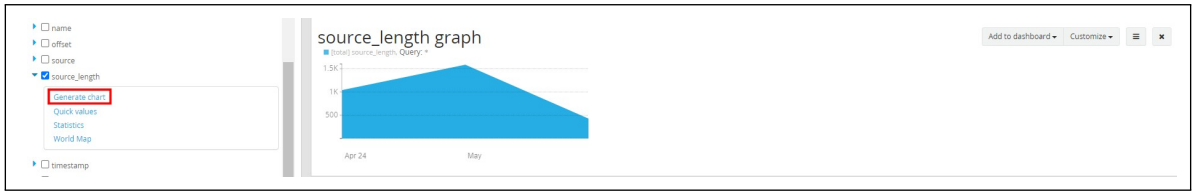

FIGURE 4.5 – Generate Chart Example

- **Quick Values :** Provides a concise summary of different values present in a specific field of your log data. Presenting a table with value frequencies allows you to quickly identify the most common values and their occurrences.

  The visualization 4.6 presents a clear representation of the log message distribution based on the timestamp field in the form of a pie chart. This chart provides a visual breakdown of the proportionate distribution of log messages across various time intervals. Additionally, a table accompanies the pie chart, highlighting the top five dates when Graylog received the highest number of messages. This table includes the percentages and counts of these dates, relative to the total number of messages received.

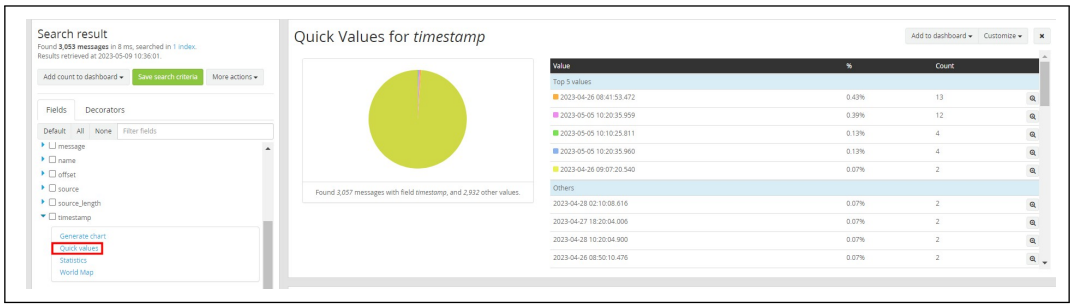

FIGURE 4.6 – Quick value Example

- **Statistics :** Gain valuable insights by calculating statistical metrics for a specific field in your log data. It provides useful information such as counts, averages, sums, and minimum/maximum values, giving you a quantitative understanding of your log events.

  The example in 4.7 figure displays statistical calculations for two fields : "source-length" and "timestamp." These calculations provide information about the messages received that contain these fields, including the total count, mean value, minimum and maximum values, and the sum of the values within these fields.

  Regarding the "source-length" field, since we are working with a single field that has the same length as its name, its variance is null. This is because there is no variation in the values of this field, resulting in cardinality of 1. In contrast, the "timestamp" field exhibits more diverse values, leading to non-null standard deviation and variance values. The cardinality of the "timestamp" field reflects the variety of timestamps observed when log data is received by Graylog.

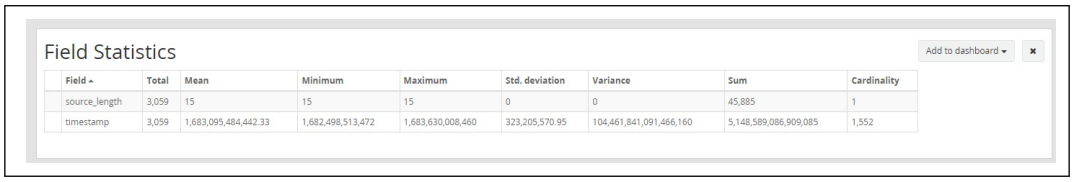

FIGURE 4.7 – Statistics Example

- **World Map :** Displays a geographic visual representation of your log data. By plotting log events on a map, you can identify geographic patterns, hotspots, or anomalies in your data.

  Notice : About the world map widget, we do not have geolocation information in log messages, such as IP addresses, latitude-longitude coordinates, or any other format that can be used to determine the location and present them in map.

These analysis options provide valuable ways to explore, summarize, and visualize log data

4.1.3 Dashboard

4.1.3.1 Dashboard Definition

In Graylog, a dashboard is a powerful tool for visualizing and summarizing log event information. It allows users to create customized visualizations that showcase various metrics and trends on a single page as depicted in the figure 4.8 . Dashboards consist of widgets that represent log data through charts, tables, and summaries based on field values. Users can drill down into the data from these widgets to explore details, analyze, and understand log event data.

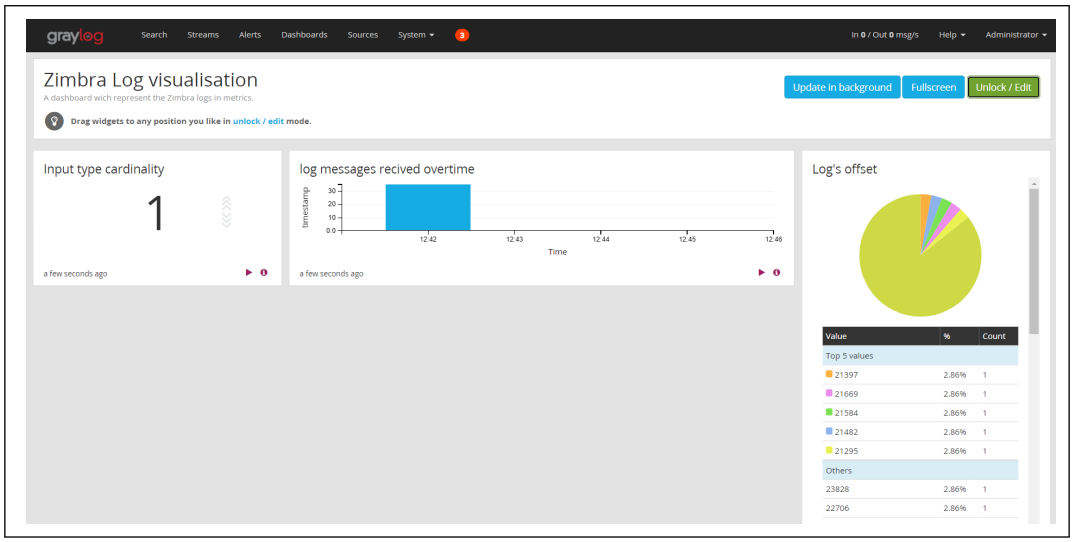

FIGURE 4.8 – Dashboard example

4.1.3.2 Dashboard Creation

To create a dashboard, the user should follow these steps :

1. In the top navigation bar of the Graylog web interface, click on "Dashboards".
2. Click on the "Create a dashboard" button to start creating a new dashboard.
3. Give your dashboard a title and description.
4. Add the widgets you created from the search section.
5. Customize the layout of the dashboard by rearranging and resizing the widgets as needed.
6. Save the dashboard by clicking the "Save" button.
7. You can optionally share the dashboard with other users or assign permissions to control access.
8. Once the dashboard is saved, you can finally view it by clicking on the "Dashboards" menu and selecting your dashboard from the list.

4.2 DATA MANIPULATION

Graylog comes with several features that allow manipulating data. these features include Streams, Pipelines, Extractors, and Alerts.

4.2.1 Streams

4.2.1.1 Streams Definition

Streams are a logical grouping of messages that are routed based on certain criteria such as the source, the type, or keywords in the log message.

4.2.1.2 Stream Rule Creation

When setting up a stream, you establish criteria called rules that dictate which messages are eligible for inclusion in that particular stream. By following the next steps, we can define a stream rule :

1. Log in to the Graylog web interface.
2. Navigate to the "Streams" section.
3. Click on the "Create stream" button to create a new stream.
4. Give the stream a name and a description.
5. Define the stream rules by selecting the "Add rule" button.
6. Select the "field" against which you wish to match, such as the message, source, or timestamp. This indicates that you want to compare the incoming message with the content of the designated message field.
7. Specify the "condition" that the field must satisfy, such as including a particular string or conforming to a regular expression pattern.
8. You have the option to input the "value" that the field should match, whether it be a specific string or a regular expression.
9. You have the flexibility to negate the condition, allowing you to specify whether the rule should "match all" or any of the defined conditions.
10. Save the stream rules.
11. Enable the stream.

The figure 4.9 shows the configuration of one stream from our project.

After configuring a stream, Graylog will automatically direct messages that meet the stream's defined rule into that stream in real-time during processing. This enables you to easily view all messages that fall under a particular category without the need for manual sorting. Subsequently, these streams can be utilized to search, filter, and analyze your log data (see Figure 4.10).

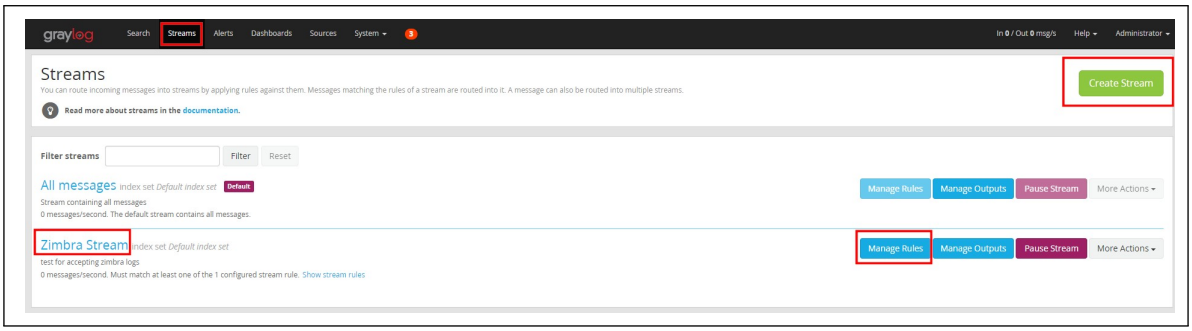

FIGURE 4.9 – Zimbra Stream

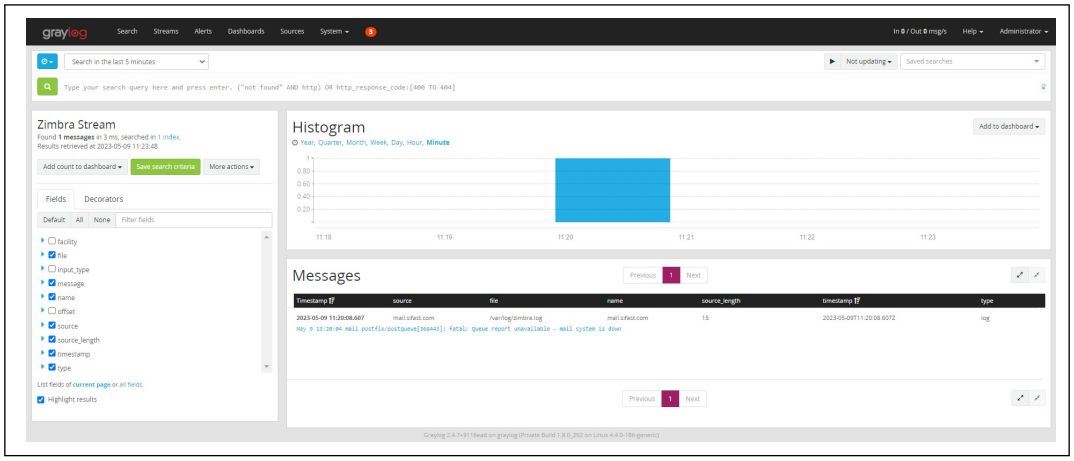

FIGURE 4.10 – Stream Messages Display

4.2.2 Extractors

Graylog's extractor feature allows users to parse and extract specific fields from log messages, enabling the identification of relevant information for data exploration and analysis. By defining extraction rules, users can separate and store desired fields for further manipulation and visualization. Graylog provides different types of extractors tailored to specific data extraction needs as depicted in the figure 4.11.

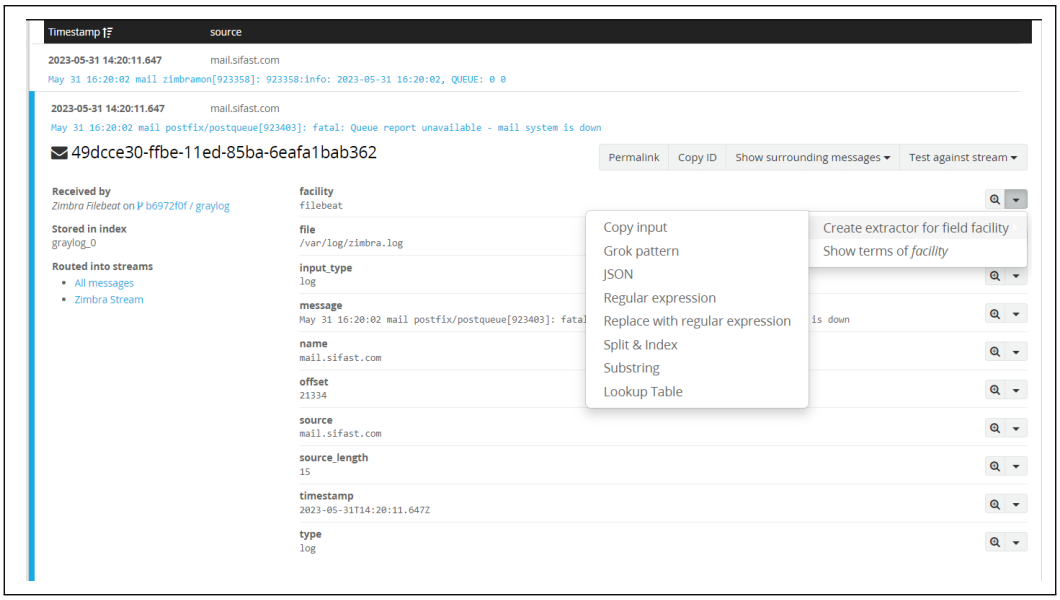

FIGURE 4.11 – Extractor

Copy Input Extractor : This extractor allows you to copy the entire log message and store it in a separate field. It is useful when you want to preserve the original log message while applying other extractors.

Grok Pattern Extractor : This extractor uses the Grok pattern syntax to match patterns in log messages and extract specific fields. It is particularly useful for structured log formats.

JSON Extractor : This extractor is designed to extract fields from log messages that are in JSON format. It can automatically parse and extract fields from JSON log messages.

Regular Expression Extractor : This extractor allows you to define custom regular expressions to match patterns in log messages and extract fields. It provides flexibility in extracting fields based on specific patterns.

Replace with Regular Expression Extractor : This extractor enables you to find and replace specific patterns or values in log messages using regular expressions.

Split and Index Extractor : This extractor allows you to split a log message into multiple parts using a delimiter and extract fields from those parts. It is useful when log messages have a hierarchical structure.

Substring Extractor : This extractor extracts fields from log messages based on fixed start and end positions. It is useful when log messages have a consistent format, and the desired fields are in fixed positions.

Lookup Table Extractor : This extractor maps the values of a field to new values based on a lookup table. It is useful for transforming specific field values into more meaningful or standardized representations.

4.2.3 Pipelines

4.2.3.1 Pipelines Definition

Graylog pipelines are a central concept of the Graylog logging platform that enables processing specific message types. They can be linked to one or more inputs, streams, or outputs, and are composed of rules.

Processing Rules :

Rules are conditions followed by tasks that include filtering, parsing, appending, and deleting, among other possibilities.

Stages :

As processing rules cannot control their execution process themselves, Graylog pipelines rely on stages to control the sequence of rule actions. Stages contain sets of conditions and actions that must be executed in order, prioritized based on their addition, and rules within the same stage are executed at approximately the same time to work with data from other rules in the stage.

4.2.3.2 Pipeline rules Creation

As rules are the core concept of pipelines, it is crucial to dive into the way they are managed. The following items list steps of pipeline rules creation :

1. Log in to the Graylog web interface.
2. Navigate to the "Pipelines" page under the "System" section.
3. Click on the "Create Pipeline" button.

4. You can give the pipeline a "name" and "description".
5. Click on the "Add Rule" button which will open a page where you can define the rule's conditions and actions.

   notice;

   The basic syntax for creating a pipeline rule is as follows :

```
rule "Rule name"
when
  <condition>
  // if these conditions are met
then
  <action>
  // do these things
end
```

   Here's an overview about the meaning behind every keyword :

   (a) "rule" : is followed by the name of the rule, which can be any string.
   (b) "when" : is followed by a condition which can be any valid expression that evaluates to a boolean value.
   (c) "then" : is the beginning of the section where we define actions which can be any valid expression that modifies the message or sets a field value.
   (d) "end" : marks the end of the rule.
6. Navigate to the "Rule Configuration" section.
7. Click on "Add Stage" to add a new stage to the rule.
8. Navigate to the "Stage Configuration" section and click on "Add Condition" to add one or more conditions to the stage such as 'Field has value', 'Message contains', 'Regex match'.
9. If you want to add more stages to the rule, repeat steps 6-8.
10. Click on "save" to save the rule.
11. Once the pipeline is created, users can connect it to one or more streams

The following figures (4.12, 4.13, and 4.14) show a guide for the steps we have mentioned.

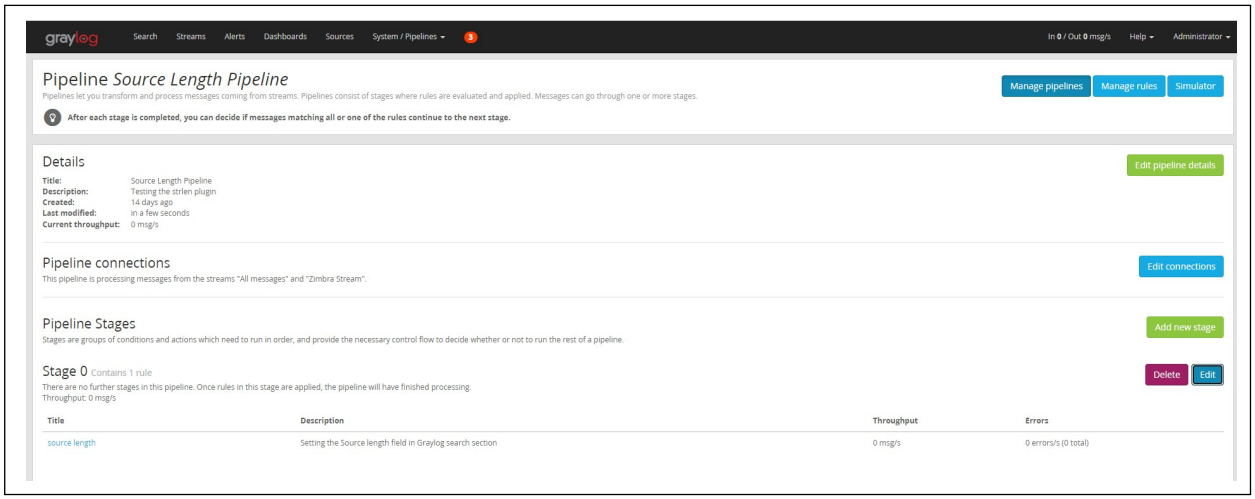

FIGURE 4.12 – Pipeline Creation

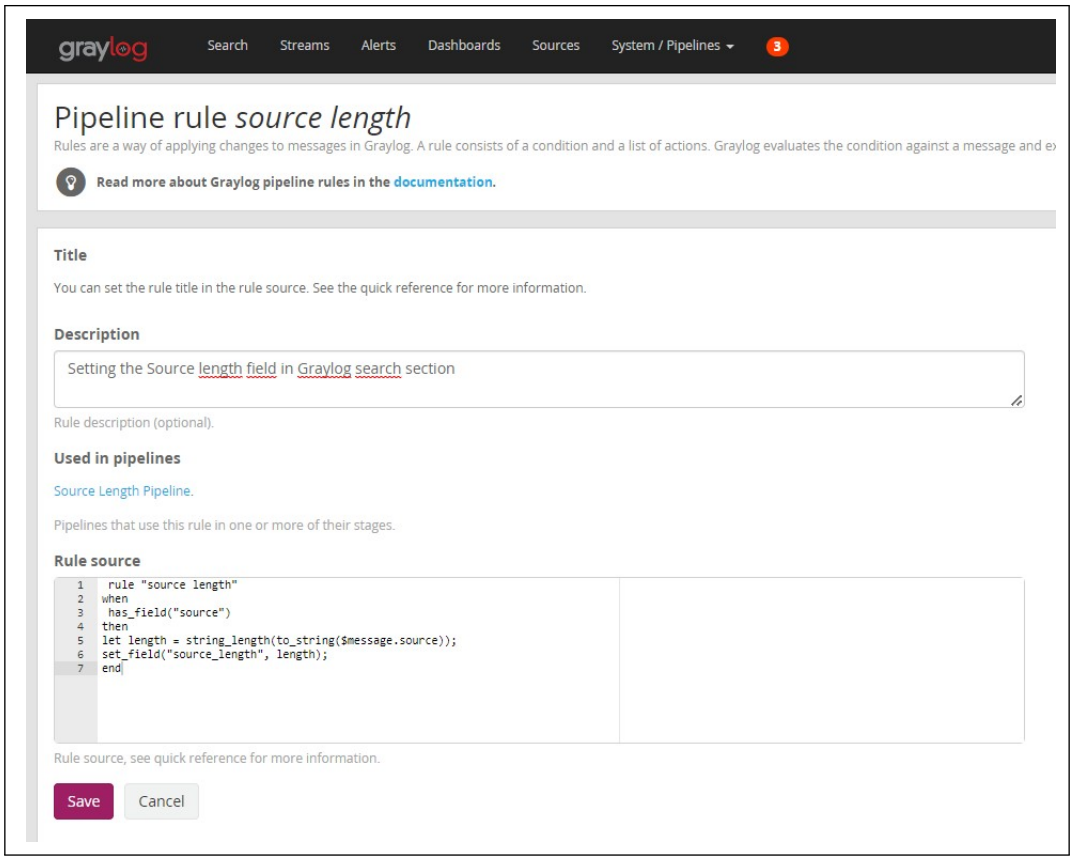

FIGURE 4.13 – Pipeline Rule Creation

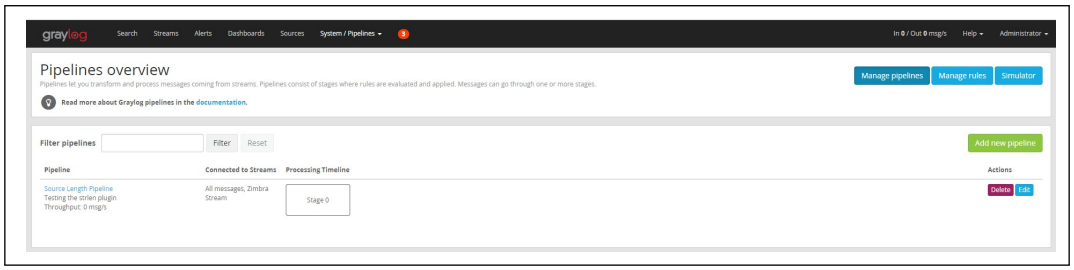

FIGURE 4.14 – Source Length Pipeline

Notice REST Method :

Alternatively, use the Graylog REST API to manage pipelines including updating, deleting, and retrieving pipeline information. For eg if you want to create a pipeline you can send a POST request to the

$$/api/plugins/org.graylog.plugins.pipelineprocessor/system/pipelines$$

endpoint with the pipeline definition in the request body which consists of the pipeline's name, description, stages, and rules.

4.2.3.3 Message Processing Simulator

The Message Processing Simulator is a Graylog feature that enables users to test their pipeline rules and preview the message processing result. To use the simulator, users can navigate to the pipelines section and click on "Simulator". However, some users have encountered problems with using the simulator, such as its limited usefulness for specific input types or extractor fields not being visible during simulation. Therefore, it is advisable to verify pipeline rules using actual messages in addition to utilizing the simulator.

4.3 ALERTS

4.3.1 Alerts Definition

Alerting is a mechanism that allows to monitor log messages in real-time and trigger notifications or actions when specific conditions or criteria are met based on log data streams. Moreover it help proactively identify important events or patterns in your log data and take appropriate actions to address them promptly.

4.3.1.1 Alert states

The alerts have two distinct states

- **Resolved Alerts :** Graylog automatically resolves alerts when the alert condition is no longer satisfied. This marks the final state of an alert, indicating that the triggering condition has returned to a normal state. After an alert is resolved, Graylog applies the grace period defined in the alert condition. During this period, Graylog waits for a specified duration before creating a new alert for the same condition. This approach allows for a controlled monitoring process, preventing excessive notifications and ensuring that alerts are only triggered when the condition persists beyond the grace period.
- **Unresolved Alerts :** Alerts enter an unresolved state when the defined condition is satisfied. This state indicates that the triggering condition for the alert is currently valid and requires attention. New alerts matching the same condition also enter the unresolved state, ensuring continuous monitoring. The figure 4.15 shows an example of an unsolved alert state

4.3.1.2 Alert components

a) Alert condition

An alert condition is a rule that determines when an alert should be triggered based on specific criteria or events detected in log data. Graylog provides different types of alert conditions that can be configured. Here are the three default alert condition types included

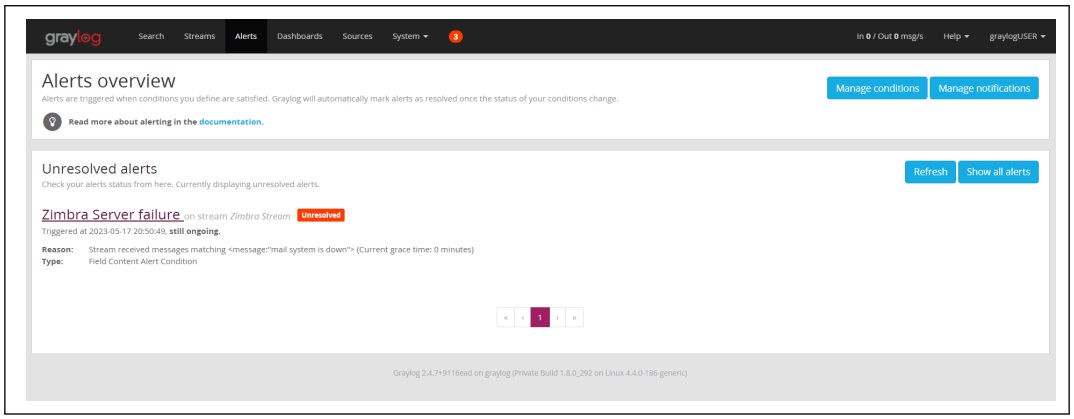

FIGURE 4.15 – Unsolved alert example

- **Message count condition** : This type of condition triggers an alert when the number of messages received in a stream exceeds a specified threshold within a defined time period. It helps monitor exceptional events or patterns in log data based on message volume.
- **Field aggregation condition** : This condition triggers alerts when a numeric field in the stream reaches a specified limit. It is particularly useful for monitoring performance issues by tracking metrics such as response time, CPU usage, or memory usage.
- **Field content condition** : This condition triggers an alert when at least one message in the stream has a specific field set to a given value. It enables you to detect specific events or patterns in the log data based on the content of a particular field.

b) Notifications

A notification is a message or alert that informs or updates users about a specific event, status, or occurrence. It play a critical role in allowing users to take appropriate action quickly.

- **Email alert Notification :** Graylog's email notification sends customized alerts via email when specific conditions are met in the log data. It allows you to define recipients, customize email content using templates, and include relevant information such as the triggered condition and associated messages. The figure 4.16 illustrates an email notification indicating the failure of the Zimbra server.
- **HTTP alert Notification :** Graylog's HTTP alerting triggers actions on external systems or services by sending a POST request to a specified endpoint. The POST payload contains

FIGURE 4.16 – Alert Notification

details about the alert, including the triggered condition and relevant message information. This notification facilitates integration with other systems for automated responses, so you can initiate remediation actions, or perform custom actions based on the events detected in your log data.

Notice : Graylog allows to extend both the alert conditions and alert notifications through Plugins, providing additional functionality and customization options.

4.3.2 Alerts Creation

4.3.2.1 Step 1 : Graylog server configuration

The first step in the process of creating an alert with Graylog is to configure the "server.conf" file precisely the email transport section. The critical configurations that need to be addressed include

- **Transport-email-enabled** :Set this option to true to enable the email transport in Graylog.
- **Transport-email-hostname** Specify the hostname or IP address of your SMTP server. This is the server that Graylog will use to send email notifications.

  Notice : SMTP, or Simple Mail Transfer Protocol SMTP, or Simple Mail Transfer Protocol is an essential protocol for sending email messages. It works on a client-server model and governs the transfer of email from the sender to the recipient. The sender's

email client communicates with the SMTP server to relay the message to the recipient's email server.The protocol uses specific commands and responses to establish a connection, validate addresses, and transfer the email content. Works mainly on port 25, but may use secure ports such as 587 and 465. By facilitating the smooth transmission of email across networks, SMTP enables reliable and efficient communication between users .

- **Transport-email-port :** Set the port number that your SMTP server uses.
- **Transport-email-use-auth :** This option determines whether authentication is required to connect to the SMTP server.
- **Transport-email-use-tls :** Set this option to true if your SMTP server supports TLS encryption, which provides a secure communication channel
- **Transport-email-use-ssl :** Set this option to true if your SMTP server uses SSL encryption. If your server doesn't use SSL, set it to false.
- **Transport-email-auth-username :** Specify the email address that Graylog will use as the sender for the email notifications
- **Transport-email-auth-password :** Provide the password for the email account . This allows Graylog to authenticate with the SMTP server when sending email notifications.

Once the configuration is done restart the graylog sever for the changes to take effect.

4.3.2.2 Step 2 : Alert creation

To create an alert in Graylog, you can follow these step-by-step instructions :

1. Log in to your Graylog web interface.
2. Navigate to the "Alerts" section.
3. Click on the "Manage Conditions" button, then select the "add new condition" button.
4. Define the condition for your alert. This includes selecting a stream, and specifying a condition type.
5. Configure the parameters for the selected condition type.
6. Configure the "Grace period" and "Backlog" settings to control when the alert triggers and how far back it looks for messages.
7. Once the condition is created, select the appropriate notification mechanism for the alert by clicking the "Manage Notifications" button.

8. Configure the specific settings for the selected notification type. For example, if you selected Email, specify the email address where you want to receive the alert and the body of the email.
9. Save the alert condition by clicking on the "Save" button.

Once the alert condition is saved, Graylog will continuously monitor the specified field and trigger notifications whenever the condition is met.

The figure 4.17 presents an example of an email being sent when the condition of Zimbra server failure is met.

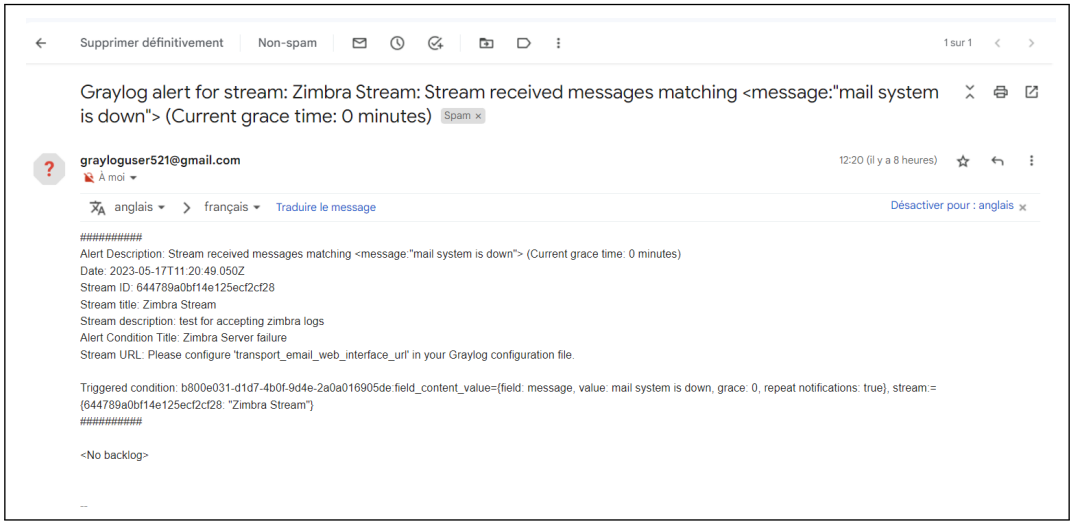

FIGURE 4.17 – Email Example

4.4 CONCLUSION

To summarize, Graylog provides powerful features for manipulating, exploring, visualizing, and alerting on log data. These capabilities enable effective log data management and proactive monitoring.

Furthermore, Graylog's functionality can be expanded through the use of plugins, which enhance log data workflows. Step up to the next chapter as we delve into Graylog plugins and discover how they optimize log data management even further.

Chapter

5

4^{th} SPRINT - Graylog Plugins

Contents

5.1 INTRODUCTION

A plugin, also known as an add-on or add-in, is a software component that extends an existing program to improve it, enhance its capability, or integrate with external systems, by adding to it a specific feature or functionality without altering its code. When a program allows for plugins, it enables users to customize it to meet their specific needs. In the context of our project, Graylog provides a plugin system that allows developers to extend the functionality of the platform by creating custom plugins.

Through this chapter, we will explore Graylog plugins, discussing not only the way they can be integrated and their development methodologies but also their problems.

5.2 GRAYLOG PLUGINS INTEGRATION

"Graylog Plugins Integration" refers to the process of installing and configuring plugins to extend the functionality of Graylog wich can help users streamline their log management processes and improve their ability to analyze log data from various sources.

Two options to install plugins on Graylog are most used.

5.2.1 Graylog Marketplace

Graylog has a marketplace where users can find various plugins developed by the community or by Graylog partners. In fact, it is a repository of plugins that can be downloaded and installed directly from the Graylog web interface. To use a plugin from the marketplace, we should follow these steps :

1. Open the Graylog web interface.
2. Navigate to System > Plugins.
3. Click on the "Find new plugins" button.
4. Browse the available plugins and read their descriptions.
5. Select the plugin you want to install and click on the "Install" button.

6. After the plugin is installed, you may need to restart the Graylog server for it to take effect.
7. Now, the functionality added to the platform can be used.

5.2.2 GitHub

GitHub is an online platform, used by millions of developers worldwide, that offers Git-based version control and software development hosting services. It allows developers to collaborate on projects, contributes to open-source software, track changes to code, and manage their Git repositories as well as software releases. Besides, GitHub provides a range of features, including bug tracking, pull requests, code review, GitHub Discussions, and project management tools. In addition to the web-based platform, GitHub offers a mobile app and GitHub Enterprise, a self-managed version of GitHub.com with similar functionality that can be run on an organization's hardware or a cloud provider. Furthermore, GitHub hosts an annual developer conference, known as GitHub Universe, that delivers updates, ideas, and inspiration to assist developers in constructing and designing software.

In the context of our project, If we cannot find the plugin we need in the marketplace, we can also search for third-party plugins on GitHub or other online sources.

Plugins from GitHub are typically provided as source code that needs to be compiled into a JAR file. Thus, we can install a plugin manually from Github by :

1. Log in to GitHub.
2. Browse the available Graylog plugins and read their descriptions in README files (check also discussions to know about plugin problems).
3. Download the source code or clone the repository using Git.
4. Configuring the plugin, if necessary, by editing the pom.xml file, updating the Graylog configuration file, or using environment variables.
5. Compile/Package the source code into a JAR file using Maven or Gradle (The build instructions are provided in the plugin's README file).
6. Place the JAR file in the plugin directory of the Graylog server. The default location of this directory is "/usr/share/graylog-server/plugin"
7. Once a plugin is installed, restart the Graylog server to take effect

5.3 GRAYLOG PLUGINS DEVELOPMENT

Graylog plugins are written in Java and built using Maven. In order to develop a Graylog plugin, we can whether write it from scratch using archetypes or contribute to an open-source developed plugin. However, It's crucial to choose the specific type of plugin you want to create depending on your need and set up your development environment before getting started.

5.3.1 Graylog Plugin Types

There are several types of Graylog plugins and each one comes with its specific set of requirements. these types include :

- **Input :** Enable Graylog to accept specific types of messages.
- **Output :** Enable Graylog to forward log messages to other systems while processing.
- **Processors :** Enable Graylog to enrich, transform, or drop incoming messages, and generate multiple new messages.
- **Event Notifications :** Enable Graylog to activate event notifications when an event alert is triggered.
- **Decorator :** Enable Graylog to modify the presentation of messages during search time.

5.3.2 Environment Setup

Graylog plugins are developed in Java using not only any Java IDE such as Eclipse, IntelliJ IDEA, or NetBeans but also Linux text editors such as nano. Then, for production, they are built and packaged using Maven or Gradle. To get started, we need to set up a development environment for Graylog plugins, which includes installing :

5.3.2.1 Integrated Development Environment (IDE)

An IDE is a software application that consolidates various tools required for writing, testing, and debugging software code through a single graphical user interface (GUI) to

improve productivity, reduce setup time, and standardize the development process. Typically, these tools include a source code editor, build automation tools, and a debugger. Additionally, IDEs are available in various forms, including those designed for a specific programming language or supporting multiple languages.

5.3.2.2 Java Development Kit (JDK)

JDK is a software package that offers a wide range of tools that are useful for a Java programmer. These tools include an interpreter, a compiler, debugging utilities, performance monitoring tools, and a Java Runtime Environment (JRE) which is the implementation of the Java Virtual Machine (JVM) and provides a platform to execute Java programs. In order to create applications in an IDE, It's crucial to have a JDK installed which can be obtained from the official Oracle website.

5.3.2.3 Apache Maven

As a project management tool, Maven is responsible for managing Java projects and can be used with or without an IDE. In fact, the JDK does not build an application, It just provides a group of classes that work together. Therefore, based on the concept of the project object model (POM), Maven provides a project structure, manages its content as well as its dependencies, and takes the group of Java files and packages them into one file. Maven follows the convention over configuration principle, meaning you can follow a certain configuration or customize it yourself. Additionally, Maven can be used for projects written in other programming languages like C#, Ruby, and Scala.

5.3.2.4 Git

Git is a robust version control system that proves invaluable in managing plugins. By establishing a Git repository for your plugin, you can easily track changes, collaborate with team members, and maintain a comprehensive history of modifications. Git's branching and merging capabilities enable parallel development, allowing individual developers to work

on specific features or experiments independently and seamlessly integrate their changes into the main plugin codebase. Additionally, with remote repository hosting platforms like GitHub or GitLab, you can effortlessly share your plugin with the community or team, facilitating contributions and ensuring efficient version control. Git's tagging feature further assists in managing releases, ensuring reproducibility, and providing a clear history of releases. Overall, Git's robust tools and features enhance the management and customization of plugins in Graylog, promoting streamlined development workflows and effective collaboration among developers.

5.3.2.5 Node & Yarn

If we are planning to write a web plugin, we also need Node and Yarn for the frontend. Node.js is a server-side JavaScript runtime built on Chrome's V8 engine, while Yarn is the primary JavaScript package manager for the Node.js that emphasizes speed, security, and consistency. Practically, developers can use Yarn to manage dependencies for their Node.js projects. In fact, it is installed globally on the system and can install packages for any Node.js project. Additionally, Yarn can be installed on multiple operating systems, including Windows, macOS, and Linux, and is compatible with Docker images. Another well-known package manager for Node.js is NPM, which developers can choose based on their preferences and needs.

5.3.3 Archetypes use case

5.3.3.1 Archetype Definition

An archetype is used in the context of software design patterns and refers to a universal or generalizable template known as the blueprint that can be used to generate new projects with a predefined structure and configuration. Typically, this motif provides a standardized approach to solving a specific design problem, avoiding common pitfalls and errors, and creating software systems that are efficient, maintainable, and scalable. Therefore, by using archetypes, developers can reduce development time, improve code quality, and increase the likelihood of success for their projects.

Examples of software design archetypes include the Model-View-Controller (MVC) pattern, the Singleton pattern, the Factory pattern, and the plugin pattern.

Note : In coding, the terms "skeleton" and "archetype" both refer to templates or frameworks that provide a starting point for software development, but they differ in their level of abstraction and complexity. In fact, a plugin archetype is a more specific type of skeleton and provides a standardized structure for the plugin project. However, a skeleton refers to a style of programming based on simple high-level program structures. More specifically, Archetypes are used to create skeleton projects, which developers can then modify and extend to build their applications.

5.3.3.2 Archetype Generation & Functionality Implementation

After setting up the development environment, the easiest way to create a new Maven project with the required files and directory structure for a Graylog plugin is by generating a Graylog archetype using the Command Line Interface (CLI)). This will provide us with a fundamental plugin framework containing all required dependencies and files to initiate the work.

This command uses the "archetype :generate" goal to create a new Maven project from a template (called an "archetype"). The options used in this command are :

- **groupId :** wich is the group ID of the project that should be in reverse domain name notation. For example, if your company's domain name is example.com, we can use com.example as the groupID
- **artifactId :** which is the ID of the project, which should be a unique name for your project. For example, we can use graylog-plugin-example.
- **archetypeArtifactId :** which is the ID of the archetype to use for the project. we can use the maven-archetype-quickstart archetype, which creates a simple Java project
- **interactiveMode :** which specifies whether Maven should ask for confirmation before generating files.

Then, to customize the plugin, we should first update the pom.xml file content such as the "parent.version" which targets the desired Graylog version. Second, edit the generated files by implementing the desired functionality by adding or writing Java code.

5.3.3.3 Plugin Deployment

In order to deploy the plugin to a repository or for installation in Graylog, we need to configure Maven to build and package the plugin using the Command Line Interface which executes the goals written in the pom.xml file.

5.3.4 Contribution use case

Another option to create a plugin is to contribute to an existing open-source project. First, the contributor can either browse the Graylog Marketplace or search for plugins on GitHub. Then, comes the step of cloning the plugin's repository using Git to make the wanted changes to the code.

The contributor can also distribute the developed plugin to other Graylog users by publishing it on the Graylog Marketplace or sharing the JAR file directly

Note : When contributing to an existing Graylog plugin, it's important to adhere to the plugin development guidelines and best practices that may differ based on the plugin we are contributing to. Thus, reading the plugin's documentation we are working on is crucial.

5.4 GRAYLOG-PLUGIN-FUNCTION-STRLEN

In this section, we will explore some details of the functional plugin we used which is named "graylog-plugin-function-strlen".

This plugin is very basic but showcases the development and integration aspects of Graylog plugins. It adds the function of calculating the string's length of a message field source and then stores the result in a new message field called source_length.

The following steps detail how we developed and integrated our plugin :

1. Using a Maven Archetype we created a complete plugin skeleton. In our Ubuntu virtual machine, a new folder called graylog-plugin-function-strlen was generated containing all the required files. The 5.1 figure showcases the project directory detailed :

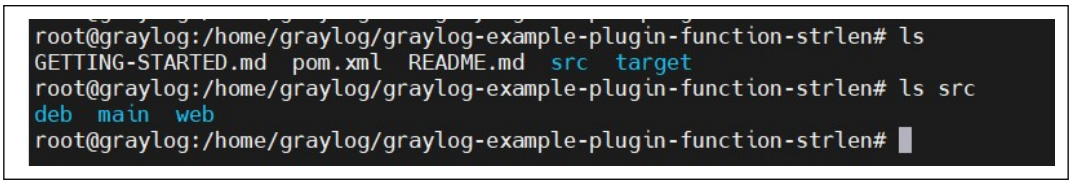

FIGURE 5.1 – Plugin's Directory

- **pom.xml :** This file contains the Maven build configuration for the plugin.
- **README.md :** A markdown file with information about the plugin.
- **src/main/java :** The directory containing the plugin's Java source code.
- **src/main/resources :** The directory containing the plugin's resource files, such as configuration files and property files
- **src/test/java :** The directory containing the plugin's unit tests.
- **src/test/resources :** The directory containing the plugin's test resources.

2. We made some changes to the pom.xml file including adding dependencies.
3. We implemented the function in the StringLengthFunction class.
4. We instructed Graylog to load the new function by editing the StringLengthFunctionModule class.
5. We navigated into the graylog-plugin-function-strlen folder and compiled the plugin using "$ mvn package".
6. The generated graylog-plugin-function-strlen-1.0.0-SNAPSHOT.jar file is located in the /target folder. We copied it to the "/usr/share/graylog-server/plugin" directory which is configured in graylog-server.conf.
7. We restarted graylog-server
8. We navigated to the Graylog web interface to create a pipeline rule which tests the plugin. The figure 5.2 below showcases the rule's body.

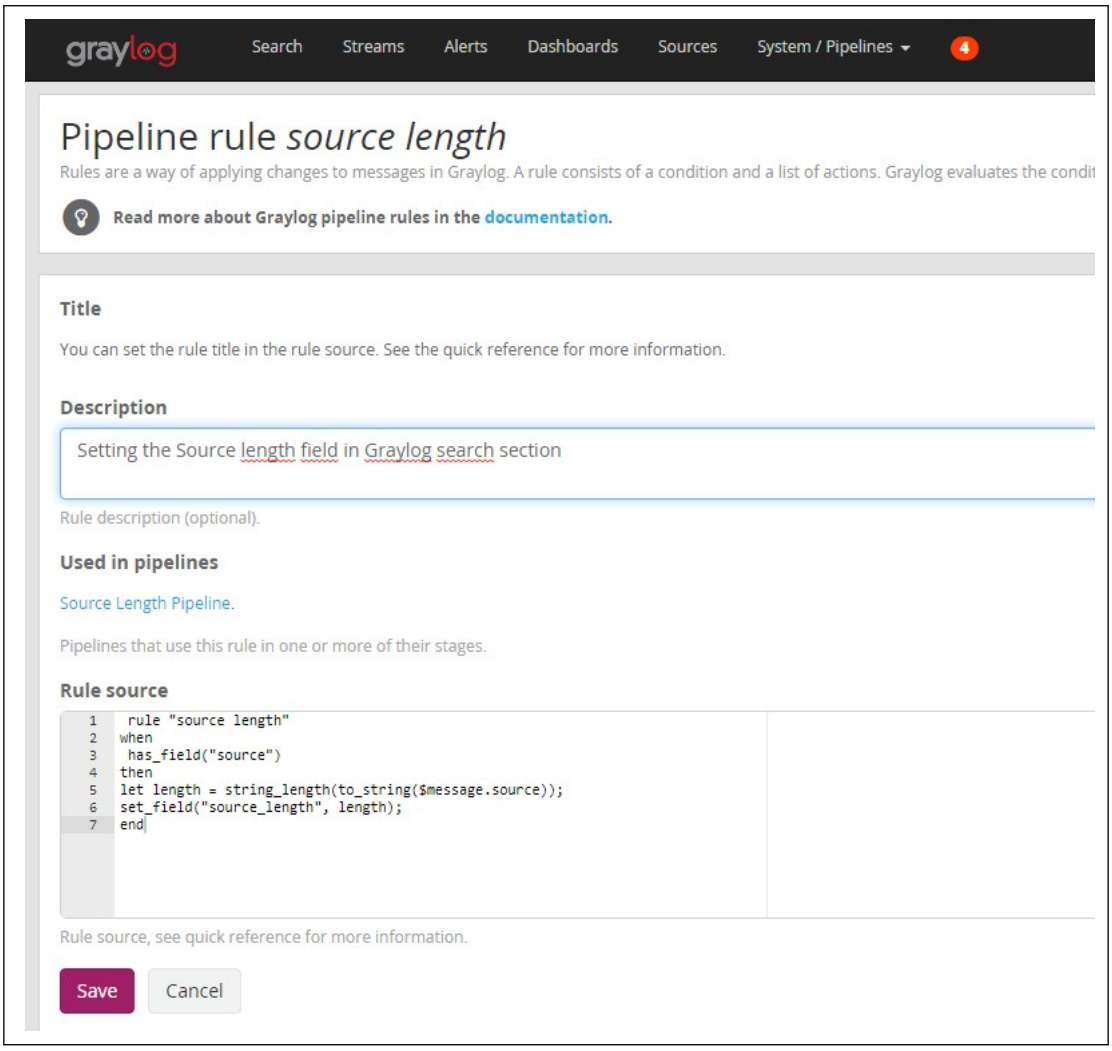

FIGURE 5.2 – Plugin's pipeline rule

9. We wired that pipeline to the default stream.
10. The result is desplayed in the search result of the default stream as shown in the figure5.3.

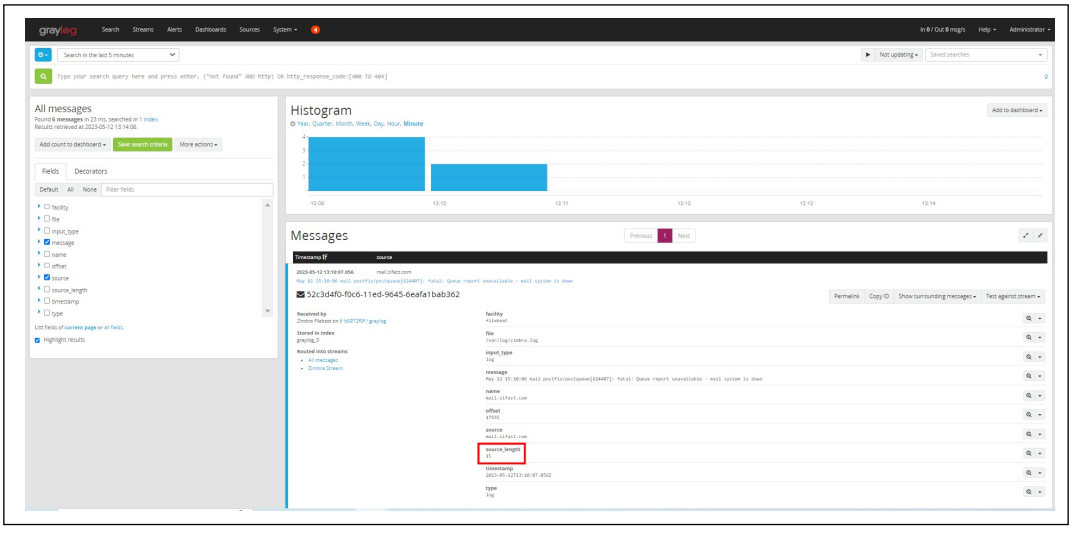

FIGURE 5.3 – Plugin's Function

5.5 PLUGINS PROBLEMS

The current sprint lasted for a long period of time during which we worked on many plugins. However, no desired result was reached due to several reasons which include :

- The third-party plugins may not be officially supported by Graylog, and they may not be compatible with the version of Graylog we are using. That's why, we should always test plugins thoroughly before using them in a production environment, and be aware that they may introduce security risks or performance issues.
- Some plugins may require additional configuration before they can be used. That's why, we should always read the documentation provided with the plugin to understand how it works and what steps are required to set it up.
- There are problems in yarn while building a web plugin. However mostly all Graylog plugins should be web-based because the Graylog platform is web-based.
- Graylog Community is so tiny. That's why, it's so hard to find a disired working plugin or coding instruction to develop a plugin. Here, this mission cannot be assigned only to a Big Data Developer, we need also the intervention of a Java Developer and maybe a Node.js Developer.

5.6 CONCLUSION

In conclusion, Graylog plugins are powerful tools that enhance the platform's functionality and cater to specific use cases. They enable users to extend Graylog's capabilities and better meet their log management and analysis needs. However, developing and integrating plugins can present several challenges. Developers may encounter issues with compatibility, installation, and configuration, which can hinder the seamless integration of plugins into the Graylog system. Despite these challenges, the benefits of Graylog plugins in addressing unique requirements and improving log management processes make them a valuable addition to the platform.

CONCLUSION & PERSPECTIVES

During our final year internship at SIFAST, our main objective was to establish a server monitoring system using Graylog and compare it with ELK stack. This system would collect, analyze, and visualize logs, as well as send alerts in case of any anomalies.

Throughout the internship, we adopted the Agile approach for project management and implemented the SCRUM methodology, dividing the project into different sprints. In the "First Sprint", we conducted research and exploration about the project technologies and the concept behind server monitoring. Then, building upon that knowledge, the "Second Sprint" focused on installing and configuring both the server and client sides, setting up the foundation for log management. With the system in place, the "Third Sprint" focused on log analysis, where we demonstrated Graylog's powerful capabilities to derive valuable insights from the logs. Finally, in the "Fourth Sprint", we advanced our expertise by implementing Graylog plugins, expanding the functionality of the system. This Agile approach allowed us to effectively manage our time and improve our communication skills and teamwork.

This experience provided us with an opportunity to learn and improve our skills in various technologies including Java, Maven, Elasticsearch, Graylog, and Filebeat, as well as to step further in the Big Data field.

Graylog proved to be a valuable tool, simplifying log management through its centralized platform, user-friendly interface, and advanced features for analysis and monitoring. It enable users to get insights from their data, swiftly detect issues, and enhance the security and performance of the infrastructure.

It's worth mentioning that the ELK project, developed by our colleagues, was also a part of our exploration. ELK stood out for its powerful capabilities and flexibility. First, Elasticsearch

provided scalable search and log analysis functionalities, while Logstash simplified log collection and aggregation from diverse sources. Last but not least, Kibana offered a visually appealing interface for log data visualization and exploration.

In comparing Graylog and ELK, we found that Graylog excelled in its simplicity of setup and intuitive user experience. It featured a streamlined configuration process and built-in alerting capabilities, enhancing its appeal for log management tasks. Additionally, Graylog offered a cost-effective solution with open-source and enterprise editions, providing flexible pricing options. However Graylog's weaknesses lay in the tiny documentation and community, making it harder to handle encountered problems.

On the other hand, the ELK project's strength lay in its comprehensive toolkit and scalability, catering to large-scale log management and complex data analysis tasks. The ELK stack, provided a robust foundation for handling extensive log data, offering scalability, high -performance indexing, and powerful visualization capabilities. However, it's important to note that the ELK stack's commercial offering, known as the Elastic Stack, introduced licensing costs based on data volume and additional features. Ultimately, the choice between Graylog and ELK depends on specific requirements including budget considerations and the scale of log management needs. Graylog's simplicity, affordability, and built-in alerting capabilities make it a solid choice for small to medium-sized enterprises. Meanwhile, the ELK project's extensive toolkit and scalability make it a preferred option for larger organizations with advanced log management and analysis needs.

Undoubtedly, this end-of-studies project has enriched our experience and increased our chances of succeeding in our professional careers. It allowed us to apply the knowledge acquired during our studies and equipped us with valuable skills and experience necessary to thrive in the workplace.

However, we acknowledge that this project is just the beginning, and there is much more to explore in this field. We have plans to implement additional features, such as containerization migration and developing custom plugins tailored to specific customer needs. Furthermore, we aim to integrate K-Means analysis into the log analysis and monitoring section. We see this

project as a promising seed in the ever-evolving realm of server monitoring, and we are excited to continue our journey and contribute to its advancement.

BIBLIOGRAPHY

[1] **Graylog, Inc.** Graylog Documentation. **[Pdf]**. Available at :https://archivedocs.graylog.org/_/downloads/en/2.4/pdf/ Release 2.4.6. Mar 01, 2019

[2] **Bealdung**.Apache Maven Guide**[Pdf]**. Available at :https://s3.amazonaws.com/baeldung.com/Apache+Maven+Guide.pdf?__s=83jowj5xht5nkcgkq50v December 28, 2022

[3] **Chamillard, G.**.Ubuntu : administration d'un systéme Linux**[Pdf]**. Available at : https://books.google.tn/books?id=vaiQzpla5aMC 2009

[4] **ȘUȘTIC, Roxana-Daniela, MORARU, Alexandra, ANDREI-BOGDAN, R. U. S., et al.** PERFORMANCE EVALUATION OF ELK STACK VERSUS GRAYLOG AS OPEN-SOURCE LOG MANAGEMENT TOOLS. Acta Technica Napocensis. Electronica-Telecomunicatii, 2022, vol. 62, no 1.**[pdf]**. Available at : https://users.utcluj.ro/~dobrota/ATN_1_2022_4.pdf

[5] **Andrew Whiteley, Julien Pollack, Petr Matous**. The origines of Agile and iterative methodes JANUARY/APRIL2021 **[Online]**. Available at : https://journalmodernpm.com/manuscript/index.php/jmpm/article/view/JMPM02502/414

WEBOGRAPHY

[1] **Graylog**. Graylog official documentation version 2.4 **[Online]**. Available at : https://archivedocs.graylog.org/en/2.4/index.htmll

[2] **MobaXterm**. MobaXterm X server and SSH client **[Online]**. Available at :https://mobaxterm.mobatek.net/

[3] **elastic.co**.Installing Elasticsearch **[Online]**. Available at : https://www.elastic.co/guide/en/elasticsearch/reference/current/install-elasticsearch.html

[4] **elastic.co**. Filebeat documentation **[Online]** version 5.6 :https://www.elastic.co/guide/en/beats/filebeat/5.6/filebeat-installation.html

[5] **coursera**.Agile-project-management **[Online]**. Available at : https://www.coursera.org/learn/agile-project-management

[6] **Openclasseooms**. Gérez du code avec Git et GitHub 5/19/22 **[Online]**. Available at : https://openclassrooms.com/en/courses/7162856-gerez-du-code-avec-git-et-github/7166041-tirez-le-maximum-de-ce-cours

« Server Monitoring System using Graylog »

الخلاصة: كجزء من مشروع التخرج الخاص بنا داخل شركة SIFAST، تعهدنا بتطوير بنية تحتية مركزية وقابلة للتطوير لإدارة السجلات والرصد باستخدام منصة Graylog. كان الهدف الرئيسي من هذا المشروع هو تحسين يقظة وإدارة الصحف داخل المنظمة من أجل زيادة أمن النظم وتوافرها بشكل عام. لتنفيذ هذا المشروع، اعتمدنا منهجية Scrum كإطار عمل، واستخدمنا Ubuntu للتثبيت والتكوين، و Filebeat للإدارة و واجهة الويب الخاصة ب Graylog لتحليل السجلات، وطورنا plugins باستخدام Java.

المفاتيح: Graylog , Elasticsearch, MongoDB, Filebeat, Ubuntu, Java, Maven, Git, Plugins, السجلات , Scrum

Résumé :

Dans le cadre de notre projet de fin d'études au sein de la société SIFAST, nous nous sommes lancés dans le développement d'une infrastructure centralisée et évolutive de gestion et de suivi des logs à l'aide de la plateforme Graylog. Notre objectif principal était d'améliorer la vigilance et la gestion des journaux au sein de l'organisation, améliorant ainsi la sécurité et la disponibilité globales du système. Pour y parvenir, nous avons adopté la méthodologie Scrum comme cadre, utilisé Ubuntu pour l'installation et la configuration, utilisé Filebeat pour la direction des journaux, installé et configuré Elasticsearch pour le stockage et la recherche des journaux, utilisé l'interface Web de Graylog pour l'analyse des journaux et même aventuré dans le développement de Graylog. plugins utilisant Maven, git et Java.

Mots clés : Graylog , Elasticsearch, MongoDB, Filebeat, Ubuntu, Java, Maven, Git, Plugins, Journaux, Scrum

Abstract:

As part of our graduation project at SIFAST company, we embarked on developing a centralized and scalable infrastructure for log management and monitoring using the Graylog platform. Our primary objective was to enhance vigilance and log management within the organization, ultimately improving overall system security and availability. To achieve this, we adopted the Scrum methodology as our framework, employed Ubuntu for installation and configuration, utilized Filebeat for log direction, installed and configured Elasticsearch for log storage and search, utilized Graylog's web interface for log analysis, and even ventured into developing Graylog plugins using Maven, git, and Java.

Keywords: Graylog , Elasticsearch, MongoDB, Filebeat, Ubuntu, Java, Maven, Git, Plugins, Logs, Scrum