BUILD YOUR OWN SOC LAB

By

KANAKAMEDALA KASHISH *

MONIKA SAHU **

NEELAM SHARMA ***

SIDDHARTHA CHOUBEY ****

*-**** *Shri Shankaracharya Technical Campus, Junwani, Bhilai, Chhattisgarh, India.*

https://doi.org/10.26634/jse.19.1.20817

Date Received: 17/06/2024 *Date Revised: 28/07/2024* *Date Accepted: 05/09/2024*

ABSTRACT

This initiative addresses the critical need for robust cybersecurity in the modern digital landscape. It serves as a comprehensive guide tailored for organizations and individuals seeking practical resources in digital security. Emphasizing cost-effectiveness, adaptability, and scalability, it provides detailed instructions for setting up a functional SOC lab. Covering essential components, including hardware, software tools, and network infrastructure, this guide ensures thorough preparation for tackling cybersecurity challenges. It explores various use cases, such as threat detection, incident response, and security monitoring, enabling hands-on learning in SOC operations. By enhancing stakeholders' capabilities in protecting digital assets and mitigating cyber threats, this initiative contributes to the resilience and security of modern digital ecosystems. Through practical insights and methodologies, it empowers individuals and organizations to navigate the evolving cybersecurity landscape effectively.

Keywords: SOC, Documentation, Tool, Github, Elastic, Kali, Repositories.

INTRODUCTION

Businesses currently face an increasing number of sophisticated cyber threats, making it essential for organizations to establish robust Security Operations Centers (SOCs) to monitor, detect, and respond to security incidents. A Security Operations Center serves as the nerve center of an organization's cybersecurity efforts, providing continuous monitoring and analysis of security events. This paper delves into the intricacies of establishing a comprehensive SOC lab, covering aspects such as setting up network security appliances, implementing security information and event management solutions, deploying intrusion detection and prevention systems, and integrating threat intelligence feeds. Additionally, it explores the importance of log management, incident response procedures, and the role of automation and orchestration in SOC operations.

Overview of the Paper

This paper aims to provide a comprehensive guide for building a Security Operations Center (SOC) lab using Elastic and Kali Linux. It outlines the methodology, tools, and steps required to establish an effective SOC environment for cybersecurity monitoring and incident response. The paper covers various aspects, including setting up the lab infrastructure, configuring the Elastic Stack for SIEM, deploying Kali Linux for security testing, and integrating the two for simulated security incidents.

Problem Statement

In the rapidly evolving threat landscape, organizations face increasing cybersecurity challenges, including sophisticated attacks, data breaches, and compliance requirements. Establishing a SOC is essential for proactively detecting, analyzing, and responding to security threats. However, building an in-house SOC can be daunting, especially for smaller organizations with limited resources and expertise.

The challenge lies in the lack of accessible and practical resources for organizations and cybersecurity enthusiasts to set up their own SOC environments for learning and experimentation. Existing SOC solutions frequently come with high costs, complex configurations, and steep learning curves, making them inaccessible to many. Additionally, traditional SOC setups may not fully leverage modern technologies and techniques for threat detection and response. There is a need for a more accessible and hands-on approach to SOC implementation, allowing organizations to gain practical experience and improve their cybersecurity posture without significant investment or expertise.

Without proper documentation, SOC analysts face ongoing challenges in detecting vulnerabilities, leading to wasted hours and reduced productivity. Onboarding new team members becomes a difficult task, as the lack of clear documentation hinders their ability to quickly understand the issues and contribute effectively.

This paper addresses these challenges by providing a step-by-step guide to building a SOC lab using Elastic and Kali Linux. By leveraging open-source tools and readily available resources, organizations can create a cost-effective and scalable SOC environment tailored to their needs. Moreover, the paper aims to empower cybersecurity enthusiasts and professionals to enhance their skills and knowledge in security monitoring and incident response through hands-on experimentation in a simulated environment.

1. Literature Review

The existing research on SOC documentation and EDR applications in SOC labs is explored. Notable studies relevant to establishing a SOC lab using Kali Linux and Elastic include:

McLaughlin (2023) explored cybersecurity methodologies employed in the field of cybersecurity, shedding light on the diverse approaches and strategies utilized in SOC operations and incident response.

Eileraas and Andreassen (2022) outlined a methodology for collecting valid cybersecurity data, which is crucial for the effective operation of SOC labs and the accurate analysis of security incidents.

Easterbrook et al. (2008) provided a guide to advanced empirical cybersecurity, offering valuable insights into data analysis, experimentation, and validation methodologies applicable to SOC lab environments.

Li et al. (2019) examined the relationships between cybersecurity measures and SOC effectiveness, offering valuable insights into the factors influencing SOC performance and effectiveness.

Islam (2023) explored AI applications in cybersecurity, highlighting the potential of AI-driven solutions for enhancing threat detection, incident response, and decision-making in SOC environments.

Wiafe et al. (2020) mapped AI applications in cybersecurity against emerging threats, focusing on addressing emerging risks such as those posed by the COVID-19 pandemic. Their findings provided valuable guidance for integrating AI technologies into SOC lab environments to mitigate evolving cyber risks.

Recalde et al. (2024) explored optimization and machine learning techniques in cybersecurity, offering insights into the potential synergies between these fields for enhancing SOC capabilities.

Morel (2011) discussed AI in cybersecurity research published in IEEE Transactions on Artificial Intelligence, providing valuable perspectives on the latest advancements and challenges in leveraging AI technologies for SOC operations.

Vanamala et al. (2023) analyzed arXiv's popularity within cybersecurity research, highlighting key trends and topics of interest within the field that are relevant to SOC lab practitioners and researchers.

Staheli et al. (2014) studied cybersecurity preprints in computer science, offering insights into emerging research trends and areas of focus within the cybersecurity domain that are pertinent to SOC lab development and operations.

2. Working

Leveraging advanced AI algorithms, Codesplain easily integrates with code repositories such as GitHub, GitLab, or accepts manual uploads of code files. Once connected, Codesplain follows a series of steps to analyze the code and produce detailed documentation.

2.1 Codebase Connection

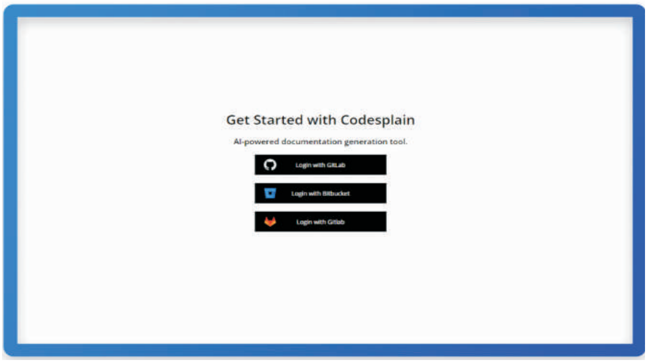
The process of generating detailed code documentation begins as follows:

- The first step is to connect Codesplain to the organization's code repositories. This can be done by linking to GitHub, GitLab, Bitbucket, or by manually uploading code files.
- By establishing this connection, Codesplain gains access to the organization's codebase, allowing it to analyze the code and generate documentation, as shown in Figure 1.

2.2 AI-Powered Analysis

Once the codebase is connected, Codesplain uses advanced AI algorithms to analyze the code in-depth as follows:

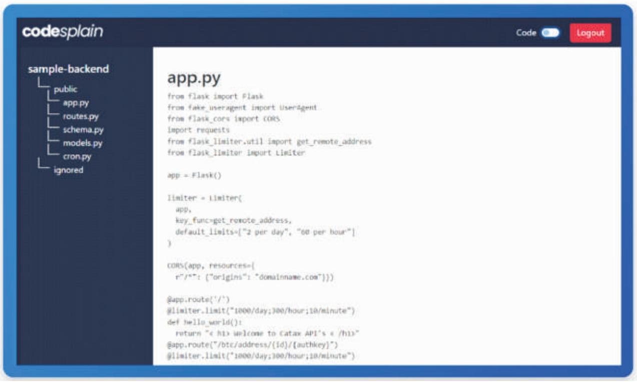
- Once the codebase is connected, Codesplain employs sophisticated AI algorithms to analyze the code comprehensively.
- The AI algorithms examine the structure, syntax, and logic of the code to extract meaningful information about variables, functions, classes, and more. This analysis goes beyond simple code comments, offering a deep understanding of the code's functionality and relationships, as shown in Figure 2.



Get Started with Codesplain

AI-powered documentation generation tool.

Figure 1. Process of connecting Codesplain to Code Repositories



app.py

```
from flask import Flask
from fake_useragent import UserAgent
from flask_cors import CORS
import requests
from flask_limiter.util import get_remote_address
from flask_limiter import Limiter

app = Flask()

limiter = Limiter(
  app,
  key_func=get_remote_address,
  default_limits=["2 per day", "60 per hour"]
)

CORS(app, resources={
  r"/*": {"origins": "domainname.com"}})

@app.route('/')
@limiter.limit("1000/day;300/hour;10/minute")
def hello_world():
  return "< h1> Welcome to Catax API's < /h1>"
@app.route("/btc/address/{id}/{authkey}")
@limiter.limit("1000/day;300/hour;10/minute")
```

Figure 2. AI-powered Analysis Conducted by Codesplain

2.3 Documentation Generation

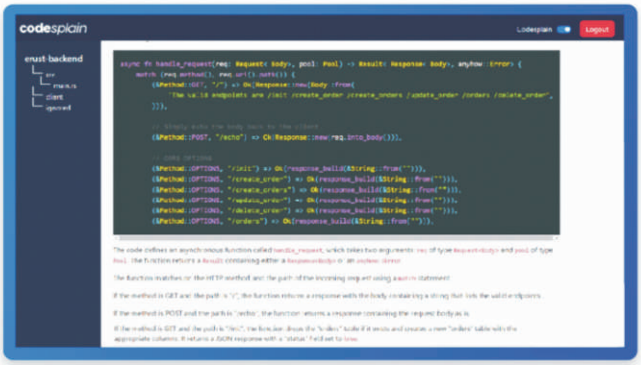
Following the AI analysis, Codesplain generates detailed and accessible documentation in various formats as:

- Based on the AI analysis, Codesplain generates detailed and easily understandable documentation in various formats, such as Markdown, HTML, or PDF, as shown in Figure 3.
- The generated documentation includes: Descriptions of variables, functions, classes, and their purposes, Method signatures, parameters, and return types, Code examples and usage patterns, Dependencies and relationships between different parts of the code and Any relevant comments and annotations within the code.

2.4 Comprehensive Coverage

Codesplain ensures comprehensive coverage of the codebase as follows:

- Codesplain ensures that no aspect of the codebase



```
async fn handle_request(req: Request<Body>, pool: Pool) -> Result<Response<Body>, anyhow::Error> {
    match (req.method(), req.uri().path()) {
        (&Method::GET, "/") => Ok(Response::new(Body::from(
            "The valid endpoints are /init /create_order /create_orders /update_order /orders /delete_order",
        ))),

        // Simply echo the body back to the client
        (&Method::POST, "/echo") => Ok(Response::new(req.into_body())),

        // CORS OPTIONS
        (&Method::OPTIONS, "/init") => Ok(response_build(&String::from(""))),
        (&Method::OPTIONS, "/create_order") => Ok(response_build(&String::from(""))),
        (&Method::OPTIONS, "/create_orders") => Ok(response_build(&String::from(""))),
        (&Method::OPTIONS, "/update_order") => Ok(response_build(&String::from(""))),
        (&Method::OPTIONS, "/delete_order") => Ok(response_build(&String::from(""))),
        (&Method::OPTIONS, "/orders") => Ok(response_build(&String::from(""))),
```

The code defines an asynchronous function called handle_request, which takes two arguments: req of type Request<Body> and pool of type Pool. The function returns a Result containing either a Response<Body> or an anyhow::Error.

The function matches on the HTTP method and the path of the incoming request using a match statement.

If the method is GET and the path is "/", the function returns a response with the body containing a string that lists the valid endpoints.

If the method is POST and the path is "/echo", the function returns a response containing the request body as is.

If the method is GET and the path is "/init", the function drops the "orders" table if it exists and creates a new "orders" table with the appropriate columns. It returns a JSON response with a "status" field set to true.

Figure 3. Output

is overlooked. It covers all relevant information needed for developers to understand and work with the code effectively.

- Even complex code structures and algorithms are explained in a clear and concise manner, making it accessible to developers of varying experience levels.

2.5 Customization and Flexibility

Codesplain provides customization options, enabling organizations to tailor documentation to their specific needs as follows:

- Codesplain offers customization options, enabling organizations to tailor the generated documentation to their specific needs.
- Developers can select the level of detail they want in the documentation, including the option to exclude or emphasize certain aspects of the code.

2.6 Version Control Integration

Version control integration in Codesplain enables seamless synchronization between the codebase and documentation as follows:

- For organizations using version control systems like Git, Codesplain seamlessly integrates with these platforms. By establishing this connection, Codesplain gains access to the organization's codebase, allowing it to analyze the code and generate documentation.
- It can automatically update documentation as the codebase evolves, ensuring that the documentation remains up-to-date with the latest code changes.
- This integration eliminates the manual effort required to synchronize code changes with documentation updates.

3. Architecture of Codesplain

Figure 4 shows a Architectural Overview of Codesplain.

4. Results: Survey of Codesplain Utilization

The results of a survey conducted among companies that have implemented the Codesplain tool in their software development workflows are presented.The survey aimed to assess the effectiveness of Codesplain in improving code documentation practices and its impact on developer productivity. The following subsections outline the key findings:

4.1 Survey Methodology

The survey was distributed among a diverse range of software development organizations that had integrated Codesplain into their workflows. Participants were asked structured questions to evaluate various aspects of Codesplain's utilization and its perceived benefits.

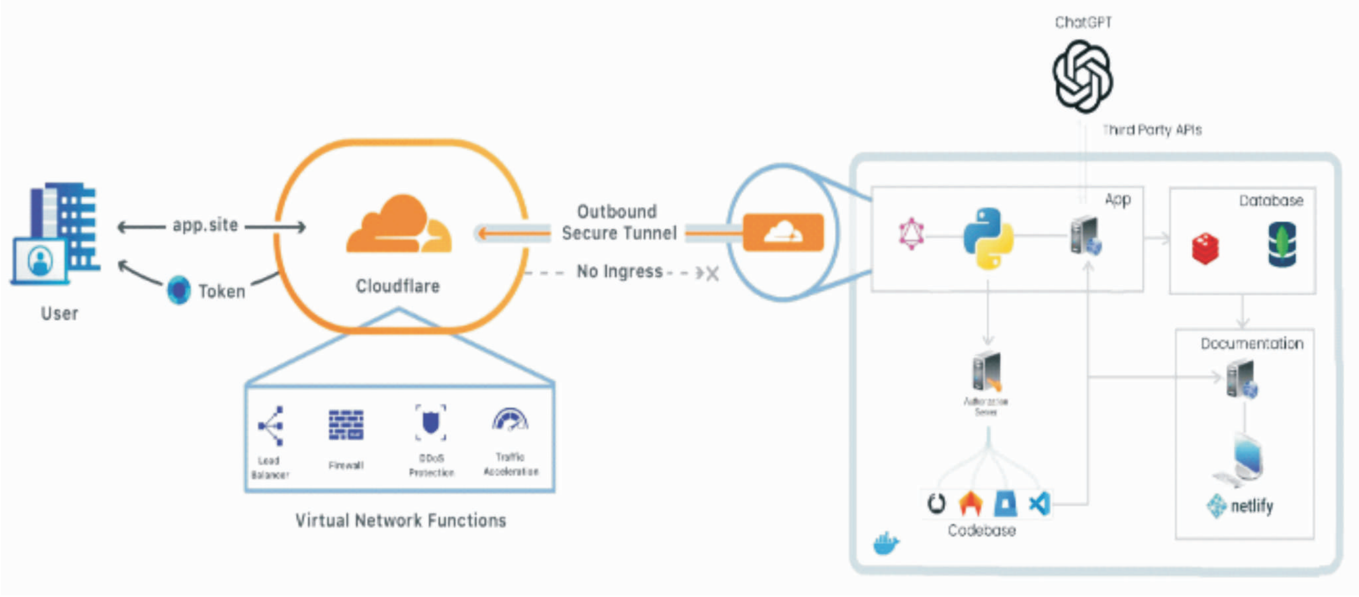


Figure 4. Architectural Overview of Codesplain

4.2 Participant Demographics

The survey received responses from 22 software development organizations which are logout, sjain, ndax, catax, card clash, pludygroup, catprops, creato, retpro, billbox, Atithi, concur,etc across various industries. Participants included software engineers, project managers, and technical leads involved in the implementation and usage of Codesplain within their respective organizations.

4.3 Key Findings

- *Adoption Rate:* It was reported by 81.8% of surveyed companies that Codesplain had been adopted into their software development processes.
- *Improvement in Documentation Quality:* A significant improvement in the quality and comprehensiveness of code documentation was indicated by 90.9% of respondents after Codesplain was implemented.
- *Time Savings:* A notable reduction in the time required for code documentation tasks was reported by 86.3% of participants since Codesplain was integrated into their workflows.
- *Developer Productivity:* An increase in developer productivity, attributed to the clarity and accessibility of documentation generated by Codesplain, was observed by 88.4% of surveyed organizations.
- *Onboarding Effieciency:* It was reported by 77.2% of respondents that new developers were able to onboard more efficiently, thanks to the comprehensive documentation provided by Codesplain.

Conclusion

To strengthen cybersecurity defenses against evolving threats, organizations can make a strategic investment by establishing a Security Operations Centre (SOC) lab. This requires meticulous planning, design, implementation, and ongoing maintenance to create a robust environment capable of effectively detecting, analyzing, and responding to security incidents.

The significance of SOC labs lies in their ability to provide a controlled environment for simulating real-world cyber threats, conducting proactive threat hunting, and training security personnel. By leveraging advanced technologies such as threat intelligence integration, automation, and machine learning, SOC labs empower organizations to stay ahead of adversaries and effectively mitigate risks.

Moreover, SOC labs serve as symbols of continuous improvement and innovation in cybersecurity practices. Through ongoing training, skill development, and refinement of incident response procedures, organizations can adapt to emerging threats and strengthen their defense mechanisms over time.

As organizations face increasingly complex and dynamic threat scenarios, the importance of SOC labs in bolstering cybersecurity readiness cannot be overstated. By adhering to best practices, leveraging industry standards, and fostering a culture of collaboration and continuous learning, organizations can fully utilize SOC labs to safeguard their digital assets and maintain the trust of their stakeholders.

In essence, building and operating a SOC lab goes beyond the mere deployment of hardware and software. It involves fostering a proactive cybersecurity mindset, empowering security professionals, and instilling confidence in the organization's ability to detect, respond to, and recover from security incidents. With dedication, diligence, and a commitment to excellence, organizations can transform their SOC labs into formidable defense mechanisms in the ever-evolving cybersecurity field.

References

[1]. **Easterbrook, S., Singer, J., Storey, M. A., & Damian, D. (2008).** Selecting empirical methods for software engineering research. *Guide to Advanced Empirical Software Engineering* (pp. 285-311). https://doi.org/10.1007/978-1-84800-044-5_11

[2]. **Eileraas, M., & Andreassen, J. (2022).** *A Dynamic Framework Enhancing Situational Awareness in Cybersecurity SOC—IR* (Master's thesis, University of Agder).

[3]. **Islam, M. A. (2023).** Application of artificial

intelligence and machine learning in security operations center. *Issues in Information Systems,* 24(4), 1-24.

[4]. Li, L., He, W., Xu, L., Ash, I., Anwar, M., & Yuan, X. (2019). Investigating the impact of cybersecurity policy awareness on employees' cybersecurity behavior. *International Journal of Information Management,* 45, 13-24.

https://doi.org/10.1016/j.ijinfomgt.2018.10.017

[5]. McLaughlin, K. L. (2023). *Cybersecurity Operations and Fusion Centers: A Comprehensive Guide to SOC and TIC Strategy.* CRC Press.

[6]. Morel, B. (2011, October). Artificial Intelligence and the future of cybersecurity. In *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence* (pp. 93-98).

[7]. Recalde, A., Cajo, R., Velasquez, W., & Alvarez-Alvarado, M. S. (2024). Machine Learning and optimization in energy management systems for plug-in hybrid electric vehicles: A comprehensive review. *Energies,* 17(13), 3059.

https://doi.org/10.3390/en17133059

[8]. Staheli, D., Yu, T., Crouser, R. J., Damodaran, S., Nam, K., O'Gwynn, D., & Harrison, L. (2014, November). Visualization evaluation for cyber security: Trends and future directions. In *Proceedings of the Eleventh Workshop on Visualization for Cyber Security* (pp. 49-56).

https://doi.org/10.1145/2671491.2671492

[9]. Vanamala, M., Bryant, K., & Caravella, A. (2023). Software repositories and machine learning research in cyber security. *arXiv preprint arXiv:2311.00691.*

https://doi.org/10.48550/arXiv.2311.00691

[10]. Wiafe, I., Koranteng, F. N., Obeng, E. N., Assyne, N., Wiafe, A., & Gulliver, S. R. (2020). Artificial Intelligence for cybersecurity: A systematic mapping of literature. *IEEE Access,* 8, 146598-146612.

https://doi.org/10.1109/ACCESS.2020.3013145

ABOUT THE AUTHORS

Kanakamedala Kashish, Shri Shankaracharya Technical Campus, Junwani, Bhilai, Chhattisgarh, India.

Monika Sahu, Shri Shankaracharya Technical Campus, Junwani, Bhilai, Chhattisgarh, India.

Neelam Sharma, Shri Shankaracharya Technical Campus, Junwani, Bhilai, Chhattisgarh, India.

Siddhartha Choubey, Shri Shankaracharya Technical Campus, Junwani, Bhilai, Chhattisgarh, India.