**HLD**

**Cyber Shield Project**

Document ID: AAA-BBB-12345

Revision History:

|  |  |  |  |
| --- | --- | --- | --- |
| Rev. | Date | Author | Details |
| 0.1 | 20.05.2023 | Asaf Rehavi | First version |
| 0.2 | 27.05.2023 | Asaf Rehavi | Second version |
| 0.3 | 10.06.2023 | Asaf Rehavi | Add EDR,System logic |

Abstract:

This High-Level Design (HLD) document presents the architecture of a new enterprise product that is designed to collect and process a significant amount of data from various sources. Moreover, the document defines the architecture for data processing and storage, ensuring the accuracy and availability of the collected data when required.

Contents

[**1.**](#_heading=h.30j0zll) **General 3**

[1.1](#_heading=h.1fob9te) Introduction 3

[1.2](#_heading=h.3znysh7) Glossary, references 3

[**2.**](#_heading=h.2et92p0) **Requirements 3**

[1.1](#_heading=h.3dy6vkm) Logical (System Functionality) 4

[1.2](#_heading=h.1t3h5sf) User Workflow 4

[1.3](#_heading=h.4d34og8) Availability and Recovery 4

[1.4](#_heading=h.2s8eyo1) Performance & Capacity Requirements 4

[1.5](#_heading=h.3rdcrjn) Scalability 4

[1.6](#_heading=h.26in1rg) Security 4

[1.7](#_heading=h.lnxbz9) Monitoring and Debugging 5

[1.8](#_heading=h.35nkun2) Deployment 5

[1.9](#_heading=h.1ksv4uv) Backward Compatibility 5

[**3.**](#_heading=h.44sinio) **High Level Design 5**

[1.1](#_heading=h.2jxsxqh) System Architecture 5

[1.2](#_heading=h.3j2qqm3) Processes 5

[1.3](#_heading=h.1y810tw) Design Rules and Principles 6

[1.4](#_heading=h.4i7ojhp) Upgradability 6

[1.5](#_heading=h.2xcytpi) Assumptions and Dependencies 6

[**4.**](#_heading=h.1ci93xb) **Time Estimation 6**

[**5.**](#_heading=h.3whwml4) **Limitations and Reservations 6**

[**6.**](#_heading=h.2bn6wsx) **Risks 6**

[**7.**](#_heading=h.qsh70q) **Open Issues 7**

1. General
   1. Introduction

The System that described at this document is designed in order to project the organization from security threats.

The system collects data from the following resources:

1. **File analysis** -scan the files from external sources using third party tool.
2. **Command and controls** – packets meta data and applies ML logic to detect command and control traffic in IP traffic.
3. **End point agents** – verify that the organization user perform allowed actions using third party tools
4. **Network forensics** - record and index all incoming traffic to an organization from

external resources

* 1. Glossary, references

1. **Backward Compatibility**: The ability of a newer version of software to work with data or configurations generated by an older version.
2. **Deployment**: The process of moving software from a development environment to a production environment.
3. **Backward Compatibility**: The ability of a newer version of software to work with data or configurations generated by an older version.
4. **Deployment**: The process of moving software from a development environment to a production environment.
5. **Encryption:** The process of converting data into a code to prevent unauthorized access.
6. **High-Level Design (HLD)**: A design approach that focuses on the overall system architecture and its components.

[opswap](https://www.opswat.com/?utm_campaign__c=&utm_source__c=google&utm_medium__c=paid&utm_term__c=opswat&utm_term=opswat&utm_campaign=EMEA+%7C+Search+%7C+Brand+(Exact)&utm_source=google&utm_medium=paid&hsa_acc=7260173240&hsa_cam=18518141759&hsa_grp=142865405260&hsa_ad=626385828167&hsa_src=g&hsa_tgt=kwd-336038014544&hsa_kw=opswat&hsa_mt=e&hsa_net=adwords&hsa_ver=3&gad=1&gclid=CjwKCAjwscGjBhAXEiwAswQqNFsKPGAD1toHSqXUu7Rcq8NuTI95_eL7KB_dcPr7FU-D76LCBUE3TxoCsegQAvD_BwE)

[End point agent explanation](https://plurilock.com/answers/endpoint-agent-what-does-endpoint-agent-mean/)

[edr eset](https://www.eset.com/il/business/solutions/endpoint-detection-and-response/?utm_source=Serach&utm_medium=ad&utm_campaign=Google&utm_content=eset-website&gad=1&gclid=CjwKCAjwpuajBhBpEiwA_ZtfhbUfzbRuamzaAWML1KPOCg1gMapB-Opatj0-OqIlywRg9I7KDjXNKBoCrooQAvD_BwE)

[edr sentinal one](https://www.sentinelone.com/surfaces/endpoint/?utm_source=google-paid&utm_medium=paid-search&utm_campaign=ukie-bau-edp&utm_term=Endpoint%20protection&campaign_id=20043535176&ad_id=656670718684&gclid=CjwKCAjwpuajBhBpEiwA_ZtfhaaP-0E25wkbBajXrdDyRCJnkYIU5YOgF_2tn2TfvtNEXnsj5H_tiRoCragQAvD_BwE)

1. Requirements

2.1 Functional requirements

2.1.1 The system should handle this artifact types:

* File analysis
* Command and Control
* End Point Agents
* Network forensics

2.1.2 -each artifact will have unique id

2.1.3 - The system will expose an investigation portal in order to display the threads to the users.

2.1.4 - The system will purge data older than 90 days or exceeding the 80% of the allocated storage.

2.2 Non-Functional requirements

2.2.1-The solution should be available 99.9% of the time

2.2.2 - The solution is installed on prem in each company

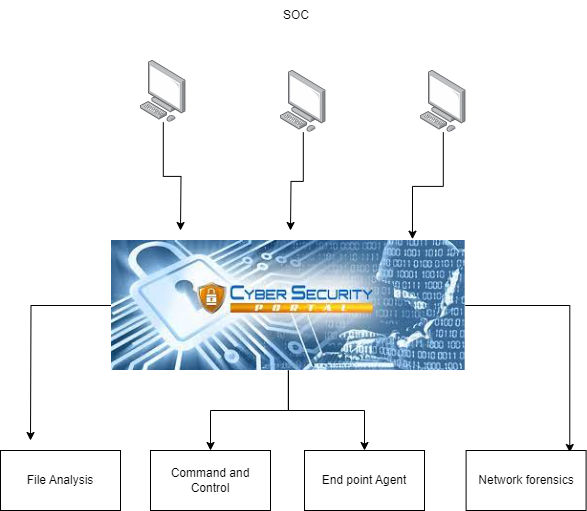
2.2.3. The solution should be deployed as a cloud native application (some parts may be deployed on VMs)

2.2.4 -Monitoring The system should monitor all of the components dashboards should be created for

2.2.5 – The system will have a central log .

2.2.6 – the solution should fit to small, medium, large and extra large organizations

* 1. Logical (System Functionality)



The purpose of the system is to protect the organization from cyber threats.

System components:

1. **Investigation portal**: echo all cyber events to the SOC analyst and enables to perform cyber

investigations over all data that was collected and analyzed

1. **File analysis**: detection of malicious malware in a file
2. **Command and control**: process takes packets meta data and applies ML logic to detect command and control.

traffic in IP traffic

1. **End point Agent:** follow user actions in order to reduce cyber risk
2. **Network forensics:** is responsible to record and index all incoming traffic to an organization from

external resources

* 1. User Workflow

*{If relevant and possible include sample of screen shots}*

[Enter your writing here]

* 1. Availability and Recovery

The product shall be designed to ensure high availability and rapid recovery in the event of system failure or other disruptions. The following measures shall be implemented to achieve these goals:

1.**Load Balancing**: The investigation portal shall be hosted on multiple servers, and a load balancer shall be used to distribute traffic evenly among them. This will ensure that if one server fails or becomes overloaded, traffic will be automatically redirected to the other servers, ensuring that the investigation portal remains available.

2.**Data Backups**: The website's data shall be backed up regularly to prevent data loss in the event of a system failure. Backups shall be stored offsite to ensure that they are not affected by local disasters or other disruptions.

3.**Redundancy**: Critical components of the product, such as the database and web servers, shall be duplicated to ensure redundancy. If one component fails, the duplicate component shall take over automatically, ensuring that the website remains available.

4.**Monitoring**: The product shall be monitored continuously to detect and diagnose problems as they occur. This will enable the development team to respond quickly and effectively to any issues that arise, minimizing downtime and ensuring rapid recovery.

5.**Disaster Recovery Plan**: A disaster recovery plan shall be developed and tested regularly to ensure that the website can be quickly restored in the event of a major disaster or other catastrophic event. The plan shall include procedures for restoring data backups, deploying redundant components, and restoring service to users as quickly as possible.

By implementing these measures, the website shall be designed to provide high availability and rapid recovery, ensuring that users can access the site and make purchases with minimal disruption.

The system will be available at 99.99% of the time.

* 1. Performance & Capacity Requirements

1.4.1 **process time**:

The data that is being processed should be stored in persistent data store up to 5-10 minutes.

1.4.2 **Caching:**

The system will use a cache solution in order to prevent many IOPS operations.

1.4.3 **Read and write processes**.

Each component will create read and write models that ensure that the data will be written to the different database efficiently, the system will use 80% of write operations and 20 % of read operations.

**1.4.4 Disk capacity**

Basic assumptions:

The disk capacity basic assumptions is that each data will be purge after 180

The disk space will be at least 20% more than the estimated data size.

Small –, 3 Gbps IP traffic analysis

Medium –10 Gbps traffic analysis

Large –50 Gbps traffic analysis

Extra large –1000 Gbps traffic analysis

|  |  |  |  |
| --- | --- | --- | --- |
| **Small org** | **Medium org** | **Large org** | **Extra large** |
| **3 \* 60\*24-180 =778 \* 1.2 = 933 GB** | **10 \* 60\*24-180 \* 1.2 = 3.1 tB** | **50 \* 60\*24-180 \* 1.2 15.5 tB** | **1000 \* 60\*24-180 \* 1.2 = 310 tB** |

Scalability

The system will have the ability to perform horizontal scale for the following issues:

* File analysis – the system should be able to handle up to 10,000 files in a minute – the scale should contains large use of opswat tool
* Command and Control – The system should support scale up to 100,000 packets per minute
* End point Agent - – The system should support scale up to 20,000 agents .
* Network forensics The system should support scale up 1000 Gbps
  1. Security

**Encryption**:

The system shall use encryption to protect sensitive data both in transit and at rest. SSL/TLS shall be used to encrypt data in transit, while strong encryption algorithms shall be used to encrypt data at rest.

Sensitive data will be stored encrypted at the different databases.

Encryption algorithm :RSA 256 key size 4096

**Authentication and Authorization**:

The system shall use secure authentication and authorization mechanisms to prevent unauthorized access to sensitive data and functionality,

Authentication will be done by the central IDP of the organization in order to control the authentication outside of the system.

and role-based access control shall be used to ensure that only authorized users can access the system.

Authorization will allow some of the users to view data based on dates and stat type for example (user x can view all of the history and other user y can view only the last week, or user x can view CNC data only )

**Security updates**

Updates for the different components will be done by on monthly basis.

Vulnerability Scanning and Penetration Testing: The website shall undergo regular vulnerability scanning and penetration testing to identify and address security vulnerabilities. Vulnerability scanning shall be performed using tools such as Nessus and OpenVAS, while penetration testing shall be performed bycertified ethical hackers.

Security updates for third party tools such as opswat will be done upon severity and quarterly based

**Secured communication** –

The traffic between the different databases will not be at default port.

Sensitive data will be passes between the different application parts encrypted.

The connection string for each database will be encrypted, database permission will be defined.

The website shall be designed and implemented with security in mind to protect user data, prevent unauthorized access, and ensure the confidentiality, integrity, and availability of the system. The following measures shall be implemented to achieve these goals:

* 1. Monitoring and Debugging

The system shall be monitored in order to detect scale up and scale down, or detect any failure of any component.

The system logs will be separated by the severity error.

The logs will be saved for 14 days.

Monitor dashboards should contain hardware usage of components.

The monitor mechanism will create alerts before failure.

The system will use monitor tools such as Grafana and Prometheus.

* 1. Deployment

Each component will be deployed independently.

The product shall be deployed using a reliable and repeatable process to ensure consistency and minimize the risk of errors. The following measures shall be implemented to achieve these goals:

2.2.1. Version Control: All code and configuration files shall be stored in a version control system, such as Git, to allow for easy tracking of changes and to enable rollbacks if needed.

2.2.2 Continuous Integration and Deployment: The website shall use a continuous integration and deployment (CI/CD) system to automate the build and deployment process. This will ensure that any changes to the code or configuration files are automatically tested and deployed to the production environment, reducing the risk of errors and ensuring consistency.

2.2.3. Testing: Automated testing shall be used to ensure that the website is functioning correctly after each deployment. This will include unit tests, integration tests, and end-to-end tests.

2.2.4. Staging Environment: A staging environment shall be used to test changes to the system before they are deployed to the production environment. This will allow for testing of new features and changes.

2.2.5. Rollback Plan: A rollback plan shall be developed and tested to ensure that the website can be quickly and easily rolled back to a previous version in the event of errors or issues.

2.2.6 Infrastructure as Code: The infrastructure used to deploy the website, including servers, load balancers, and databases, shall be defined using code. This will allow for easy replication of the infrastructure and reduce the risk of errors.

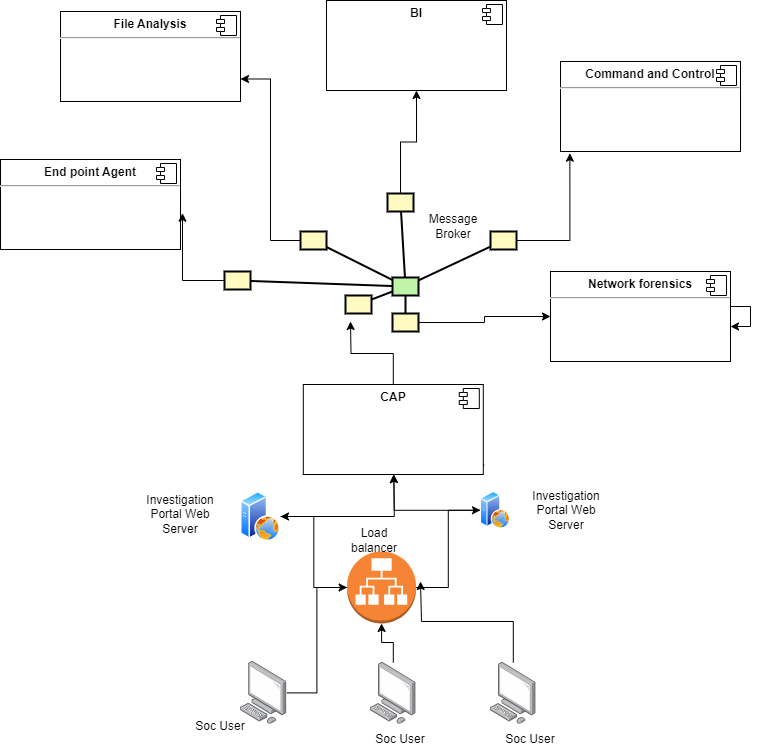
2.2.7. Security: The deployment process shall be designed to ensure that the system is deployed securely. This will include measures such as encrypting sensitive data, limiting access to production environments, and using secure protocols for data transfer.

* 1. Backward Compatibility

TO Be Define

1. High-Level Design

3.1 System Architecture



**Soc users**: The Users that will get alarms from all components using the investigation portal.

**CAP**: This is the “Brain” of the cyber defense system.

**File Analysis:** This is a component that runs 3rd party software – opswat – it is a process that runs 16 antivirus engines to scan for malware over each file.

**Command and Control:** The process takes packets meta data and applies ML logic to detect command and control traffic in IP traffic.

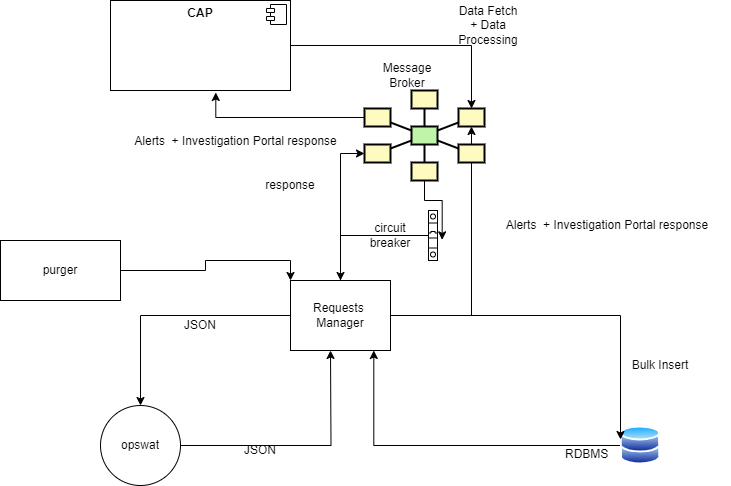
**End point Agent:** These agents are sending events to a centralized location in the CAP for processing.

**Network forensics:** This component is responsible for recording and index all incoming traffic to an organization from external resources.

**Investigation Portal:** The investigation portal echo all cyber events to the SOC analyst and enables to perform cyber investigations over all data that was collected and analyzed

Processes

**3.2 File Analysis:**

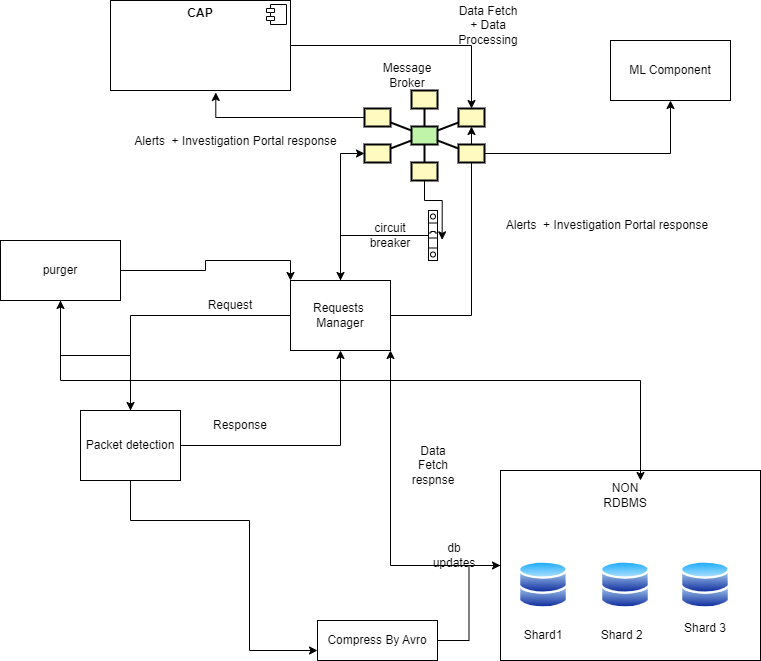


Scan File :

A diagram of a computer program

Description automatically generated with low confidence

3.3 Command and control



Packages union:

In order to improve the ability of search packets from the same connection we need to save those packets union by connection here is an explanation how to union the packets

Identify the protocol: Determine the network protocol being used for packet fragmentation and reassembly. For example, in the case of TCP/IP, the Transmission Control Protocol (TCP) provides reliable and ordered delivery of packets, including reassembly.

Track connections: Establish a mechanism to track and identify connections between packets. This typically involves examining fields within the packet headers that indicate the source and destination addresses, ports, and sequence numbers.

Receive packets: Collect the individual packets that belong to a specific connection. This can be done using network socket programming or by listening to network traffic on a specific port.

Sort packets: Sort the received packets based on their sequence numbers. This ensures that the packets are reassembled in the correct order.

Reassembly process: Use the protocol-specific reassembly mechanism to reconstruct the original data from the packets. For example, in TCP/IP, the sequence numbers and acknowledgement numbers are used to reorder and reassemble the packets into the correct sequence.

Deliver the data: Once the packets have been successfully reassembled, deliver the reconstructed data to the appropriate application or layer that is responsible for processing it.

**CQRS:**

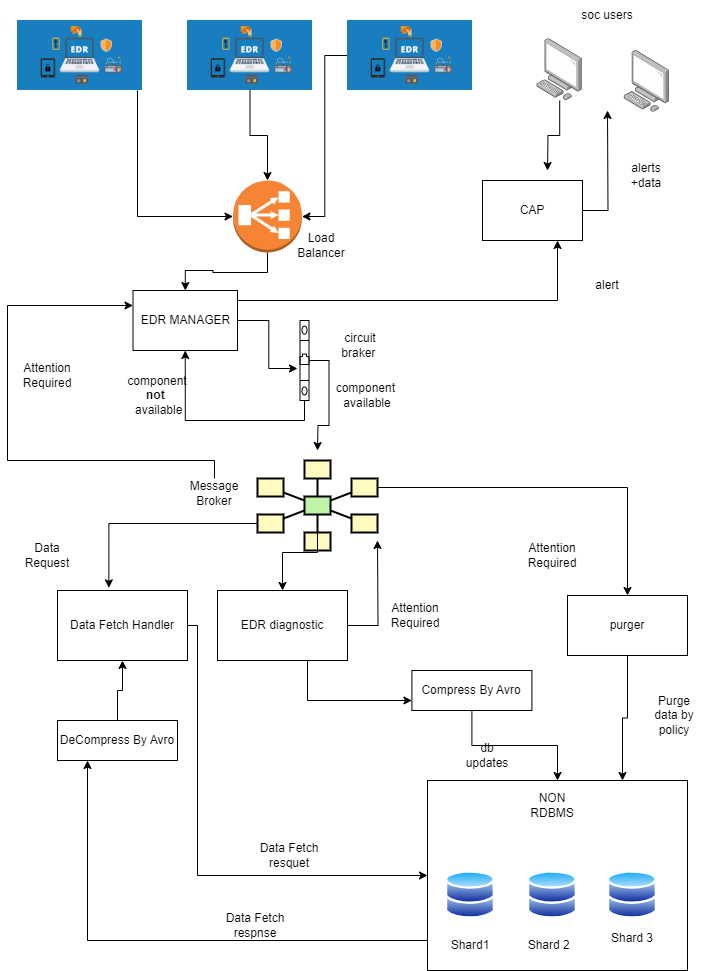
At this component the data will be design by the CQRS principle

The data for read will be designed at different schema from the data for write in order to improve performance [CQRS explanation](https://developer.confluent.io/learn-kafka/event-sourcing/cqrs/?utm_medium=sem&utm_source=google&utm_campaign=ch.sem_br.nonbrand_tp.prs_tgt.dsa_mt.dsa_rgn.emea_lng.eng_dv.all_con.confluent-developer&utm_term=&creative=&device=c&placement=&gad=1&gclid=CjwKCAjwyeujBhA5EiwA5WD7_UJ5zWUMqal5ezgqLHV21giMV50adOmZ6AfN5kEvXztupcCDjz3B0hoCrwUQAvD_BwE)

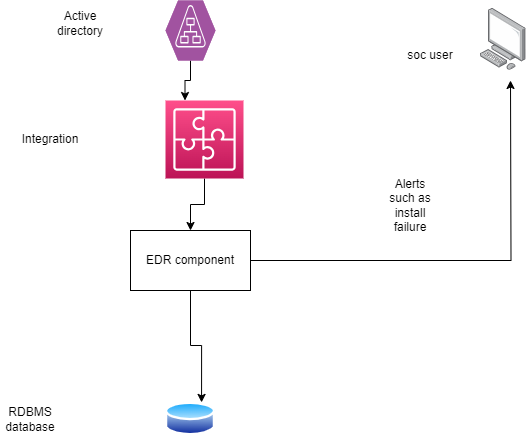
**Use of non-RDBMS DATABASE:**

At this component we need to handle large data for this reason we will use non-RDBMS with sharding

**3.4 EDR**



**Handle computer EDR installation and upgrade:**



There will be integration between active directory and the EDR component and the active directory.

This integration will run two times a day and provide to the soc users alerts such as: users that required manually install, users that should upgrade the EDR etc..

**Different EDR support:**

The component will be designed to support different EDR providers in minimal time and changes.

**CQRS:**

At this component the data will be design by the CQRS principle

The data for read will be designed at different schema from the data for write in order to improve performance [CQRS explanation](https://developer.confluent.io/learn-kafka/event-sourcing/cqrs/?utm_medium=sem&utm_source=google&utm_campaign=ch.sem_br.nonbrand_tp.prs_tgt.dsa_mt.dsa_rgn.emea_lng.eng_dv.all_con.confluent-developer&utm_term=&creative=&device=c&placement=&gad=1&gclid=CjwKCAjwyeujBhA5EiwA5WD7_UJ5zWUMqal5ezgqLHV21giMV50adOmZ6AfN5kEvXztupcCDjz3B0hoCrwUQAvD_BwE)

**Use of non-RDBMS DATABASE:**

At this components we need to handle large data for this reason we will use non-RDBMS with sharding

1.3 Design Rules and Principles

**1.3.1 Microservices**: The system should be designed as a set of microservices, with each service responsible for a specific functionality. Each microservice should be designed to be independent and loosely coupled, communicating with other services through well-defined APIs.

**1.3.2 Tests**: Developer work must be done with automated unit and integration tests.  
E.g if you write a feature you shouldn’t go to the browser and manually test it out, a test runner must be active on watch instead, doing that job automatically and structurally.

**1.3.3 Async flows:** use async processes as much as possible to create performance.

**1.3.4 Simplicity**: Keep the design simple and intuitive, ensuring that other developers can easily understand the functionality. Avoid cluttered layouts and excessive use of complex elements.

**1.3.5 Error Handling**: Design error messages and validation feedback to be clear, concise, and helpful. Provide users with actionable guidance to resolve errors or mistakes.

1.4 Upgradability

The product shall be deployed using a reliable and repeatable process to ensure consistency and minimize the risk of errors. The following measures shall be implemented to achieve these goals:

1.4.1. Version Control: All code and configuration files shall be stored in a version control system, such as Git, to allow for easy tracking of changes and to enable rollbacks if needed.

1.4.2. Continuous Integration and Deployment: Each component of the system shall use a continuous integration and deployment (CI/CD) system to automate the build and deployment process. This will ensure that any changes to the code or configuration files are automatically tested and deployed to the production environment, reducing the risk of errors and ensuring consistency.

1.4.3. Testing: Automated testing shall be used to ensure that the Each component of the system is functioning correctly after each deployment. This will include unit tests, integration tests, and end-to-end tests.

1.4.4. Staging Environment: A staging environment shall be used to test changes to Each component of the system before they are deployed to the production environment. This will allow for testing of new features and changes .

1.4.5. Rollback Plan: A rollback plan shall be developed and tested to ensure that Each component of the system can be quickly and easily rolled back to a previous version in the event of errors or issues.

1.4.6. Infrastructure as Code: The infrastructure used to deploy Each component of the system, including servers, load balancers, and databases, shall be defined using code. This will allow for easy replication of the infrastructure and reduce the risk of errors.

1.4.7. Security: The deployment process shall be designed to ensure that Each component of the system is deployed securely. This will include measures such as encrypting sensitive data, limiting access to production environments, and using secure protocols for data transfer.

1.4.8.upgrade of each third-party tool will be done once in a quarter

Assumptions and Dependencies

[Enter your writing here]

1. Time Estimation

|  |  |
| --- | --- |
| Subsystem/team | Workdays |
|  |  |

1. Limitations and Reservations

*5.1.1* **Limited Control over Third-Party Systems:**

The product's ability to defend against threats may be limited when interacting with third-party systems or services. The security posture of those systems, adherence to best practices, or response to threats may impact the overall effectiveness of the product.

5.1.2 **Performance Impact:**

While the product aims to minimize performance impact, certain configurations or high traffic scenarios may result in a slight reduction in system performance. It is important to monitor the product's performance and fine-tune configurations as needed to maintain an optimal balance between security and system efficiency.

5.1.3 **Human Error and Insider Threats**: The product's effectiveness may be compromised by human error or malicious actions from insiders with privileged access. Adequate user access controls, employee training, and continuous monitoring should be implemented to mitigate such risks.

1. Risks

6.1 **Third-Party Risks:**

The risk associated with the use of third-party vendors or services that may have security weaknesses or fail to meet security standards.

Be aware of software updates for each of the vendors

6.1.2 : **Compliance and Regulatory Risks**:

The risk of non-compliance with industry or legal regulations, resulting in penalties, fines, or legal consequences.

There will be consulting in order to find regulation changes every quarter.

6.1.3 **Resource and Budget Constraints**:

The risk of insufficient resources or budget allocations for implementing and maintaining robust cyber security measures.

The project development will be measured every sprint and there will be a discussion with the company management in order to ensure that Budget limits are not violated. .

1. Open Issues

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Description | Subsystem | Responsible |
| 1 | Define Sensitive data | All | Asaf |
| 2 | Define EDR provider | EDR | Asaf |
| 3. | Add Network forensics design | Network forensics | Asaf |
| 4. | Add Backward Compatibility | General | Asaf |
| 5. | Disk capacity of EDR  , Network forensics AND File analytic component | General | Asaf |
| 6. | Time estimation | General | Asaf |