ADVISORY REPORT

Audit programme upscaling and chain implementation
Hansken

Management summary

Hansken is a platform that enables digital investigators and analysts to search confiscated digital equipment for forensic traces. Hansken has since been used by various other government and investigative agencies to assist them in digital forensic investigations.

In order to optimise Hansken, we were asked to draw up a recommendation concerning the infrastructural requirements, the extent to which Hansken's current products are fit for purpose, Hansken's requirements in a (private) cloud environment, the requirements of the monitoring solution and how a multi-tenancy design can be used.

To determine the infrastructure requirements, several models were created that provide insight into the (required) speed and capacity of Hadoop, Elasticsearch and Cassandra clusters. For an accurate predictive outcome, these need to be tuned to the environment.

Within Hansken, all products were found to be *fit for purpose* except Cassandra. It is recommended to investigate the benefits of a *relational database management system* (RDBMS).

A greenfield design has been made in which an environment is described that not only meets government security and detection standards, but also deploys *industry best practices* and offers a solution for possible bottlenecks. Important spearheads are the environment-wide deployment of authentication, authorisation and encryption. Also, disconnecting object storage from the compute cluster is one way to create more scalability. Finally, it can add a lot of value to containerise certain services in order to better distribute the system load over the service(s).

In order to effectively monitor all components within Hansken, recommendations have been made regarding the metrics and log files to be monitored. Best practices have also been defined that contribute to the effective monitoring of the landscape.

The multi-tenancy design is elaborated in two additional scenarios besides the single tenant with a dedicated hardware scenario; multi-tenant with shared external object storage and multi-tenant shared infrastructure with external object storage. To further explain this, three architectural plates were developed with a description of the possibilities and limitations. Finally, the individual software components were examined for their technical deployability within a multi-tenancy environment. Per component the possibilities are described in the field of authentication, authorisation, data encryption, user spaces and performance with quotas.

Table of contents

1 Introduction

This is the advisory report on the infrastructural design of Hansken. Hansken is a powerful platform that enables digital investigators and analysts to search confiscated digital equipment for forensic traces. The platform was developed and put into use by the Netherlands Forensic Institute (NFI). Hansken has since been deployed at various other government and investigative agencies to assist them in digital forensic investigations.

The request for the research report arose from a collaboration between the NFI, the police and the FIOD. The parties involved want to know how Hansken should be optimally set up technically and what the infrastructural requirements are for a scalable and future-proof application landscape.

The research report consists of a number of different parts: (1) a recommendation on the infrastructural requirements, (2) a test for the installation of Hansken in a (private) cloud environment and (3) a recommendation on the set-up of a multi-tenant environment.

Prior to the study, a questionnaire was made available. Based on these questions, interviews were conducted with the parties involved in the study. These interviews have been summarised and the initial conclusions have been included in chapter 3 of this report. Based on the interviews conducted and the documentation provided, we proceeded to answer the questions in the report.

Chapter 2 contains a description of the reason and objective for the study. Chapter 3 discusses the approach to the study. Chapter 4 shares the results gathered during the preliminary investigation. Chapter 5 describes the (infrastructural) requirements for a small, medium and large Hansken environment. Chapter 6 discusses various security aspects that influence the design of Hansken. Chapter 7 describes the optimal set-up of monitoring within Hansken. Chapter 8 contains the details of the investigation into the individual applications present within Hansken and a greenfield situation of the Hansken platform.

In chapter 9, a design is made for the deployment of the Hansken platform in various multi-tenant situations. In chapter 10, an advice is given based on the research carried out and in this chapter follow-up steps are defined based on the advice.

2 Case study

In order to create a clear picture of the assignment carried out, this chapter describes the reason and objective of the Hansken audit programme.

2.1 Reason

Hansken is a platform that enables digital investigators and analysts to search confiscated digital equipment for forensic traces. The Netherlands Forensic Institute works with a variety of parties to improve the Hansken software and the accompanying documentation. To be able to use Hansken adequately, documentation is supplied with the product that describes the requirements and set-up of the infrastructure. This is made up of several clusters for: Hadoop, Elasticsearch, Cassandra and various Hansken services, each with their own requirements and setup[1]

The confidence in the application depends on the results that the forensic data yields and on the performance of the product. In the future, people would like to optimise Hansken's product further, and for this purpose, a request was made to draw up a recommendation regarding the infrastructural requirements, the extent to which the current products within Hansken are fit for purpose, a test of the installation of Hansken in a (private) cloud environment and how a multi-tenancy design can be deployed.

2.2 Objective

Capgemini has been commissioned by the NFI as an independent party to conduct research into the technical deployability of Hansken and to advise on the scalability of the software. The goal is to answer the question whether there are currently optimisation possibilities within the Hansken environment. To this a greenfield situation is added in order to map out how Hansken can be optimised technically within current standards and possibilities within the market for parties that Hansken provides its services to now and in the future.

[1] NFI - MOP - audit advice infra Hansken 14 May - v1.1 - Page 6

3 Research approach

The following research methods were used for the study:

Research Method	Specification	Note
Field research	- Open interviews - Capgemini best practices	- Gain insight into: ○ Business processes, applications and IT infrastructure ○ Bottlenecks ○ Environmental requirements and wishes - Consult Capgemini (big data) best practices
Desk research	- Literature study - Hansken architecture summary - SIG Hansken assessment - Capgemini best practices	- Gain insight into: ○ Business processes, applications and IT infrastructure ○ Bottlenecks - Identifying factors for theoretical model

By combining the results of these research methods, an integral structure per knowledge area follows. The aim is to answer the research questions using the above research methods.

3.1 Assumptions

This section describes various assumptions made during the study to answer the various research questions.

- **Assumption 1:** environments are basically identical as described in the Hansken architecture summary;
- **Assumption 2:** Hadoop is the largest component within the Hansken platform, consuming the most resources and incurring the most hardware costs;
- **Assumption 3:** Elasticsearch is the second largest component within the Hansken platform after Hadoop;
- **Assumption 4:** Information gathered from interviews serves as a reliable source of information;
- **Assumption 5:** Not all security and investigative standards have been deemed relevant to answering the main questions within the Hansken investigation;
- **Assumption 6:** In order to develop a functional multi-tenancy design, no components and/or functionalities should be excluded in advance.

3.2 Research questions

The research is based on research questions defined by the parties involved. The overview below shows which chapters answer these research questions.

Research question	Chapter
1. What are the technical requirements that the Hansken software suite imposes on the environment in which the product must be installed. Such that the infrastructure is scalable and the Hansken software can perform optimally and also complies with the Dutch security government and investigation standards.	5, 6, 8
2. Advice on how to adequately set up the monitoring of the platform (Hansken software suite implemented on the infrastructure). What (additional) requirements does this place on the software?	7
3. An advice on the type(s) of configuration of the infrastructural environment (including which clusters to distinguish) and what are the main characteristics of this environment? Which configuration types are excluded in advance?	8
4. Advice on the design of the individual software components. In combination with the Hansken software.	8
5. What (additional) requirements does a multi tenancy design impose on the setup of the individual software components/products in combination with the Hansken software?	9
6. What (additional) requirements does a multi tenancy design place on Hansken's software architecture?	9
7. What (additional) requirements does a multi-tenancy design impose on the organisation of the infrastructure?	9
8. Provide advice on where the cut-off between the common/central part and the individual tenants can best be made and more specifically where to make it (in the code or the infra).	9
9. Assess the multi-tenant design of the FIOD and advise on its optimal implementation, based on the current Hansken architecture. Include the way in which the NFI has implemented it. Assess this implementation and, if possible, define options for improvement.	9

4 Preliminary investigation

Within the research, interviews were conducted to find out the needs of the parties involved. During these interviews, attention was also paid to exploring the landscape of Hansken.

Below is a summary of the results of the interviews conducted with all parties involved. It is possible that not every party will (fully) recognise themselves in every result that may have arisen from information obtained from other parties. The results are presented in the form of key-takeaways, described per relevant part within the research. The complete questionnaire for use during the interviews conducted can be found in Appendix I interview questions.

4.1 Performance Hansken

Respondents' experiences of Hansken on both the front-end and back-end components were investigated.

In the front-end, performance issues are experienced when a large number of users are active. The magnitude of this issue increases as the number of users of Hansken within one organisation increases. In addition, users experience performance issues when opening larger files within Hansken. It has been indicated that this is probably due to the web application used to open documents and/or files (Document viewer).

Within the survey, it was found that respondents mainly want to know whether hardware is efficiently utilised by the components within the back-end. A recurring issue was the limitation imposed by Hansken to performing 5 extractions simultaneously within the landscape. One extraction equals one job within MapReduce, which creates multiple containers for processing the data. When a single job takes a long time to perform an extraction (at the end of a job, there is often a so-called "tail": one container that is still active for hours and therefore occupies one job slot), no other extraction can be started at the same time. As a result, extractions are experienced as slow.

4.2 Components Hansken

Hansken uses different components. The following components were examined during the interviews: Cassandra, Hadoop (HDFS, YARN, MapReduce), Elasticsearch, Ansible and Kafka and the Hansken core services.

Cassandra

Cassandra has been used as a repository for both relational and unstructured data. Research is currently being done into the possibility of using a relational database as repository for the relational data within Cassandra. Relational data is the (meta) data that is related to a forensic investigation. Unstructured data within Cassandra includes OCR output generated during an extraction.

Respondents experience Cassandra as burdensome in the sense that the product is not correctly applied for the right purpose. It has been indicated that alternatives are available for storing relational data that require less maintenance, are less complex and are not inferior in terms of scalability and functionality.

Hadoop

Hadoop is used to store images and to perform extractions. Topics covered about this application were the ability to perform rolling updates and the previously mentioned point about the formation of a "tail" within Mapreduce during the end of a job. No recurring problems were identified during the interviews.

Elasticsearch

Elasticsearch is used to index data for various services within Hansken. No recurring performance incidents were observed during the interviews. Comments were observed around a lack of authentication, compatibility incidents with Ubuntu, and the need to enable roll-up upgrades.

Plug-ins have been developed for Elasticsearch. During the interviews, no comments were observed regarding these plug-ins.

Ansible

Ansible is used within Hansken for the installation and configuration of Hansken services. Playbooks are delivered by the NFI, some with an accompanying installation manual. Customers have control over the method in which playbooks are distributed and executed. Examples include a CI/CD pipeline (Jenkins pipeline) or deployment directly from an Ansible machine to a host. Hansken customers adjust variables within the Ansible roles to ensure correct operation within their own environment.

Kafka

Kafka is a message streaming and queuing tool offered within Hansken for monitoring the progress of an extraction job. During the interviews it was mentioned that Kafka is also used for queuing the Hansken extraction job itself. When the maximum number of extraction jobs is reached which is set within the Hansken software, new extraction jobs are queued in Kafka.

4.3 S/M/L environment

The respondents' interpretation of the S/M/L issue was diverse. Answers to the question of what they saw as a small, medium and large environment varied. Sometimes the answer was limited to the size of cases, some also looked at it from the point of view of the number of users and type of load. The biggest difference that was noted was the factors mentioned by respondents to define an environment. The following factors were mentioned by the respondents:

- Data size: what is the average size of an image?
- Volume of data: how much data is stored?
- Number of users: How many people use Hansken?
- Load type: What kind of load is placed on the front-end?

4.4 Monitoring

Hansken is offered with the Grafana and Prometheus applications to monitor the platform. Monitoring and logging are provided by the NFI via Ansible playbooks. Grafana is used for visualisation of metrics, which are stored centrally by Prometheus. Observations show that the set-up of monitoring within Hansken differs among the parties involved, which affects the extent to which customers are able to control and manage their environment.

4.5 Governance/security

Governance

The infrastructural set-up of the Hansken platform is highly dependent on the various government standards and security requirements that are present at customers. During the interviews, the following government standards were classified as important:

- General Data Protection Regulation (GDPR/AVG);
- Decree on information security regulations for the civil service (VIR-BI);
- Police Data Act (WPG).

During the investigation, a number of adaptations were observed within Hansken that comply with the above rules and regulations. Specific requirements are implemented in interaction between the customer and the NFI. Hansken customers are ultimately responsible for complying with the rules that apply to each organisation.

Security

The organisations involved in the research have specific security requirements imposed by the organisation itself in addition to government standards. During the interviews, the following security requirements were observed:

- Insight into what data is transported between the various components;
- Access to Hansken only from specific locations (workstations in the office) and remotely via VPN to the workstations;
- Prohibiting access to case data by the IT organisation: administrators are not allowed to access the content data of the case that is loaded as an image;
- Encryption for *data in flight* between components;
- Encryption for *data at rest* on each component;
- Authentication and authorisation on all components;
- Network segregation.

4.6 Infrastructure

The Hansken platform is installed on its own dedicated hardware within a shared data centre. The hardware used was selected on the basis of requirements established by the organisations themselves. It was observed that respondents also asked for advice from the NFI and from suppliers of the individual components within the platform. Advice was sought regarding the hardware required, based on the expected load.

4.7 Needs assessment

Fit for purpose

One of the most discussed needs was that surrounding the current components within Hansken. Respondents want to know if the components that are used are still the best choice for Hansken. It should be examined whether the current components within Hansken are efficient and functional in comparison to software that is available today. In addition, the applications must be scalable, in the sense that it will continue to operate well when the load on Hansken's servers increases.

In terms of efficiency, respondents want to know if Hadoop's HDFS is currently the best choice for storing images. For Cassandra, respondents would like to see an alternative where a relational database is regularly mentioned. Functionalities that received attention during the interviews are roll-up updates, always-on capabilities, real-time streaming, alternative search suggestions and better conversation displays for mail, for example, as is possible in Outlook.

Infrastructure

Respondents want to know whether the requirements drawn up by the NFI regarding infrastructure are still sufficient today to make use of production clusters within their organisation. It should be taken into account that the various parties have specific requirements due to the presence of security standards and organisational requirements. Because of this, it is not possible to give an unequivocal answer to what requirements should be set for hardware and what the configuration should look like. The requirements drawn up for this purpose should be examined in broad outline in order to subsequently issue a recommendation for each use case.

Monitoring

Observation shows that respondents have a need for optimisation of their current monitoring set-up. Respondents need more monitoring options, provided directly by the NFI, with accompanying instructions for setting this up correctly.

Multi-tenancy

Some of the Hansken users want to use the infrastructure as broadly as possible when servicing multiple clients. For this purpose, they have been asked to create a multi-tenancy design, indicating what additional requirements this imposes on the infrastructure, software architecture and the organisation of the individual software components.

5 Infrastructure requirements

Theoretical models are used to compile the infrastructure requirements. These models are based on the most important characteristics per component within Hansken's architecture and provide insight into the required capacity per established KPIs as described in section 4.3:

Business KPIs	**Description**	**Technical KPIs**
Data processing (speed)	Data processing in MB/s	• Storage throughput • Network throughput • CPU utilization
Number of users	Quantity of users	• Front-end utilisation • Number of simultaneous MapReduce jobs
Storage volume	Quantity of data	• Total storage

Within this chapter, the following research question is answered:

> *1.* ***What are the technical requirements of the Hansken software suite for the environment in which the product must be installed.*** *Such that the infrastructure is scalable and the Hansken software can perform optimally and also complies with Dutch government security and investigation standards.*

5.1 Hadoop

The Hadoop cluster is used within the Hansken platform for forensic analysis and consists of Zookeeper, HDFS and YARN (MapReduce). After a device is imaged and placed on HDFS, it is analysed by a MapReduce job. Traces found are then made searchable for researchers via the front-end UI.

Within Hadoop environments where MapReduce plays the main role in the execution of analyses, technical requirements for the environment are determined in part by the content of the MapReduce jobs. A MapReduce job that performs many operations on a small amount of data may be more intensive than a MapReduce job that performs few operations on many data. Similarly, analysis on large amounts of small files may produce a higher system load per byte processed than analysis on small amounts of large files.

Within Hadoop, there are several methodologies available that focus on testing a specific part of the system and are capable of creating a baseline of system performance.

To get an accurate picture of how well a specific MapReduce job scales, it is important to ensure consistency in the following areas:

- The analysis (the job).
- The dates.
- The hardware.
- The software.
- The environment (such as network load).

5.1.1 Impact of infrastructure on the performance of a platform

Within a platform, many components determine the performance of software running within that platform. For example, it is likely that software within a virtual machine running on an already (over)loaded CPU will perform less well than when it is allocated cores from a CPU with 0% utilisation. However, these (external) influences on the hardware are often not visible to the OS and monitoring solutions that measure at OS level, which can give a wrong picture of the load and performance of a software solution or platform. When determining the size of an environment, it is therefore important to have insight into the influence of (individual) components on performance.

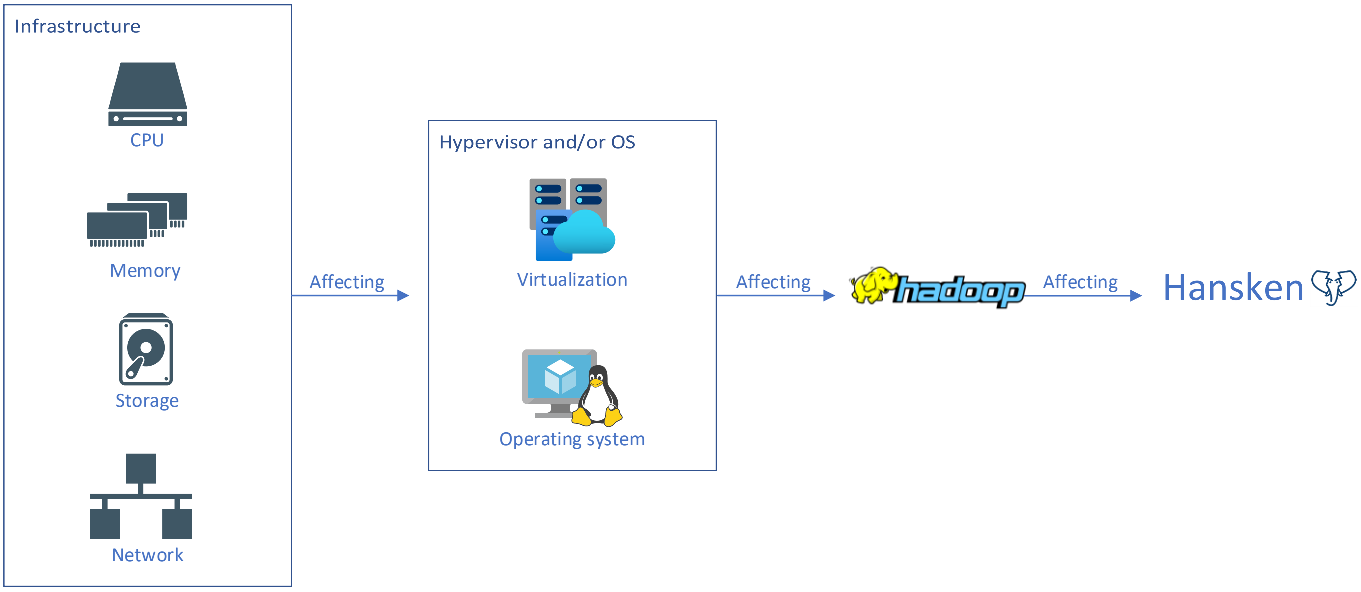

Image 1: Components Hadoop

In order to determine what performance a platform should have, it is important to identify a number of issues:

1. When does a software solution perform "well"? What is the range of "good"?
2. What KPIs are important in assessing this performance?
3. Which components (in)directly influence these KPIs?

In the following section, a model is elaborated for the KPI "data processing".

5.1.2 Model - data processing

In order to determine the data processing speed for Hadoop, a model was created that calculates the expected HDFS throughput based on infrastructure components and aspects. This model was designed based on metrics, standardised benchmarks and experience in the field. As the impact of an extraction on YARN (MapReduce) on a cluster can be so different from a standardised benchmark, it is recommended that a specific benchmark be developed that accurately reflects the workload of an extraction.

Factors

The following factors play an important role in determining the speed of data processing:

Sym	Factor	Explanation and example value

A	Number of nodes	20
B	Number of discs per node	5
C	Maximum sequential throughput per disk	100MB/s
D	I/O workload pattern	The following values can be used: HDD: • 100% sequential: 1.0 • 100% random: 0.025 SSD: • 100% sequential: 1.0 • 100% random: 0.75
E	CPU utilisation impact	The scale runs from 0 to 1 with the following meanings: - 0: maximum impact - 1: no impact Example value: 0.8
F	Network utilization impact	The scale runs from 0 to 1 with the following meanings: - 0: maximum impact - 1: no impact Example value: 0.8
G	Virtualization impact	The scale runs from 0 to 1 with the following meanings: - 0: maximum impact - 1: no impact Example value: 0.8
H	Replication factor	Example value: 3
I	Other impact through utilisation	The scale runs from 0 to 1 with the following meanings: - 0: maximum impact - 1: no impact Example value: 0.9 Impact of: • OS • Storage controller • Background processes • "Noisy neighbours"

Impact factor

Because it can be very complex to estimate the impact of individual factors (such as CPU and network utilisation) on the throughput, it is also possible to work with a generic impact factor. This impact factor corrects the theoretical maximums to realistic values in practice.

This impact factor can be determined by calculating the ratio between the theoretical maximum and the maximum in practice. The maximum in practice can be obtained by subjecting the cluster to a performance benchmark, for example using TestDFSIO.

Sym	Factor	Explanation and example value
a	Number of nodes	20
b	Number of discs per node	5
c	Maximum sequential throughput per disk	100MB/s
h	Replication factor	3
j	Impact factor	0,8 Impact of: • CPU utilization • I/O workload pattern • Network utilization • Virtualization • OS • Storage controller • Background processes • "Noisy neighbours" • Encryption

Formulas

In order to calculate the speed of data processing, the following formulas have been formulated:

1. **Simple - node storage throughput**

This allows the storage throughput per node to be calculated.

$$Simple\ node\ storage\ throughput = (number\ of\ disks * throughput\ per\ disk)$$

2. **Simple - cluster storage throughput**

This makes it possible to calculate the raw storage throughput per cluster without including external factors that may influence the throughput. The input is the result of the formula "1. Simple - node storage throughput".

$$Simple\ cluster\ storage\ throughput = (number\ of\ nodes * node\ storage\ throughput)$$

3. Complex - cluster storage throughput

This allows an estimation of the storage throughput of a cluster using HDFS as file system. It also takes into account the most important environmental factors that can influence the throughput.

$$Complex - \ Cluster\ storage\ throughput:$$

$$(Cluster\ storage\ throughput * I/O\ workload\ type\ on\ disk\ type\ * CPU\ utilization\ impact * network\ utilization\ impact * \left(\frac{1}{replication\ factor}\right) * virtualization\ impact * other\ utilization\ impact)$$

4. Complex - cluster storage throughput - impact factor

Just as with formula #3, this can be used to estimate the storage throughput of a cluster where HDFS is used as a file system. The difference with formula #3 is that a consolidated correction factor is taken into account to bring the total throughput more in line with practice.

$$\boldsymbol{Complex - \ Cluster\ storage\ throughput}:$$

$$(Cluster\ storage\ throughput * \left(\frac{1}{replication\ factor}\right) * impactfactor)$$

Definitions of S/M/L

The defined sizings S, M and L will be used as input for the model. S, M and L are defined as follows:

Environment	Nodes	(Average) data size	Data volume	Number of users	Type of load front-end
S(mall)	50	800GB	200TB	25	N/A
M(edium)	100	2TB	500TB	50	N/A
L(arge)	200	4TB	1PB	100	N/A

Assumptions:

- In a smaller environment, images of a smaller size are stored on average;
- The type of load on the front-end does not affect the outcome of this calculation.

Performance (storage throughput) is considered good when the following condition is met:

- 25% of the total data volume should be analysed within 8 hours.

The acceptable range for this is 20% - 30%.

S/M/L as input for model

In this section, for each size (S, M and L) based on the definition from the previous section, the model is filled in and the cluster storage throughput calculated.

Case 1 - Small

Factor	Value

Number of nodes	50
Number of discs per node	8 HDDs
Maximum sequential read throughput per disk	100MB/s
I/O workload pattern	Mixed random + sequential: 0.75
CPU utilisation impact	0,8
Network utilization impact	0,95
Virtualization impact	0,95
Replication factor	3
Other impact through utilization	0,95

$$\boldsymbol{Simple\ node\ storage\ throughput} = (\boldsymbol{number\ of\ disks} * \boldsymbol{throughput\ per\ disk})$$

$$Simple\ node\ storage\ throughput = (8 * 100MB/s) = 800MB/s$$

$$\boldsymbol{Simple\ cluster\ storage\ throughput} = (\boldsymbol{number\ of\ nodes} * \boldsymbol{node\ storage\ throughput})$$

$$Simple\ cluster\ storage\ throughput = (50 * 800MB/s) = 40GB/s$$

$$\boldsymbol{Complex - \ Cluster\ storage\ throughput}:$$

$$(Cluster\ storage\ throughput * I/O\ workload\ type\ on\ disk\ type\ * CPU\ utilization\ impact * network\ utilization\ impact * \left(\frac{1}{replication\ factor}\right) * virtualization\ impact * other\ utilization\ impact)$$

$$\boldsymbol{Complex - \ Cluster\ storage\ throughput}: (40GB/s * 0{,}75 * 0{,}8 * 0{,}95 * \left(\frac{1}{3}\right) * 0{,}95 * 0{,}95) = 6{,}86GB/s$$

The table below shows an overview of how much data (TB) can be processed in a given amount of time:

Time	**Volume (TB)**
1 minute	0,4
60 minutes	24,1
4 hours	96,5
8 hours	192,9
24 hours	578,8

The acceptable bandwidth translates into the following data volumes:

Criterion	Outcome (TB)
20%	40
25%	50
30%	60

This means that the 25% criterion can be achieved within the following amount of time:

$$\frac{51200GB}{(6{,}86GB/s)} = 7463{,}55\ seconds = 124{,}39\ minutes = 2{,}07\ hours$$

Case 2 - Medium

Factor	Value
Number of nodes	100
Number of discs per node	8 SSDs
Maximum sequential read throughput per disk	300MB/s
I/O workload pattern	Mixed random + sequential: 0.95
CPU utilisation impact	0,6
Network utilization impact	0,8
Virtualization impact	0,95
Replication factor	3
Other impact through utilisation	0,95

$$\boldsymbol{Simple\ node\ storage\ throughput} = (\boldsymbol{number\ of\ disks} * \boldsymbol{throughput\ per\ disk})$$

$$Simple\ node\ storage\ throughput = (8 * 300MB/s) = 2400MB/s$$

$$\boldsymbol{Simple\ cluster\ storage\ throughput} = (\boldsymbol{number\ of\ nodes} * \boldsymbol{node\ storage\ throughput})$$

$$Simple\ cluster\ storage\ throughput = (100 * 2400MB/s) = 240GB/s$$

$$\boldsymbol{Complex -\ Cluster\ storage\ throughput}:$$

$$(Cluster\ storage\ throughput * I/O\ workload\ type\ on\ disk\ type\ * CPU\ utilization\ impact * network\ utilization\ impact * \left(\frac{1}{replication\ factor}\right) * virtualization\ impact * other\ utilization\ impact)$$

$$\boldsymbol{Complex -\ Cluster\ storage\ throughput}: (240GB/s * 0{,}95 * 0{,}6 * 0{,}8 * \left(\frac{1}{3}\right) * 0{,}95 * 0{,}95) = 32{,}92GB/s$$

The table below shows an overview of how much data (TB) can be processed in a given amount of time:

Time	Volume (TB)
1 minute	1,9
60 minutes	115,7
4 hours	462,9
8 hours	925,9
24 hours	2777,6

The acceptable bandwidth translates into the following data volumes:

Criterion	Outcome (TB)
20%	100
25%	125
30%	150

This means that the 25% criterion can be achieved within the following amount of time:

$$\frac{128.000GB}{(32{,}92GB/s)} = 3888{,}21\ seconds = 64{,}80\ minutes = 1{,}08\ hours$$

Case 3 - Large

Factor	Value
Number of nodes	200
Number of discs per node	8 HDDs
Maximum sequential read throughput per disk	100MB/s
I/O workload pattern	Mixed random + sequential: 0.85
CPU utilisation impact	0,8
Network utilization impact	0,85
Virtualization impact	0,95
Replication factor	3
Other impact through utilisation	0,95

$\boldsymbol{Simple\ node\ storage\ throughput = (number\ of\ disks * throughput\ per\ disk)}$

$Simple\ node\ storage\ throughput = (8 * 100MB/s) = 800MB/s$

$\boldsymbol{Simple\ cluster\ storage\ throughput = (number\ of\ nodes * node\ storage\ throughput)}$

$Simple\ cluster\ storage\ throughput = (200 * 800MB/s) = 160GB/s$

$\boldsymbol{Complex -\ Cluster\ storage\ throughput}$:

$$(Cluster\ storage\ throughput * I/O\ workload\ type\ on\ disk\ type\ * CPU\ utilization\ impact * network\ utilization\ impact * \left(\frac{1}{replication\ factor}\right) * virtualization\ impact * other\ utilization\ impact)$$

$\boldsymbol{Complex -\ Cluster\ storage\ throughput}$: $(160GB/s * 0{,}85 * 0{,}8 * 0{,}85 * \left(\frac{1}{3}\right) * 0{,}95 * 0{,}95) = 27{,}82GB/s$

The table below shows an overview of how much data (TB) can be processed in a given amount of time:

Time	**Volume (TB)**
1 minute	1,6
60 minutes	97,8
4 hours	391,2
8 hours	782,4
24 hours	2347,3

The acceptable bandwidth translates into the following data volumes:

Criterion	**Outcome (TB)**
20%	204,8
25%	256
30%	307,2

This means that the 25% criterion can be achieved within the following amount of time:

$$\frac{262.144GB}{(27{,}82GB/s)} = 9422{,}86\ seconds = 157{,}05\ minutes\ = 2{,}62\ hours$$

Application of model to other environments

The model created for calculating the speed of data processing in Hadoop is designed to be used for various environments. In order to remain as close as possible to the KPI "data processing", TestDFSIO was used as a benchmark method during the design of the model. To be able to accurately estimate for Hansken how many nodes are needed to achieve a certain performance (speed of data processing), it is important that a consistent and replicable test is used. Subsequently, the factors within the model need to be attuned to the values associated with the environment and the workload.

Maximum cluster speed determined by bottleneck(s)

The maximum speed of data processing within the cluster is determined by bottlenecks in the environment. Two main situations can be distinguished, each with their own bottlenecks:

1. **Environments running on HDDs**

Main bottlenecks:

- HDDs (mainly with random I/O);
- Network (mainly with sequential I/O).

In environments that use HDDs for HDFS, this does not mean that CPUs cannot form a bottleneck, as this depends entirely on the workload involved in a MapReduce job. However, often the bottleneck will be found with other components.

2. **Environments running on SSDs**

Main bottlenecks:

- CPUs (both random and sequential I/O);
- Network (mainly with sequential I/O).

Because SSDs very rarely form a bottleneck when used for HDFS, bottlenecks are more likely to be found on the CPU and network side.

5.1.3 Model - storage volume

In order to determine the required raw storage capacity for Hadoop, a model was created that calculates this from the amount of data and the replication factor. This model was designed based on documentation and field experience.

Factors

The following factors play an important role in determining the required storage capacity:

Sym	Factor	Explanation and example value
a	Data volume	200TB
b	Replication factor	3

Formulas

In order to calculate the required amount of cluster storage, the following formula has been formulated:

$\boldsymbol{Cluster\ storage\ requirement}: (data * replication\ factor)$

S/M/L as input for model

In this section, the model is filled in for each size (S, M and L) based on the definition from the previous section and the cluster storage requirement is calculated.

Case 1 - Small

Factor	Value
Data volume	200TB
Replication factor	3

$\boldsymbol{Cluster\ storage\ requirement}: (200TB * 3) = 600TB$

Case 2 - Medium

Factor	Value
Data volume	500TB
Replication factor	3

$\boldsymbol{Cluster\ storage\ requirement}: (500TB * 3) = 1500TB$

Case 3 - Large

Factor	Value
Data volume	1PB
Replication factor	3

$\boldsymbol{Cluster\ storage\ requirement}: (1PB * 3) = 3PB$

5.1.4 Conclusion

Within Hansken, Hadoop is responsible for performing analyses on data. Since Hadoop uses a relatively large portion of the platform's capacity, it can be valuable, for example from a cost perspective, to be able to estimate the required performance and thus the required hardware before setting up a cluster. Even after a cluster has been included in the operation, it can be valuable to carry out benchmarks and to compare these with other clusters to gain insight into the (performance) differences between clusters. The framework included in this chapter provides guidance for carrying out these studies.

5.2 Elasticsearch

Elasticsearch is used to index data for various services within Hansken. The technical requirements of Elasticsearch are strongly influenced by a combination of different factors. To estimate the required capacity for an Elasticsearch cluster, the following topics are covered:

1. Use case;
2. Working memory;
3. Networking between nodes;
4. Storage requirements;
5. Snapshots;
6. Shards;
7. Data Nodes.

5.2.1 factors

Use case

When estimating the required capacity of an Elasticsearch cluster, it is important to consider the type of load. An Elasticsearch cluster consists of a number of nodes with specific roles that process or store indexed data, distributed over "shards" (duplicates of data sets). This mainly depends on the usage scenario for Elasticsearch. In addition, the various operations you perform on the data are also an important factor that affects the load on a cluster. These factors are very different for each usage scenario.

For example, a query is a responsive function that can be scoped and therefore can generally be expressed in milliseconds. An ingest command does not have the ability to do this and is therefore more likely to place a heavier load on the cluster.

Working memory

Working memory (RAM) is the primary driver when it comes to performance within search and ingest. When there is insufficient working memory on all nodes due to, for example, an excessive number of user queries, this can cause delays in the system or out of memory and affect responsiveness. Because Elasticsearch is scalable, increasing working memory can directly improve responsiveness and prevent further slowdowns or downtime.

Elasticsearch's architecture is geared towards nodes with an average size of 32GB to 64GB so that no extremely expensive hardware is required to improve performance and responsiveness.

Networking between nodes

The speed at which nodes can communicate with each other over their network has a strong impact on performance within heavy-load clusters. Because Elasticsearch constantly splits and balances indices, a fast network is an important requirement to ensure low latency when executing jobs. The minimum recommended bandwidth for communication between nodes within a multi-petabyte Elasticsearch cluster is 1Gbit/s. Depending on the performance within the cluster, it can be decided to increase this to 10Gbit/s or higher.

Snapshots

When planning for storage capacity, a backup strategy should be considered. Within Elasticsearch, snapshots can be used. A snapshot is a full backup of an Elasticsearch cluster. Snapshots can be used to make regular backups of a cluster without downtime and to restore data from all or part of the cluster. Snapshots are stored in a separate snapshot repository outside the cluster. This means that sufficient storage space must be taken into account for at least one full copy of the Elasticsearch cluster that is being used.

Number of shards

When estimating the capacity, the number of shards also influences the infrastructure requirements. By default, Elasticsearch divides a single index into a specified number of primary and replica shards. All shards within a cluster are more or less a burden on the CPU and working memory. Too many small and/or large shards can cause performance issues and out-of-memory errors.

There are no hard limits on the size of shards. However, shards typically range from 10GB to 50GB. too many small shards can be inefficient, as 10 shards of 1GB will take longer to search than one 10GB shard. A shard larger than 50GB would result in too long a recovery time in case of failure. This should ultimately be decided based on the usage scenario.

Data nodes

The number of nodes is again highly dependent on factors that are ultimately decided by the scenario for which Elasticsearch is deployed. To estimate this, you need to decide how much data is processed, how many resources are made available to a single node (CPU/RAM/DISK), among other things. As mentioned before, the amount of working memory has a big impact on the performance of Elasticsearch. Therefore, the amount of working memory per node can have a big impact on the number of nodes.

Storage requirements

Disk space is a critical factor in the health of an Elasticsearch cluster. Documents are stored within Elasticsearch in the form of an index. The storage of a single index can be distributed over several shards on multiple disks. When an index grows over time through updates or insertions, index rollover can be used. When a certain threshold is reached, a new index is created to continue writing to it.

Storage requirements again depend on the respective usage scenario. Factors such as the expected size of a single index, its retention time and the extent to which it needs to be replicated need to be considered. Finally, Elasticsearch also reserves a portion of disk storage for overhead. This is 5 or 10% for possible error margin and 10% to 15% of the total memory of a node to stay below the watermark limit, leaving space for e.g. logs and internal operations on disk.

5.2.2 Model - storage volume

Formulas

To give an indication of the required storage, the following formula can be applied.

$$Amount\ of\ data\ (GB) = \big(Data\ (GB)\ per\ day * Number\ of\ days\ stored * (number\ of\ replications + 1)\big)$$

$$Total\ storage\ \ (GB) = Total\ data\ (GB) * (1 + 0.15\ disk\ watermark\ treshold + 0.1\ error\ margin)$$

Definitions of S/M/L

The defined sizings S, M and L will be used as input for the model. S, M and L are defined as follows:

Environments	**Nodes**	**(Average) data per day**	**Number of users**	**Type of load front-end**
S(mall)	17	800GB	25	N/A
M(edium)	34	2TB	50	N/A
L(arge)	52	4TB	100	N/A

S/M/L as input for model

Case 1 - Small

Factor	**Value**
(Average) data per day	800GB
Number of days saved	365
Replication factor	1

$$Amount\ of\ data\ (GB) = \left(800GB * 365\ Days * (1\ replication + 1)\right) = \text{584,000GB}$$

$$Total\ storage\ \ (GB) = \ 584.000GB\ * (1 + 0.15\ disk\ watermark\ treshold + 0.1\ error\ margin) = \text{730,000GB}$$

Case 2 - Medium

Factor	**Value**
(Average) data per day	2TB
Number of days saved	365
Replication factor	1

$$Amount\ of\ data\ (GB) = \left(20GB * 365\ Days * (1\ replication + 1)\right) = 1.460.000GB$$

$$Total\ storage\ \ (GB) = 1.460.000GB * (1 + 0.15\ disk\ watermark\ treshold + 0.1\ error\ margin) = 1.825.000GB$$

Case 3 - Large

Factor	**Value**
(Average) data per day	4TB
Number of days saved	365
Replication factor	1

$$Amount\ of\ data\ (GB) = \left(4000GB * 365\ Days * (1\ replication + 1)\right) = 2.920.000GB$$

$$Total\ storage\ \ (GB) = 2.920.000GB * (1 + 0.15\ disk\ watermark\ treshold + 0.1\ error\ margin) = 3.650.000GB$$

5.2.3 Conclusion

Within Hansken, Elasticsearch is responsible for indexing data. Performance of Elasticsearch is influenced by a combination of the described factors and by the functional deployment of Elasticsearch within the Hansken platform. Based on interviews with stakeholders and further research, Elasticsearch has proven to be a suitable component for use within the Hansken environment. Because the described factors for technical requirements are part of a theoretical approach, it can be valuable to test the calculated capacity in practice. This can be achieved by performing a benchmark on the implemented environment. The official Rally benchmark can be used for this[2] . This benchmark is applied by Elasticsearch itself to investigate the performance of an Elasticsearch installation and subsequently implement performance improvements within an environment. In addition, to further optimise Elasticsearch, an investigation can be performed on the configuration with regard to functional deployment. This can identify possible bottlenecks related to the use of Elasticsearch in order to optimise this component within the Hansken architecture.

[2] https://github.com/elastic/rally

5.3 Cassandra

Cassandra is a so-called NoSQL database management system that is highly scalable and can offer high performance within an application landscape. It handles unstructured data well and can handle a high volume of incoming read and write transactions. This is mainly because it uses flexible schedules and no logical categories to handle large numbers of data. It also offers high availability because databases are distributed and no traditional master-slave architecture is used. Because data is stored in different locations (nodes), it does not have a single point of failure, as in a cluster with masters or leaders.

The hardware requirements of Cassandra depend heavily on the use case for which it is used. For example, memory, CPU, disk and network capacity are different in an environment with relatively much static data than in an environment with much volatile data, which is often approached. To estimate the required capacity for a Cassandra cluster, the following topics are covered in particular:

- CPU usage;
- RAM (working memory);
- Disk space;
- Network capacity;
- Type of data;
- Number of nodes.

This section describes the minimum recommended requirements for a Cassandra cluster. Based on performance monitoring, tuning and performing benchmarks, the amount of memory, CPU (cores) and/or disk space may need to be increased.

5.3.1 Factors

Working memory (RAM)

The more working memory a Cassandra node has, the better its read performance becomes. In addition, more working memory ensures that memory tables hold more recently written data in memory, which in turn leads to fewer SSTables that need to be written to disk and fewer files that need to be scanned from disk during a read. Within Cassandra, an SStable is a *sorted string table* which is an immutable data file for each database table stored on disk to which the memory tables periodically write.

The amount of RAM needed depends very much on the anticipated size of the required data that is read relatively often, or *hot* data. It also depends on the structure of the data scheme, but at least 4 GB should be dedicated to Cassandra per node.

For production environments, a minimum of 8 GB per Cassandra node is recommended.

CPU usage

Workloads for writing data will impact CPU rather than working memory. All write operations go to the *commit log* where the database is so efficient that the CPU is the limiting factor. The Cassandra database makes simultaneous use of the CPU cores that are available.

For production environments, a processor with at least 4 cores is recommended, but for data with many read/write operations, an 8-core CPU is preferred.

Network bandwidth

The minimum recommended bandwidth for a Cassandra cluster is 1Gbit/s.

A distributed database system ensures that there are many read and write operations and that a lot of data is also replicated between nodes. For this, it is important that the network between nodes allows traffic without restrictions from (incorrectly) configured firewall rules, for example.

Disk space

Disk space requirements depend heavily on the use and configuration of the databases. The databases write data to disk when data is added to the *commit log* for consistency and when memory tables *are flushed* to an SSTable data file for permanent storage. The *commit log* has different access patterns (read/write ratio) than the pattern for reading data from the SSTables. This is even more important with HDDs compared to SSDs.

The SStables are periodically *compacted*, whereby *compacting ensures* that the performance improves. This is because data is merged and rewritten. In addition, old unnecessary data is discarded. However, depending on the type of *compact* chosen and the size of the *compactions*, the disk usage and volume on the data directory will be temporarily increased during the *compaction* process. Therefore, it is important to reserve some disk space for this on a node. Normally, a limit of up to 80% of the maximum available disk space is used.

When choosing disks for the nodes, it is important to look at the total data capacity required and the expected IOPS (input/output operations per second). In some cases, lower performance disks will be sufficient and in other cases, more nodes with more working memory will need to be added.

For a production cluster, at least two disks per node are recommended, one for the *commit* log and one for the actual data. The *commit* log should be put on its own partition. For both HDDs and SDDs, it is recommended to split commit logs and data directories for performance and reliability. The amount of data disks can be one or more per node and should be large enough for the expected amount of data and fast enough for read operations that are not in memory and for the *compaction* process.

RAID is not required because data is already replicated across the cluster according to the replication factor chosen. Cassandra uses a disk management function (JBOD) that can deal with the failure of a disk by stopping the node or blacklisting the failed disk. No RAID is required on the *commit log* disk either.

To estimate how much usable disk capacity is required to scale Cassandra, a formula that takes the above factors into account is described in the next section.

5.3.2 Disk space per node

The formula for calculating the required writing space per node is as follows:

$$Disk\ space\ per\ node = \left(\frac{Data\ size * \ Replication\ factor}{Number\ of\ nodes}\right) * Overhead\ * \ Compression$$

- **Data size.** This is the (expected) data volume.
- **Replication factor**. In most cases, a minimum of 2 is used to create sufficient redundancy. For a production cluster, 3 is common, to ensure both high availability and a high cluster lifetime.
- **Number of nodes**. Is the number of Cassandra nodes within a cluster. This must be at least as large as the replication factor, usually a minimum of 3. In addition, this number of nodes will need to be increased to reach an acceptable level of available disk space per node.
- **Overhead.** A multiplier for the *overhead* of indexing and non-merged SSTables on the disk. A value between 1 and 2, depending on the amount of indexing and type of data. With a value of 1 there is no overhead because there is (almost) no indexing and it concerns very stable data.
- **Compression.** If compression is enabled, it can be taken into account as a factor in the formula. Normally compression will reduce the data by about half. However, indexing is not compressed so a value of "0.6" should be used as a factor when compression is on. If compression is switched off then a value of "1" can be used.

Definitions of S/M/L

The defined sizings S, M and L will be used as input for the model. S, M and L are defined as follows:

Environments	**Nodes**	**Expected Data Volume**	**Number of users**
S(mall)	5	10TB	25
M(edium)	10	20TB	50
L(arge)	20	50TB	100

Assumptions:

- Data volatility and frequency of data change are not included;
- Two data discs per node are assumed;
- Utilisation limit is 80% for *cluster health*, internal operations and the *compaction* process.

S/M/L as input for model

In this section, for each size (S, M and L) based on the definition from the previous section, the model is filled in and the disk space per node is calculated as well as the usable disk space.

Case 1 - Small

Factor	Value
Number of nodes	5
Replication factor	3
Overhead multiplier	1.6
Compression	0.6
Discs per node	2
RAM per node	8 GB
CPU per node	4
File system overhead	0.1
Utilisation limit	0.8

$$\boldsymbol{Data\ volume\ per\ node} = \left(\frac{\boldsymbol{Total\ data\ volume\ /\ Number\ disks}}{\boldsymbol{Number\ of\ nodes}}\right)$$

$$Data\ volume\ per\ node = \frac{10\ /\ 2}{5} = 1\ TB$$

$$\boldsymbol{Disk\ space\ = Data\ volume * Replication\ factor *\ Overhead\ multiplier *\ Compression}$$

$$Disk\ space\ = 1 * 3\ *\ 1.6 * 0{,}6 = 2{,}88\ TB$$

$$\boldsymbol{Gross\ Capacity} = (\boldsymbol{number\ of\ disks * disk\ space})$$

$$Gross\ capacity\ = (10 * 2{,}88\ TB) = 20{,}88\ TB$$

$$\boldsymbol{Formatted\ disk\ space}\ = (\boldsymbol{Bruto\ capacity} * (\boldsymbol{n} - \boldsymbol{system\ layout\ overhead}))$$

$$Formatted\ disk\ space = (20{,}88\ * 0{,}9) = 18{,}79\ TB$$

$$\boldsymbol{Total\ usable\ disk\ space} = (\boldsymbol{Formatted\ disk\ space * percentage})$$

$$\boldsymbol{Total\ usable\ disk\ space}\ = (18{,}792 * 0{,}80) \approx 15{,}03\ TB\ of\ 20{,}88\ TB$$

Case 2 - Medium

Factor	**Value**
Number of nodes	10
Replication factor	3
Overhead multiplier	1.4
Compression	0.6
Discs per node	2
RAM per node	8 GB
CPU per node	4
File system overhead	0.1
Utilisation limit	0.8

$$\boldsymbol{Data\ volume\ per\ node} = \left(\frac{\boldsymbol{Total\ data\ volume\ /\ number\ disks}}{\boldsymbol{Number\ of\ nodes}}\right)$$

$$Data\ volume\ per\ node = \frac{20\ /\ 2}{10} = 1\ TB$$

$$\boldsymbol{Disk\ space} = (\boldsymbol{Data\ volume} * \boldsymbol{Replication\ factor} * \boldsymbol{Overhead\ multiplier} * \boldsymbol{Compression})$$

$$Disk\ space = 1 * 3 * 1.40 * 0{,}60 = 2{,}52\ TB$$

$$\boldsymbol{Gross\ capacity} = (\boldsymbol{number\ of\ disks} * \boldsymbol{Disk\ space})$$

$$Gross\ capacity = (20 * 2{,}52\ TB) = 50{,}40\ TB$$

$$\boldsymbol{Formatted\ disk\ space} = (\boldsymbol{Gross\ capacity} * (\boldsymbol{n} - \boldsymbol{File\ system\ overhead}))$$

$$Formatted\ disk\ space = (50{,}40 * 0{,}90) = 45{,}36\ TB$$

$$\boldsymbol{Total\ usable\ disk\ space} = (\boldsymbol{Formatted\ disk\ space} * \boldsymbol{Percentage})$$

$$\boldsymbol{Total\ usable\ disk\ space} = (45{,}36 * 0{,}80) \approx 36{,}29\ TB\ of\ 50{,}40\ TB$$

Case 3 - Large

Factor	Value
Number of nodes	20
Replication factor	3
Overhead multiplier	1.5
Compression	1 (no compression)
Discs per node	2
RAM per node	8 GB
CPU per node	4
File system overhead	0.1
Utilisation limit	0.8

$$\boldsymbol{Data\ volume\ per\ node} = \left(\frac{\boldsymbol{Totaal\ data\ volume\ /\ Number\ disks}}{\boldsymbol{Number\ of\ nodes}}\right)$$

$$Data\ volume\ per\ node = \frac{50\ /\ 2}{20} = 1{,}25\ TB$$

$$\boldsymbol{Disk\ space} = (\boldsymbol{Data\ volume} * \boldsymbol{Replication\ factor} * \boldsymbol{Overhead\ multiplier} * \boldsymbol{Compression})$$

$$Disk\ space = 1{,}25 * 3 * 1{,}50 * 1 = 5{,}63\ TB$$

$$\boldsymbol{Gross\ capacity} = (\boldsymbol{Number\ of\ disks} * \boldsymbol{Disk\ space})$$

$$Gross\ capacity = (40 * 5{,}63\ TB) = 225\ TB$$

$$\boldsymbol{Formatted\ disk\ space} = (\boldsymbol{Gross\ capacity} * (\boldsymbol{n} - \boldsymbol{File\ system\ overhead}))$$

$$Formatted\ disk\ space = (225 * 0{,}90) = 202{,}50\ TB$$

$$\boldsymbol{Usable\ disk\ space} = (\boldsymbol{Formatted\ disk\ space} * \boldsymbol{Percentage})$$

$$\boldsymbol{Usable\ disk\ space} = (202{,}5 * 0{,}80) = 162\ TB \text{ of } 225\text{ TB}$$

5.3.3 Conclusion

For optimal *workload*, a maximum disk size per node of around 1 terabyte is recommended, depending on the amount of I/Os. For newer versions of Cassandra (1.2 +) this can be up to 5 TB per node including replication. If the disk size is significantly higher, the number of *nodes* in a cluster should be increased. However, a larger amount of disk space per disk is still possible if sufficient hardware performance can be delivered (e.g. SSDs) and/or if a lot of work is done with static data and relatively little is read or written.

However, there are potential drawbacks if the disk size on a node exceeds 1 terabyte:

- Extreme long times for new nodes due to replication to other nodes;
- Reduces efficiency when remedial actions are required such as when replacing and adding nodes;
- Nodes with more capacity work better when the data is static and low *data* access times;
- Increases *compacting* per node substantially.

A minimum Cassandra production configuration requires 4 cores with 8 GB of working memory per node. However, in cases where Cassandra is used for multiple purposes, such as using the data models and meeting security standards (storing the encryption keys), it may be necessary to increase the working memory per node to 16 or even 32 GB.

6 Security

The environment where Hansken is installed must meet various security and detection standards.

The requirements drawn up on the basis of these standards frameworks influence the final infrastructure and software design. For these reasons, the following standards frameworks are included in the study:

- The Baseline Information Security Government (BIO);
- Decree on information security regulations for the civil service (VIR-BI);
- Police Data Act (WPG).

A selection is made on the standards within the BIO, VIR-BI and WPG that have the most influence on the design of the architecture. In order to formulate an opinion on the infrastructural design, a selection is made of the most decisive measures. The selection was based on interviews with the parties involved and a literature study of the three standards frameworks. [3][4][5]

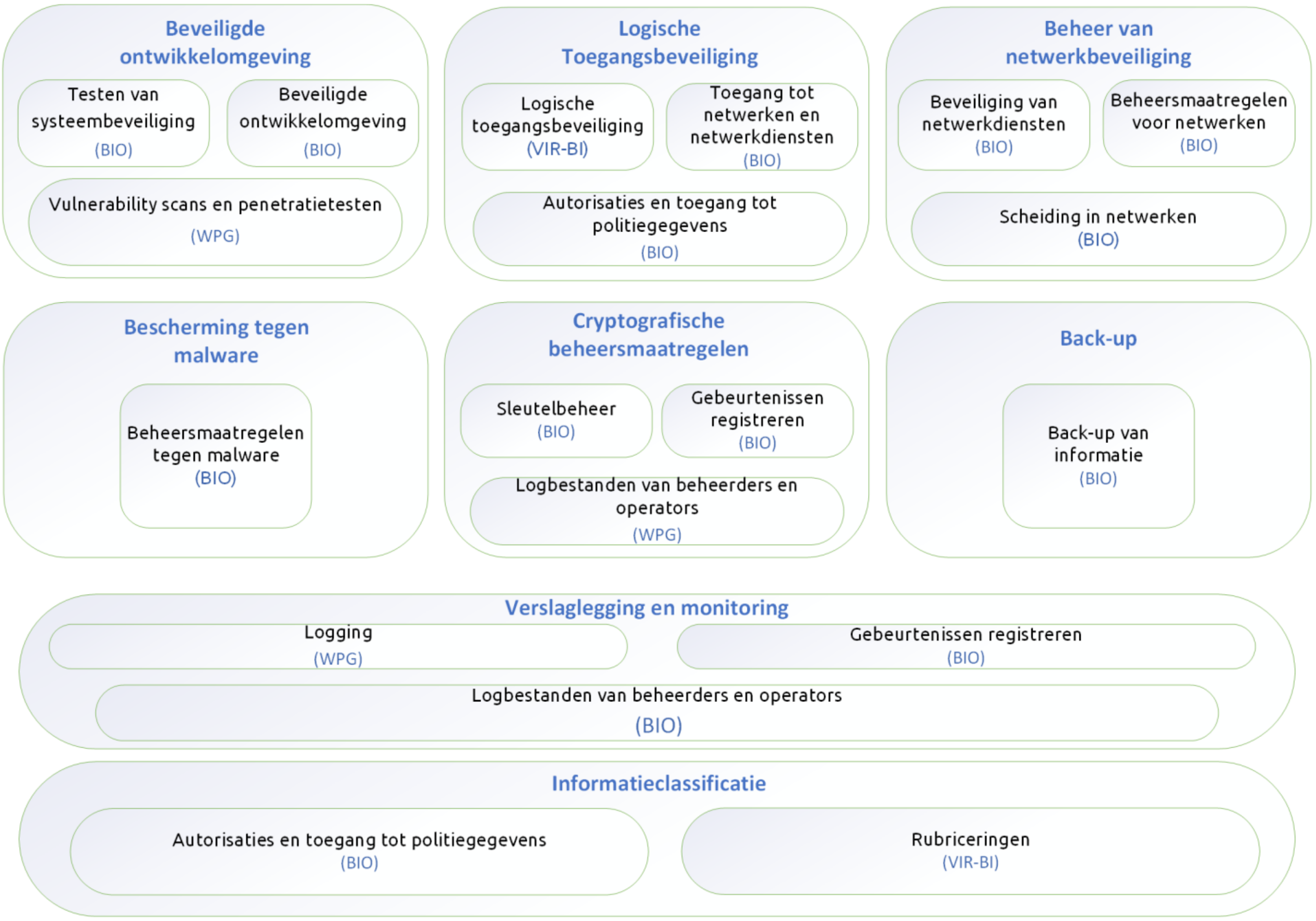
Beveiligde ontwikkelomgeving
Testen van systeembeveiliging (BIO)
Beveiligde ontwikkelomgeving (BIO)
Vulnerability scans en penetratietesten (WPG)
Logische Toegangsbeveiliging
Logische toegangsbeveiliging (VIR-BI)
Toegang tot netwerken en netwerkdiensten (BIO)
Autorisaties en toegang tot politiegegevens (BIO)
Beheer van netwerkbeveiliging
Beveiliging van netwerkdiensten (BIO)
Beheersmaatregelen voor netwerken (BIO)
Scheiding in netwerken (BIO)
Bescherming tegen malware
Beheersmaatregelen tegen malware (BIO)
Cryptografische beheersmaatregelen
Sleutelbeheer (BIO)
Gebeurtenissen registreren (BIO)
Logbestanden van beheerders en operators (WPG)
Back-up
Back-up van informatie (BIO)
Verslaglegging en monitoring
Logging (WPG)
Gebeurtenissen registreren (BIO)
Logbestanden van beheerders en operators (BIO)
Informatieclassificatie
Autorisaties en toegang tot politiegegevens (BIO)
Rubriceringen (VIR-BI)

Figure 2: Overview of security and government standards

[3] Government Departments Special Information Security Regulations Decree - VIRBI (2013 - present)

[4] Police Data Act - WPG (2020 - present)

[5] BIO Version 1 (2020)

Within this chapter, the following research question is answered:

1. What are the technical requirements of the Hansken software suite for the environment in which the product must be installed. Such that the infrastructure is scalable and the Hansken software can perform optimally ***and also complies with the Dutch security government and investigation standards.***

6.1 Logical access security

In a secure application landscape, users should only be given access to what they are authorised to do. One way to achieve this is by applying digital access security within the applications. This means that applications must offer the possibility of doing this by offering it within the selected application.

Logical access protection must be applied to all layers of the environment to make Hansken completely resistant to unauthorised access on the landscape. This means that the possibilities for this must be considered for each component. Within the framework of multi-tenancy, the possibility of authorisation for each component is investigated in chapter 9.2 separate software components.

Measure	**Standard**	**Topic**
Access to an account is blocked after a number of directly consecutive failed login attempts.	VIR-BI	Logical access security
Access to systems can be determined at the group level.	VIR-BI	Logical access security
Access to special information is determined at the individual level.	VIR-BI	Logical access security
Users should only be granted access to the network and network services for which they are specifically authorised.	BIO	Access to networks and network services
Measures have been identified and implemented that verify a user's identity and access rights and ensure legitimate access to data.	WPG	Authorisations and access to police data

6.2 Information classification

Information should be classified according to legal requirements, value, importance and sensitivity to unauthorised disclosure or modification. To ensure this, measures such as the use of a digital classification system that labels information according to these factors should be taken.

In the field of information classification, various processes and measures can be implemented. As these processes may vary from one organisation to another, this may have an impact on the applicability of Hansken as it is developed. The level at which data should be classified can have an impact on the intensity with which information is classified and how this is achieved by processes that are available or still need to be developed.

Measure	Standard	Topic
To label information, an appropriate set of procedures should be developed and implemented in accordance with the information classification scheme established by the organisation.	BIO	Labelling information
Information which must be kept secret because of the interests of the State, its allies or one or more ministries must be given an appropriate level of classification.	WPG	Headings

6.3 Secure development environments

An environment must be made available that is independent of the production environment that is operational and available at the time. This is to prevent exposure to vulnerabilities that may adversely affect the security level and/or use of the platform. Testing should not take place in the production environment unless it has been approved in writing, in which case it may be deviated from.

The possibilities with regard to setting up a secure development environment are influenced by the extent to which applications can be virtualised and how the various components communicate with each other. This was investigated per component in chapter 8.1.1 Front-end components. Because a development environment should be independent of the production environment and may not hinder each other, consideration should also be given to the capacity limits set for a development environment. The design of a secure development environment may therefore differ for each customer.

Measure	**Standard**	**Topic**
Organisations shall establish and appropriately secure secure development environments for system development and integration operations covering the entire system development life cycle.	BIO	Secure development environment
Security functionality should be tested during development activities.	BIO	Testing of system security
Penetration tests and vulnerability scans are performed in a process-based and procedural manner, supported by guidelines, on the infrastructure of the systems in which police data are processed.	WPG	Vulnerability scans and penetration tests

6.4 Protection against malware (BIO)

Applications within the platform must be adequately protected against malware. To achieve this, digital and procedural management measures must be applied for detection, prevention and recovery. An important principle here is that the measures implemented are periodically updated to guarantee the security level.

The possibilities of protection against malware are influenced by the features provided by the individual software components. The various forms of malware require their own control measures. For example, it is essential that applications are kept up-to-date and that data is sent and stored in encrypted form. When a combination of applications hampers the available functionality, the decision can be made to develop customised software, but this can have an impact on performance within the platform. Therefore, research must be carried out beforehand into the practical application and the available protection that the application offers.

Measure	Standard	Topic
To protect against malware, management measures for detection, prevention and recovery should be implemented, together with appropriate user awareness.	BIO	Anti-malware measures
The anti-malware software and associated recovery software used is current and supported by periodic updates.	BIO	Anti-malware measures

6.5 Back-up

To prevent irreparable loss of data, measures must be implemented around data recovery. A backup function must be made available within the platform and a corresponding backup policy must be drawn up and complied with. BIO sets out a number of strict requirements: (a) "data loss is maximum 28 hours" and (b) "recovery time in case of incidents is maximum 16 working hours (two days of 8 hours) in 85% of the cases ". In addition, the recovery procedure must be tested annually or after a major change in order to ensure proper operation.

Due to the large volume of data processed within Hansken, making periodic backups can have a significant impact on the capacity required for the platform. The applications Hansken has implemented allow for data to be backed up, but consideration must be given to the content of the backup, the frequency and the replication factor. This is for the benefit of the large amount of storage capacity required for this purpose.

Measure	Standard	Topic
Back-up copies of information, software and system images should be made and tested regularly in accordance with an agreed back-up policy.	BIO	Backup of information

6.6 Network security management

Security mechanisms should be applied to the network environment in which the platform is implemented. This means that data traffic entering or leaving the organisation must be monitored/analysed by means of detection facilities and that, for wireless connections outside the controlled area, use must be made of encryption means for which the NBV has issued a positive deployment recommendation. In addition, it is necessary to separate networks to prevent access to network segments that a user or service is not authorised to access.

Hansken's network security measures can be implemented by separating network environments and securing these network environments with a firewall that actively manages them. It should be considered in advance which components need to be managed by the firewall and what capacity and throughput speed is required for this, as securing the network can be a bottleneck when processing large amounts of data.

Measure	**Standard**	**Topic**
Networks should be managed and controlled to protect information in systems and applications.	BIO	Management measures for networks
For wireless connections such as WiFi and for wired connections outside the controlled area, encryption means are used for which the NBV has issued a positive deployment recommendation.	BIO	Security of network services
Groups of information services, users and systems should be separated in networks.	BIO	Separation in networks

6.7 Cryptographic control measures

Cryptographic management measures must be used properly to guarantee the confidentiality, authenticity and integrity of information by applying encryption to data. To transfer data that is unreadable for unauthorised users, encryption can be used for *data at rest* and for *data in transit*.

The degree of encryption is in principle strongly influenced by the applications selected within Hansken, per component the possibilities for encryption should be investigated. Within the framework of multi-tenancy, the possibility of data encryption is investigated per component in chapter 9.2 individual software components.

Measure	**Standard**	**Topic**
Policies should be developed and implemented regarding the use, protection and lifetime of cryptographic keys, covering the entire lifetime of the cryptographic keys.	BIO	Key management
To protect information, a policy for the use of cryptographic control measures should be developed and implemented.	BIO	Policy on the use of cryptographic control measures
To protect police data, a policy for the use of cryptographic control measures should be developed and implemented.	WPG	Cryptography

6.8 Reporting and monitoring

In the context of reporting and monitoring, it is important to take measures to ensure that changes to data and actions that are sensitive in nature are recorded and retained for a predefined period of time.

One way of implementing measures for reporting and monitoring is to use a monitoring system. This allows events to be recorded and saved for the purpose of an audit, among other things.

Measure	Standard	Topic
Activities of system administrators and operators should be recorded and the logs protected and reviewed regularly.	BIO	Log files of administrators and operators
The information processing environment is monitored by a SIEM and/or SOC using detection facilities, such as the National Detection Network (only for central government organisations). These are deployed based on a risk assessment, partly based on the nature of the data and information systems to be protected, so that attacks can be detected.	BIO	Recording events
The data controller and the processor are responsible for the logging of processing operations as included in Article 32a, paragraph 1. The organisation uses the logging solely for the purpose of verifying the lawfulness of the data processing, internal checks, guaranteeing the integrity and security of police data and for criminal proceedings.	WPG	Logging

6.9 Conclusion

Customers of Hansken have to deal with various security and detection standards. Implementing measures in accordance with the standards frameworks studied can have such an impact on various architectural principles. Within this study, three standards frameworks were examined: BIO, VIR-BI and WPG. Within these standards frameworks, a selection has been made of the standards that can have the greatest influence on Hansken's infrastructural requirements.

Based on the results of the investigation of the connection of standards within the architecture of Hansken, there is no need to further adapt the technical requirements as described in chapter 5. In the context of infrastructural requirements, it is mainly necessary to reserve sufficient storage capacity for replicating data within Hansken and to set up network security by means of firewalls, whereby these should not form a bottleneck in the communication between components due to a lack of available bandwidth. In addition, for the purpose of reporting and monitoring, it is important to deploy a monitoring system that makes it possible to collect log files generated within the environment. In conclusion, the security and detection standards present mainly relate to the functional deployment and configuration of the software components within Hansken and the processes that have been put in place to guarantee various standards.

7 Monitoring

Part of a professional platform is an adequately designed monitoring solution to guarantee confidentiality, integrity and availability. Disruptions to infrastructure or applications can have such a large impact that business processes are impeded or even rendered impossible in their execution. A properly designed monitoring solution is also crucial for the correct implementation of (predictive) incident and problem management and facilitates the possibility to perform (security) audits. This chapter answers the following research question:

2. Advice on how to adequately set up the monitoring of the platform (Hansken software suite implemented on the infrastructure). What (additional) requirements does this place on the software?

7.1 Types of infrastructure and application monitoring: logs and metrics

Infrastructure and application monitoring can be divided into two components: log files and metrics. The purpose of each component is explained below.

Log files

The purpose of monitoring log files is to identify events that may have an impact on the confidentiality, integrity and availability of infrastructure, applications and business processes. Within log files, events of varying severity can be found that provide a first estimate of the severity of an event. Within the system, it is possible to configure the minimum visible severity of events that can be found in log files.

Metrics

The purpose of monitoring metrics is to identify events in hardware or software measurement data that, like log files, can impact the confidentiality, integrity and availability of infrastructure, applications and business processes.

7.2 Best practices

Within big data environments, the following best practices are considered important for both monitoring metrics and log files. These best practices are often used to optimise platform services within Capgemini.

Determine what is important to monitor based on business processes

Identifying components within the IT landscape that support business processes is a crucial first step in being able to ensure (high) availability, confidentiality and integrity. A top-down approach can help to understand which technical KPIs are important in order to properly execute a business process and ensure control within an environment.

The successful execution of a business process is not only related to the uptime of IT services, but also the timely detection of a large number of login attempts from unknown sources can prevent damage to the confidentiality and integrity of IT components and thus the continuity of a business process.

Perform smoke tests on a regular basis

Regularly performing so-called smoke tests, which functionally test the entire infrastructure and application landscape for the purpose of simulating a business process, provides a reflection of the actual feasibility of these processes within a platform.

Alerting based on thresholds and sudden changes

Taking measurements within a platform is a first step towards knowing what is going on. The next step is to ensure that action is taken when necessary on the basis of measurement data, for example by sending out an alert in response to a threshold being exceeded or a large percentage change. An alert sent out on the basis of certain criteria in one or more log files can also make people aware that an incident is taking place and speed up the implementation of a possible solution.

Define responsibilities within the team

When a threshold exceeded or criterion detected in a log file triggers an alert, it is important that it is clear who should take action and who should be informed of the potential impact. A RACI matrix can help to create clarity in all roles, actions and responsibilities.

Configure a retention time that is long enough

A long retention time can be useful for, among other things:

- Proper execution of the capacity management process (ability to extrapolate growth within the environment based on metrics that provide insight into storage usage);
- Identifying and mitigating security incidents based on metrics or access logs;
- Solving problems based on past incidents.

A short retention period can be cost-effective in the short term, but often does not help (enough) to put information into perspective and to understand a platform sufficiently.

Evaluate on a regular basis whether everything that is important is being monitored

Evaluating on a regular basis whether stored metrics and log files of components within the IT landscape are still adequate and provide a complete enough picture is an important process for monitoring quality. It is often the case that after a certain period of time, some of the things being monitored are less relevant than initially thought and other things are more effective to keep an eye on.

Integrate the monitoring solution within service management processes

The integration of a monitoring solution within the incident management process ensures traceability of (potential) incidents and structure in the follow-up of an alert. In addition, this offers advantages for the problem management process and for a faster solution in the event of a reoccurrence of the incident because a solution from the past can be deployed again (faster). Also, when following up a problem through the change management process, relationships can easily be established with past events.

Create dashboards per role

Both for the display of metrics and the frequency of events, dashboards per role can contribute greatly to the clarity of (crucial) KPIs. By being able to see the status of the most important KPIs at a glance, one can then quickly zoom in on areas that require more attention (for example by switching to dashboards per IT component).

7.3 Current situation

The architecture summary shows that within Hansken Prometheus is used to retrieve metrics and that these metrics are subsequently visualised (by default) using Grafana. These two components are thus jointly responsible for platform monitoring.

All applications and services generate local application and audit logs in which events can be found about behaviour and things that happen. These logs can be read centrally if a Hansken customer has made the necessary arrangements. The deployment of a SIEM solution is also the customer's responsibility.

7.3.1 Metrics - Grafana dashboards

Hansken comes standard with a set of Grafana dashboards that provide insight into the metrics of services that are part of the platform. Below is a list of the missing metrics in the current dashboards:

All services

- Alert: threshold on CPU utilization: 30 min average higher than 90%
- Alert: threshold on memory utilization: 15min average higher than 80%
- Process hangs
- Process restarts

Expert UI, Ganesha

- Number of logged in users
- Top 5 most logged-in users
- Number of login attempts last 24h
- Average number of login attempts per 24 hours
- Number of successful registrations in the last 24 hours
- Number of unsuccessful applications in the last 24 hours

Smoke test: automated login via user in OpenLDAP

Apache Hadoop - overall

- Service health
- Host swap rate

Apache Zookeeper

- Open connections
- JVM heap memory usage
- Garbage collections
- Garbage collection time

Apache Hadoop - YARN (MapReduce)

- Total vcores
- Available vcores
- Running containers
- Pending containers
- Running applications
- Pending applications
- Failed applications
- Garbage collections
- Garbage collection time
- Total memory available
- Total memory in use

Apache Hadoop - HDFS

- Total HDFS storage
- Available HDFS storage
- Used HDFS storage
- Missing blocks
- Under-replicated Blocks
- Garbage collections
- Garbage collection time
- JVM heap memory usage
- Used non-heap memory

Elasticsearch

Health cluster

- Count of active shards
- Relocating shards
- Initialising shards
- Unassigned shards

Request performance

- Query load
- Number of fetches currently in progress
- Total number of queries
- Total time spent on queries
- Total number of fetches
- Total time spent on fetches

Index performance

- Total refreshes

- Total time spent refreshing
- Current merges
- Total merges
- Total time spent merging

Thread pools

- Active
- Queue
- Rejected

Elasticsearch caches

- Field data cache size
- Node query cache size
- Shard request cache size

JVM health

- Memory usage
- Threads
- Garbage collection

Kafka

- Messages in/out
- Network handler idle time
- Under-replicated partitions
- CPU Idle time

Cassandra

- Health cluster
- Total Compactions Completed
- CompletedTasks (commit log)
- Pending Tasks (commit log)
- Compression Ratio

Hansken overview

- Process health
- CPU utilization

Hansken core services

No additions.

OpenLDAP

- Top 5 most logged-in users
- Number of login attempts last 24h
- Average number of login attempts per 24 hours
- Number of successful registrations in the last 24 hours
- Number of unsuccessful registrations in the last 24 hours

- Operations
- Connections (total/current)
- Waiters read
- Waiter write

Keycloak

- Top 5 most logged-in users
- Number of login attempts last 24h
- Average number of login attempts per 24 hours
- Number of successful registrations in the last 24 hours
- Number of unsuccessful registrations in the last 24 hours

Document viewer

No additions.

Grafana

No additions.

7.3.2 Log files

Monitoring log files is an important part of the overall monitoring of the platform. By mapping which events are important indicators for the (dis)functioning of the platform, it can be measured how often and when they occur. This way, a quick response can be given after an event has taken place and an alert is sent out.

It can be valuable to have an overview of (a certain type of) events when:

- The severity is at least WARN(ING) or ERR(OR);
- An event that occurs more frequently and requires intervention to prevent (or resolve) a disruption;
- It is related to security (e.g. login attempts);
- This information provides information about the load on the platform.

By having charts (based on predefined queries), placed on dashboards, automatically updated by the monitoring solution, the user gets a more accurate picture of the current state of (a part of) the platform.

7.4 Requirements for monitoring solution

A good monitoring solution adds value to the platform in many ways. These include increasing stability and allowing audit logs to be searched quickly. In particular, the following features are very important to have within a monitoring solution in order to monitor effectively:

- Enables users to perform (automated) queries on (a selection of) log files;
- Has a retention time that matches the requirements of the business;
- Is able to send out alerts;
- Is able to archive data.

It is also important that sufficient resources are allocated to the (virtual) machines on which the monitoring software runs.

7.5 Requirements for Hansken

In order to guarantee the proper functioning of Hansken, a number of aspects are important to monitor. It is important to determine where possible bottlenecks (may) occur and to draw up KPIs based on these bottlenecks. Subsequently, these KPIs must be monitored by means of metrics or events.

The following KPIs are leading within the models:

1. Speed of data processing
2. Required storage capacity
3. The possible number of users
4. The type of load (on the front-end)

These KPIs can be shaped using the following metrics:

7.5.1 Speed of data processing

- Average processing time per GB (of all operations within a case)
- Occupancy of MapReduce jobs queue
- How many containers (vcores) per (type of) MapReduce job in use

7.5.2 Required storage volume

- Average retention period of a case
- Average case size
- Total used storage capacity of a case

7.5.3 The possible number of users

- Number of simultaneously logged on users
- How quickly documents or media are opened (document viewer)

7.5.4 The type of load (on the front-end)

- Characterisation of logged on users (so that load on the rest of the system can be related)

Finally, frequent smoke tests can provide insight into both the health and the performance of components within a platform. By being aware of poor performance or a component behaving strangely at an early stage, (larger) incidents at peak times can be avoided.

7.6 Conclusion

A good monitoring solution is crucial for gaining insight into how a platform is performing. By having insight into metrics as well as log files, users know better and faster what the current state is. A number of best practices have been defined for (big data) platforms that are (or can be) important for effective monitoring. It is recommended to make a mapping based on these best practices in order to validate the current situation and subsequently to investigate which of the missing best practices can still add value.

Within Hansken, Prometheus and Grafana are used to centralise and visualise metrics. Hansken provides dashboards that give insight into the load on individual components of the platform. These dashboards have been validated and a list of valuable metrics that are currently missing from the dashboards has been compiled for each component.

To centralise log files, the customer must deploy its own (monitoring) solution. It is important that this solution has functionalities that enable users to easily and effectively monitor the health of the platform.

8 Greenfield approach

In this chapter, the greenfield approach is explained. After a thorough analysis of the current situation, (potential) bottlenecks have been identified and optimisations have been implemented to bring the architecture closer to industry best practices. Within this chapter, the following research questions are answered:

> *1. What are the technical requirements that the Hansken software suite imposes on the environment in which the product must be installed.* ***Such that the infrastructure is scalable and the Hansken software can perform optimally*** *and also complies with Dutch government security and investigation standards. "*
>
> *3. An advice on the type(s) of configuration of the infrastructural environment (including which clusters to distinguish) and what are the main characteristics of this environment? Which types of configuration are excluded in advance?".*
>
> *4. An advice on the design of the individual software components. In combination with the Hansken software."*

The multi-tenancy design is not part of this chapter and will be discussed later in the document. It does discuss whether or not a component should be virtualised, whether the product is still *fit for purpose* and any recommendations.

8.1 Virtualisation matrix

The tables below maintain the subdivision shown in Figure 8 (Hansken infrastructure (production example)) on page 26 of the Hansken Architecture Summary. The tables indicate which products it is recommended to virtualise.

8.1.1 Front-end components

Product	**Virtualise?**
Document Viewer	Yes
Grafana	Yes
Keycloak	Yes
Gatekeeper	Yes
Nginx	Yes
Keystore	Yes

8.1.2 Platform core components

Product	**Virtualise?**
Apache Zookeeper	Yes
Kafka	Yes
Prometheus	Yes
OpenLDAP	Yes
Repository server	Yes

Project case	Yes
Trace	Yes
Data	Yes
Extraction	Yes
Resource	Yes
Lobby	Yes
Preference	Yes
Keystore	Yes

8.1.3 Back-end components

Product	**Virtualise?**
Apache Hadoop	Yes
Elasticsearch	Yes
Cassandra	Yes
RDBMS	Yes

8.1.4 Other components

Product	**Virtualise?**
Ansible	Yes
LUKS keystore	Yes
OpenStreetMap server	Yes

8.2 Product vision

8.2.1 Apache Hadoop

Virtualisation

It is often recommended to deploy Hadoop on bare metal in order to eliminate as much overhead as possible, including that caused by virtualisation. However, there are also ways to optimise Hadoop's performance in a virtualised environment while taking advantage of the benefits of virtualisation. Decoupling compute and storage is one of the elements of a successful virtualised Hadoop environment. By using physical servers purely for supplying CPU power (compute), extra capacity can be deployed within the hypervisor as needed for specific purposes, such as extra YARN/MapReduce nodes. (Object) storage must then come from a storage array (that has the HDFS API) that is accessible via the network, for example. In this way, the following advantages, among others, can be realised:

More efficient use of storage capacity

- Data protection through erasure coding significantly reduces storage requirements;
- Data compression reduces storage requirements;
- With smaller Hadoop clusters, the storage performance immediately increases.

More effective use of computing power

- Automated deployment/activation of Hadoop VMs based on demand;
- Hardware can be deployed multi-tenant;
- Unused hardware can more easily be used for other purposes where there is more demand.

The main drawback of such a solution is that the software on the storage array can be restrictive when rolling out new features within Hadoop. For example, the latest version of Hadoop may also require a software update of the storage array to update the HDFS API. You also need to be aware of the support for (Hadoop) applications in combination with the storage array: YARN and HDFS will often be fully supported without any problems, but in the case of more specific products such as Ranger or Sentry, the use of ACLs may mean that they are not supported.

Deployment of other storage APIs

Within Hadoop, HDFS is traditionally the most widely used solution when it comes to providing object storage. Besides HDFS, other forms are possible, such as ABFS, S3 and Ceph (which partly uses the S3 API). These other forms are aimed at external storage clusters that are managed outside of Hadoop and can therefore take advantage of the benefits that cloud solutions bring.

The success of the deployment of another storage API is often closely linked to the extent to which the API is used within the market (gets exposure). With more exposure, an API will be given more time and attention, which means more features will be available (or will become available) and bug fixes will be implemented more quickly. Lesser used APIs may not be compatible with the latest versions of Hadoop, or things like encryption, authentication or authorisation may not be (fully) supported or may offer less performance. For these reasons, within production environments, technology is almost always chosen that has already proven itself or for which the exposure is so high that it is very likely that new versions (containing new features and bug fixes) will be made available quickly.

Fit for purpose

Hadoop plays a crucial role in the Hansken landscape by storing image data and performing extractions. Extractions are performed by MapReduce, and the applications for these are developed by the development team. Extractions fulfil the need to extract traces from data and store them so that they can be searched. On a regular basis, a new bundle of the Hansken suite is made available in which new ways of extracting data have become part of the application and can be used when necessary.

There are various ecosystems and applications available in the market that can handle both the processing and analysis of data. It is important that an ecosystem or application is chosen that its developers can use to its full potential. This means that a potentially more efficient application is (potentially) less effective when developers have little experience with or knowledge of it than a potentially less efficient application. It is therefore important to choose an ecosystem, application or even programming language that the developers can handle well or quickly build up maturity in. For Hansken, this maturity seems to exist for the development of MapReduce applications.

Potentials

1. Scalability through dynamic deployment of virtual machines and storage centralisation

By using Hadoop cluster scaling out, a significant amount of processing power can be added to a cluster in a short period of time. At a later time, the same virtual machines can be scaled down again (stopped or removed) making capacity available for other purposes.

An important prerequisite for this to happen is that the object storage that Hadoop uses for HDFS is provided via a centralised storage solution. When this is separate, stopping and/or removing virtual machines has no impact on data availability.

2. More effective queuing within YARN

Within Hansken, there is currently a limit of five simultaneous YARN jobs. This limit is set to prevent a job within YARN from having too few resources and therefore taking longer than desired for an extraction. One of the side effects of this is that when a job is in the last phase of the extraction, it can only use a few percent of the total calculation capacity and therefore leaves a lot of calculation capacity unused.

The graph below shows the *vcore usage* of a job:

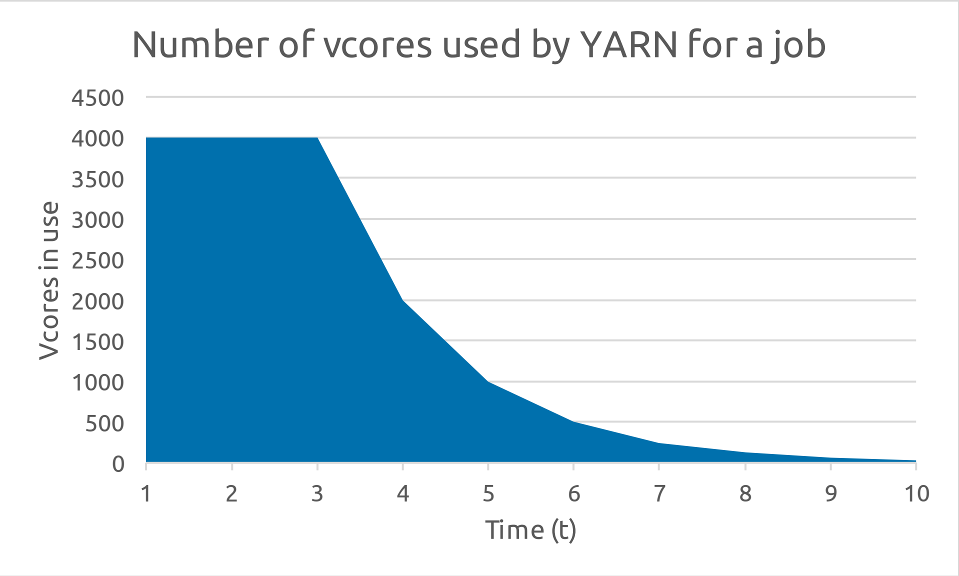

Image 3Use vCores YARN

Optimising the application by looking at whether more parallelisation is possible in the last stages of the job can shorten the so-called "tail". Also, dynamically determining the limit on the number of simultaneous YARN jobs can ensure that the total computing capacity of the cluster is used more effectively. By looking, among other things, at the ratio of the number of *vcores* available for containers within YARN to the total number of *vcores*, one can, for example, choose to admit a new job when the previous job is at t=5. The already running jobs within YARN must be protected by the Capacity Scheduler against taking away too many resources by new jobs.

3. Apache Ambari as a management tool for Apache Hadoop

Apache Ambari provides the ability to *provision*, manage and monitor Apache Hadoop clusters through a web UI and multiple REST APIs. As Hadoop plays a central role in the Hansken landscape, it can be a valuable addition to the overall monitoring solution to anticipate potential problems or subsequently resolve them more effectively.

8.2.2 Hansken front-end components

Within Hansken, Document Viewer is used as an intermediary between the front and back-end. The software ensures that raw data from HDFS, for example, is converted into readable text in the front-end. Document Viewer is often heavily loaded, as a result of which users do not always have an optimal experience during use.

Since heavy load on this service leads to a noticeable drop in performance, it can be interesting to have several instances of it active in combination with a load balancer. The containerisation of this service can also help to respond intelligently to sudden peaks in demand.

8.2.3 Hansken microservices

The following components are categorised as Hansken microservices:

- Trace service;
- Extraction service;
- Data service.
- Case project service;
- Resource service.

The microservices act as intermediaries between the front and back-end. Depending on the nature of an operation, it runs through one of these microservices so that a single click in the front-end can result in several actions in the back-end.

At busy moments, for example during extractions, these services are subjected to such a load that it can be interesting to think about dynamic scalability and the load balancing algorithm. This means that at busy moments an extra service node becomes active within a few minutes, after which it will become inactive again at a quieter moment. A container orchestration platform such as Kubernetes can take care of this process. The use of a different load balancing algorithm can ensure that a new request will go to the node with the lowest load (for example, CPU load or number of simultaneous requests) instead of to the next available node. This prevents one specific node from being overloaded (for example 100% CPU load) and the other available nodes from being underloaded.

8.2.4 Elasticsearch

Within the Hansken platform, all traces found in Hadoop (MapReduce) are indexed by Elasticsearch. A *case investigator* can then search on these traces, where Elasticsearch will perform a query that refers to the data on HDFS.

Fit for purpose

Elasticsearch still plays an essential role within the Hansken landscape when searching for traces in criminal cases. Furthermore, no remarkable bottlenecks emerged in the study. For on-premise environments, Elasticsearch is still market leader when it comes to search and index functions.

Potentials

However, Elasticsearch is increasingly used in the market in combination with Kibana, Beats Logstash, all part of the Elastic Stack. This could be interesting for the Hansken project to investigate additional ingest and visualisation possibilities.

8.2.5 Cassandra

Within the Hansken platform, Cassandra is used for several purposes, namely storing specific case project data, trace data resulting from a case and/or extraction and storing encryption keys. All these purposes were also discussed during the interviews. In addition, Cassandra is also used for storing user preferences within the Hansken application and as a resource database for storing data models, as also mentioned in the Hansken Architecture Summary.

Fit for purpose

Cassandra is currently also largely used for data that is relational in nature, such as specific project data and trace data. It would therefore be better to replace Cassandra, a NoSQL solution, with a relational database management system. This is currently being worked out in more detail by the parties as indicated during the interviews.

Potentials

Cassandra can continue to be used for the keystore and possibly the data models. However, it is necessary to However, it should also be investigated whether HBase offers a possible alternative solution for the required NoSQL functionality.

8.2.6 Configuration management

Within the platform, Ansible is used for configuration management. Because Ansible often has a lot of privileges to enable installation and configuration of software and/or the OS, there is a chance that unauthorised or even authorised people, intentionally or unintentionally, can cause a lot of damage. The likelihood of this happening increases when a single Ansible instance is used and has access to all environments. Isolation, whereby only one Ansible instance is used per environment or per type of environment (e.g., for all development environments), can increase the confidentiality, integrity, and availability of an environment.

8.2.7 Security

Within the platform, various security measures have been taken on several levels as standard. Using *data at rest* and *data in flight* encryption as much as possible can make an important contribution to guaranteeing confidentiality and integrity within an environment. Even if parts of an environment are separated in terms of the network and only allow traffic on specific ports, an infection, vulnerability or malicious party can manipulate or intercept network traffic.

8.3 Greenfield design

The greenfield design was drawn up after a careful analysis of the current situation. The architecture summary, SIG code review and interviews were used to arrive at a design that takes *industry best practices into account* and tries to provide a solution for possible bottlenecks that have emerged.

Important focal points within this design are:

- Authentication and authorisation across the board.
- Encryption for both *data in flight* and *data at rest*.
- Network segmentation will be maintained at least as it is today.
- A decoupling of object storage from the rest of the (*compute*) hardware.
- Scalability (*burstability*) plays an important role where it is needed.

- The use of components for their intended purpose (*fit for purpose*).

No insurmountable technical problems are anticipated, provided the above spearheads can be met. For an optimal cloud implementation, it is therefore important that the object storage is disconnected from the compute hardware, which is also the optimal solution within a (public) cloud environment.

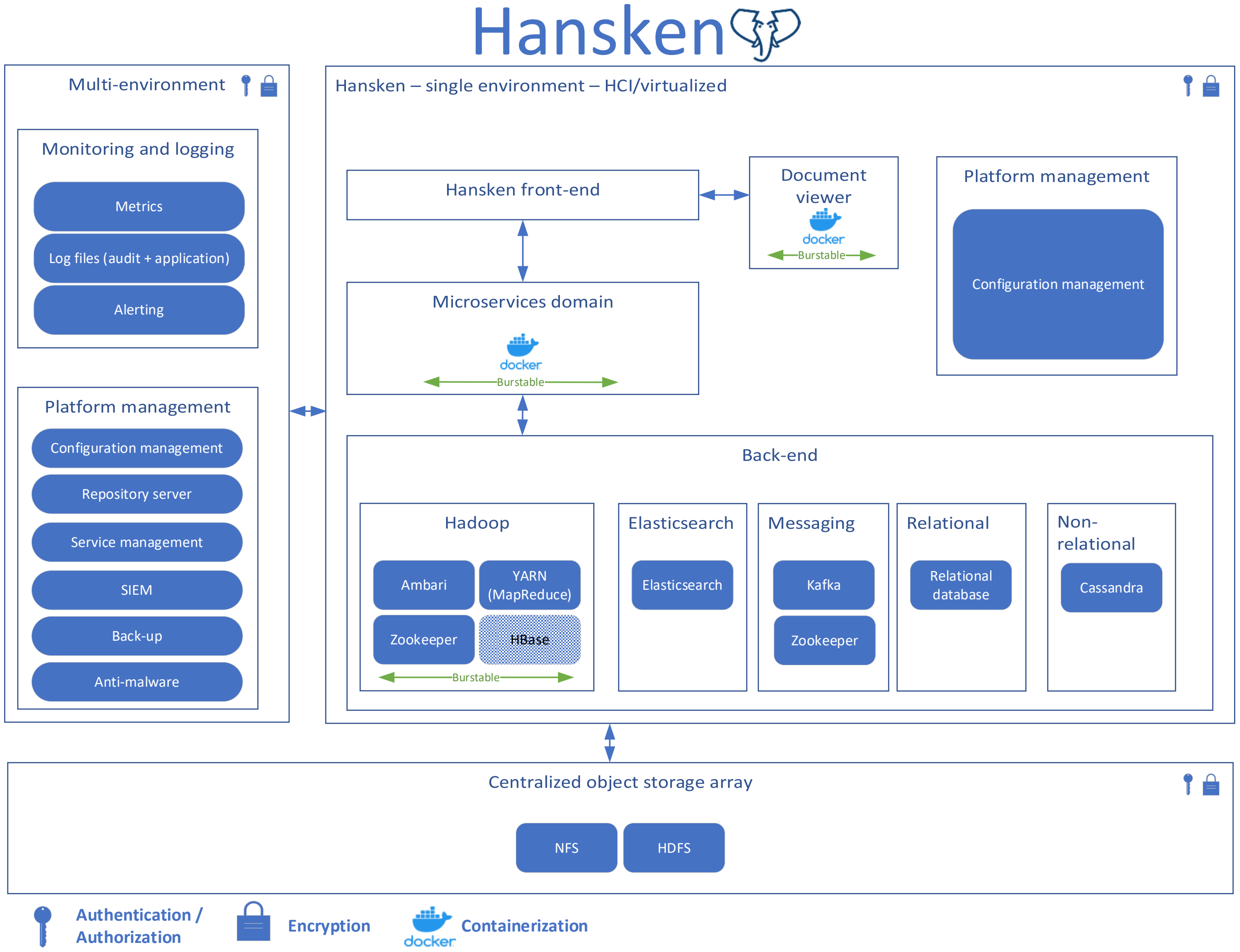

Image 4Greenfield

8.4 Looking to the future

The *to-be* situation as depicted in the previous paragraph reflects parts of an architecture that is often used by organisations to host professional big data environments. In addition to the matters that are reflected in this design, there are also a number of other matters that may add value in the (near) future:

1. **Deployment of Spark to (partially) replace MapReduce**

Spark is part of the Hadoop ecosystem and, like MapReduce, uses YARN as a resource manager/negotiator. Since the deployment of Spark can lead to faster and/or more efficient processing of data, it may be valuable to see if Spark can contribute to optimisation of extractions.

2. **Deployment of containerisation for more components within the platform**

In addition to the (additional) use of virtualisation, the use of containerisation for more components can offer a number of extra advantages. It can provide (more) standardisation within a platform, but also an even better coordination of the supply of (available) resources with demand. It is important that a container orchestration platform (for example, Kubernetes) is also part of the total solution.

9 Multi-tenancy

Some of Hansken's stakeholders have the desire to use the infrastructure as broadly as possible when providing a service to multiple customers. This chapter answers the following research questions:

5. What (additional) requirements does a multi tenancy design impose on the setup of the individual software components/products in combination with the Hansken software?

6. What (additional) requirements does a multi tenancy design place on Hansken's software architecture?

7. What (additional) requirements does a multi-tenancy design impose on the organisation of the infrastructure?

8. Provide advice on where the cut-off between the common/central part and the individual tenants can best be made and more specifically where to make it (in the code or the infra).

9. Assess the multi-tenant design of the FIOD and advise on its optimal implementation, based on the current Hansken architecture. Include the way in which the NFI has implemented it. Assess this implementation and, if possible, define options for improvement.

The possible advantages of using the platform as multi-tenancy include:

- It is relatively cheaper compared to a dedicated tenant architecture, where each customer environment has its own hardware and has to be managed;
- A way to set up pay-per-use models for individual customers with less overhead;
- Customers are relieved of their worries because many processes (such as update management, lifecycle management, etc.) are taken care of by the service provider;
- Tenants do not need to worry, or worry less, about the hardware on which their data is hosted;
- The architecture is relatively easy to scale.

However, a multi-tenancy platform may also have its drawbacks:

- Multi-tenant applications tend to be less flexible due to dependencies on the provider;
- Multi-tenancy is generally more complex than single-tenancy;
- Multi-tenant applications have more stringent requirements when it comes to authentication and access control to prohibit access to (someone else's) data;
- A tenant environment can be troubled by another customer in another tenant, so-called *noisy neighbours, which* can of course have an impact on performance and accessibility of services;
- Maintenance windows of the service provider may not always match the wishes of customers, resulting in downtime or subsequent updates/upgrades.

Of course, these disadvantages can be mitigated by good governance and careful consideration of how hardware and software components relate to multi-tenancy deployment. For a safe realisation of a multi-tenant environment, a number of additional requirements must be set for the following components:

- Setting up the underlying infrastructure;
- Set-up of the individual platform components (e.g. HDFS, Elasticsearch and Promotheus);
- The software architecture of Hansken Core itself.

These components will be discussed further in the following two subsections.

9.1 Infrastructure set-up

With regard to infrastructure, we distinguish three possibilities for a hardware architecture, namely; single tenant with dedicated hardware, multi-tenant hyperconverged and multi-tenant with external object storage. The three possibilities are explained in more detail below on the basis of three architectural plates with a description of the advantages and disadvantages.

Scenario 1 - single tenant - dedicated hardware

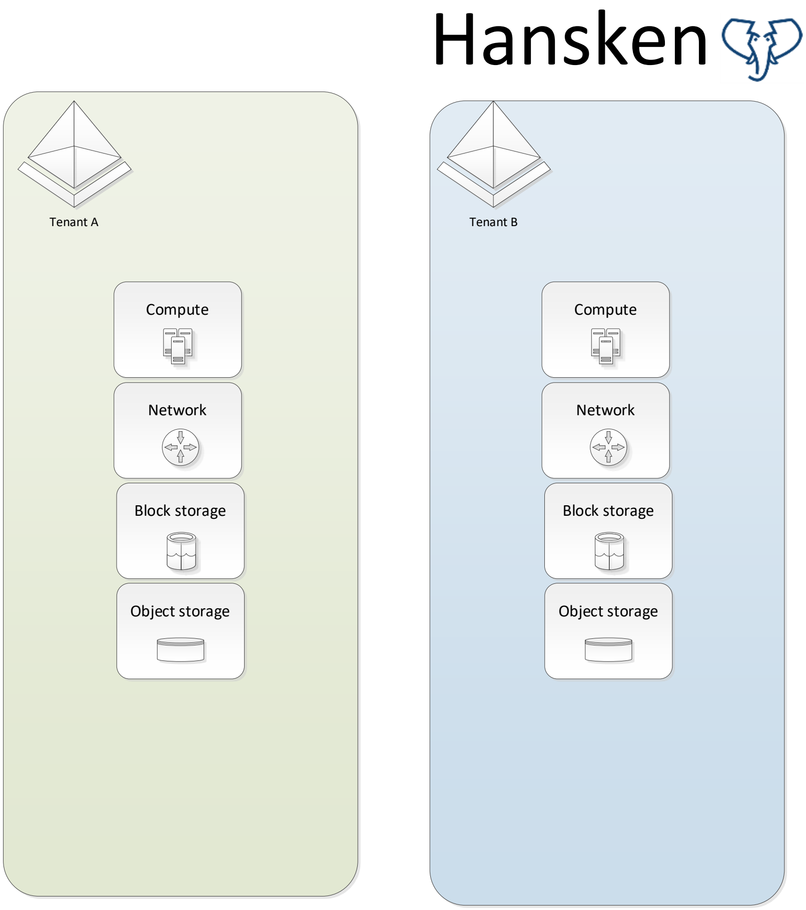

Figure 5Single tenant - dedicated hardware

The first scenario is not a multi-tenant platform, but concerns completely isolated environments in terms of infrastructure. This is similar to the current set-up used by the various parties. Each tenant has its own *dedicated* hardware and no infrastructure is shared between the tenants.

Benefits

- **Stability** because there are no dependencies between tenants and organisations.

Disadvantages

- **Higher costs** because hardware is deployed for each individual tenant and organisation;
- **More difficult to manage** because software runs on different hardware from different suppliers;
- **Relatively higher operational costs** because each tenant requires its own management team.

Scenario 2 - multi-tenant - external object storage

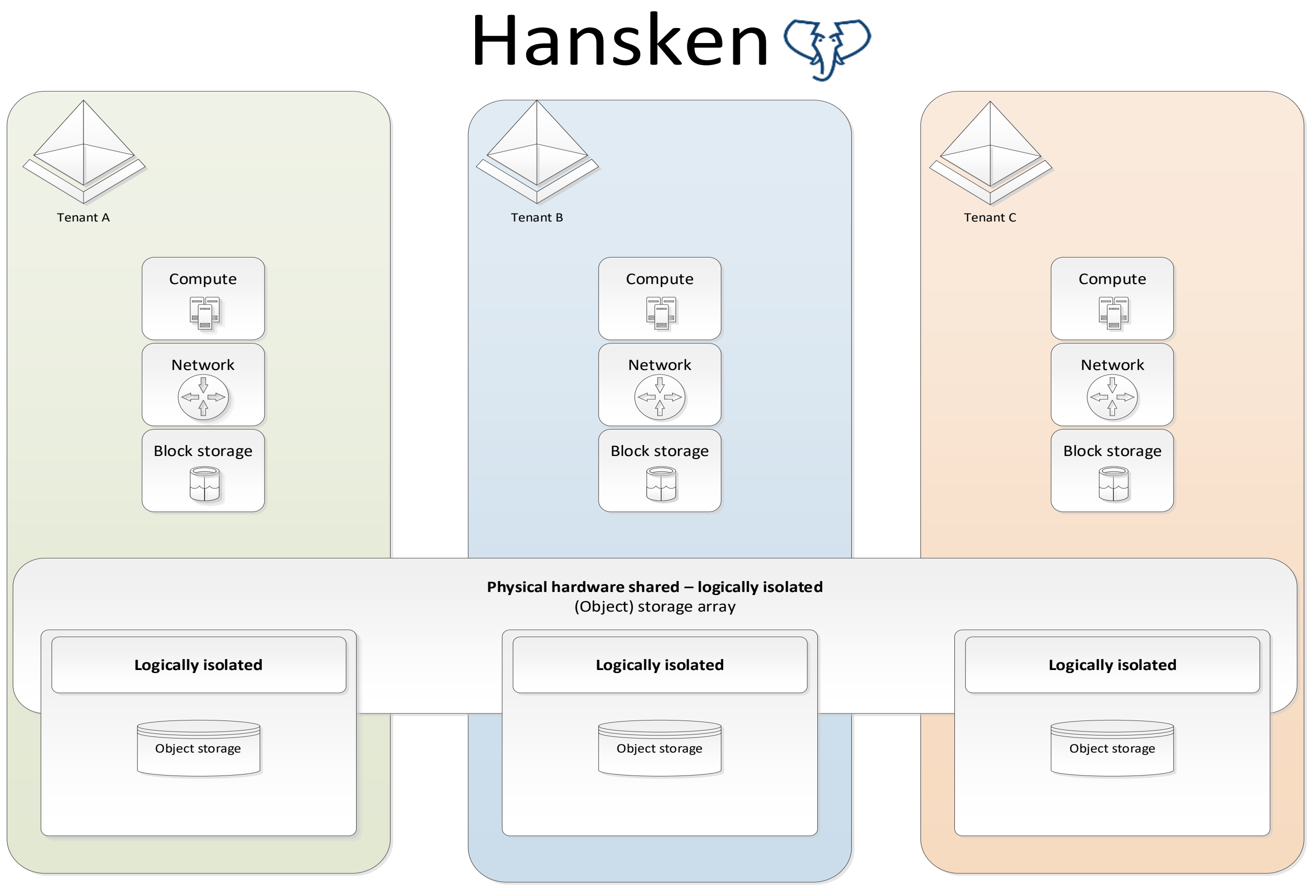

Figure 6 *Multi tenant - external object storage*

In the second scenario, there is a multi-tenant platform where each tenant still has its own physical servers with their own block storage with a compute and own network layer. However, the object storage is shared externally across the tenants.

Benefits

- **Reduced operational management costs** through shared external object storage;
- **Scalability,** making it easier to expand storage as data volume grows;
- **Lower costs** than a traditional infrastructure solution.

Disadvantages

- **Relatively more expensive than** a fully hyper-converged infrastructure;
- **Complexity** due to integration of different tenants with various vendors' hardware with external object storage.

Scenario 3 - multi-tenant - shared infrastructure - (hyper)converged

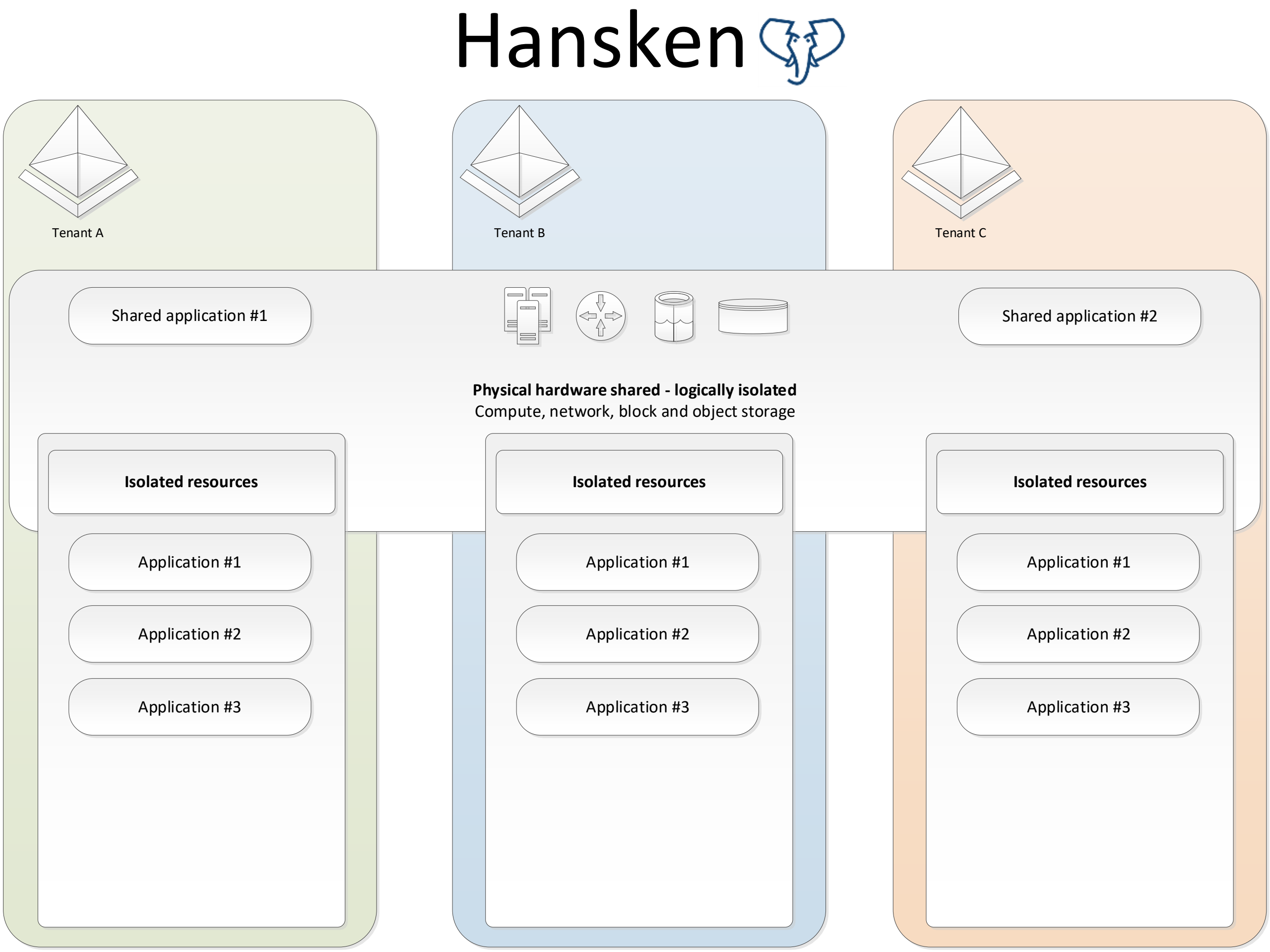

Figure 7Multi tenant - Shared infrastructure - (hyper)converged

The last scenario and also the one we recommend is a multi-tenancy (hyper)converged infrastructure, where there is a logical separation of computing power, block storage and networks between tenants. Also, there is still an object storage shared that is external, just like in scenario 2.

Benefits

- **Operationally easier to** manage through automated management and a central administration interface;
- **Scalability,** makes it easier to grow flexibly in terms of resources on all components;
- **Lower costs** than a traditional infrastructure solution.

Disadvantages

- **Vendor lock-in**, because of the (hyper)converged infrastructure it is possible to become dependent on one hardware supplier.

9.2 Individual software components

The requirements for making a component multi-tenant can be estimated along three main lines:

- Isolation of data with regard to authentication, authorisation and encryption;
- Isolation of user work space (name spaces);
- Isolation of performance with associated quotas.

The components will be rated high, medium or low according to these criteria. The reasons for this assessment are described in more detail. Per relevant component or component group, a table follows with the criteria and assessment. The assessment is based on best practices in the market, information from the supplier, our expertise with the product and how it is implemented within Hansken according to the various interviews and the architecture summary. In addition, a number of other factors are mentioned that may play a role in whether a component is suitable for multi-tenancy or whether it should remain available dedicated to a tenant.

Type of Isolation	Explanation
Authentication	User cannot authenticate with a domain from another tenant.
Data authorisation	Users can only access data for which they are authorised and not data from another tenant.
Data Encryption	Data is both non-crypted *in transit* and *at rest* and cannot be decrypted between tenants.
User spaces	Users have their own workspace (name space) and no access to anyone else's workspace.
Performance with quotas	Users are not troubled by *noisy neighbours*, as good performance quotas are in place. "Tenant X users are not allowed to use more than X% of the total capacity.

Hadoop

Isolation	Possibility	Component Hadoop (HDFS)
Authentication	High	Kerberos is one possibility.
Data authorisation	High	Authorisation can be performed by means of ACLs.
Data encryption	High	*Data at rest, data in transit* (flow encryption of Hansken itself and Hadoop encryption)
User spaces	High	User directories can be created.
Performance with quotas	High	Through YARN, quotas can be set.

Conclusion

Using Kerberos, it is possible to have a Hadoop cluster authenticate with multiple tenants. In addition, *ACLs can* be assigned at the file or folder level to force permissions for specific roles.

YARN can be used to maintain performance quotas. Queues can be created, where one or more queues can be linked to various tenants, each with its own weight, a performance quota with a maximum limit on computing power and memory usage. With YARN, it is possible to enforce *scheduling* so that each tenant on average receives an equal share of resources over time.

Our advice: **do** deploy multi-tenant if Kerberos, ACLs and scheduling are set up properly.

Elasticsearch

Isolation	Possibility	Component Elasticsearch
Authentication	Average	Limited with the free licence. Possibility of LDAP, AD and SAML authentication, among others.
Data authorisation	High	This is possible both at document and field level.
Data encryption	High	Index search: can use an encrypted file system but ES indexes themselves are not encrypted. Encryption *in transit*: Node inter-communication supports TLS and certificates. TLS is also supported between the client and the Elasticsearch cluster. All plugins can also use TLS.
User spaces	Average	Two options are possible; each tenant gets its own index or several tenants share one index.
Performance with quotas	High	Data tiering is possible via index lifecycle management.

Conclusion

When dividing Elasticsearch to multiple tenants, it is possible to create separate indices per tenant or to let multiple tenants share one index. Both options are explained with advantages and disadvantages.

When using the single index per tenant option, there is a minimum working memory cost that needs to be taken into account for each shard. This basic working memory cost applies even if the shard itself does not contain any documents (data). And because each index needs at least 1 shard, this can cost a lot of working memory. The advantage is that a separate index can be used for each tenant and not too many adjustments have to be made.

For environments with smaller tenants, it may be better to have several tenants share one index. The advantage here is that working memory resources can be shared. However, the queries must be adapted to this, whereby filtering can be done on a Tenant ID. Because Elasticsearch searches every shard by default, regardless of the size of a query, it is important to use custom routing. With custom routing, the index settings can be adjusted so that the Tenant ID can be used as the routing key.

Our advice: **do** deploy multi-tenant if you have thought carefully about how user spaces are distributed in terms of indexes. There are also licence fees for more extensive authentication possibilities.

Prometheus

Isolation	Possibility	Component Prometheus
Authentication	Low	Only basic authentication with username and password.
Data authorisation	Low	No authorisation available without a customised setup.
Data encryption	Low	No TLS support for data in transit, no other encryption for data at rest.
User spaces	Low	No possibility for user directories.
Performance with quotas	Low	No native multi-tenant support and therefore no standard possibility to set a quote for performance for the different tenants.

Conclusion

From Promotheus, there are no or few *native* functionalities to support multi-tenancy.

Our advice: **do not** deploy multi-tenant but use one Promotheus server per tenant for the environment.

Grafana

Isolation	Possibility	Component Grafana
Authentication	High	SAML, AD and LDAP authentication is supported.
Data authorisation	High	Folder permissions, dashboard permissions and various permission levels (Admin, View and Edit) can be assigned.
Data encryption	High	Grafana supports TLS for encryption in transit.
User spaces	High	Different directories can be created out-of-the-box in Grafana.
Performance with quotas	Average	Only if Grafana is made highly available by a load balancer.

Conclusion

Our advice: **do** deploy multi-tenant and create a separate organisation in Grafana per tenant with a standard set of available dashboards. This can then be expanded per tenant with additional more specific dashboards for the environment.

Cassandra

Isolation	Possibility	Component Cassandra
Authentication	Low	Cassandra supports only basic authentication with password and username
Data authorisation	Average	Database roles can be created in a YAML file across different *namespaces*. However, authentication scores low, which is a requirement for high authorisation.
Data encryption	Average	*Data in transit* with SSL/TLS, not *data at rest*.
User spaces	High	Different *key spaces can* be created per tenant in Cassandra.
Performance with quotas	Low	Due to the design of Cassandra - distributed database system - it is not possible to set a limit per tenant without impacting the rest of the tenants.

Conclusion

Strictly speaking, it is possible to use multi-tenancy by creating keyspaces per individual tenant. However, because it is not possible to set a limit, you will always have the "noisy neighbour" with Cassandra. For example, if one tenant is writing a lot of data to the same table, the rest of the tenant environments will suffer in terms of performance. Moreover, it is not possible to guarantee a comparable read/write ratio for every tenant.

In addition, upgrading the Cassandra cluster becomes an issue where coordination across multiple teams and/or organisations is required.

Our advice: **do not** deploy multi-tenant but set up a separate cluster for each tenant, where data and user preferences remain stored.

Hansken front-end

Isolation	Possibility	Component Hansken front-end
Authentication	Average	One OpenLDAP *instance* per tenant, but must be adapted to cope with multiple organisations.
Data authorisation	High	Permissions can be set with roles.
Data encryption	High	Is handled in the keystore with its own encryption key.
User spaces	Average	Application must be adapted to create multiple customer directories.
Performance with quotas	High	This is a web application, so load balancing would be a requirement.

Conclusion

In the Hansken front-end, authentication and authorisation takes place via Keycloak and OpenLDAP. By using a keystore, encryption keys are generated with which an image data is encrypted. This allows a user to request *trace* data after logging in (authentication) and determines which actions that person may perform on the data based on authorisations. After authorisation, a user can only access the data if he has the encryption key of the data.

Our advice: **do** deploy multi-tenant if multiple organisations can be created in Hansken's web application (UI).

Hansken core

Isolation	Possibility	Component Hansken Core
Authentication	High	Authentication for gatekeeper, lobby etc.
Data authorisation	High	Verifies the required permissions via OpenLDAP.

Isolation	Possibility	Component Hansken Core
Data encryption	Average	Encryption for data *at rest*, no encryption (yet) for data *in transit*.
User spaces	High	The *RPC/API calls* can be distinguished.
Performance with quotas	High	By means of load balancing and scheduling (RPC and Kafka) in the lobby.

Conclusion

In the Hansken core components, a microservices architecture has been put in place that should be generic and scalable. Authentication takes place on the Gatekeeper that converts the API calls to RPCs and via OpenLDAP the permissions are checked. In addition, requests coming from the Hansken front-end can be load-balanced and scheduled in the lobby, which functions as *microservices manager* and *job scheduler*.

Our advice: **do** deploy multi-tenant, provided that adjustments are made to the Hansken services. Encryption for *data in transit* must also be implemented everywhere.

Kafka & Zookeeper

Isolation	Possibility	Component Kafka & Zookeeper
Authentication	High	By means of SASL, authentication can be enabled between brokers, between clients and brokers and between brokers and Zookeeper.
Data authorisation	High	ACLs can be applied to the more recent versions of Kafka in a shared Kafka cluster.
Data encryption	Average	Encryption for data in transit (SSL/TLS), no functionality out-of-the box for encryption for data at rest.
User spaces	High	By means of MirrorMaker it is possible to activate replications between two Kafka clusters.
Performance with quotas	High	Various quotas can be set on the producer and consumer side.

Conclusion

For Kafka, Access Control Lists (ACLs) can be configured so that multi-tenant commands can be executed on a shared Kafka cluster. Multiple Kafka clusters can share a common Zookeeper. Please note that upgrading Zookeeper may affect the availability and compatibility with Kafka.

In order to enable multi-tenancy, use should be made of MirrorMaker, which is part of Apache Kafka. MirrorMaker allows Kafka to replicate data between two Kafka clusters, creating a consumer and producer cluster just like

Kafka Brokers, moving data from a source cluster to the target cluster. This makes it possible to make Kafka *highly available* and at the same time isolate data per use case.

Our advice: **do** deploy multi-tenant but MirrorMaker must also be (correctly) implemented to make this possible.

9.3 Multi-tenancy advice

Based on the previous section, we looked at each component to see if it would be suitable for multi-tenancy or not, or to a lesser extent; these are shown in the design below. On top are the components such as the deployment and virtual management layer, repository and monitoring (Grafana) that are managed across tenants. Then there are components such as the Hansken services and Hadoop that can be shared across tenants with logical separations and a few adjustments in terms of resources. Furthermore, there are a number of components such as Cassandra and Promotheus that will get their own instance per tenant because they are not or less suitable for multi-tenancy.

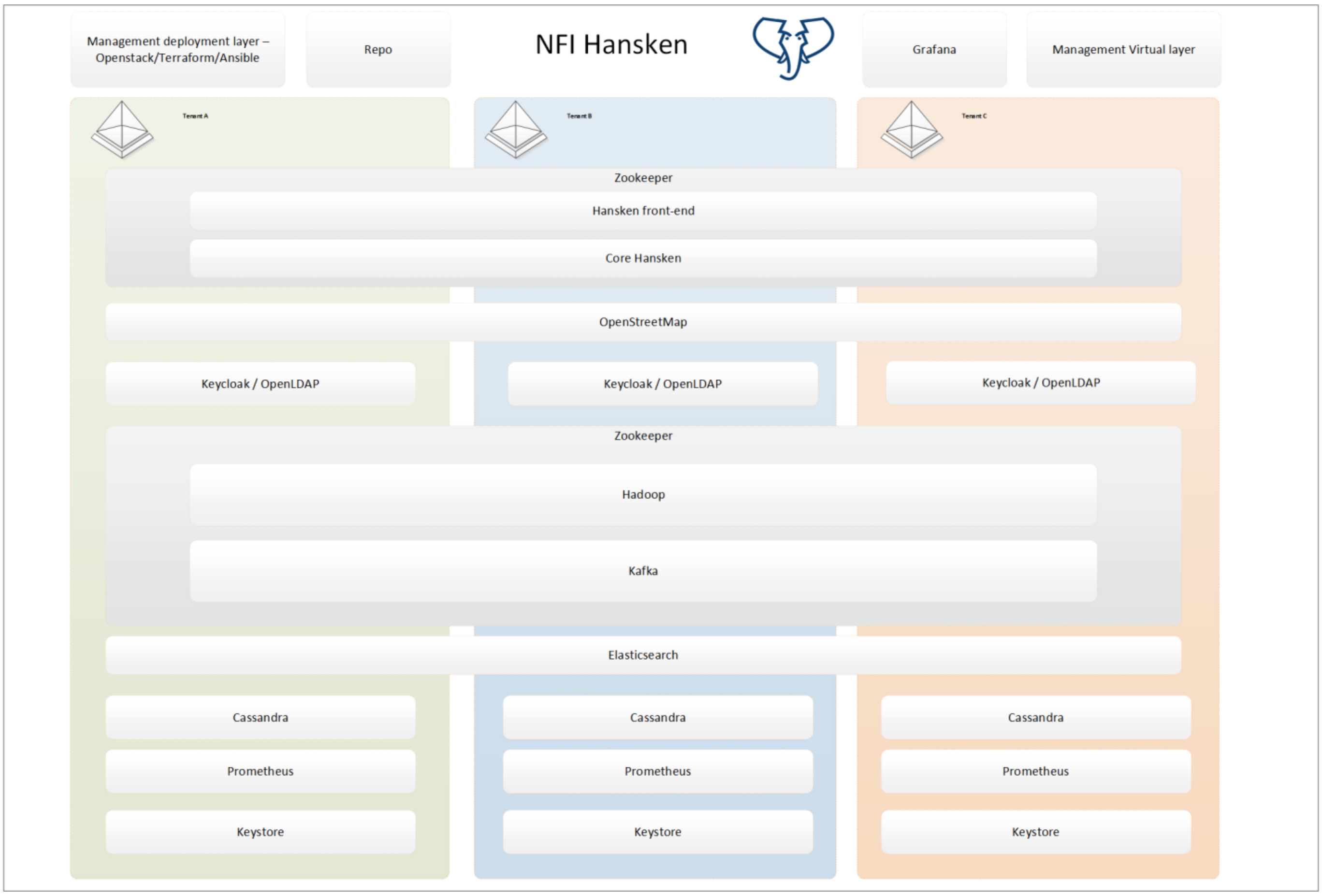

Figure 8Software components multi-tenancy overview

For the infrastructural hardware deployment, we recommend scenario 3, in which a hyper-converged infrastructure with external object storage ensures that the Hansken platform is scalable and will remain so in the future. Among other things, it is possible to quickly expand object storage when the amount of data increases significantly by expanding within the storage cluster. It is also possible to significantly reduce costs when resources such as processing power, storage, network and management (time) can be spread across multiple tenants. However, it is important that the management of the multi-tenant platform is taken into account and a risk analysis must be carried out to prevent possible (hardware) vendor lock-in and due diligence must be conducted on the choice of hardware and suppliers.

In the area of system management, a distinction must be made between two types of management: the management of the platform as a whole across tenants, and the management of the individual tenants themselves. *Separation of duties is* an important principle here, where no single user or administrator has enough privileges to use the system in an undesirable way. In addition, an environment must of course be secure, but on the other hand it must also be easy to manage by the various parties or teams; this is a continuous consideration that must be made. This makes it all the more important that good agreements are made about who is responsible for what and that it is always clear who has access to what. It is also important that it is impossible for administrators to access data for which they do not have permission. In reality, this is already mitigated by the use of a data encryption key generated by the platform operator.

For each individual software component, we looked at whether or not it should be used as a multi-tenant application. Five criteria were used to assess whether the software is sufficient to be used for multiple tenants simultaneously. An overview can be found above at figure 7: Software components multi-tenancy overview. Of course the actual development, configuration and implementation of the software determines to what extent it is suitable for multi-tenancy.

10 Advice

During the execution of the research, several key takeaways were defined that served as focal points when answering the research questions in the report. As a result, the following matters are recommended:

Apache Hadoop

- Apache Ambari is highly recommended for managing the Hadoop cluster for monitoring purposes to effectively resolve any incidents. It also lends itself to preventive identification and prevention of potential problems.
- Increase the effectiveness of available processing power by making the limit on the number of simultaneous extractions dynamic, thus allowing optimal use of the platform's capacity.
- Use a separate storage cluster to replace HDFS, this will improve the scalability, cost-effectiveness and stability of the platform.
- Make Hadoop more scalable by enabling dynamic scaling of nodes.
- Investigate the possibility of deploying Apache Spark to replace Apache MapReduce or as an addition to existing functionality.

Apache Cassandra

- Use a relational database for relational data such as project and trace data.
- It should be further investigated whether HBase can be used for the required NoSQL functionality.

Configuration management

- Use an Ansible instance per environment or per type of environment, this will benefit the confidentiality, integrity and availability of an environment.

Monitoring

- Add additional metrics to the included Grafana dashboards.
- Ensure that customers of Hansken monitor the log files.
- Check the monitoring situation against the *best practices*.

Containerisation

- Conduct further research for the deployment of *containerisation* and *load balancing* for services that provide a specific service and where (performance) bottlenecks may occur (such as with the Hansken core services and the Document Viewer).

Fit for purpose

- Except for Cassandra, all components have been found *fit for purpose*, Cassandra needs to be replaced (partially), this is discussed in section 8.2.5.

Multi-tenancy

- Consider adapting some components, as described in the multi-tenancy advice, so that they become suitable for use in a multi-tenant environment.
- To enable cost reductions and scalability for the platform, it is important to move to a shared infrastructure with external object storage.

- To enable multi-tenancy in a secure manner, encryption, authorisation and authentication must be correctly implemented everywhere.

In addition to the above, the following is also recommended:

- Carrying out an audit of the installation at a customer site for the purpose of identifying and potentially resolving environment-specific problems.
- Adjusting the models for calculating storage requirements and the speed of data processing to environment-specific factors for the sake of the accuracy of the outcome.
- Based on the greenfield design, carry out a technical implementation that tests the functionalities and exploits the intended performance benefits.

11 Source reference

Institute, N. F. (2021, May 14). Multiple negotiated bid "audit programme Scaling up and Chain Implementation (OK) Hansken".

Government. (2013, June 1). *Laws Government* . Retrieved from Government Information Security Regulations Decree 2013: https://wetten.overheid.nl/BWBR0033507/2013-06-01

Government. (2020, January 01). *Police Information Act*. Retrieved from Laws of Government: https://wetten.overheid.nl/BWBR0022463/2020-01-01

Capgemini Netherlands. (2022, 2 4). Capgemini best practices. *Knowledge Management System*. Utrecht, the Netherlands: Capgemini Nederland B.V.

Cassandra . (2022, 4 2). *Apache Cassandra*. Retrieved from Cassandra Apache: https://cassandra.apache.org/_/index.html

Datastax. (2022, 4 2). *DataStax Docs*. Retrieved from Datastax: https://docs.datastax.com/

Elastic (2022, 4 2). *Elastic Stack and Product Documentation*. Retrieved from Elasticsearch: https://www.elastic.co/guide/index.html

Elastic Rally. (2022, April 01). Retrieved from Github: https://github.com/elastic/rally

Grafana Labs. (2022, 4 2). *Grafana*. Retrieved from Grafana: https://grafana.com

Institute, N. F. (2021, May 14). Multiple negotiated bid "audit programme Scaling up and Chain Implementation (OK) Hansken".

Government. (2013, June 1). *Laws government* . Retrieved from Government Information Security Regulations Decree 2013: https://wetten.overheid.nl/BWBR0033507/2013-06-01

Government. (2020, June 17). *BIO Government*. Retrieved from 17-06-2020: https://bio-overheid.nl/media/1572/bio-versie-104zv_def.pdf

Government. (2020, January 01). *Police Data Act*. Retrieved from Laws Government: https://wetten.overheid.nl/BWBR0022463/2020-01-01

12 Annex I: Interview questions

This document serves as input for the preliminary investigation. The focus of the questions is mainly exploration and inventory. The questions are used during the execution of a number of semi-structured interviews with the different actors within the process as described in the Hansken Architecture Summary.

1. Architecture principles

a. What are the architectural principles that you took as a starting point during the design process? How did you fill them in?
b. What is the definition of an S/M/L environment?
c. What quantities of data are contained in the various environments?
d. What data growth is expected in the various environments, and how will this be handled?
e. Is the AVG relevant and what principles have you included?

2. Infrastructure/(physical) storage

a. What underlying hardware is used for the environment?
b. Which network components are between the VMs? What bandwidths are available here?
c. In which physical locations is data expected to be stored? What conditions apply?
d. From which locations can data be retrieved by users, and through which connection?
e. What data sources are used besides physical data devices?
f. Are the images created block or file based?

3. Service management

a. What is the SLA for data availability?
b. What is the SLA on availability of the services provided?

4. Development

a. What is the test and development environment used for?Is it for testing algorithms for research or for testing improvements to the platform, or both?
b. Are there any segregations of rights within the platform?
c. How is it ensured that users do not get in each other's way when developing data analyses and the end user performing them?
d. How are you still involved with the Hansken system now that it is in production, is it continuously improved and what is your role in it?

5. Security/performance

a. Which elements of security have you built in, and how does this affect the performance of the system? What considerations have you made in this respect?

6. Infrastructure

a. What underlying hardware is used for the environment?
b. What is the load on this?
c. Which network components are between the VMs?
d. What ranges are available here?
e. Are there any known bottlenecks here?

7. Service management

a. What are common incidents within the landscape?
b. Are there any known problems in the area?
c. Is there a capacity planning process?
d. Have support contracts been concluded for the software and hardware used?

8. Cases/Images

a. How many cases are created in a day?
b. What is the average size of a case?
c. Which software is used to create and upload an image
d. What is the average size of an image on the HDFS/Ceph store?
e. How many cases do you have in total per S/M/L environment?
f. How fast is project data currently loaded per environment? How fast is it now, and how fast should it be able to be written away?

9. CEPH/HDFS

a. How are the CEPH and HDFS store deployed?
b. How is encryption set up for all elements within the process flow. Both data in flight and data at rest.

10. Ansible

a. How can Ansible access other machines?
b. What rights does the Ansible VM have on the hosts running playbooks?
c. Which hosts does the Ansible VM have access to?
d. Who all has access to the Ansible VM?

11. Hadoop

a. How do you monitor the application, what data do you use for this? What metrics and issues do you look at?
b. What distribution of Hadoop is used (Cloudera?)
c. What services are running on the Hadoop cluster?
d. Are there MapReduce queues in which workloads fall?
e. How many vcores and memory are/are available for YARN containers? Is it different for S/M/L?
f. Where is the bottleneck when new source data is loaded?
g. Where is the bottleneck when the source data is analysed by MapReduce?
h. Are there any other known bottlenecks in Hadoop at this time?
i. At what other times is MapReduce used?
j. Do warnings and/or alerts arise under load?
k. Does periodic maintenance take place? If so, what work is carried out, for example?
i. How much time do MapReduce jobs take during the extraction process?
j. How are authentication and authorisation organised?
m. How are authentication and authorisation organised? Is Kerberos used?
n. Have any parameters within Hadoop been adjusted to optimise the Hansken platform?

12. Cassandra

a. How do you monitor the application, what data do you use for this? What metrics and issues do you look at?
b. What is the volume for the Cassandra application? How much is written and how much is retrieved?
c. How large are the databases within Cassandra?
d. Are there any known bottlenecks within Cassandra?
e. Do warnings and/or alerts arise under load?
f. How are authentication and authorisation organised?

13. Elasticsearch

a. How do you monitor the application, what data do you use for this? What metrics and issues do you look at?
b. Are there any known bottlenecks within Elasticsearch?
c. Do warnings and/or alerts arise under load?
d. How are authentication and authorisation organised?

14. Hansken Security

a. Have security audits ever been performed on the environment?

15. Business continuity

a. Are servers and data backed up?
If so, where to?
What are the RPO and RTO?

16. Inventory/performance

a. Can we receive an overview of all version numbers of the applications in use?
b. Is the following measurement information available at physical and VM level for the past 30 days?
CPU: utilisation and zoom in on peak moments
Storage: I/O per second, throughput
Memory: utilization
Network: average throughput and peak throughput
c. Is the following measurement information available within Cassandra, Hadoop and Elasticsearch?
CPU: utilisation and zoom in on peak moments
Storage: I/O per second, throughput
Memory: utilization
Network: average throughput and peak throughput
Request count
Garbage collections

17. case investigator

a. Can you tell us what you use the system for (what roles you have)?
b. What steps do you carry out to create a Hansken case project?
c. Can you tell us what you use the system for (what roles you play)?
d. How do you experience the performance during your work?
e. What things are performing well now?
f. What things could perform better?

g. Is it noticeable when others (colleagues) use the system at the same time?
h. Can you tell us something about the stability of the system? Does anything ever get stuck?
i. Looking at the work you do within hansken, what possibilities for improvement do you see within it?
J. Based on your current experience with the system, are there any opportunities for improvement?

18. Platform operator

a. Can you tell us what you use the system for (what roles you have)?
b. What steps do you carry out in order to upload the data to hansken?
c. What device is used to upload the data?
d. Is it ever the case that you cannot upload a device to hansken, what kind of error message do you get, and what do you do in that case?
e. How do you experience the performance during your work?
f. Which things are performing well now?
g. What things could perform better?
h. Is it noticeable when others (colleagues) use the system at the same time?
i. Can you tell us something about the stability of the system? Does anything ever get stuck?
j. Looking at the work you do within hansken, what possibilities for improvement do you see within it?

Job specific

a. What do you consider an acceptable lead time for obtaining data per environment? S/M/L
b. What do you consider an acceptable turnaround time for uploading an image of a device? Per specific data source/data device
c. How often do you create a report in Hansken?
d. The process shows that you extract traces from the data of the image, can you tell us what you are doing and what kind of data you are extracting?
e. As a platform operator, are you also involved in setting up the underlying system? If so, what responsibilities do you have in this regard (Hadoop, Cassandra, Ceph)?
f. Based on your current experience with the system, are there any opportunities for improvement?

19. Forensic experts

a. Can you tell us what you use the system for (what roles you have)?
b. How do you experience the performance during your work?
c. What things are performing well now?
d. What things could perform better?
e. Is it noticeable when others (colleagues) use the system at the same time?
f. Can you tell us something about the stability of the system? Does anything ever get stuck?

Job specific

a. What actions do you carry out within hansken to develop a case report? What steps do you carry out for this?
b. When you have developed a report case, what do you do with this document, is it closed by you or is someone else involved after your actions in this process?
c. What is an acceptable lead time for obtaining data per environment? S/M/L
d. Based on your current experience with the system, are there any opportunities for improvement?

20. Digital expert

a. Can you tell us what you use the system for (what roles you have)?
b. How do you experience the performance during your work?
c. What things are performing well now?
d. What things could perform better?
e. Is it noticeable when others (colleagues) use the system at the same time?
f. Can you tell us something about the stability of the system? Does anything ever get stuck?

Job specific

a. Can you tell us what steps you take to perform big data analysis?
b. Are there different kinds of analyses that can be performed, and how do you know which one to perform for the desired result? Is there any difference in terms of load on the system?
c. Can you tell us what is possible with the python API used to perform big data analysis?
d. Are you also responsible for developing the algorithms, or do you just execute them?
e What exactly does the MapReduce job do? What things does it extract from the images?
f. Based on your current experience with the system, are there any opportunities for improvement?

13 Annex II: Research Scope

1. Infrastructure requirements

"What are the technical requirements of the Hansken software suite for the environment in which the product is to be installed."

1A. Are the requirements for a basic production cluster as described in the architecture summary document and installation manual up to date, complete and correct for using a Hansken instance?

Actions:

- Inventory requirements from architecture summary;
- Request installation manual and assess its applicability and correctness.

Purpose: To demonstrate whether the current documentation is sufficient for the use of a basic production cluster.

Scope: Judgement is made on the basis of the specifications and installation manuals provided by NFI. On this basis, a judgement is made about the completeness and correctness of the infrastructural requirements. We hereby restrict ourselves to the documentation as issued by the NFI. This will be assessed from the point of view of industry best practices.

"Such that the infrastructure is scalable and the Hansken software can perform optimally, while also meeting Dutch security and government investigation standards."

1B: Are the requirements as described in the documentation supplied by the NFI correctly applied and is this still sufficient today, compared to the three defined environments? *Focused on the different environments: S: NFI | M: FIOD | L: Police. Determining environment size: number of users, amount of data, type of load per production cluster based on the metrics: CPU, memory, storage, network.*

Actions:

- Request architectural principles and security requirements from parties;
- Conduct interviews to gather experience on performance and scalability;
- Retrieve and analyse metrics;
- Assess whether infrastructural requirements as described are sufficient: is the environment scalable and to what extent is security[6] described for all three environments?

Objective: To demonstrate whether the current requirements of Hansken are still sufficient for the environments of the Police, FIOD and NFI.

Scope: We limit our research to the minimum, most critical architectural principles and organisational requirements for running an application. These requirements are drawn from interviews with the parties involved. During this research we do not look at the efficiency of the code with a view to scalability, as this has been previously investigated and found to be good.

[6] Authentication, authorisation and encryption.

"An advice on the type(s) of configuration of the infrastructural environment (including which clusters to distinguish) and what are the main characteristics of this environment? Which configuration types are excluded in advance?"

1C. Which components (individual servers) and clusters (groups of servers) exist within the environment? What are the key characteristics (virtual or physical) of these components and clusters and which issues should be avoided?

Actions:

- Inventory current configurations;
- Compare current situation with supplier requirements;
- Advise on better alignment with vendor requirements (Hadoop, Elasticsearch and Cassandra).

Aim: To test whether the current physical/virtual configurations as supplied by the NFI meet the requirements for a basic production cluster.

Scope: The answer to the research question is limited to Hadoop, Elasticsearch and Cassandra clusters.

2. Hansken components

"An advice on the design of the individual software components. In combination with the Hansken software."

2A. Which of the current components within Hansken are currently no longer fit for purpose and what possible improvements are recommended?

Actions:

- Conduct interviews, identify most talked-about applications for research[7] ;
- Examining components and comparing them to current industry standards;
- Advise on possible improvements within the landscape of Hansken (other component - improve current component - leave it as it is).

Purpose: To test whether the current components within Hansken serve the platform optimally.

Scope: Advice is given on improvements, but there is no assistance with in-depth issues currently present within Hansken. No parameters are examined when the advice is given and no specific vendor solutions are proposed.

[7] Cassandra, Hadoop (HDFS + MapReduce), Elasticsearch and Kafka.

"Advice on how to adequately set up the monitoring of the platform (Hansken software suite implemented on the infrastructure). What (additional) requirements does this impose on the software?"

2B. How can the components within Hansken be optimally monitored and what requirements does this place on the software in order to set this up adequately?

Actions:

- Conduct interviews and collect information on current monitoring efforts;
- Investigate functional possibilities for improving the current monitoring.

Purpose: To provide advice on how to better align monitoring with the software and validate KPIs that are currently defined.

Scope: Only scalability and security1 are considered for the purpose of proper separation of tenants.

3. Multi-tenancy

"What (additional) requirements does this place on the set-up of the individual software components/products (e.g. Cassandra, Hadoop HDFS, Hadoop MapReduce and Elasticsearch) in combination with the Hansken software."

3A. What (additional) requirements does this place on the design of the individual software components/products (e.g. Cassandra, Hadoop HDFS, Hadoop MapReduce and Elasticsearch) in combination with the Hansken software?

Actions:

- Inventory best practices per component;
- Draw up a blueprint of all components that takes into account the best practices of all components along the axis of scalability and security.

Goal: Mapping the additional requirements when deploying multi-tenancy.

Scope: Only scalability and security1 are considered for the purpose of proper separation of tenants.

"What (additional) requirements does this place on Hansken's software architecture?"

3B. What (additional) requirements does this place on Hansken's software architecture?

Actions:

- Identify the requirements for Hansken's software architecture;
- Draw up recommendations for the software architecture.

Goal: To check where the gaps are within the current architecture in order to achieve a multi-tenancy solution.

Scope: Only scalability and security1 are considered for the purpose of proper separation of tenants.

"What (additional) requirements does this place on the design of the infrastructure?"

3C. What (additional) requirements does this impose on the design of the infrastructure?

Actions:

- Advise on the impact on the infrastructure.

Goal: To check where the gaps are within the current architecture in order to achieve a multi-tenancy solution.

Scope: Only scalability and security1 are considered for the purpose of proper separation of tenants.

"Please advise on where the cut-off between the common/central part and the individual tenants can best be made and more specifically where to make it (in the code or the infra)."

3D. Which divisions need to be made for the correct deployment of multi-tenancy based on the results regarding the individual components (3A), architecture (3B) and infrastructure (3C)?

Actions:

- Advise on where the cut should be made and why.

Objective: To provide advice on the correct and secure separation of customers within a multi-tenancy solution for Hansken.

Scope: Criteria for the recommendation are that parties are not allowed to access each other's data (AVG/Wpg), but also that from a performance perspective each party has as little impact on the other (scheduling).

"Assess the multi-tenant design of the FIOD and advise on its optimal implementation, based on the current architecture of Hansken. Include the way in which the NFI has implemented it. Assess this implementation and, if possible, define options for improvement."

3E. Is the multi-tenant design of the FIOD adequate and where do we see improvements based on the dependencies found within questions 3A, 3B, 3D and 3C.

Actions:

- Compare the current design of FIOD with the advice we have prepared;
- Indicate the impact of differences found.

Objective: To review a multi-tenant design for Hansken.

Scope: In assessing the design, we limit ourselves to the following criteria: that parties are not allowed to access each other's data (AVG/Wpg) but also, from a performance perspective, that they cause minimum inconvenience to each other (scheduling).

About Capgemini

Capgemini is a global leader in partnering with companies to transform and manage their business by harnessing the power of technology. The Group is guided everyday by its purpose of unleashing human energy through technology for an inclusive and sustainable future. It is a responsible and diverse organization of 270,000 team members in nearly 50 countries. With its strong 50 year heritage and deep industry expertise, Capgemini is trusted by its clients to address the entire breadth of their business needs, from strategy and design to operations, fueled by the fast evolving and innovative world of cloud, data, AI, connectivity, software, digital engineering and platforms. The Group reported in 2020 global revenues of €16 billion.

Get the Future You Want | www.capgemini.com

This document contains information that may be privileged
or confidential and is the property of the Capgemini Group.