Decentralized Identity Management for a Maritime Digital Infrastructure
- With focus on usability and data integrity

Decentraliserad Identitetshantering för en Digital Maritim Infrastruktur - Med fokus på användbarhet och dataintegritet

J. Theodor Fleming

Supervisor: Zeinab Ganjei
Examiner: Mikael Asplund

External supervisor: David Marklund
Upphovsrätt


Copyright

The publishers will keep this document online on the Internet – or its possible replacement – for a period of 25 years starting from the date of publication barring exceptional circumstances. The online availability of the document implies permanent permission for anyone to read, to download, or to print out single copies for his/hers own use and to use it unchanged for non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional upon the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility. According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement. For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its www home page http://www.ep.liu.se/.

© J. Theodor Fleming
Abstract

When the Internet was created it did not include any protocol for identifying the person behind the computer. Instead, the act of identification has primarily been established by trusting a third party. But, the rise of Distributed Ledger Technology has made it possible to authenticate a digital identity and build trust without the need of a third party. The Swedish Maritime Administration are currently validating a new maritime digital infrastructure for the maritime transportation industry. The goal is to reduce the number of accidents, fuel consumption and voyage costs. Involved actors have their identity stored in a central registry that relies on the trust of a third party.

This thesis investigates how a conversion from the centralized identity registry to a decentralized identity registry affects usability and the risk for compromised data integrity. This is done by implementing a Proof of Concept of a decentralized identity registry that replaces the current centralized registry, and comparing them. The decentralized Proof of Concept’s risk for compromised data integrity is 95.1% less compared with the centralized registry, but this comes with a loss of 53% in efficiency.
Acknowledgments

I would like to thank my examiner Mikael Asplund for the ideas and the positive inspiration. Thanks to my supervisor Zeinab Ganjei for the feedback on the work. Thanks to David Marklund, Combitech, for the management of assets and meetings with Combitech, Linköpings universitet and the Swedish Maritime Administration, and also the supportive feedback. I would also like to thank Per Löfbom, Mattias Johansson and Per de Flon at the Swedish Maritime Administrator for all of your time and assisting inputs. Lastly I would like to thank my lovely friends and family for being there.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Abstract</strong></td>
<td>iii</td>
</tr>
<tr>
<td><strong>Acknowledgments</strong></td>
<td>iv</td>
</tr>
<tr>
<td><strong>Contents</strong></td>
<td>v</td>
</tr>
<tr>
<td><strong>List of Figures</strong></td>
<td>viii</td>
</tr>
<tr>
<td><strong>List of Tables</strong></td>
<td>ix</td>
</tr>
<tr>
<td><strong>1 Introduction</strong></td>
<td>1</td>
</tr>
<tr>
<td>1.1 Motivation</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Objective</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Motivation of Analyzed Factors</td>
<td>2</td>
</tr>
<tr>
<td>1.3.1 Usability</td>
<td>2</td>
</tr>
<tr>
<td>1.3.2 Risk for Compromised Data Integrity</td>
<td>2</td>
</tr>
<tr>
<td>1.4 Research Questions</td>
<td>3</td>
</tr>
<tr>
<td>1.5 Method Overview</td>
<td>3</td>
</tr>
<tr>
<td><strong>2 Theory</strong></td>
<td>5</td>
</tr>
<tr>
<td>2.1 Digital Identities</td>
<td>5</td>
</tr>
<tr>
<td>2.1.1 Centralized</td>
<td>5</td>
</tr>
<tr>
<td>2.1.2 Federated</td>
<td>6</td>
</tr>
<tr>
<td>2.1.3 User-Centric</td>
<td>6</td>
</tr>
<tr>
<td>2.1.4 Self-Sovereign</td>
<td>6</td>
</tr>
<tr>
<td>2.2 Self-Sovereign Identity</td>
<td>7</td>
</tr>
<tr>
<td>2.2.1 Definition</td>
<td>7</td>
</tr>
<tr>
<td>2.2.2 Decentralized Identifier</td>
<td>7</td>
</tr>
<tr>
<td>2.3 Distributed Ledgers</td>
<td>9</td>
</tr>
<tr>
<td>2.3.1 Distributed Ledger Technology</td>
<td>9</td>
</tr>
<tr>
<td>2.3.2 Hyperledger Indy</td>
<td>10</td>
</tr>
<tr>
<td>2.4 Public Key Management</td>
<td>10</td>
</tr>
<tr>
<td>2.4.1 Public Key Infrastructure</td>
<td>10</td>
</tr>
<tr>
<td>2.4.2 Decentralized Public Key Infrastructure</td>
<td>11</td>
</tr>
<tr>
<td>2.5 Analyzed Factors</td>
<td>12</td>
</tr>
<tr>
<td>2.5.1 Efficiency of a System</td>
<td>12</td>
</tr>
<tr>
<td>2.5.2 Risk in Cybersecurity</td>
<td>13</td>
</tr>
<tr>
<td><strong>3 Maritime Connectivity Platform</strong></td>
<td>14</td>
</tr>
<tr>
<td>3.1 Overview of Maritime Connectivity Platform</td>
<td>14</td>
</tr>
<tr>
<td>3.2 Centralized Identity Registry</td>
<td>14</td>
</tr>
<tr>
<td>3.3 Web Application</td>
<td>15</td>
</tr>
<tr>
<td>3.4 Roles and permissions in Maritime Connectivity Platform</td>
<td>15</td>
</tr>
</tbody>
</table>
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Visualization of a system that uses PKI</td>
<td>11</td>
</tr>
<tr>
<td>2.2</td>
<td>Visualization of a system that uses DPKI</td>
<td>12</td>
</tr>
<tr>
<td>4.1</td>
<td>Overview of IMS’s components and their relation</td>
<td>18</td>
</tr>
<tr>
<td>4.2</td>
<td>Overview of Identity Manager</td>
<td>19</td>
</tr>
<tr>
<td>4.3</td>
<td>Overview of DID’s parts and their relationship</td>
<td>20</td>
</tr>
<tr>
<td>4.4</td>
<td>Overview of implementation of Identity Manager</td>
<td>22</td>
</tr>
<tr>
<td>6.1</td>
<td>Sequence diagram over MCP’s registration procedure</td>
<td>28</td>
</tr>
<tr>
<td>6.2</td>
<td>Sequence diagram over IMS’s registration procedure</td>
<td>31</td>
</tr>
<tr>
<td>6.3</td>
<td>Total number of required actions to register in each system for the ship</td>
<td>30</td>
</tr>
<tr>
<td>6.4</td>
<td>Total count of entities that are authorized to modify identity data for each system</td>
<td>32</td>
</tr>
<tr>
<td>6.5</td>
<td>Number of total possible identity modification actions for each registry</td>
<td>33</td>
</tr>
<tr>
<td>6.6</td>
<td>The risk for compromised data integrity for each system</td>
<td>34</td>
</tr>
</tbody>
</table>
### List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Summary of the design goals and principles of DID architecture</td>
<td>8</td>
</tr>
<tr>
<td>3.1</td>
<td>Roles and their permissions in MCP’s identity registry</td>
<td>15</td>
</tr>
<tr>
<td>4.1</td>
<td>Overview of distributed ledgers</td>
<td>17</td>
</tr>
<tr>
<td>4.2</td>
<td>Roles and their permissions in MCP’s identity registry</td>
<td>21</td>
</tr>
<tr>
<td>5.1</td>
<td>The entities with their corresponding role in each system</td>
<td>26</td>
</tr>
<tr>
<td>6.1</td>
<td>Which entities that are authorized to modify an entity’s identity data in MCP</td>
<td>32</td>
</tr>
<tr>
<td>6.2</td>
<td>Which entities that are authorized to modify an entity’s identity data in IMS</td>
<td>32</td>
</tr>
<tr>
<td>6.3</td>
<td>The potential actions on entities identity data with different entities’ keys in MCP</td>
<td>33</td>
</tr>
<tr>
<td>6.4</td>
<td>The potential actions on entities identity data with different entities’ keys in IMS</td>
<td>33</td>
</tr>
<tr>
<td>6.5</td>
<td>The risk for compromised data integrity for both systems, relative to each other</td>
<td>34</td>
</tr>
</tbody>
</table>
1 Introduction

This chapter begins with motivating the initiative for this thesis and describes the problem it tries to solve. This is followed by a presentation of the objectives for this thesis. Then the research questions are presented and lastly a discussion on how this thesis will try to answer them.

1.1 Motivation

Nowadays, maritime transportation makes up more than 90% of the worldwide trade volume \[22\] and the shipping industry continues to expand globally \[6\]. According to Akten \[2\], the environment ships operate in is unsafe and the increasing number of marine vessels may lead to an increase in maritime hazards and accidents \[6\].

To reduce the number of accidents, as well as lowering fuel consumption and voyage costs, the project MONALISA was initiated in 2010 with Swedish Maritime Administration (SMA) as the lead partner. This project has been further developed and is now tested on large-scale in the Sea Traffic Management Validation Project (STM). The test bed consists of the Nordic and Mediterranean Seas, around 300 vessels, 13 ports and 5 shore based services centres.

To be a part of this project and utilize its services all actors must be registered in an identity register on a platform called Maritime Connectivity Platform (MCP). This allows ships and ports to retrieve information required to authenticate each other and initiate communication, among other things.

If new actors wish to join MCP their requests has to be managed by a central organization. However, according to SMA, a centralized solution requires more administration like governance and an organization which in turn drives cost and possibly has trust issues. If an actor does not trust the organization, or the other way around, if the organization does not trust the actor, there is no other way for it to join MCP. Also, if the organization has any kinds of issues with approving new applications no new actors will be able to join until the organization has found a solution to their problems. Accordingly, this organization will act as a single point of failure for the registration in MCP.

From a cybersecurity perspective, an attacker could perform a successful attack against a single node in a centralized system to gain access to all identity data in the registry. On the
other hand, in a decentralized system, a successful attack against just one node may not be enough to obtain all the identity data.

Also, the experience from STM points to a more successful adoption, and an increased number of interested actors in using a decentralized architecture as opposed to a centralized dito. This is because the involved actors are in full control of their data and can chose whom to share it with instead of trusting a third party.

A possible solution to this problem could be a decentralized self-sovereign identity registry with the use of Distributed Ledger Technology (DLT). A self-sovereign identity is an identity where the user is in control of its own data. There is no organization that can remove or alter the identity, only the user has those privileges. The user can also decide what data to share about their identity and to whom [1].

Decentralized identity management has been difficult to realize in the past because one of the core requirements of functional identity is discovery: if an identifier is given, it needs to be looked up somewhere. This has always led to centralized directories, which led to centralized identity systems. But, the evolution of DLT has made it possible to decentralize identity systems. Self-sovereign identity systems utilize DLT in a way that decentralized identifiers can be looked up without involving a central directory. This would make the identity registry decentralized and no single organization would have to manage new identities. Also, a single successful attack against that registry would not give access to all identity data.

1.2 Objective

The objective of this thesis is to investigate how the usability and the risk of compromised data integrity for MCP are affected when transitioning from a centralized identity registry to a self-sovereign decentralized identity registry for a maritime digital infrastructure.

1.3 Motivation of Analyzed Factors

This section motivates the choice of analyzing the usability and the risk for compromised data integrity when comparing the two registries.

1.3.1 Usability

Since the goal of STM is to integrate the whole maritime logistics chain, the system’s usability might be a determining factor. If the system is too difficult or time consuming to utilize, actors may refuse to use it. When converting to another type of identity system, the process of registering new users may change.

Usability refers to how well the system meets the requirements of a user in terms of ease of use, efficiency and effectiveness [24]. Meeting qualities such as ease of use, results in the system being viewed in favour by users [31]. In this thesis the focus of usability is on efficiency - the estimated cost of resources used when achieving a specific goal. The resources counted in this thesis were the total number of actions required by a ship to register in the registries.

1.3.2 Risk for Compromised Data Integrity

Within STM an attribute analysis has been performed where 31 important attributes for STM has been ranked. To simplify, number one can be seen as the most important and 31 as the least - integrity is ranked as number 4. The integrity of data can be compromised by not preventing unauthorized access to the data. About a year ago, the Equifax data breach compromised the personal data from a centralized database of up to 143 million people [28]. These kinds of breaches are not often caused by hackers or other malicious efforts, but because of
1.4. Research Questions

Weak security that does not prevent unauthorized access. Many centralized identity systems have security issues and this case highlights the vulnerability of centralized databases.

Data integrity is defined as the prevention of unauthorized or improper modification of information [13]. This means that data with high integrity is data that has been created or modified, and stored in a way, that makes it trustworthy. Integrity can be maintained by regulating who are authorized to modify data. This has been shown in the security model Biba Model, also known as Biba Integrity Model. In this model subjects and objects are given different levels of integrity. A subject can only modify objects with lower integrity level than itself and thus retain the integrity of objects with a high level [13]. But, even if the system has appropriate safeguards that prevents unauthorized access, there is always a risk that an attacker obtains an authorized entity’s key and modifies data, and thus the integrity of the data cannot always be guaranteed [18]. Although, there are approaches that could reduce the impact.

MCP’s centralized identity system uses a Public Key Infrastructure (PKI) to manage keys. This implies that a central authority issues digital certificates which are used to authenticate entities in the identity register. With the transition from the centralized registry to a decentralized registry, a shift to a Decentralized Public Key Infrastructure (DPKI) follows. When switching from a PKI to a DPKI, the way keys are managed is changed. This might reduce the potential damage an attacker could cause if getting hold of private keys and gaining access.

In cybersecurity risk can be seen as the combination of the likelihood that something occurs and the consequence of the occurrence. In this thesis the likelihood of compromised data integrity and the following consequence of it was analyzed and in combination used to present a result of the risk for compromised data integrity.

1.4 Research Questions

This thesis evaluated how usability and data integrity were affected when transforming from a centralized identity system to a decentralized identity system, by answering the following questions:

1. How is the usability for users affected when they register in a decentralized identity registry instead of a centralized, measured as the number of total actions required to register?

2. How does a change from a centralized identity registry to a decentralized affect the likelihood of modifications of an entity’s identity data, measured as the number of actors that has authorized access to it?

3. How does a transition from a centralized identity registry to a decentralized affect the consequence of stolen keys, measured as the number of possible modifications of identity data that can be accomplished with them?

1.5 Method Overview

The usability and the risk are assessed by creating a Proof of Concept, named Identity Management System (IMS), for how a decentralized self-sovereign identity registry could be implemented if it was used in MCP to replace the centralized identity registry that is currently in use. And, then compare these attributes for the current system and the IMS to determine how a decentralization of the identity system would affect them.

IMS is created in Java together with Hyperledger Indy, an open source distributed ledger that is created for self-sovereign identities. It is built and run on a computer where a decentralized network is established in a Docker container that illustrates how the network would be resembled and behave if it were implemented in MCP.
A phrase SMA strives to achieve is: "Episodic Tight Coupling, i.e. a ship that calls a port for the first time shall on a secure way integrate with the port without trouble." This phrase was taken into consideration when implementing the IMS. Additionally, SMA has a scenario that they use as a guideline. In this scenario a ship departs from a port, navigates according to its route and lastly arrives at a port it has never been to before. In this thesis it is called Voyage Scenario and has been used in the analyses to put them in a real context.

To analyze the changes in usability of the system, the procedure where one ship advance from not being registered to being registered in the identity registry, is implemented in both MCP and IMS. Each action required is counted separately for each system and then compared.

The conversion from a centralized identity registry to a decentralized identity registry could reduce the number of actors that have authorized access to the entities’ identity data, and this could increase the integrity of data that is stored. Therefore, the number of actors that are authorized to modify entities’ identity data are counted and compared for both identity registries.

The transition from PKI to DPKI could potentially reduce the impact of lost keys. This is studied by investigating which actions that modify entities’ identity data are possible with different entities’ private keys, and comparing the number of possible modifications for each system.

By combining the results from the count of number of actors that are authorized to modify entities’ identity with the results from the analysis of which actions a potential hacker are able to do with different entities’ keys, the risk for compromised integrity regarding entities’ identity data is calculated relative to each identity registry.
2 Theory

In this chapter a review of digital identities is presented. This is followed by a definition of self-sovereign identity and an introduction to DLT - the technology that has made self-sovereign identity possible. Next follows a presentation of two different approaches for managing public keys. And lastly, the two factors that are analyzed in this thesis are explained.

2.1 Digital Identities

When the Internet was created it did not include an identity layer. The Internet's addressing system was and is still based on identifying physical endpoints in the network, such as computers. The person behind the computer was not taken into consideration and therefore the Internet lacked a way to uniquely identify people [29]. Since the Internet protocols were not able to identify persons, neither could the users of the Internet. A solution to this was the creation of digital identities. This began with applications and websites who offered their local accounts, generally with usernames and passwords. Even though other models for online identity have been implemented since, this is still the dominating solution for identities online. However, the progress of DLT has made it possible to change the way digital identities are managed [29].

As mentioned, applications and websites offered their local accounts, but since anyone could create and distribute these digital identities they could not always be trusted. Therefore, centralized organizations began to issue and authenticate digital identities. And, further improvement to help Internet commerce sites to prove their identity was done in 1995, when Certificate Authorities (CAs) stepped in to assist [3].

Since the birth of the Internet, According to Christopher Allen [3], the model for online identity has seen four phases: Centralized Identity, Federated Identity, User-Centric Identity, and now Self-Sovereign Identity [3].

2.1.1 Centralized

Centralized Identity is the administrative control by a single authority or hierarchy. A small change from centralized identity some organizations did was to create hierarchies. An organization as root controller could give permission to another organization to manage their own
2.1. Digital Identities

hierarchy. The root was still in control though, they were only creating new centralizations below them with less power [3]. Problems with giving the control of the digital identity to centralized authorities are the same as in the physical world: users are bound to a single authority who can deny their identity. Centralization removes the control of the identity from the user and gives the authority to the centralized entities [4]. This is today the dominating solution for managing online identities [29]. However, there has been an increased interest to return the control of the digital identity to the people for the last two decades [3].

2.1.2 Federated

Federated Identity is the administrative control by multiple, federated authorities. Microsoft had visions for federated identity and in 1999 they introduced Microsoft’s Passport. This initiative allowed users to create an identity that could be used on multiple sites. But, in the end this was almost as centralized as before since it put Microsoft at the center of the federation [3].

In response, Sun Microsystems, with the purpose of defying centralized authority, formed the Liberty Alliance (2001). In their effort to create a “true” federation they ended up with an oligarchy. The power of centralized authority was instead spread among several powerful entities. Federated identity made it possible for users to go from site to site under the system. However, each site still remained an authority [3].

2.1.3 User-Centric

User-Centric Identity is where the individual or administrative has control across multiple authorities without requiring a federation. The Augmented Social Network (ASN) group felt with assumptions as: “[…] every individual ought to have the right to control his or her own online identity”, that Microsoft’s Passport and the Liberty Alliance did not strive for same goals because the “business-based initiatives” sees users as consumers. ASN posted an extensive white paper [17] where they proposed building “persistent online identity”. This became groundwork for a new kind of digital identity [3].

With decentralization as focus, The Identity Commons, a community that works for open identity layer for the Internet, began to improve the new work on digital identity. Together with the Identity Gang, a Working Group of Identity Commons, they created the Internet Identity Workshop (IIW). The IIW’s work has supported several new methods for creating digital identity such as OAuth and OpenID. Two principles User-Centric Identity methodologies usually focus on is user consent and interoperability, and these two principles can together give the opportunity for a user to decide to share an identity between multiple systems [3].

The dream of the User-Centric Identity community was to give users complete control of their digital identity. But, still today User-Centric Identities’ ownership is placed at the entities that register them. According to Christohper Allen, co-author of TLS Security Standard, “Being user-centric isn’t enough” [3].

2.1.4 Self-Sovereign

The next step of digital identity demands user independence. This is the main focus of Self-Sovereign Identity [3]. An identity that the individual is in control of, without the need for trusted third parties.
2.2 Self-Sovereign Identity

This section presents a definition of Self-Sovereign Identity and an introduction to Decentralized Identifiers (DIDs) that are used as identifiers for Self-Sovereign Identities.

2.2.1 Definition

Self-Sovereign Identity is the concept of letting users have full control over their digital identities [26]. To define Self-Sovereign Identity this thesis will use Christopher Allen’s ten principles of Self-Sovereign Identity [3]:

1. Existence. Users must have an independent existence.
2. Control. Users must control their identities.
3. Access. Users must have access to their own data.
4. Transparency. Systems and algorithms must be transparent.
5. Persistence. Identities must be long-lived.
6. Portability. Information and services about identity must be transportable.
7. Interoperability. Identities should be as widely usable as possible.
8. Consent. Users must agree to the use of their identity.
10. Protection. The rights of users must be protected.

Self-sovereign Identity gives users the chance to create identities that they own themselves. They can control what data they want to share, and to whom they want to share this data. Also, they do not have to trust a third party that would otherwise be the real possessor of their digital identity.

2.2.2 Decentralized Identifier

To create and manage Self-Sovereign Identities a new type of identifiers that are called DIDs are used [32]. DIDs provide a standard for entities to create and store unique identifiers that are cryptographically verifiable [25]. One of the biggest distinctions of DIDs in comparison with centralized digital identities such as phone numbers or Google accounts, that are basically determined by a service provider, is that DIDs are fully owned by the entity that has created them. Following is an example of how a DID can look: did:example:123456789abcdefghi

DIDs are associated with DID Documents that define the DID and how to interact with it. A DID Document should contain at least three things: cryptographic material, authentication suites, and service endpoints. With the use of the cryptographic material and the authentication suites (e.g., public keys) the identity owner can authenticate their DID. This is an example of DID Document describing the example DID:

```json
{
    "@context": "https://w3id.org/did/v1",
    "id": "did:example:123456789abcdefghi",
    "publicKey": [ {
        "id": "did:example:123456789abcdefghi#keys-1",
        "type": "RsaVerificationKey2018",
        "owner": "did:example:123456789abcdefghi",
        "publicKeyPem": "———BEGIN PUBLIC KEY...END PUBLIC KEY——
```
2.2. Self-Sovereign Identity

```
Listing 2.1: Example of DID Document

Each identity owner can create and possess as many DIDs as necessary. This ability is
required to follow the claim for privacy according to DID design goals that are presented
in Table 2.1. By letting the identity owner posses multiple DIDs it can create a DID for a
specific purpose [25]. In a situation where the identity owner is required to verify its age to
a bartender it can use one DID that only verifies the age. It does not even have to expose
the exact age, only that the person is old enough for consuming alcohol legally. Nowadays
people can show their driving license to authenticate their age. The reason why driving licenses
can be used to verify the age is because the issuer of the driving license is an authority with
established trust. But, these ID documents contains more information than what is needed
to verify the age, such as name, nationality, place of birth and so forth. Instead, the person
create a verifiable claim bound to its DID, stating that he or she is qualified to consume
alcohol. Then, by letting a trusted authority approve and sign this claim, the bartender can
verify this with the use of the organization’s public key. And, the bartender never discovers
(i.e., has “zero knowledge of”) the actual age or birth day - this is called a zero-knowledge
proof.

<table>
<thead>
<tr>
<th>Decentralization</th>
<th>Should remove the need for centralized authorities or single points of failure in identifier management. Include creation of globally unique identifiers, public keys, service endpoints, and other metadata.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Sovereignty</td>
<td>Should give the identity owners control over their digital identity without the need to trust third parties.</td>
</tr>
<tr>
<td>Privacy</td>
<td>Should give the identity owners control of what data they share with others.</td>
</tr>
<tr>
<td>Proof-based</td>
<td>Should give the identity owner the ability to authenticate and authorize through cryptographic proof.</td>
</tr>
<tr>
<td>Discoverability</td>
<td>Should be able to search and find other DIDs to interact with and receive information about.</td>
</tr>
<tr>
<td>Interoperability</td>
<td>Should use interoperable standards, making DID’s infrastructure able to use existing tools designed for interoperability.</td>
</tr>
<tr>
<td>Portability</td>
<td>Should be system and network-independent so that identity owners can use their digital identity with any system that supports DIDs.</td>
</tr>
<tr>
<td>Simplicity</td>
<td>Should be (to paraphrase Albert Einstein) “as simple as possible but no simpler”, in order to meet these goals.</td>
</tr>
<tr>
<td>Extensibility</td>
<td>Should be extensible as long as it does not act as an obstacle for interoperability, portability, or simplicity, whenever possible.</td>
</tr>
</tbody>
</table>

Table 2.1: Summary of the design goals and principles of DID architecture

All kinds of identifiers are usually bound to their network and can in most cases only
be used within that network. For example, a student’s ID card at Linköpings university
can be used to authenticate a student for entrance at the university but, it cannot be used
2.3. Distributed Ledgers

to authenticate entrance at Combitech’s office in Linköping. They also have their own set of rules for how the identities are managed. The same applies to DIDs for each ledger or network they are created in or associated with. Therefore, a so called DID Method is required which specifies, for each distributed ledger or network, how the DID is managed. It specifies how a DID is created, accepted, updated and removed on that specific ledger or network. Whereas some specifications allows all DIDs added to the ledger, some might require an identity owner with authority over the ledger to verify the new DID before it is added.

The design of DIDs removes the dependency for CAs and centralized registries for identifiers. Since each identity owner can act as its own root authority, a DPKI architecture is applied instead of the standard key management architecture, PKI [25]. Even though DIDs eliminate the need for centralized authorities, DID methods can be implemented to be interoperable with centralized and federated identifiers.

As a result of removing the need for centralized registries, DIDs are instead stored with the use of DLT. In distributed ledgers every transaction has a digital signature that requires a private key. This makes distributed ledgers a clear choice to use for storage of associated public keys which is used by identity owners to prove ownership. DIDs can be used with any particular type of distributed ledger as long as it is capable of fulfilling basic principles [25].

2.3 Distributed Ledgers

This section gives an introduction to DLT and a short explanation of how it works. This is followed by an introduction to Hyperledger Indy, which is a distributed ledger that was used in this thesis.

2.3.1 Distributed Ledger Technology

A detailed explanation of how distributed ledger technology operates is out the scope of this thesis, but a brief summary is presented.

Blockchain was introduced in 2008 with the purpose of creating an electronic cash system that did not rely on a trusted third party [21]. It was done by creating a network where all transactions are broadcast and visible to everyone and, when validated, added into a block. A block of these approved transactions is then chained to the previous block of approved transactions creating a chain of blocks, hence the name Blockchain.

Blockchain is the underlying technology for Distributed Ledger Technology (DLT) [14]. And, the possibility for fully decentralized identity is thanks to the emergence of DLT. A distributed ledger is a database that is divided geographically and spread across multiple places. The data that it holds is consensus agreed on upon the majority of participants of the network, and all participants can have their copy of the ledger. If someone alters the data in the ledger the changes are applied to all copies of the ledger. However, this does not mean that anyone can change the data however they please. Distributed ledgers are divided into two base categories: permissioned and permissionless. Blockchain is a permissionless distributed ledger which means that anyone is allowed to alter the ledger by adding a new block of transactions (as long as a Proof of Work is done and valid). A permissioned ledger, as the one used for analysis in this thesis, demands some sort of authority clearance before transactions are added to the ledger. No central authority is in control of what changes are made to the ledger. Instead, the accuracy and security is established cryptographically with the use of keys and digital signatures.
2.3.2 Hyperledger Indy

Hyperledger Indy is a permissioned ledger that was used in this thesis. It is built for decentralized identities and provides libraries and tools to manage them. The identities are stored in the ledger as DIDs and users can create as many of them as they want to establish the privacy. Further, Hyperledger Indy provides all necessary tools and components to create decentralized networks. Hyperledger Indy’s code base also provided all the functionality to run validators in the network that cryptographically validates transactions before they are added to the ledger, where each validator is a node in the network. The procedure for how the transactions were validated is out of the scope for this thesis and is not discussed. Hyperledger Indy also provides a Software Development Kit (SDK) that assists with the implementation of their distributed ledger and the usage of its functions.

Hyperledger Indy is open-source and gives users the ability to create their own private ledger on top of the public one. This means that a shipping company could have their private ledger that only contains the identities of their staff, ships and ports. And, the shipping company could be stored on the public ledger, and thus if you trust the shipping company you can trust all identities on the shipping companies private ledger.

2.4 Public Key Management

Authentication is the act of verifying an identity. Today, the most accepted mechanism for proving ownership of an identity is by providing a password (private key) that corresponds to the identity. This however requires that the identity must first be looked up in a registry that associates identities with keys. The most common approach for this is called PKI and is used in MCP. DPKI is another approach and it was used in the IMS. This section introduces both these different approaches to manage public keys.

2.4.1 Public Key Infrastructure

Currently, PKI is the dominating way for managing authentication and distribution of public keys. Basically, it is a centralized database that stores pairs of identities and public keys. To prove ownership of a pair, in other words, proving the ownership of an identity, the private key (typical a password) corresponding to the public key is required. This private key should only be known to the owner of the identity, and be able to provide when needed. However, even if an identity can prove its ownership with the private key, the organization that is in control of the database is the actor that must be trusted in the first place. These organizations have responsibilities, such as accurate registration and identity retention.

The first responsibility mentioned, guarantees that the identity certainly belongs to the entity that is registering. The second mentioned is to make sure that not any user should be able to impersonate an already registered identity.

Today, the majority of trust on the Internet is built upon the use of CAs - trusted third parties. This is because the use of CAs is the most common approach to PKI. Nearly all of the online identities such as usernames, are rented through certificates and according to Allen et al. “This results in severe usability and security challenges Internet-wide.”. One security challenge PKI convey to is the creation of single points of failure, such as MCP’s centralized identity registry as mentioned in Section 1. Conner and Dragos indicates that too much trust is being placed in CAs and several incidents demonstrate this. For example, the DigiNotar incident, where a certificate for google.com was issued mistakenly, which made it possible for customers to act as CAs themselves - a violation against Google’s identity retention. In addition to single points of failure, since multiple CAs exist they might, without knowledge of other CAs actions, certify different public keys corresponding to the same identity, and once again the CA system violates the identity retention.

1https://www.hyperledger.org/projects/hyperledger-indy
2.4. Public Key Management

Figure 2.1 presents a simplified version of how a system that uses PKI could be resembled. The entities connect to the network and look up other entities in the database that a trusted third party control and manage. Consecutively, if the trusted third party would be compromised, the integrity of the whole database could be jeopardized. And also both the confidentiality and the availability of the data are at risk to be compromised.

In a cybersecurity perspective, decentralization of trust can have a positive effect, since the single point of failure centralized systems represent can be eliminated. Even though CAs themselves, hopefully, does not exploit the access of their users’ data, one successful attack against a CA can compromise all of their users’ data. According to Allen et al. [4], “When centralized systems go down, they take all their users with them.”

In efforts to remove, or at least reduce, the trust being placed in CAs and centralized systems, studies with focus on removing the need for a trusted third party and improving the way keys are managed on the Internet have been done. One resulting approach that has been proposed is called DPKI [4]. This alternative transfers the control of the identities from the CAs to the identity owners themselves. Thus, this system addresses many of the security challenges that comes along with PKI. And, according to Allen et al. [4], DPKI can improve each phase of the PKI life cycle.

2.4.2 Decentralized Public Key Infrastructure

The goal of DPKI is to remove the ability for single trusted third parties to compromise the integrity of the system, and instead decentralizing the trust [4]. This is done by allowing the identity owners to have complete control of their identity data instead of central actors. One requirement for DPKI is that the private keys must be generated in a decentralized way. This can be fulfilled if a user’s private key is created on a device only the user holds. Registration where a service generates a key pair on a device not in possessions of the user, a server for example, is not allowed. After all, if the key is generated on this server, the user must trust the central actor who manages it, and we are back where we started, trusting a third party. And likewise, this is why the keys cannot be stored on any other location than on a device that is in the user’s custody. Immediately when the keys are not exclusively stored within the user’s command, the whole idea that the user is in full control is shattered. The trust is transferred to whomever that is storing it, and it is no longer a decentralized system since we trust, once again, a third party. Another requirement is that the private keys must be managed in a secure way. For this reason they must never be transmitted in a insecure manner. With
2.5. Analyzed Factors

This section presents background and definition of the two factors that are used in this thesis, efficiency and risk, and how they are measured in this thesis.

2.5.1 Efficiency of a System

According to Metzker and Seffah [24], engagement in usability offers many advantages and there are several cost-benefits studies that have shown the significance of usability. However, usability has been defined in many different ways by different groups of people. To establish a shared definition this thesis has followed ISO/IEC 25010 definition for quality as: “the quality of a system is the degree to which the system satisfies the stated and implied needs of its various stakeholders, and thus provides value.”. Further, usability is generally defined with a set of factors and the one assessed in this thesis is efficiency. It is defined as: “The capability of the software product to enable users to expend an appropriate amount of resources
2.5. Analyzed Factors

in relation to the results achieved in a specified context of use.". This can be measured by estimating the cost, usually the time, of actions performed. A higher cost results in a less efficient system, while a lower cost would result in a more efficient system. For this thesis, the estimated cost was measured in the total number of actions required for a ship to join an identity registry.

2.5.2 Risk in Cybersecurity

According to Refsdal, Solhaug and Stølen [23], risk is defined as: the likelihood of an incident and its consequence for an asset. There are a few concepts that are necessary to be familiar with to understand the definition of risk. The first is likelihood, the chance that something happens. It can be described both in general terms or in mathematically terms such as probability or frequency. Probability is presented as a number between zero and one and is a measurement of the chance that something occurs. Where on the other hand, frequency is expressed as the number of occurrences per unit of time. The second concept is incident, which is an harmful action that destroys or reduces the value of an asset. The third one is consequence, which is the negative impact on an asset caused by an incident. The last is asset, which is something of value for an entity or unit.

For this thesis, the asset is the entities’ identity data that is stored in the identity registries. The incident is if an unauthorized entity would get access to this asset and modify it. The likelihood is the chance that this would occur and the consequence is that the integrity of the data is compromised.

One method that has been used to analyze the identity registries’ difference of risk for compromised data integrity is the likelihood-consequence matrix. It is a method that represent the risk for specific incidents [7]. It is a two-dimensional matrix, with one axis representing the consequence and the other axis represents the likelihood. The analyzed incidents are then placed within the matrix where its likelihood and consequence match both corresponding axes. An incident that has an almost certain likelihood of occurrence and catastrophic consequences is an incident with a high risk. While an incident that rarely ever occurs and at the same time has a negligible impact has a low risk. The result of each incident can be used to decide whether or not any action should be made to reduce the risk.

Using a mathematical function is one way to measure risk [23], and in this thesis the risk was calculated as the multiplication of likelihood and consequence:

\[
Risk = Likelihood \times Consequence
\]  

(2.1)
3 Maritime Connectivity Platform

In this chapter an overview of STM’s current approach to manage identity is presented. Its accompanied web application is presented, and how identities are stored and managed within it. This chapter is presented to give the reader insight about how the current identity management operates, and also highlight the vulnerability of this centralized registry.

3.1 Overview of Maritime Connectivity Platform

MCP is an online platform where actors involved in STM can log in to manage their organizations, ships and utilize services. It is accessed through a web application where the users are authenticated by providing their email and private key (password). The users are granted different permissions in MCP depending on their roles, which are set by an admin of MCP when a user’s registration is approved. All registered entities, such as persons, ships and ports, are stored in a central database and the system uses a PKI.

3.2 Centralized Identity Registry

Currently, all identities of organizations and ships involved in STM are managed in MCP. MCP is a communication framework created in STM for secure and reliable communication between authorized maritime actors. Apart from the identity registry, MCP is composed of two other core components: service registry and maritime messaging service. However, the focus is only on the identity registry of this thesis. Its objective is to ensure identification and authorization for MCP and its users.

This system utilizes a PKI and acts as a CA itself, which can issue and sign certificates used for authentication of organizations and ships in MCP. In consequence, the root of trust is established in MCP. And, all identities in MCP are managed and stored in a centralized database. This means that involved actors got to trust the administrators of MCP with their data, they are not in control of it, and they cannot decide who to share it with.
3.3 Web Application

MCP’s functionality is available for use through a web application, MCP Management Portal, where its services can be utilized by authorized users. Prior to gaining access to the services as a user, the organization the user belongs to must be registered. This is done by having a person who holds the legal rights to act on behalf of the organization applying to MCP through a web form. If an administrator of MCP approves the application the organization is registered. The administrator of the registered organization now has the authority to add new users to MCP under their organization. New users are added simply by filling out a form with fields such as Maritime Resource Name (MRN), name and role. MCP requires a MRN for every entity in the identity registry and different roles grant permission for different actions that can be performed in MCP.

An administrator for a registered organization can issue certificates, that follows the X.509 standard, for their ships. The first time a certificate is issued it contains a private key, which should be stored locally by the one that issued it. After the certificate is issued it is available, without the private key, for download at MCP Management Portal for those who are registered in that organization. For this reason, it is only the one issuing the certificate for the first time that is in possess of the private key.

3.4 Roles and permissions in Maritime Connectivity Platform

MCP employ a centralized identity system with an hierarchical approach. There are three main roles: MCP administrator, organization administrator and user in organization. A MCP administrator has the most privileges and can essentially modify identity data for every entity. MCP administrators can also add new organizations. These organizations has their own administrators, users, ships and ports. The organization administrators can add new users, ships and ports to their own organization. A user in an organization cannot modify any identity data, not even their own.

In Table 3.1 the three main roles in MCP are presented and what permissions they have. And as can be observed, someone with the login credentials for a MCP administrator can basically create and delete whatever identity it demands. The single point of failure is a fact for MCP. And also, by decentralizing this registry, actors will be in control of their data and will not have to trust the administrators of MCP. And the experience from STM indicates that more actors are willing to be involved in this project if they are handed the government of it. Another thing that can be observed is that users is not allowed to perform any identity data modifications, but they are still able to utilize services and look up other entities in the registry. Those actions are however out of this thesis scope and analysis and will not be discussed.

<table>
<thead>
<tr>
<th>Role</th>
<th>Add Org</th>
<th>Add to Org</th>
<th>Modify Org Users</th>
<th>Modify Org Ships</th>
<th>Delete Org</th>
</tr>
</thead>
<tbody>
<tr>
<td>Site Admin</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Organization Admin</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>User</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.1: Roles and their permissions in MCP’s identity registry
This Chapter begins with explaining the choice of the distributed ledger used in IMS. This is followed by an overview of IMS and explanations of used components. Then it explains what makes the system decentralized, and then it introduces the roles and permissions in the system. Finally, it gives a description of how the system was implemented.

4.1 Choice of Distributed Ledger

This section explains the choice of the distributed ledger that was used to build IMS. It begins with presenting some requirements that the ledger should satisfy. Then four open source distributed ledgers are presented. The reason these four was examined was because they were according to the author the most developed open source distributed ledgers with focus on decentralized identity. Lastly the choice of the ledger is explained.

4.1.1 Requirements of the Ledger

There are a couple of established initiatives for decentralized identities with the use of distributed ledgers. Most of them are quite similar, but there are some factors that differ between them. When choosing which distributed ledger to use for self-sovereign identity in STM there are some desired features for the ledger. The desired features the ledger should have for this thesis are:

- Transactions to the ledger should be permissioned
- The transactions should be private
- The identities of participants should available for discovery

The reason for having a permissioned ledger is because a permissioned ledger requires some authority clearance before transactions are added to the ledger and this makes it possible to have some sort of restraint on whom are added instead of allowing every applying entity to join. This system is not supposed to be for everyone to use, only actors within the maritime sector. And the reason why the transactions should be private is because according to SMA, actors involved in STM wants to be able to keep data regarding business a secret for
not authorized actors. The last requirement is needed for making it possible for entities to look up other entities on the ledger to initiate interaction between them.

4.1.2 Hyperledger Indy

Hyperledger Indy is a permissioned ledger where every actor creates their own identity and builds its own web of trust. The transactions are private and new DIDs for every interaction can be created to establish privacy.

4.1.3 uPort

uPort is a distributed ledger that focuses on self-sovereign identities and is open for everyone. It is built on top of Ethereum, an open source distributed ledger, and is permissionless and public. uPort utilizes DIDs as identifiers on the ledger and is similar to Hyperledger Indy but it is not permissioned.

4.1.4 ShoCard

ShoCard uses DLT to provide identities by binding an identity together with an existing trusted credential (e.g., passport) and cryptographic hashes that are stored in permissionless Bitcoin transactions. ShoCard does not only verify identities through online interactions, but also face-to-face.

4.1.5 OneName

OneName uses the Namecoin blockchain and is a system for a decentralized identity system. They compare the identity one OneName with Blockchain, where private keys are used to acquire complete control over the Bitcoins - no one but the owner of the private key can move them. And it works in the same way for the identities. OneName is open-source, is permissionless and made for everyone to use.

4.1.6 Choosing a Ledger

Table 4.1 lists the chosen distributed ledgers together with their features.

<table>
<thead>
<tr>
<th>Ledger</th>
<th>Permission Feature</th>
<th>Transaction Type</th>
<th>Discoverable Identities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hyperledger Indy</td>
<td>Permissioned</td>
<td>Private</td>
<td>Yes</td>
</tr>
<tr>
<td>uPort</td>
<td>Permissionless</td>
<td>Public</td>
<td>Yes</td>
</tr>
<tr>
<td>ShoCard</td>
<td>Permissionless</td>
<td>Both</td>
<td>Yes</td>
</tr>
<tr>
<td>OneName</td>
<td>Permissionless</td>
<td>Public</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 4.1: Overview of distributed ledgers

The only self-sovereign identity initiative examined in this thesis that is permissioned is Hyperledger Indy, the rest are created to be open and available for everyone. This means that there are no constraints on whom are joining the registry. ShoCard has another problem, the trust is established in the provider of the credentials which is a trusted third party.

The most essential factor to choose Hyperledger Indy to use in this thesis was because it is a permissioned ledger. Also, the transactions and the data stored in the ledger is not available for the public, but the identity of the participants is still known. In comparison with

1https://github.com/hyperledger/indy-node
2https://github.com/uport-project/specs
3https://github.com/drwasho/onename
Blockchain and Ethereum, where all transactions are available and public for everyone, but participant identity is almost impossible to interpret. The combination of acknowledging the identity of the participants and preserving the data and the interactions between them secret, makes Hyperledger Indy a good choice for this system. Also, Hyperledger Indy provided all functionality to create a self-sovereign identity system and used DIDs as identifiers on the ledger. And, related work often mentions Sovrin [25] as a matured self-sovereign identity initiative, and Sovrin is built on top of Hyperledger Indy’s codebase. According to Abraham [1], “Out of all the evaluated technologies, Sovrin checks most of the boxes when developing a self-sovereign identity system” and “Sovrin got the best result especially because its design is made to realize a self-sovereign identity system”. By using an already proven ledger for identity management, errors in the implementation were avoided and more focus could be put on the analysis of the system.

4.2 Component Overview

This Section presents an overview of IMS and its components.

IMS consists of four main components: a network, a distributed ledger, Identity Manager, and Hyperledger Indy’s SDK. A visual presentation of IMS can be seen in Figure 4.1. Hyperledger Indy provides the functionality to run nodes that create the network that establishes a self-sovereign identity ecosystem on a distributed ledger. The network runs in a Docker container and consists of four nodes where each node has its own copy of the distributed ledger. This network can be connected to through the Identity Manager, to interact with entities (DIDs) and the ledger. Identity Manager is a component that every entity has an instance of and is represented as a Java class that the author implemented and it uses Hyperledger Indy’s SDK to perform operations for managing DIDs and communication with other entities.

![Figure 4.1: Overview of IMS’s components and their relation](image)

4.2.1 Identity Manager

All entities that connect to the network and utilize the decentralized identity registry has an instance of the Identity Manager. A visual representation of it can be seen in Figure 4.2. The black arrows represent inheritance and the white show data flow. It is represented as a Java class.
4.2. Component Overview

A class that contains two other classes: Configuration and Entity. The Configuration class holds the information required to connect to the network and Entity holds information regarding the entities, such as name, role, DIDs and the digital wallet. In IMS, all users have their own so called digital wallet. This wallet contains the user’s keys. Most importantly, it contains the user’s private key, which is used to prove ownership of an identity in the ledger. The wallet is a software that is stored on a device that the user holds. In IMS the wallet is saved in the local storage of the computer on which the Identity Manager is run. HyperLedger Indy’s SDK provides a class, named Wallet, that is used to interact with the locally stored wallet and its keys. Every Entity has a DID class that the SDK provides, which manages everything regarding their DIDs, such as the creation and modification of them.

![Figure 4.2: Overview of Identity Manager](image)

4.2.2 Hyperledger Indy

Hyperledger Indy make use of DIDs that were introduced in Section 2.2.2 to manage identities. They can be used in different ways in Hyperledger, but in this thesis they are used in two ways: First as the identity for a user on the ledger, this is called a Verinym DID. The other way is called a Pairwise-Unique Identifier, where two DIDs are used pairwise by two users, and is only used for secure interaction between them.

In IMS there are three significant parts of the DIDs: id, public key and private key. Id is the public identifier for the DID. It is used to look up the DID in the ledger. Public key is the key of the DID that was stored on the ledger together with the id of the DID. The private key is the key the entity uses to prove ownership of a DID and is stored in the entity’s digital wallet. Figure 4.3 displays an overview of these parts and their relationship.

To establish a secure way for two entities to communicate, they need to exchange a connection request and a corresponding connection response and create one Pairwise-Unique Identifier each. They are used to set up and maintain privacy of the connection. The first entity creates a connection request that contains the id of the entity’s Pairwise-Unique Identifier.
4.3 Decentralized Identity Registry

Figure 4.3: Overview of DID’s parts and their relationship

that was created for this specific connection. The connection request is added to the ledger and then sent to the second entity, which answers with a connection response. This connection response contains the id and the public key of the Pairwise-Unique Identifier that the second entity created for this session. When the first entity receives the connection response, it adds the second entity’s Pairwise-Unique Identifier to the ledger. The two entities can now use these Pairwise-Unique Identifiers to maintain a secure connection.

After a session between two entities are finished, it is possible to use the same Pairwise-Unique Identifier to communicate with the same entity again. But, if a session is to be established with a new entity, a new Pairwise-Unique Identifier should be created. DIDS, and other globally unique identifiers can be used for correlation [32]. This can be mitigated by using different Pairwise-Unique Identifiers for every relationship. The Pairwise-Unique Identifier can be seen as a pseudonym for the verinym DID.

Hyperledger Indy comes with default roles and rules for the entities, from Sovrin’s design, and none of them were modified in any way. The three roles, from least privileged to most privilege, that are used in IMS are: common user, trust anchor and steward. There are several different permissions that each role has, but there is only one permission that is essential for data integrity and this thesis, and that is the permission to add new users to the ledger. As common user you can not add any new users, as trust anchor you can only add users as common user and as steward you can add users both as common user and as trust anchor. Only the creator of the system can add users with the role as steward.

4.3 Decentralized Identity Registry

Since every pair of id and public key for the DIDs are stored on a distributed ledger a PKI is not utilized in IMS, hence neither any CA that assists with authentication of identities. Instead a DPKI is employed where no single trusted party can compromise the integrity of all users’ identity data. And, instead of trusting one central authority, actors can choose who to trust. In the Voyage Scenario, the arrival port observes that the departure port added the ship to the ledger. And, since it trusts that port it chooses to also trust the ship. The arrival port can decide who to trust and therefore build its own web of trust. If the arrival port does not trust the departure port it does not trust the ship on the fact that the departure port added the ship in the registry. Nonetheless, the ship could have gathered verifiable claims, mentioned in Section 2.2.2 from other ports, which the arrival port trusts, and by doing so established trust of the port. These verifiable claims are however not used in IMS since they are out of the scope for this thesis.
4.4 Roles and permissions in Identity Manager System

In IMS there is no central authority that is in control of the users’ identity data. The identities in IMS are managed by the identity owners themselves and are stored in a distributed ledger. The ledger is however permissioned, which means that there are actors with various permissions for adding new identities to it. It is important to note that even though the system at a first glance might appear centralized because of these actors with permission, the identity data is saved in a distributed ledger which no one has full control over. And, actors can choose who to trust instead of blindly trust one central organization.

One difference from MCP’s role structure is that ships and ports has their own role instead of just an identity. The difference here is that ships and ports can add other ships and ports, if they have roles that gives them that permission.

In Table 4.2 the three roles used in IMS and their permissions are displayed. Notice that roles are only given permission to add new users. They can not remove or alter any of the other roles’ identity data in any way.

<table>
<thead>
<tr>
<th>Role</th>
<th>Add User as Steward</th>
<th>Add User as Trust Anchor</th>
<th>Add User as Common User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steward</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Trust Anchor</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Common User</td>
<td></td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2: Roles and their permissions in MCP’s identity registry

4.5 How the Identity Manager System was Implemented

The system that serves as Proof of Concept for this thesis was written in Java and with the use of Hyperledger Indy and its included SDK [27]. The goal when building the system was to create a decentralized identity system that would live up to the phrase and scenario, mentioned in Section 1.2, where a ship leaves a port and arrives at another port the ship has never visited in a secure manner.

Hyperledger Indy’s SDK provides several Java classes and their respective methods. The significant ones used to create the IMS were: Did, Wallet, Ledger, Pool and Crypto. Did was used for creation, storing and manipulation of DIDs. Wallet was used for storing and usage of keys. Ledger was used for building and sending request to the distributed ledger. Pool contained the distributed network and validators for the ledger. Crypt was used for encryption, decryption and authentication.

In excess of these classes three other Java classes were implemented: IdentityRegistry, Configuration and Entity. An overview of the implementation is presented in Figure 4.4. IdentityRegistry is the main class that performs all operations with assist of the SDK and contains the instantiated instances of the two other classes. The class Configuration contains a list of all the instances of initiated Entity and the instance of the Pool class, which contains the validators. The Entity class consists of a name and a Wallet.

Four objects of the class Entity were instantiated to act as the ship, the two ports and the organization SMA. Each Entity had a name (e.g. arrivalPort) and a Wallet for managing their DIDs. Each entity was added to the class Configuration, which also contained the decentralized network (Pool). This network was established with the use of Hyperledger Indy and composed of four nodes. Each node contained their own copy of the distributed ledger, which from the start only contained the verinym DID for SMA as the role steward.

A note: As mentioned in Section 2.4.2 there are some requirements for DPKI that needs to be met, one being that private keys must be generated in a decentralized way. Since the IMS was created on a single computer all entities’ DIDs were created on the same device. This
4.5. How the Identity Manager System was Implemented

is not allowed as it means that all identities have access to the same device, which implies that they have physical access to each other’s private keys. Therefore, in this thesis each instance of Entity, which has its Wallet containing private keys, is treated and observed as an independent device.

Figure 4.4: Overview of implementation of Identity Manager
5 Evaluation method

This chapter begins with describing how the Voyage Scenario was redefined. This is followed by an explanation of how the registration procedures were composed for each system. Then it presents how the likelihood of modified identity data was counted for each identity system. This is followed by a presentation of how the consequence of stolen keys for each system was counted. And finally, it explains how the result from the likelihood and consequence analyses were combined into an analysis of the risk for compromised data integrity.

5.1 Redefinition of The Voyage Scenario

In Section 1.5 the Voyage Scenario that SMA use as a guideline is introduced. This scenario begins with a ship leaving a port and ends with the ship arriving at another port it has never visited. This scenario was modified, and the action of registration was added, to make the scenario more adapted for this thesis and usable in all analyses. Since this scenario has earlier been used in the establishment of MCP the author believes the analysis is put in a context that is well fitted for this thesis. And by adding the registration procedure in the Voyage Scenario the systems are evaluated in a presumably realistic setting.

The redefinition of The Voyage Scenario: There are two ports, departure port and arrival port, that belongs to the same organization and they are both registered in the identity registry. One unregistered ship is moored at the departure port. The ship undergoes the registration procedure and is added in the identity system. The ship initiates communication with the arrival port and both parts authenticate each other.

A few preconditions where added to the Voyage Scenario to define it even further:

- The departure port and the ship trust each other.
- Both ports trust each other (they are in the same organization).
- The departure port and the ship has already established a safe connection to interact over.

This scenario was used as base to give the systems same reference and to put the procedures in a realistic context.
5.2 Composing Registration Procedure

As mentioned in Section 2.5.1, the efficiency of the systems was measured in the total number of actions required for a ship to join the identity registry. This section explains how the required actions for a ship, from the process where a ship goes from not being registered in the registry, to it being registered, were acquired. These actions were used to compare the efficiency of the systems’ registration procedure when a ship registers. First, this section describes all the preconditions that concern both systems, then it presents how the registration procedures were formed for each system. And lastly, how the two systems’ registration procedure was compared.

5.2.1 Precondition for Registration Procedure

For both systems, the least possible amount of entities and actions were used to complete both the Voyage Scenario and the registration procedure. The least amount of entities required was obtained by only including and adding entities to the registry that were necessary for the ship to register. E.g., the departure port was needed because the ship registered through it, and the arrival port was needed because the ship had to authenticate and be authenticated by it. The same was applied to the actions, only the actions necessary to complete the registration of the ship was used. E.g., the ship had to initiate contact with some entity to start the registration procedure and thus that action was included.

The ports and the ship was set to belong to the same organization and therefore only one organization admin was needed instead of one for each ship. This was done to make the comparison as fair as possible, since a lower amount of entities that can modify data result in a lower risk for compromised data.

The author acted as every involved entity except for SMA in both registration procedures.

5.2.2 Registration in Maritime Connectivity Platform

MCP provides a guide [5] that explains how entities are authenticated in MCP, and another guide [19], for how to register and use MCP. By using this information together with following the steps in the Voyage Scenario, every action of every entity that had to be executed was noted and put together after each other. The result gave the registration procedure in MCP. Then, by selecting the actions performed by the ship, all actions required by the ship to register was obtained.

5.2.3 Registration in Identity Manager System

Four instances of Identity Manager were created, each for every entity in the Voyage Scenario: system admin, ship and the two ports. The system administrator was added with the role Steward. Both ports were added to the registry. The departure port was given the role as Trust Anchor to give it authorization to approve new entities. The arrival port was given the role as Common User.

Hyperledger Indy’s codebase can be found at [github.com/hyperledger/indy-node](https://github.com/hyperledger/indy-node), where they also provide a "Getting Started Guide". This guides presents a scenario where one entity adds another entity to the ledger. Within this scenario there are steps such as creating and adding Pairwise-Unique Identifiers to the ledger, creating connection response/request and encrypting/decrypting data. These steps were extracted and used together with the Voyage Scenario and implemented in IMS as one Java function called run, to create all of the necessary actions for the Voyage Scenario to be completed. Every action of every entity was noted in the same way as for MCP and it resulted in the registration procedure in IMS. Again, all actions performed by the ship was filtered out to obtain all the actions the ship was required to execute in the registration procedure.
5.2.4 Comparing the Procedures

To analyze how the registration procedure efficiency differed between MCP and IMS, the procedure where a ship registered in each system was analyzed and each action required by the ship was counted and summarized. Each interaction with another entity was defined as an action, e.g., when the ship fetched data from the database or sent a message to a port it was observed as one action. And each function in IMS that was used and obtained from Hyperledger Indy’s SDK was also defined as an action. E.g., a function that creates a Pairwise-Unique Identifier was observed as an action. The count of actions for each system was then compared.

As mentioned in Section 2.5.1, the efficiency of the systems were measured in the total number of actions required for a ship to join an identity registry. The reason the total count of actions was used was because it was a factor that could be measured in the same way for both systems, resulting in a fair comparison. IMS does not have a UI and therefore measuring the time for each action would not have been a fair comparison. Even though each action’s time could have been approximated it would probably not give a trustworthy result, since the design of the UI can alter the time of each action [17].

5.3 Risk for Compromised Data Integrity

This section begins with presenting how the likelihood for modified identity data was analyzed for both systems. Then it presents how the impact of lost keys was analyzed for each system. Lastly, it explains how the results from these two analyses were used in combination to analyze the risk for compromised data integrity in comparison to each system.

5.3.1 Likelihood of Modified Identity Data

To investigate how the likelihood for compromising the integrity of identity data for the two systems differ, a comparison of the systems’ entities permissions to modify entities’ data were made. The entities from the Voyage Scenario were also used in this analysis to put the comparison in a realistic context. Also, the analysis was made after the ship in the Voyage Scenario had completed the registration procedures. And, the roles the entities had was based on which role they had before or were given in the registration procedures. That means that any entity that was not used or involved in neither the Voyage Scenario nor the registration procedure for a system was not added to the registry.

Each entity and its corresponding role are presented in Table 5.1. As can be observed for the roles in MCP, there are four entities that does not have a role. The captain of the departure port was not involved in neither the Voyage Scenario nor in the registration procedure and was thus not added to the registry and does therefore not have a role in MCP. Regarding the ports and the ship, MCP does not have roles for these kinds of entities even though they are in the registry. And, they do not have the ability to perform any data modification operations, but since they were still used they were added in the registry.

IMS way of managing entities differ a bit from MCP and users do not have to belong to an organization to be added in the registry. That is the reason why no organization admin were involved in neither the Voyage Scenario or in the registration procedure for IMS. Also, none of the captains had to be added in the registry for any of the scenarios in IMS.

The difference in how the registries manage their entities and roles results in more involved entities for MCP than IMS when implementing the Voyage Scenario. Even though the author added preconditions such as putting the ports in the same organization, resulting in only one organization admin for MCP instead of two, MCP still demands more entities than IMS. This could have an negative impact of the likelihood of compromised identity data for MCP since there are more involved actors which could increase the likelihood that identity
data is modified. This emphasize the hierarchical of centralized identity and its single point of failure thus MCP requires entities to have some sort of guardians to be in the registry.

<table>
<thead>
<tr>
<th>Entity</th>
<th>Role in MCP</th>
<th>Role in IMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Admin</td>
<td>Site Admin</td>
<td>Steward</td>
</tr>
<tr>
<td>Organization Admin</td>
<td>Organization Admin</td>
<td>None (not added)</td>
</tr>
<tr>
<td>Captain of Ship</td>
<td>User</td>
<td>None (not added)</td>
</tr>
<tr>
<td>Captain of arrival port</td>
<td>User</td>
<td>None (not added)</td>
</tr>
<tr>
<td>Captain of departure port</td>
<td>None (not added)</td>
<td>None (not added)</td>
</tr>
<tr>
<td>Departure port</td>
<td>None</td>
<td>Trust Anchor</td>
</tr>
<tr>
<td>Arrival port</td>
<td>None</td>
<td>Common User</td>
</tr>
<tr>
<td>Ship</td>
<td>None</td>
<td>Common User</td>
</tr>
</tbody>
</table>

Table 5.1: The entities with their corresponding role in each system

The permissions MCP’s different roles had were obtained from STM’s information about MCP’s implementation \[5\]. IMS’s roles and permissions were obtained from Sovrins Foundation’s network roles and permissions \[11\].

For both systems and for each entity, every occurrence of an entity that was authorized to modify an entity’s data were counted.

5.3.2 Consequence of Stolen Keys

To investigate the difference for the systems regarding the consequence of stolen keys, every entity’s permission involving any modification were counted. This was done to examine which actions an unauthorized entity would be able to do if it would gain access of an entity in the registry’s private key. The same entities that were used in the likelihood analysis in Section 6.2.1 that can be seen in Table 5.1 were also used in this analysis. Every authorized action that changed an entity’s identity data was counted and summarized for all analyzed entities.

In this analysis the entities of the Voyage Scenario were used to put the comparison in a realistic context. Also this analysis was made after the ship in the Voyage Scenario had done the registration procedures. And, the roles the entities had was based on which role they had or were given in the registration procedures.

5.3.3 Combining the Likelihood and the Consequence

As mentioned in Section 2.5.2 the risk is a combination of the likelihood that something occurs and the consequence of the occurrence. By using the results from the likelihood for modified identity data analysis in Section 5.3.1 and the results from the consequence of stolen keys analysis in Section 5.3.2 the risk for compromised data integrity for each identity registry was calculated, relative to each other.

Since the risk was calculated relative to each other, the lowest and highest result received from both analyses was used as thresholds in a risk-matrix. Both registries were added to the matrix which gave a visual representation of how the risk for compromised data integrity were compared to each registry.

Both the likelihood and consequence analysis resulted in an integer for each registry. The integers received from the likelihood analysis were divided with each other to give a percentage representation of how the difference in likelihood differ. The same was done with the integers received from the consequence analysis.
This chapter presents the results from the analyses presented in Chapter 5. It is divided into two sections, the first presents the results from the registration procedure and the second presents the results of the risk analysis. Overall, IMS sustain both a lower likelihood for modified identity data and a much lower consequence of lost keys compared to MCP. This results in that IMS sustains a much lower risk for compromised data integrity, but the efficiency for IMS is almost halved compared with MCP.

6.1 Registration Procedure

This section begins with presenting the results from the registration procedure analysis and the total count of actions for MCP. Then it presents the these results for IMS. Lastly it presents the results from the registration comparing analysis. As mentioned in Section 5.2.1, the author of this thesis acted as every entity except for SMA during the registration procedure for both systems.

6.1.1 Registering Procedure in Maritime Connectivity Platform

Figure 6.1 presents a sequence diagram that represents the following bullet point list which explains the necessary actions performed in MCP to implement the Voyage Scenario.

- The captain of the ship downloads MCP’s root certificate
- Then it contacts the administrator of the organization it belongs to and requests to be added in MCP, in this case SMA.
- SMA’s administrator logs on MCP Management Platform with its credentials - username and password. There it registers the captain as user in the organization and the ship (without any role).
- The login credentials (username and password) are sent in a mail to the captain of the ship.
- The captain uses these to log in on MCP Management Platform, where it issues a certificate for the ship.
6.1. Registration Procedure

- After the ship’s certificate is downloaded it is sent in a message used to initiate contact with, to the arrival port.

- The captain of the port authenticates the ship, together with the received certificate and MCP’s root certificate.

- The captain of the port sends their certificate to the ship and the captain of the ship authenticates it with MCP’s root certificate.

![Sequence diagram over MCP’s registration procedure.](image)

6.1.2 Registration Action Count for Maritime Connectivity Platform

The total number of actions that was required by the ship to register in MCP was 8. The following list present each action that the ship was required to perform to register in MCP:

1. Download MCP’s root certificate
2. Contact administrator of organization
3. Fetch login credentials
4. Log on MCP Management Platform
5. Issue ship’s certificate
6. Create message containing certificate
7. Send message containing certificate to arrival port
8. Authenticate the arrival port with root certificate
6.1.3 Registration Procedure in Identity Manager System

Figure 6.2 presents a sequence diagram that represents the following bullet point list with the required actions executed in IMS to complete the Voyage Scenario.

- The ship initiates communication with the departure port and requests to be added in IMS. This is done outside of IMS and can be done with a mail, phone call etc.

- The port approves and a Pairwise-Unique Identifier, for interaction with the ship, is created in the departure port’s instance of IdM. This Pairwise-Unique Identifier is added to the ledger.

- The departure port creates a connection request that contains the id of the Pairwise-Unique Identifier. This is sent to the ship.

- In the ship’s IdM, with the use of the id in the connection request, the public key for the Pairwise-Unique Identifier is fetched from the ledger.

- A Pairwise-Unique Identifier is then created in the ship’s IdM to be used for interaction with the departure port.

- A connection response containing the ship’s Pairwise-Unique Identifier is encrypted with the use of the fetched public key. This connection response is then sent to the departure port’s IdM.

- The connection response is decrypted with the departure port’s Pairwise-Unique Identifier private key, obtained from the departure port’s wallet, and the Pairwise-Unique Identifier is added to the ledger.

- Now both Pairwise-Unique Identifiers are added to the ledger and can be used for secure communication between the departure port and the ship.

- A Verinym DID for the ship is created in the ship’s IdM and put in a message that is encrypted with the departure port’s Pairwise-Unique Identifier public key. This message is sent to the departure port’s IdM.

- This message is decrypted with the departure port’s Pairwise-Unique Identifier private key. This encryption/decryption process also delivers the sender’s public key, in this case the ship’s. The ship’s public Pairwise-Unique Identifier key is looked up in the ledger with the use of the id from the Pairwise-Unique Identifier. These two public keys are compared to verify the sender. Since they correspond, the ship’s Verinym DID is added to the ledger from the departure port’s IdM with the role Common User.

- The ship begins the procedure to initiate contact with the arrival port. In the ship’s IdM the arrival port’s public key is fetched.

- In the ship’s IdM a Pairwise-Unique Identifier for communication with the arrival port is created and added to the ledger.

- This Pairwise-Unique Identifier is put in a connection request, the request is encrypted with the use of the arrival port’s public key and then sent to the arrival port’s IdM.

- In the arrival port’s IdM the connection request is decrypted, this decryption algorithm also returns the public key of the sender. The ship’s public key is fetched from the ledger and used to discover which entity it was that added the ship to the ledger - it was the departure port.

- In the arrival port’s IdM a comparison of the public key from the decryption of the connection request and the ship’s fetched public key is done, they correspond.
• Since the arrival port trusts the departure port, it decides to trusts the ship. In the arrival port’s IdM a Pairwise-Unique Identifier for interaction with the ship is created and added to the ledger.

• The Pairwise-Unique Identifier is put in a connection request. The connection request is encrypted with the ship’s Pairwise-Unique Identifier’s public key and then sent to the ship’s IdM.

• In the ship’s IdM the connection request is decrypted with the ship’s Pairwise-Unique Identifier’s private key. The ship and arrival port now have each other’s Pairwise-Unique Identifiers which they can use to communicate in a safe manner.

6.1.4 Registration Action Count For Identity Manager System

The total number of actions that was required by the ship to register in IMS was 17. The following list present each action that the ship was required to perform to register in IMS:

1. Sends request to departure port
2. Fetch Pairwise-Unique Identifier’s public key from ledger
3. Create Pairwise-Unique Identifier
4. Create connection request
5. Encrypt connection request
6. Send connection request to departure port
7. Create verinym DID
8. Create message containing verinym DID
9. Encrypt the message
10. Send the message to the departure port
11. Fetch arrival port’s public key from ledger
12. Create Pairwise-Unique Identifier
13. Add Pairwise-Unique Identifier to ledger
14. Create connection request
15. Encrypt connection request
16. Send connection request
17. Decrypt connection response

6.1.5 Registration Comparison

The total number of actions required for a ship to register in MCP is 8, while it requires 17 in IMS and is displayed in Figure 6.3. The number of required actions increased with 9, which result in an decrements of efficiency with 53% when registering in IMS compared with MCP.

6.2 Risk for Compromised Data Integrity

This section first presents the result from the likelihood of modified identity data analysis. Then it presents the results from the consequence of stolen keys analysis. And lastly, it
Figure 6.2: Sequence diagram over IMS's registration procedure.
6.2. Risk for Compromised Data Integrity

presents the result of the risk for compromised data integrity analysis, which is a combination of the two previous mentioned results.

6.2.1 Likelihood of Modified Identity Data

The count of how many entities that were authorized to modify one specific entity in MCP was 13. Its analyzed entities and the entities that were authorized to modify are presented in Table 6.1. The same count but for IMS was 4 and its analyzed entities are presented in Table 6.2. The total count for each system is presented in Figure 6.4. As a result, the likelihood for modified identity data is 69.2% less in IMS than in MCP.

<table>
<thead>
<tr>
<th>Entity to be modified</th>
<th>Entity Authorized to Modify</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Admin</td>
<td>System Admin</td>
<td>1</td>
</tr>
<tr>
<td>Organization Admin</td>
<td>System Admin / Organization Admin</td>
<td>2</td>
</tr>
<tr>
<td>Captain of Ship</td>
<td>System Admin / Organization Admin</td>
<td>2</td>
</tr>
<tr>
<td>Captain of Arrival Port</td>
<td>System Admin / Organization Admin</td>
<td>2</td>
</tr>
<tr>
<td>Departure Port</td>
<td>System Admin / Organization Admin</td>
<td>2</td>
</tr>
<tr>
<td>Arrival Port</td>
<td>System Admin / Organization Admin</td>
<td>2</td>
</tr>
<tr>
<td>Ship</td>
<td>System Admin / Organization Admin</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 6.1: Which entities that are authorized to modify an entity’s identity data in MCP.

<table>
<thead>
<tr>
<th>Entity to modify</th>
<th>Entity Authorized to Modify</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Admin</td>
<td>System Admin</td>
<td>1</td>
</tr>
<tr>
<td>Departure Port</td>
<td>Departure Port</td>
<td>1</td>
</tr>
<tr>
<td>Arrival Port</td>
<td>Arrival Port</td>
<td>1</td>
</tr>
<tr>
<td>Ship</td>
<td>Ship</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 6.2: Which entities that are authorized to modify an entity’s identity data in IMS.

![Figure 6.4: Total count of entities that are authorized to modify identity data for each system.](image)

6.2.2 Consequence of Stolen Keys

MCP received a total of 25 possible actions that modifies an entities identity data somehow. If one entity can modify one entity’s data it is counted as one action. If it can both modify and delete an entity’s data it is counted as two actions. Table 6.3 presents which actions that was possible to perform on entities with different entities’ keys in MCP. For IMS the total count of possible actions was 4. Table 6.4 present the possible actions in IMS. The total number of actions for both registries are displayed in Figure 6.5. The consequence of lost keys is 84.0% less in IMS than in MCP.

<table>
<thead>
<tr>
<th>Action</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Admin</td>
<td></td>
</tr>
<tr>
<td>Departure Port</td>
<td></td>
</tr>
<tr>
<td>Arrival Port</td>
<td></td>
</tr>
<tr>
<td>Ship</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.3: Which actions were possible to perform on entities with different entities’ keys in MCP.

<table>
<thead>
<tr>
<th>Action</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Admin</td>
<td></td>
</tr>
<tr>
<td>Departure Port</td>
<td></td>
</tr>
<tr>
<td>Arrival Port</td>
<td></td>
</tr>
<tr>
<td>Ship</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.4: Which actions were possible to perform on entities with different entities’ keys in IMS.
### 6.2. Risk for Compromised Data Integrity

<table>
<thead>
<tr>
<th>Entity’s key</th>
<th>Actions on System Admin</th>
<th>Actions on Org Admin</th>
<th>Actions on Cpt. of Ship</th>
<th>Action on Cpt. of Arr. Port</th>
<th>Actions on Dep. Port</th>
<th>Actions on Arr. Port</th>
<th>Actions on Ship</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Admin</td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>14</td>
</tr>
<tr>
<td>Org Admin</td>
<td>Modify</td>
<td>Delete</td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>11</td>
</tr>
<tr>
<td>Cpt. of Ship</td>
<td>Modify</td>
<td></td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>0</td>
</tr>
<tr>
<td>Cpt. of Arr. Port</td>
<td>Modify</td>
<td></td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>Modify</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 6.3: The potential actions on entities identity data with different entities’ keys in MCP.

Table 6.4 presents which actions that was possible to perform on entities with different entities’ keys in IMS.

<table>
<thead>
<tr>
<th>Entity</th>
<th>Actions on System Admin</th>
<th>Actions on Departure Port</th>
<th>Actions on Arrival Port</th>
<th>Actions on Ship</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Admin</td>
<td>Modify</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Departure Port</td>
<td>Modify</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Arrival Port</td>
<td>Modify</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Ship</td>
<td>Modify</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

Table 6.4: The potential actions on entities identity data with different entities’ keys in IMS.

![Figure 6.5: Number of total possible identity modification actions for each registry.](image-url)

Figure 6.5: Number of total possible identity modification actions for each registry.
6.2. Risk for Compromised Data Integrity

6.2.3 Combining the Likelihood and the Consequence

Table 6.5 presents the systems in a risk-matrix. Important to note that the consequence and risk levels are relative to each of the systems. The lowest likelihood is 4, it is set by IMS and the highest, 13 is set by MCP. The lowest consequence is set by IMS with 4 and the highest is 25, set by MCP. The risk for compromised data integrity was calculated for each system by multiplying the likelihood of data integrity compromise with the consequence of stolen keys. The results are shown in Figure 6.6. The risk for compromised data integrity is 95.1% less in IMS compared to MCP.

Table 6.5: The risk for compromised data integrity for both systems, relative to each other

<table>
<thead>
<tr>
<th>Consequence</th>
<th>High (19-25)</th>
<th>Medium (12-17)</th>
<th>Low (4-10)</th>
<th>Low (4-7)</th>
<th>Medium (8-10)</th>
<th>High (11-13)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Likelihood</td>
<td>MCP</td>
<td>IMS</td>
<td></td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
</tr>
</tbody>
</table>

Figure 6.6: The risk for compromised data integrity for each system.
This chapter begins with discussing the results obtained from the analysis. Then it discusses the methods used in this thesis. And lastly, it discusses the work in a wider context.

### 7.1 Results

This section begins with discussing the results from the registration procedure comparison. This is followed by a discussion of the results from the likelihood of compromised data integrity analysis. Then it discusses the results regarding the consequence of stolen keys analysis. Finally, it discusses the results from the risk for compromised data integrity.

#### 7.1.1 Registration Procedure

In IMS, the amount of required actions for a ship to register were more than doubled, increasing from 8 to 17. Since efficiency is defined as the software’s capability to allow users to use a suitable amount of resources in relation to the outcome [24], the outcome for both registries is the same. The correlation between number of actions required for each system gives the difference in efficiency, measured in number of total actions required. This means that the efficiency of the system when an entity is registering in the decentralized registry is 53% lower than the centralized registry. And thus, the usability for the users decreases when registering in the decentralized identity registry instead of the centralized.

No related work used in this thesis acknowledged that a decentralized identity registry would be less efficient than a centralized. Neither did any related work state that it would be more efficient, and therefore no noticeable difference in efficiency and usability for both registries was expected. However, one reason for this result is because the communication in MCP is done through a service that is called SeaSWIM Connector. This service is used for authenticating requests, it is used to send certificates and it encrypts/decrypts all communication. None of these functions are implemented in IMS and it has to be done “manually” by the users. For example, the users themselves have to actively decrypt/encrypt the communication and compare keys to authenticate the sender.

Another reason that increases the number of required actions is because in IMS it is mandatory to create a new Pairwise-Unique Identifier for each interaction with an entity. This “channel” is already established in MCP and is not needed to create for each interaction.
Yet another reason that could explain the difference in required actions is that centralized registries has been around since the beginning of digital identities. They have had more time to adapt to the systems we use today and developed in a way that makes them more efficient, while decentralized registries are quiet new and have not had the same time to mature. Also, some of the required actions in IMS registration procedure could possibly be combined for increased efficiency. E.g, the encrypt and send a connection request actions could be designed so that the connection request is sent at the same time as its encrypted.

### 7.1.2 Likelihood of Compromised Data Integrity

MCP sustains more than a three times higher chance of being exposed by unauthorized identity data modifications than IMS. The likelihood for compromised data integrity is 325% greater for MCP than for IMS and the reasons for this is because MCP has more entities that could potentially have their identity data modified.

In MCP, an unregistered ship’s certificate must be issued by a person before the ship can use it for authentication. Therefore, one person is required to be registered in the same organization. While in IMS, a ship can create its own identity, without any registered person. This lowers the amount of entities that are needed and thus the likelihood for compromised data integrity. Also, another reason for MCP to have a higher likelihood is that there are two entitites that are authorized to modify all other entities, except for System Admin which can only be modified by itself. Meanwhile, IMS’s entities are only authorized to modify themselves.

The reduction of required entities in a decentralized system was not expected and no related work provided any facts about that. And, there is nothing that says that a centralized system requires more entities than a decentralized. The reason for this disparity is caused by how the systems were designed. The architecture of MCP’s identity registry demands more entities registered for these scenarios than IMS’s. But, for example, IMS could have been designed so that a ship’s captain is required to be registered before it can interact through its ship. However, even with the same amount of entities, MCP’s likelihood for compromised data integrity would always be higher, or in some rare situations the same as IMS e.g., if there are only one system admin in the registry. Because of the fact that MCP allows some entities to modify other entities whilst IMS does not.

The likelihood for compromised data integrity is higher for a centralized identity registry than a decentralized identity registry because it allows some entities to modify other entities, while the decentralized system does not allow these kinds of interactions. And generally this is always the case for every centralized identity registry, since the registry must maintain its trustworthiness which it does by retaining their responsibilities mentioned in Section 2.4.1. To guarantee that the identity really belongs to the identity it must be able to modify or remove an invalid identity and therefore some entity has the authority to modify other entities. In a decentralized identity registry this trust is instead established in other users’ actions and an invalid entity would not be modified or deleted. This means that it could exist entities in the decentralized registry that are invalid, which decreases the integrity of the registry. This was however not taken into consideration in this thesis and is something that would have been interesting to investigate.

### 7.1.3 Consequence of Stolen Keys

The impact of lost keys measured as possible actions, is more than six times higher for MCP than IMS. The high amount of measured actions in MCP in comparison with IMS is because of more entities, but also because MCP allows entities to perform more identity data modification actions. Table 6.3 clearly demonstrates the single point of failure that centralized systems experience as mentioned in Section 2.4.1. If someone unauthorized would gain access of the System Admin’s private key it would be possible to modify and remove every entity in the registry which could have devastating outcomes.
As mentioned in Section 2.4.2, the goal with DKPI is to remove the ability for a single party to compromise the integrity of the system and this is undoubtedly achieved in IMS. The only entity that can be modified is the entity of which the private key is accessed. To be able to perform the same actions possible with MCP’s Admin’s private key, you must obtain every entity’s private key. And even then, it is not possible to delete any identity.

### 7.1.4 Risk for Compromised Data Integrity

As can be observed in Table 6.5, the maximum threshold for both the impact and the likelihood are set by MCP, and the minimum thresholds are set by IMS. And, the risk for compromised data integrity for identity data is drastically higher in the centralized system. This was expected, especially for the impact of lost keys where it is 625% greater for MCP in comparison with IMS. As mentioned in Section 2.4.2, a system that use a DPKI approach removes any single point of failure, that a centralized system would contribute to. The reason for this is because entities in the decentralized system are never given authority to modify another entity, while some entities are authorized to modify other entities in the centralized system.

The likelihood for being subject of unauthorized identity data modifications is 325% higher for MCP in comparison with IMS. And by multiplying the consequence with the likelihood for each of the systems we get the calculated risk for compromised data integrity. By diving MCP’s risk with IMS’s risk we can observe that the risk that an unauthorized entity modifies any of the 8 entities used in the analyses is 2031.25% greater in MCP than in IMS. This is if all other parameters are constant and the same for both systems.

### 7.2 Method

This section presents a discussion about the choice of methods in this thesis. And then, a discussion concerning the choice of analyzed factors.

#### 7.2.1 Choice of Methods

This thesis compared the usability and data integrity of a centralized and a decentralized identity registry by building a Proof of Concept for a decentralized registry and comparing it with MCP. It was implemented with the use of Hyperledger Indy and the entities’ roles and their permissions where inherited from Hyperledger Indy’s definitions. If the analyses would have been done with another set of roles and rules for IMS, different results would probably have been observed.

Throughout all of the analyzes, SMA’s phrase and the Voyage Scenario was used as foundation. It was used to put the systems in the same situation and in a realistic context when analyzing them. This is fair and all good, but what is not, is that IMS was implemented to only accomplish these two. MCP on the other hand, was created to operate during the whole cycle of the STM. If IMS was created to perform all the actions MCP can, it is possible that these implementations in IMS would affect the design and thus the results in the analyses.

Since the focus of the usability analysis was on the users of the system, the analysis in this thesis was made only from the ship’s perspective. But how are the other actors affected by a decentralization of the identity registry? Also, the assignment to administer applications that a central actor did, is distributed between the users of the system. How does that affect them? There are a lot of other aspects that could have been considered in the analyses that would have been interesting to examine, to name a few. Also, the analysis would have been fairer if IMS was designed the same way as MCP. There are more actions that are required to register in IMS than in MCP, but all of them do not directly relate to the difference of registering in a centralized and a decentralized system. Some actions such as decrypting/encrypting are done in the SeaSWIM Connection service in MCP, but in IMS the users have to perform these. However, conclusively there are more actions required in IMS because of all requests to the
7.2.2 Choice of Analyzed Factors

According to Metzker and Seffah [24], efficiency is usually measured by estimating the cost in time. But, for this thesis the efficiency of the registration procedure was evaluated and compared for both registries by counting the number of actions that were required. The choice to use number of actions as the measurement for cost was made because IMS was implemented without a UI. Thus, it was not possible to make a fair comparison of the registries’ registration procedure regarding the amount of time it would take a user to register. However, there is no proof that the performed actions required a similar amount of time and for this reason the number of actions does not reflect the actual time the registration required. It is important to notice that the efficiency was measured with the number of actions as the cost, therefore a higher number of action does not have to imply that the system is less efficient timewise. But, even if the time of the actions were to be measured, there are more factors involved than only the time it takes to perform these actions, such as delays in the network. This could have been interesting to investigate, especially to observe the difference of a centralized and a decentralized network’s performance.

The analysis of the risk for compromised data integrity, identified which entities that were authorized to modify an already registered entity and how. But, what this analysis did not examine was the risk of adding invalid entities in the registries. In Section 1.2, integrity is defined as the prevention of unauthorized or improper modification of information. This does not only concern data that is already stored and new entities could affect it. However, the reason that this detail was not included was because in a decentralized system many entities are authorized to add new entities. But, only because an entity can be found in the registry does not mean that another entity blindly trusts it. On the other hand, that is exactly how a centralized system generally work. Consequently, the author believed that the comparison would become more fair and straightforward if only already saved data was used in the analysis. After all, since authorized entities can add invalid data, it associates with the trust in the entities rather than the design of the system.

7.3 The work in a wider context

Even though DLT creates a lot of possibilities which can have ethical benefits, there are also drawbacks. Different distributed ledgers use different consensus algorithms. These consensus algorithms are used to guarantee the order of the transactions and to validate the block of transactions in the network [15]. Blockchain use, what is called, a Proof of Work consensus algorithm that consumes a huge amount of energy. The energy consumption is so enormous that a paper [30] was published where they researched and explored fiscal policy options and climate change mitigation considering Blockchain technology. This is something to keep in mind when deciding which distributed ledger and consensus algorithm to use. Fortunately, Hyperledger Indy does not use Proof of Work to establish consensus. It is instead achieved by a permissioned, voting-based strategy where each node in the network vote for the order of the transactions. This does not require massive amounts of energy and reduces the impact on the environment. This method does however require that messages gets transferred to all of the nodes on the network [15]. As a consequence, a large network with many nodes demands more time, but this might be a worthy trade-off.

Allowing entities to be in control of their own identity and data online is something to strive for, but there are those who exploit these perks for illegal activities. An example is Bitcoins, an online virtual currency that can be used without identifying the buyer or the payee [20]. This has opened up the possibility for people to buy products and services online anonymously. This opportunity is unfortunately abused to perform payments for illegal
material. One negative thing this anonymously perk the cryptocurrencies bring, is the sexual exploitation of innocent children. It is possible to witness a downward trend in credit card card payments for Child Abuse Material (CAM). But, that does not mean that the overall payments have decreased, the payments realized with digital currencies are instead filling that gap. It is important to understand what possibilities a new and unexplored technology contribute with, both good and bad, before using or implementing it.
8 Conclusion

According to SMA, centralized solutions require more administration, and they have experienced a more successful adoption in using a decentralized architecture. Today, they are using a centralized identity registry in the STM which adopt to the first phase of model for online identity - centralized identity. A possible solution to many of the problems centralized identity has is a self-sovereign identity, but how would a change to a decentralized self-sovereign identity registry affect the system? To answer this question, this thesis compared a centralized identity registry with a decentralized identity registry on usability and data integrity. This was realized by taking the already implemented centralized registry, creating a Proof of Concept of a decentralized identity registry, and then compare them to answer the research questions introduced in Section 1.4.

A self-sovereign identity system uses DIDs as identifiers and stores them in a decentralized manner by utilizing DLT. In this thesis, the Proof of Concept used Hyperledger Indy as the ledger for the registry. This choice was based on that Hyperledger Indy is a private and permissioned ledger which has the ability to implement restraints on whom are added to the registry, and related work express that Sovrin, that is built on top of Hyperledger Indy’s codebase, checks most of the boxes when developing a self-sovereign identity system.

8.1 Results

A transfer from a centralized registry to a decentralized registry has the possibility to alter the registration procedure and thus affect the usability of the system. By comparing the procedure when a ship registers in the two registries, the usability, more particular, the difference in efficiency of the systems was determined. The results showed that a transitioning from a centralized identity registry to a decentralized identity registry the registration procedure is affected and the efficiency is decreased. The registration procedure in the decentralized registry demands more interactions and actions. This is mostly caused by the increased interactions with the ledger, and the fact that a new Pairwise-Unique Identifier needs to be created for every session with a new actor. However, there is room for improvement of the decentralized registry’s architecture that could reduce the amount of required actions.

A transition to a decentralized registry affects how the keys in the system are managed. Instead of using the traditional approach with a PKI that the centralized registry apply, the decentralized registry uses a DPKI. This technique removes the need of trust in CAs and
hand over the control to the users of the system. Maintaining the integrity is an important feature for STM and by adopting to a DPKI the integrity of the system could be affected. By reducing the number of entities that has access to an asset, the likelihood for unauthorized changes to it could be lowered. This was examined by implementing the same scenario in both systems and through analyzing the difference of how many entities in each system that were authorized to make any kind of identity data modification on entities in the registry. The analysis showed that the total count of entities authorized to modify was reduced from MPC’s 13 to IMS’s 4 when transitioning from a PKI to a DPKI. The low number for IMS have somewhat to do with the fact that it requires a lesser amount of entities for this analyzed scenario, but essentially because a DPKI only allows entities to modify their own data and no one else’s, which PKI does.

A conversion from a PKI to a DPKI may not only affect the number of entities that are authorized to modify, but it might also affect which actions they are authorized to perform. How would that affect the consequence if someone unauthorized got hold of an entity’s private key? This is another factor that contributes to the system’s integrity. By counting, for each system, the total number of actions that entities were authorized to perform, we determined how the consequence of lost keys differed between the registries. The results from the analysis was as expected in favour of DPKI. The goal of DPKI is to remove single trusted third parties that could compromise the integrity of the system, and the amount of actions each entity was authorized to perform was reduced from 25 (in MCP) to 4 (in IMS). This analysis confirms the goal of DPKI and demonstrates that the single point of failure in PKI is eliminated.

8.2 Future Work

Self-sovereign identity and DLT is still new areas that can be improved in many ways. One big problem DLT faces is the scalability issues, and mass adoption will never happen if the ledgers cannot scale. Since data cannot be removed from distributed ledgers they can only continue to grow in size. This has not been taken into consideration in this thesis, but for future work it is important to be aware of this problem before implementing a solution that is based on a distributed ledger.

DLT is a technique with a wide field of application and therefore there are multiple different ledgers out in the field that are specified in certain areas. Hyperledger Indy’s focus is on self-sovereign identity and is one of the most developed ledgers for this purpose. Two of the most important factors are that it is permissioned and that it uses Pairwise-Unique Identifiers. This both makes it possible to discover public entities and initiate interaction with them, but at the same time the Pairwise-Unique Identifiers makes it hard for unwanted entities to interfere and discover the identity of two interacting entities. For future work, Hyperledger Indy provides a good open source codebase for self-sovereign identities, with the ability to design the system somewhat after preference. However, the issue with scalability still remains and has to be taken into consideration. When that problem is solved, distributed ledgers have a very interesting future ahead.
Bibliography


[32] W3C. Decentralized Identifiers (DIDs) v0.10. 2018. URL: https://w3c-ccg.github.io/did-spec/