KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Computer Science and Engineering

Second cycle, 30 credits

Trust in your friends, on the ledger

Safer reproducible builds through decentralized distribution of .buildinfo files

JOHAN MORITZ

Stockholm, Sweden, 2022

Trust in your friends, on the ledger

Safer reproducible builds through decentralized distribution of .buildinfo files

JOHAN MORITZ

Degree Programme in Computer Science and Engineering
Date: February 28, 2022

Supervisor: Giuseppe Nebbione
Examiner: Mads Dam
School of Electrical Engineering and Computer Science
Host company: Företaget AB
Swedish title: Detta är den svenska översättningen av titeln
Swedish subtitle: Detta är den svenska översättningen av undertiteln

Abstract

All theses at KTH are **required** to have an abstract in both *English* and *Swedish*.

Exchange students many want to include one or more abstracts in the language(s) used in their home institutions to avoid the need to write another thesis when returning to their home institution.

Keep in mind that most of your potential readers are only going to read your `title` and `abstract`. This is why it is important that the abstract give them enough information that they can decide is this document relevant to them or not. Otherwise the likely default choice is to ignore the rest of your document.
A abstract should stand on its own, i.e., no citations, cross references to the body of the document, acronyms must be spelled out, ….
Write this early and revise as necessary. This will help keep you focused on what you are trying to do.

Write an abstract that is about 250 and 350 words (1/2 A4-page) with the following components::

- What is the topic area? (optional) Introduces the subject area for the project.
- Short problem statement
- Why was this problem worth a Bachelor's/Master's thesis project? (*i.e.*, why is the problem both significant and of a suitable degree of difficulty for a Bachelor's/Master's thesis project? Why has no one else solved it yet?)
- How did you solve the problem? What was your method/insight?
- Results/Conclusions/Consequences/Impact: What are your key results/ conclusions? What will others do based upon your results? What can be done now that you have finished - that could not be done before your thesis project was completed?

The following are some notes about what can be included (in terms of LaTeX) in your abstract. Note that since this material is outside of the scontents environment, it is not saved as part of the abstract; hence, it does not end up on the metadata at the end of the thesis.

Choice of typeface with \textit, \textbf, and \texttt: *x*, **x**, and `x`

Text superscripts and subscripts with \textsubscript and \textsuperscript: A_x and A^x

Some useful symbols: \textregistered, \texttrademark, and \textcopyright. For example, copyright symbol: \textcopyright Maguire 2021, and some superscripts: ^{99m}Tc, A*, A^{\textregistered}, and A\texttrademark : ©Maguire 2021, and some superscripts: ^{99m}Tc, A*, A®, and A™. Another example: H\textsubscript{2}O: H_2O

Simple environment with begin and end: itemize and enumerate and within these \item

The following macros can be used: \eg, \Eg, \ie, \Ie, \etc, and \etal: *e.g.*, *E.g.*, *i.e.*, *I.e.*, *etc.*, and *et al.*,

The following macros for numbering with lower case roman numerals: \first, \second, \third, \fourth, \fifth, \sixth, \seventh, and \eighth: (i), (ii), (iii), (iv), (v), (vi), (vii), and $(viii)$.

Equations using \(xxxx \) or \[xxxx \] can be used in the abstract. For example: $(C_5O_2H_8)_n$ or

$$\int_a^b x^2\,dx$$

Even LaTeX comments can be handled, for example: % comment at end

Keywords

Keyword 1, Keyword 2, Keyword3

Choosing good keywords can help others to locate your paper, thesis, dissertation, …and related work.

Choose the most specific keyword from those used in your domain, see for example: the ACM Computing Classification System (https://www.acm.org/publications/computing-classification-system/how-to-use), the IEEE Taxonomy (https://www.ieee.org/publications/services/thesaurus-thank-you.html), PhySH (Physics Subject Headings) (https://physh.aps.org/), …or keyword selection tools such as the National Library of Medicine's Medical Subject Headings (MeSH) (https:

//www.nlm.nih.gov/mesh/authors.html) or Google's Keyword Tool (https://keywordtool.io/)

Mechanics:

- The first letter of a keyword should be set with a capital letter and proper names should be capitalized as usual.
- Spell out acronyms and abbreviations.
- Avoid "stop words" - as they generally carry little or no information.
- List your keywords separated by commas (",").

Since you should have both English and Swedish keywords - you might think of ordering them in corresponding order (*i.e.*, so that the n^{th} word in each list correspond) - this makes it easier to mechanically find matching keywords.

Sammanfattning

Alla avhandlingar vid KTH **måste ha** ett abstrakt på både *engelska* och *svenska*.
Om du skriver din avhandling på svenska ska detta göras först (och placera det som det första abstraktet) - och du bör revidera det vid behov.

If you are writing your thesis in English, you can leave this until the draft version that goes to your opponent for the written opposition. In this way you can provide the English and Swedish abstract/summary information that can be used in the announcement for your oral presentation.
If you are writing your thesis in English, then this section can be a summary targeted at a more general reader. However, if you are writing your thesis in Swedish, then the reverse is true – your abstract should be for your target audience, while an English summary can be written targeted at a more general audience.
This means that the English abstract and Swedish sammnfattning or Swedish abstract and English summary need not be literal translations of each other.

The abstract in the language used for the thesis should be the first abstract, while the Summary/Sammanfattning in the other language can follow

Nyckelord

Nyckelord 1, Nyckelord 2, Nyckelord 3

Nyckelord som beskriver innehållet i uppsatsrapporten

Note that you may need to augment the set of language used in polyglossia or babel (see the file kththesis.cls). The following languages include those languages that were used in theses at KTH in 2018-2019, except for one in Chinese.
Remove those versions that you do not need.
If adding a new language, when specifying the language for the abstract use the three letter ISO 639-2 Code – specifically the ”B” (bibliographic) variant of these codes (note that this is the same language code used in DiVA).

Use the relevant language for abstracts for your home university.

Acknowledgments

It is nice to acknowledge the people that have helped you. It is also necessary to acknowledge any special permissions that you have gotten – for example getting permission from the copyright owner to reproduce a figure. In this case you should acknowledge them and this permission here and in the figure's caption.
Note: If you do **not** have the copyright owner's permission, then you **cannot** use any copyrighted figures/tables/…. Unless stated otherwise all figures/tables/…are generally copyrighted.

I would like to thank xxxx for having yyyy.

Stockholm, February 2022
Johan Moritz

Contents

List of Figures

List of Tables

Listings

If you have listings in your thesis. If not, then remove this preface page.

List of acronyms and abbreviations

API	Application Programming Interface
DLT	Distributed Ledger Technology
RDBMS	Relational Database Management System

The list of acronyms and abbreviations should be in alphabetical order based on the spelling of the acronym or abbreviation.

Chapter 1

Introduction

svensk: Introduktion

Ofta kommer problemet och problemägaren från industrin där man önskar en specifik lösning på ett specifikt problem. Detta är ofta ”för smalt” definierat och ger ofta en ”för smal” lösning för att resultatet skall vara intressant ur ett mer allmänt ingenjörsperspektiv och med ”nya” erfarenheter som resultat. Fundera tillsammans med projektets intressenter (student, problemägare och akademi) hur man skulle kunna använda det aktuella problemet/förslaget för att undersöka någon ingenjörsaspekt och vars resultat kan ge ny eller kompletterande erfarenhet till ingenjörssamfundet och vetenskapen.
Examensarbetet handlar då om att ta fram denna nya ”erfarenhet” och på köpet löser man en del eller hela delen av det ursprungliga problemet. Erfarenheten kommer ur en frågeställning som man i examensarbetet försöker besvara med tidigare och andras erfarenhet, egna eller modifierade metoder som ger ett resultat vilket kan användas för att diskutera ett svar på undersökningsfrågan.
Detta stycke skall alltså, förutom det ursprungliga ”smala” problemet, innehålla vad som skall undersökas för att skapa ny ingenjörserfarenhet och/eller vetenskap.

The first paragraph after a heading is not indented, all of the subsequent paragraphs have their first line indented.

This chapter describes the specific problem that this thesis addresses, the context of the problem, the goals of this thesis project, and outlines the structure of the thesis.

Give a general introduction to the area. (Remember to use appropriate references in this and all other sections.)

We use the *biblatex* package to handle our references. We use the command `parencite` to get a reference in parenthesis, like this \parencite{heisenberg2015} resulting in [**heisenberg2015**]. It is also possible to include the author as part of the sentence using `textcite`, like talking about the work of \textcite{einstein2016} resulting in **einstein2016**.
This also means that you have to change the include files to include biblatex and change the way that the reference.bib file is included.

Use the glossaries package to help yourself and your readers. Add the acronyms and abbreviations to lib/acronyms.tex. Some examples are shown below:

1.1 Background

svensk: Bakgrund

Present the background for the area. Set the context for your project – so that your reader can understand both your project and this thesis. (Give detailed background information in Chapter 2 - together with related work.) Sometimes it is useful to insert a system diagram here so that the reader knows what are the different elements and their relationship to each other. This also introduces the names/terms/… that you are going to use throughout your thesis (be consistent). This figure will also help you later delimit what you are going to do and what others have done or will do.

Discussions on how to verify the lack of malicious code in binaries go at least as far back as to Ken Thompson's Turing award lecture [1] where he discusses the issues of trusting code created by others. In recent years, several attacks on popular packages within the Free Open Source Software (FOSS) have been executed [2] where trusted repositories have injected malicious code in their released binaries. These attacks question how much trust in such dependencies is appropriate. In an attempt to raise the level of trust and security in Free Open Source Software (FOSS), the reproducible builds projects [3] was started within the Debian community. Its goal was to mitigate the risk that a package is tampered with by ensuring

that its builds are deterministic and therefore should be bit-by-bit identical over multiple rebuilds. Any user of a reproducible package can verify that it has indeed been built from its source code and was not manipulated after the fact simply by rebuilding it from the package's .buildinfo file. These metadata files for reproducible builds include hashes of the produced build artifacts and a description of the build environment to enable user-side verification. .buildinfo files are by this notion the crucial link to ensure reproducibility, which also means that a great deal of trust is assumed when using them. Current measures for validating .buildinfo files and their corresponding packages involve package repository managers and volunteers running rebuilderd [4] instances that test the reproducibility of every .buildinfo file added to the relevant package archive. This setup allows users to audit the separate build logs, thus confirming the validity of a particular package. However, because this would be a manual process and the different instances do not coordinate their work, it relies on the user judging on a case-by-case basis whether to trust a package or not. This project seeks to reduce some of that burden from the user while increasing their trust in the software they use by investigating possible decentralized solutions for distributing and proving the correctness of .buildinfo files.

1.2 Problem

svensk: Problemdefinition elle Frågeställning
Lyft fram det ursprungliga problemet om det finns något och definiera därefter den ingenjörsmässiga erfarenheten eller/och vetenskapen som kan komma ur projektet.

Equivalences between human readable source code and binaries are hard to prove **cite**. Likewise is it if the comparison is between source and a hash of a binary. A more easily proved variation of this problem is whether multiple binaries have been built from the same, potentially unknown, source. If the binaries are identical and we trust that the builder is not forging their results, we can be confident in that the binaries were all built the same way. The proof, though, is only as strong as our trust in the builder; an actor which could be compromised without us knowing. With multiple builders, we reduce this risk and our trust can increase likewise.

Distributing the workload creates the need for a system where the build results can be aggregated. Because users have different needs, they should be able to choose their own trust models and use the packages they trust based on

the build results from the different builders. With this as background, we ask how such a system can be designed and implemented in order to maximize user trust in that the packages they use have been derrived from the correct source code.

1.2.1 Original problem and definition

Ursprungligt problem och definition

Some text

1.2.2 Scientific and engineering issues

Vetenskaplig och ingenjörsmässig frågeställning

some text

1.3 Purpose

Syfte

State the purpose of your thesis and the purpose of your degree project. Describe who benefits and how they benefit if you achieve your goals. Include anticipated ethical, sustainability, social issues, etc. related to your project. (Return to these in your reflections in Section 7.4.)

Skilj på syfte och mål! Syfte är att förändra något till det bättre. I examensarbetet finns ofta två aspekter på detta. Dels vill problemägaren (företaget) få sitt problem löst till det bättre men akademin och ingenjörssamfundet vill också få nya erfarenheter och vetskap. Beskriv ett syfte som tillfredställer båda dessa aspekter.
Det finns även ett syfte till som kan vara värt att beakta och det är att du som student skall ta examen och att du måste bevisa, i ditt examensarbete, att du uppfyller examensmålen. Dessa mål sammanfaller med kursmålen för examensarbetskursen.

1.4 Goals

Mål

Skilj på syfte och mål. Syftet är att åstakomma en förändring i något. Målen är vad som konkret skall göras för att om möjligt uppnå den önskade förändringen (syfte).

State the goal/goals of this degree project.

The goal of this project is XXX. This has been divided into the following three sub-goals:

1. Subgoal 1

   för att tillfredsställa problemägaren – industrin?

2. Subgoal 2

   för att tillfredsställa ingenjörssamfundet och vetenskapen – akademin)

3. Subgoal 3

   eventuellt, för att uppfylla kursmålen – du som student

In addition to presenting the goal(s), you might also state what the deliverables and results of the project are.

1.5 Research Methodology

Undersökningsmetod

Här anger du vilken vilken övergripande undersökningsstrategi eller metod du skall använda för att försöka besvara den akademiska frågeställning och samtidigt lösa det e v ursprungliga problemet. Ofta kan man använda ”lösandet av ursprungsproblemet” som en fallstudie kring en akademisk frågeställning. Du undersöker någon intressant fråga i ”skarpt” läge och samlar resultat och erfarenhet ur detta.
Tänk på att företaget ibland måste stå tillbaka i sin önskan och förväntan på projektets resultat till förmån för ny eller kompletterande ingenjörserfarenhet och vetenskap (ditt examensarbete). Det är du som student som bestämmer och löser fördelningen mellan dessa två intressen men se till att alla är informerade.

Introduce your choice of methodology/methodologies and method/methods – and the reason why you chose them. Contrast them with and explain why you did not choose other methodologies or methods. (The details of the actual methodology and method you have chosen will be given in Chapter 3. Note that in Chapter 3, the focus could be research strategies, data collection, data analysis, and quality assurance.)
In this section you should present your philosophical assumption(s), research method(s), and research approach(es).

1.6 Delimitations

Avgränsningar

Describe the boundary/limits of your thesis project and what you are explicitly not going to do. This will help you bound your efforts – as you have clearly defined what is out of the scope of this thesis project. Explain the delimitations. These are all the things that could affect the study if they were examined and included in the degree project.

1.7 Structure of the thesis

Rapportens disposition

Chapter 2 presents relevant background information about xxx. Chapter 3 presents the methodology and method used to solve the problem. …

Chapter 2

Background

Bakgrund

When you do your literature study, you should have a nearly complete Chapters 1 and 2.
You may also find it convenient to introduce the future work section into your report early – so that you can put things that you think about but decide not to do now into this section.
Note that later you can move things between this future work section and what you have done as you may change your mind about what to do now versus what to put off to future work.

What does a reader (another x student – where x is your study line) need to know to understand your report? What have others already done? (This is the ”related work”.) Explain what and how prior work / prior research will be applied on or used in the degree project /work (described in this thesis). Explain why and what is not used in the degree project and give valid reasons for rejecting the work/research.

This chapter provides basic background information about xxx. Additionally, this chapter describes xxx. The chapter also describes related work xxxx.

Vilken viktig litteratur och (forsknings-)artiklar har du studerat inom området (litteraturstudie)?

2.1 Trust

While subjective in an absolute sense, trust can be both described and reasoned about. In [5] the authors modell trust as two directed graphs. Each node can

choose to trust another node, represented by an edge between them in one of the graphs. They can also choose to trust another node as a recommender, thus adding an edge between them in the second graph. Every node that the recommender trusts is also counted as trusted by the original node. This is the foundation for a web-of-trust **glossary entry**, such as used by the PGP protocol for cryptographic signing **verify and cite**.

2.1.1 Public Key Infrastructure (PKI)

2.1.2 Web of trust

2.1.3 Pretty Good Privacy (PGP)

2.1.4 Certificate Authority (CA)

2.2 Confidentiality, Integrity, Availability (CIA)

Within information security, the terms confidentiality, integrity and availability are at the core of how researchers and security auditors describe the security of information systems [6]. They each relate to the respective security risk where an actor can read, write or hinder information when they should not have been able to do so.

2.3 Reproducible builds

In a response to supply chain attacks on package archives for open source software, several projects have started within the linux community in order to raise build reproducibility [3]. Traditionally, linux distributions come with package managers (such as apt **cite** (apt) or pacman **cite** (pacman)) that help users installing and managing programs. While many packages have their source code available online and can be built directly from it by each user, package managers commonly have the functionality to download pre-built programs from an archive. This is convenient for the user but comes with security risks. Using pre-built packages relies on trusting the builder to use the correct source code and that any dependencies needed to build the package are themselves correct.

Building a package reproducibly means it is bit-by-bit identical every time it is built [2]. Verifying its correctness can therefore rely on multiple parties, each building it separately, instead of trusting a single builder. Each builder

can supply a hash of the built software which, if everything has been done correctly, should all be the same. Reproducible builds allows a separation between distributing the software artifact and its verification. Different efforts to make builds reproducible have used various strategies, but a core similarity **is this true??** between them is the use of some kind of specification for the build-environment.

2.3.1 Nix

2.3.2 Debian

2.3.3 Package archive

2.3.4 .buildinfo

In order for builds to be reproducible in different computing environments, the Debian project uses .buildinfo files to describe the necessary parts of the environment in which a package was first built. By recreating this environment on a different machine, build artifacts become identical (if the package is reproducible). .buildinfo files include, among other, name and version of the source package, architecture it was built on, checksums for the build artifacts and other packages available on the system [2]. The .buildinfo files origin and authenticity is given by the builder signing it with their private PGP key. A user can verify that a package has been built from source by comparing its checksum from **hash example** with the one in a corresponding .buildinfo file from a trusted source.

Currently, .buildinfo files are distributed in a centralized archive **cite**. As this is a single-point-of-failure, if a malicious actor takes control of this archive, it could be very hard for users to know whether or not a package should be trusted.

2.3.5 Rebuilding Debian packages

2.4 Distributed Ledger Technology (DLT)

Storing and managing data is commonly done in databases such as Relational Database Management Systems (RDBMSs) or key-value stores **cite**. Because of these solutions' often centralized nature, they come with both integrity and availability risks **cite**. They can become single-point-of-failures. If that data storage is interrupted or manipulated, a system relying on it is at risk.

Distributed Ledger Technologies are an alternative solution to data storage, mitigating the shortcomings of traditional, centralized methods. DLT is an umbrella term for several different technologies which rely on decentralized append-only logs [7] . The data stored in such a network cannot be changed by a central node. Instead there has to be a consensus over the participants on how a change is to be made, followed by that change being propagated to all nodes in the network. Depending on the application, different solutions to how consensus is made and how the ledger itself is represented have been designed, each with its strengths and weaknesses.

The term is sometimes used interchangably with blockchain, but while the latter uses a specific shape on its ledger, the former is more ambiguous. Other examples of DLTs are Certificate Transparency logs **cite** and peer-two-peer networks.

2.4.1 Merkle trees

Patent approved in 1982 [8] as a method for managing digital signatures, Merkle trees have since then been used for applications amongst file sharing and peer-to-peer communication [9], auditing certificate authorities [10] and running blockchains [11]. Merkle trees are directed acyclical graphs where each node's value is a hash based on the values of its child nodes. The leaves of the graph contain the data (or a hash thereof) relevant for a particular application, while the other nodes enable efficient proof mechanisms for validating the integrity of the data. For example given a subgraph (*i.e.,* one with less data), verifying that its supergraph contains a certain value relies only on a subset of their differing nodes. This makes Merkle trees applicable to distributed systems where sending entire graphs between clients would be too expensive.

2.4.2 Consensus

When multiple systems or processes cooperate on a shared state, any change to this state needs to be agreed upon between the different entities. If no agreement, or consensus, can be found, the entities different views of the state can drift away from each other. This can lead to an in-valid system from which no meaningful progress can be made. The problem of creating consensus can be further complicated by assuming that entities can crash and be revived at any time, or even be malicious in the messages they send to the network.

A number of consensus algorithms exists, serving various applications.

One way to differentiate them is whether they are proof of voting based. In a proof based consensus algorithm, only the party that has provided a certain proof is allowed to change the data. Such algorithms can be found in some public ledger blockchains, such as bitcoin. The proof itself can be, for example, finding a number given certain constraints, which is known as proof-of-work. With proof-of-work, the greater computational investment any one participant makes, the greater is the probability that they will be allowed to change the blockchain. However, the greater the computational power is in the whole network, the more limited is any one participants possibility control it or use it maliciously. Other proof based algorithms exist, but they are all centered on connecting responsibility with some type of resource investment. Voting based consensus algorithms on the other hand relate more to a more intuitive understanding of agreement, *i.e.,* democratic voting. Agreement is made only when a certain fraction of the nodes have voted in acknowledgment to a certain decision. This relies on knowing how many nodes there are on the network in total, making voting based consensus less usable in certain scenarios. While simple in idea, a voting based consensus algorithm can become complicated in practice. The algorithm should not only be able to find consensus in perfect conditions, instead a realistic solution should work even if some nodes on the network crashes and, perhaps, even if some nodes are malicious. A consensus algorithm that can handle both of these kinds of issues is called Byzantine resilient [12] or that it has Byzantine fault tolerance [13]. If the algorithm only handles crashes but not malicious actors it has Crash fault tolerance.

Paxos??

2.4.3 Blockchain

Originally described for Bitcoin [14, 15], blockchain is a technology based on Distributed Ledger Technology for storing transactions without needing a centralized organization. Transactions are represented as simple strings of characters which allows them to model essentially anything. This is why blockchains can be used for a broad spectrum of applications; *e.g.,* currencies, ownership contracts etc. **cite**. The name stems from the setup of a blockchain ledger where groups of, closely related in time, transactions are appended together as a block to the current chain by including a hash of the previous block in the latest one. By grouping transactions together, the throuput of the network improves. A consensus algorithm is used to create a total ordering of the blocks so that every node on the network eventually holds the same ledger.

TODO: Add figure for how one block connects to the next.

2.4.4 Peer-to-peer

2.5 Hyperledger Fabric

Run under the umbrella of the Linux Foundation **cite**, the Hyperledger Fabric, or Fabric, is a permissioned blockchain framework with a novel and flexible approach to Distributed Ledger Technology (DLT). A Fabric network can for example choose a consensus algorithm suitable for that particular use case, and define specific requirements for when a particular change to the ledger may be allowed [16]. This section will describe and discuss the main components of a Hyperledger Fabric network and how they work together.

2.5.1 Overview

The Hyperledger Fabric ledger is permissioned which, compared to a public one (such as Bitcoin **cite**), means that it is only available to certain participants. This is regulated by Membership Service Providers (MSPs) on the network, verifying the identities of nodes through Certificate Authorities (CAs) **cite**. Besides MSPs, the nodes on the network can take on one of the roles of *peer* or *orderer*. Every node on network belongs to a some organization whos' MSP determines its role. In this sense, a Fabric network is really a network of organizations rather than one of nodes.

Every peer stores' the networks entire blockchain ledger and validates transactions and changes to the ledger. Orderers, on the other hand, clump together transactions into blocks and delivers them to the peers. This separation of concern is one of the unique features of Hyperledger Fabric, and makes consensus algorithm selection possible. Changes onto the ledger are made by invoking smart contracts, called chaincodes within Fabric. Chaincodes are authorized programs that are run by peers on the network. If multple peers (according to its endorcement policy) get the same result from running the chaincode, any updates are written to the ledger. For performance reasons, a key-value store representing the current ”world-state” is continuously derrived from the ledger and stored on the peers. This allows both reading and writing to happen without going through the entire ledger itself.

TODO: Bootstraping ordering service with a genesis block containing a configuration transaction [16]

TODO: Descibe how Endorcement Policies work

2.5.2 Transaction flow

A transaction in Hyperledger Fabric goes through a number of steps before a change to the ledger happens. First, a client application invokes a particular chaincode by (as of version 2.4) sending a transaction proposal to the Fabric gateway on the network. The gateway is a service running on a peer which takes care of the transaction details, allowing the client to focus on application logic [17]. After recieving the proposal from the client, the gateway finds the peer within their own organization with the longest ledger, the *endorsing* peer, and forwards it to them. The endorsing peer runs the transaction (*i.e.,* chaincode) and notes what parts of the world-state it had to read from and which it will write to (the *read-write set*). This information together with the chaincode's Endorcement Policy informs which organizations has to accept, or endorse, the transaction for it to be valid. Only at the time when every necessary organization has endorsed the transaction can any change be made to the ledger.

The Fabric gateway is responsible for forwarding endorcement requests with the transaction proposal to peer's of each necessary organization, and gather their responses. Each of these peers will then run the transaction proposal and sign their endorsment for it with their private key, if they deem it correct. The gateway recieves the read-write sets and endorcements, validates them, and sends a final version of the transaction to the ordering service. The actual transaction contains the read-write set as well as the endorcements from the different organizations.

As the ordering service is run on other, orderer, nodes on the network, how the ordering service is implemented is completety separated from the functionality of the peers. Its purpose is to group transactions into blocks, order and distribute them to all the peers on the network. By default, Fabrics' ordering service uses Raft, which is voting based crash fault tolerant consensus algorithm [18]. Attempts have been made to add a Byzantine-fault tolerant ordering service to Fabric [19] but so far none has been added to the project.

When a peer recieves a block from the ordering service they add it to their locally stored ledger, validates each transaction and, if valid, updates its local world-state according to the transaction write set. They also notify the client application the status of the transaction. Validation has two parts. First, the transaction must fulfill its endorcement policy and, secondly, the subset of the world-state contained in the read set must not have changed. All transactions, valid and invalid, are added to the ledger, but they are marked to know which ones are which.

```
OR('Org1.member', AND('Org2.member', 'Org3.member'))
```

Figure 2.1: Endorsement policy syntax example

TODO: Add figure of transaction flow.

2.5.3 Endorsement Policies

An update to the Fabric ledger is only possible if the endorsement policy relevant to a transaction has been fulfilled. Endorsement policies describe which and how many organizations or peers that need to run and endorse a transaction proposal. They are defined as logical formulas referencing relevant organization and role for the peers that have to endorse it. Figure 2.1 shows an example endorsement policy with three organizations called Org1, Org2 and Org3 where all endorsing peers must have the *member* role.

An endorsement policy can be defined on three different granularity levels; chaincode-, collection- and key-level. Chaincode- and collection-level policies are decided on when a chaincode is commited to the network. The former is a general policy which always has to be fulfilled everytime the chaincode submitted as a transaction. Collection-level policies on the other hand are rules for when a chaincode reads or writes from a private collection. A chaincode can have multiple private collections, each with its own collection-level policy. When a chaincode accesses such a collection, the additional policies have to be satisfied as well. Key-level policies are similar to collection-level ones in that they only become relevant when a chaincode touches a subset of the world-state. A difference being that key-level policies are declared in the execution of a chaincode. A usecase for this is when adding a new asset to the ledger with a specific owner. By setting a key-level endorsement policy to the key of the asset, any transaction that tries to change the asset will have to be endorsed by, for example, its owner before it is commited to the ledger.

2.5.4 Chaincode

Many current blockchains have some notion of a smart contract [15], *i.e.,* methods to programatically change the ledger only when certain properties, or contracts, have been fulfilled. Smart contracts makes a blockchain more general purpose as they can be used to model many different protocols and applications. **examples plz**

In Hyperledger Fabric, smart contracts are called chaincode. As there is no inherent asset on the Fabric ledger over which peers can make transactions, chaincodes are the only way the ledger can be changed. They are in other words the only way to create, update and remove assets. Chaincodes has to follow the *fabric-contract* Application Programming Interface (API) to be run by peers on the network, but what language they can be written in is flexible. The official language options are Java, Javascript and Go, but other languages can be supported in theory. A chaincode smart contract interacts, through the fabric-contract API, with the world-state key-value representation of the ledger and can query, update and set values in it.

Running a chaincode on the network does not update the ledger directly (see 2.5.2 for how ledger modifications are implemented). Instead a read-write set of the keys that have been queried from and the changes to the ones that have been set during the execution of the chaincode are generated. The result of the chaincode is this read-write set, which is then further used to update the ledger itself.

TODO: Add chaincode example

TODO: Find some good references for chaincode besides the official documentation.

2.6 Formal verification

2.6.1 Temporal logic

2.6.2 TLA+

2.7 Related work area

Relaterande arbeten

2.8 Summary

Sammanfattning

Det är trevligt att få detta kapitel avslutas med en sammanfattning. Till exempel kan du inkludera en tabell som sammanfattar andras idéer och fördelar och nackdelar med varje - så som senare kan du jämföra din lösning till var och en av dessa. Detta kommer också att hjälpa dig att definiera de variabler som du kommer att använda för din utvärdering.

It is nice to have this chapter conclude with a summary. For example, you can include a table that summarizes other people's ideas and benefits and drawbacks with each - so as later you can compare your solution to each of them. This will also help you define the variables that you will use for your evaluation.

Chapter 3

Method or Methods

Metod eller Metodval

This chapter is about Engineering-related content, Methodologies and Methods. Use a self-explaining title.
The contents and structure of this chapter will change with your choice of methodology and methods.

Describe the engineering-related contents (preferably with models) and the research methodology and methods that are used in the degree project.
Give a theoretical description of the scientific or engineering methodology are you going to use and why have you chosen this method. What other methods did you consider and why did you reject them.
In this chapter, you describe what engineering-related and scientific skills you are going to apply, such as modeling, analyzing, developing, and evaluating engineering-related and scientific content. The choice of these methods should be appropriate for the problem . Additionally, you should be consciousness of aspects relating to society and ethics (if applicable). The choices should also reflect your goals and what you (or someone else) should be able to do as a result of your solution - which could not be done well before you started.

Vilka vetenskapliga eller ingenjörsmetodik ska du använda och varför har du valt den här metoden. Vilka andra metoder gjorde du överväga och varför du avvisar dem. Vad är dina mål? (Vad ska du kunna göra som ett resultat av din lösning - vilken inte kan göras i god tid innan du började) Vad du ska göra? Hur? Varför? Till exempel, om du har implementerat en artefakt vad gjorde du och varför? Hur kommer ditt utvärdera den. Syftet med detta kapitel är att ge en översikt över forsknings metod som används i denna avhandling. Avsnitt 3.1 beskriver forskningsprocessen. Avsnitt 3.2 detaljer forskningen paradigm. Avsnitt 3.3 fokuserar på datainsamling tekniker som används för denna forskning. Avsnitt 3.4 beskriver experimentell design. Avsnitt 3.5 förklarar de tekniker som används för att utvärdera tillförlitligheten och giltigheten av de insamlade uppgifterna. Avsnitt 3.6 beskriver den metod som används för dataanalysen. Slutligen, Avsnitt 3.7 beskriver ramverket valts för att utvärdera xxx.
Ofta kan man koppla ett antal följdfrågor till undersökningsfrågan och problemlösningen t ex
(1) Vilken process skall användas för konstruktion av lösningen och vilken process skall kopplas till denna för att svara på undersökningsfrågan?
(2) Hur och vilket resultat (storheter) skall presenteras både för att redovisa svar på undersökningsfrågan (resultatkapitlet i denna rapport) och redovisa resultat av problemlösningen (prototypen, ofta dokument som bilagor men vilka dokument och varför?).
(3) Vilken teori/teknik skall väljas och användas både för undersökningen (taxonomi, matematik, grafer, storheter mm) och problemlösning (UML, UseCases, Java mm) och varför?
(4) Vad behöver du som student leverera för att uppnå hög kvaliet (minimikrav) eller mycket hög kvalitet på examensarbetet?
(5) Frågorna kopplar till de följande underkapitlen.
(6) Resonemanget bygger på att studenter på hing-programmet ofta skall konstruera något åt problemägaren och att man till detta måste koppla en intressant ingenjörsfråga. Det finns hela tiden en dualism mellan dessa aspekter i exjobbet.

3.1 Research Process

Undersökningsrocess och utvecklingsprocess

Figur **??** visar de steg som utförs för att genomföra
Beskriv, gärna med ett aktivitetsdiagram (UML?), din undersökningsprocess och utvecklingsprocess. Du måste koppla ihop det akademiska intresset (undersökningsprocess) med ursprungsproblemet (utvecklingsprocess) denna forskning.
Aktivitetsdiagram från t ex UML-standard

3.2 Research Paradigm

Undersökningsparadigm
Exempelvis
Positivistisk (vad/hur fungerar det?) kvalitativ fallstudie med en deduktivt (förbestämd) vald ansats och ett induktivt(efterhand uppstår dataområden och data) insamlade av data och erfarenheter.

3.3 Data Collection

This should also show that you are aware of the social and ethical concerns that might be relevant to your data collection method.)

Datainsamling
(Detta bör också visa att du är medveten om de sociala och etiska frågor som kan vara relevanta för dina data insamlingsmetod.)

3.3.1 Sampling

Stickprovsundersökning

3.3.2 Sample Size

Provstorleken

3.3.3 Target Population

Målgruppen

3.4 Experimental design/Planned Measurements

Experimentdesign/Mätuppställning

3.4.1 Test environment/test bed/model

Describe everything that someone else would need to reproduce your test environment/test bed/model/… .

Testmiljö/testbädd/modell
Beskriv allt att någon annan skulle behöva återskapa din testmiljö / testbädd / modell / …

3.4.2 Hardware/Software to be used

Hårdvara / programvara som ska användas

3.5 Assessing reliability and validity of the data collected

Bedömning av validitet och reliabilitet hos använda metoder och insamlade data

3.5.1 Validity of method

How will you know if your results are valid?

Giltigheten av metoder
Har dina metoder ge dig de rätta svaren och lösning? Var metoderna korrekt?

3.5.2 Reliability of method

How will you know if your results are reliable?

Tillförlitlighet av metoder
Hur bra är dina metoder, finns det bättre metoder? Hur kan du förbättra dem?

3.5.3 Data validity

Giltigheten av uppgifter
Hur vet du om dina resultat är giltiga? Har ditt resultat mäta rätta?

3.5.4 Reliability of data

Tillförlitlighet av data
Hur vet du om dina resultat är tillförlitliga? Hur bra är dina resultat?

3.6 Planned Data Analysis

Metod för analys av data

3.6.1 Data Analysis Technique

Dataanalys Teknik

3.6.2 Software Tools

Mjukvaruverktyg

3.7 Evaluation framework

Utvärdering och ramverk
Metod för utvärdering, jämförelse mm. Kopplar till kapitel 5.

3.8 System documentation

If this is going to be a complete document consider putting it in as an appendix, then just put the highlights here.

Systemdokumentation
Med vilka dokument och hur skall en konstruerad prototyp dokumenteras? Detta blir ofta bilagor till rapporten och det som problemägaren till det ursprungliga problemet (industrin) ofta vill ha. Bland dessa bilagor återfinns ofta, och enligt någon angiven standard, kravdokument, arkitekturdokument, designdokumnet, implementationsdokument, driftsdokument, testprotokoll mm.

Chapter 4

What you did

Choose your own chapter title to describe this

[Vad gjorde du? Hur gick det till? – Välj lämplig rubrik (”Genomförande”, ”Konstruktion”, ”Utveckling” eller annat]

What have you done? How did you do it? What design decisions did you make? How did what you did help you to meet your goals?

Vad du har gjort? Hur gjorde du det? Vad designen beslut gjorde du? Hur kom det du hjälpte dig att uppnå dina mål?

4.1 Hardware/Software design …/Model/Simulation model & parameters/…

Hårdvara / Mjukvarudesign ... / modell / Simuleringsmodell och parametrar / …

Figur **??** visar en enkel ikon för en hemsida. Tiden för att få tillgång till den här sidan när serveras kommer att kvantifieras i en serie experiment. De konfigurationer som har testats i provbänk listas ini tabell **??**.
Vad du har gjort? Hur gjorde du det? Vad designen beslut gjorde du?

4.2 Implementation …/Modeling/Simulation/…

Implementering … / modellering / simulering / …

Chapter 5

Results and Analysis

svensk: Resultat och Analys

Sometimes this is split into two chapters.
Keep in mind: How you are going to evaluate what you have done?
What are your metrics?
Analysis of your data and proposed solution
Does this meet the goals which you had when you started?

In this chapter, we present the results and discuss them.

I detta kapitel presenterar vi resultatet och diskutera dem.
Ibland delas detta upp i två kapitel.
Hur du ska utvärdera vad du har gjort? Vad är din statistik?
Analys av data och föreslagen lösning
Innebär detta att uppnå de mål som du hade när du började?

5.1 Major results

Huvudsakliga resultat

Some statistics of the delay measurements are shown in Table **??**. The delay has been computed from the time the GET request is received until the response is sent.

Lite statistik av mätningarna fördröjnings visas i Tabell **??**. Förseningen har beräknats från den tidpunkt då begäran GET tas emot fram till svaret skickas.

5.2 Reliability Analysis

Analys av reabilitet
Reabilitet i metod och data

5.3 Validity Analysis

Analys av validitet
Validitet i metod och data

Chapter 6

Discussion

This can be a separate chapter or a section in the previous chapter.

Diskussion
Förbättringsförslag?

Chapter 7

Conclusions and Future work

Slutsats och framtida arbete

Add text to introduce the subsections of this chapter.

7.1 Conclusions

Describe the conclusions (reflect on the whole introduction given in Chapter 1).

Slutsatser

Discuss the positive effects and the drawbacks.
Describe the evaluation of the results of the degree project.
Did you meet your goals?
What insights have you gained?
What suggestions can you give to others working in this area?
If you had it to do again, what would you have done differently?

Träffade du dina mål?
Vilka insikter har du fått?
Vilka förslag kan du ge till andra som arbetar inom detta område? Om du hade att göra igen, vad skulle du ha gjort annorlunda?

7.2 Limitations

What did you find that limited your efforts? What are the limitations of your results?

Begränsande faktorer
Vad gjorde du som begränsade dina ansträngningar? Vilka är begränsningarna i dina resultat?

7.3 Future work

Describe valid future work that you or someone else could or should do. Consider: What you have left undone? What are the next obvious things to be done? What hints can you give to the next person who is going to follow up on your work?

Vad du har kvar ogjort?
Vad är nästa självklara saker som ska göras?
Vad tips kan du ge till nästa person som kommer att följa upp på ditt arbete?

Due to the breadth of the problem, only some of the initial goals have been met. In these section we will focus on some of the remaining issues that should be addressed in future work. ...

7.3.1 What has been left undone?

The prototype does not address the third requirment, i.e., a yearly unavailability of less than 3 minutes, this remains an open problem. ...

7.3.1.1 Cost analysis

The current prototype works, but the performance from a cost perspective makes this an impractical solution. Future work must reduce the cost of this solution, to do so a cost analysis needs to first be done. ...

7.3.1.2 Security

A future research effort is needed to address the security holes that results from using a self-signed certificate. Page filling text mass. Page filling text mass.

...

7.3.2 Next obvious things to be done

In particular, the author of this thesis wishes to point out xxxxxx remains as a problem to be solved. Solving this problem is the next thing that should be done. ...

7.4 Reflections

What are the relevant economic, social, environmental, and ethical aspects of your work?

Reflektioner

Vilka är de relevanta ekonomiska, sociala, miljömässiga och etiska aspekter av ditt arbete?

One of the most important results is the reduction in the amount of energy required to process each packet while at the same time reducing the time required to process each packet.

xxxx.

In the references, let Zotero or other tool fill this in for you. I suggest an extended version of the IEEE style, to include URLs, DOIs, ISBNs, etc., to make it easier for your reader to find them. This will make life easier for your opponents and examiner.
IEEE Editorial Style Manual: https://www.ieee.org/content/dam/ieee-org/ieee/web/org/conferences/style_references_manual.pdf

Låt Zotero eller annat verktyg fylla i det h r för dig. Jag föreslår en utökad version av IEEE stil - att inkludera webbadresser, DOI, ISBN etc. - för att göra det lättare för läsaren att hitta dem. Detta kommer att göra livet lättare för dina motståndare och examinator.

References

[1] Ken Thompson. “Reflections on trusting trust.” In: 27.8 (1984), p. 3.

[2] Chris Lamb and Stefano Zacchiroli. “Reproducible Builds: Increasing the Integrity of Software Supply Chains.” In: *IEEE Software* (2021). Conference Name: IEEE Software, pp. 0–0. ISSN: 1937-4194. DOI: 10.1109/MS.2021.3073045.

[3] *Reproducible Builds — a set of software development practices that create an independently-verifiable path from source to binary code*. URL: https://reproducible-builds.org/ (visited on 02/07/2022).

[4] *Public rebuilderd instances*. URL: https://rebuilderd.com/ (visited on 02/07/2022).

[5] Aifarez Abdui-Rahman, Stephen Hailes, and S Hailes. “A Distributed Trust Model.” In: (), p. 13.

[6] Spyridon Samonas and David Coss. “THE CIA STRIKES BACK: REDEFINING CONFIDENTIALITY, INTEGRITY AND AVAILABILITY IN SECURITY.” In: (), p. 25.

[7] Niclas Kannengießer et al. “Trade-offs between Distributed Ledger Technology Characteristics.” In: *ACM Computing Surveys* 53.2 (Mar. 31, 2021), pp. 1–37. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3379463. URL: https://dl.acm.org/doi/10.1145/3379463 (visited on 02/16/2022).

[8] Ralph C. Merkle. “Method of providing digital signatures.” U.S. pat. 4309569A. Univ Leland Stanford Junior. Jan. 5, 1982.

[9] Erik Daniel and Florian Tschorsch. “IPFS and Friends: A Qualitative Comparison of Next Generation Peer-to-Peer Data Networks.” In: *arXiv:2102.12737 [cs]* (Jan. 18, 2022). arXiv: 2102.12737. URL: http://arxiv.org/abs/2102.12737 (visited on 02/02/2022).

[10] Ben Laurie, Adam Langley, and Emilia Kasper. *Certificate Transparency*. Request for Comments RFC 6962. Num Pages: 27. Internet Engineering Task Force, June 2013. DOI: 10.17487/RFC6962. URL: https://datatracker.ietf.org/doc/rfc6962 (visited on 02/17/2022).

[11] Nazanin Zahed Benisi, Mehdi Aminian, and Bahman Javadi. "Blockchain-based decentralized storage networks: A survey." In: *Journal of Network and Computer Applications* 162 (July 2020), p. 102656. ISSN: 10848045. DOI: 10.1016/j.jnca.2020.102656. URL: https://linkinghub.elsevier.com/retrieve/pii/S1084804520301302 (visited on 02/02/2022).

[12] Michael J. Fischer. "The consensus problem in unreliable distributed systems (a brief survey)." In: *Foundations of Computation Theory*. Ed. by Marek Karpinski. Red. by G. Goos et al. Vol. 158. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 1983, pp. 127–140. ISBN: 978-3-540-12689-8 978-3-540-38682-7. DOI: 10.1007/3-540-12689-9_99. URL: http://link.springer.com/10.1007/3-540-12689-9_99 (visited on 02/17/2022).

[13] Giang-Truong Nguyen and Kyungbaek Kim. "A Survey about Consensus Algorithms Used in Blockchain." In: *Journal of Information Processing Systems* 14.1 (Feb. 28, 2018), pp. 101–128. DOI: 10.3745/JIPS.01.0024. URL: https://doi.org/10.3745/JIPS.01.0024 (visited on 02/11/2022).

[14] Satoshi Nakamoto. "Bitcoin: A Peer-to-Peer Electronic Cash System." In: (), p. 9.

[15] Massimo Di Pierro. "What Is the Blockchain?" In: *Computing in Science Engineering* 19.5 (2017). Conference Name: Computing in Science Engineering, pp. 92–95. ISSN: 1558-366X. DOI: 10.1109/MCSE.2017.3421554.

[16] Elli Androulaki et al. "Hyperledger fabric: a distributed operating system for permissioned blockchains." In: *Proceedings of the Thirteenth EuroSys Conference*. EuroSys '18: Thirteenth EuroSys Conference 2018. Porto Portugal: ACM, Apr. 23, 2018, pp. 1–15. ISBN: 978-1-4503-5584-1. DOI: 10.1145/3190508.3190538. URL: https://dl.acm.org/doi/10.1145/3190508.3190538 (visited on 02/21/2022).

[17] *Fabric Gateway — hyperledger-fabricdocs main documentation*. URL: https://hyperledger-fabric.readthedocs.io/en/release-2.4/gateway.html (visited on 02/22/2022).

[18] Diego Ongaro and John Ousterhout. "In Search of an Understandable Consensus Algorithm." In: (), p. 16.

[19] Artem Barger et al. "A Byzantine Fault-Tolerant Consensus Library for Hyperledger Fabric." In: *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. 2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC). May 2021, pp. 1–9. DOI: 10.1109/ICBC51069.2021.9461099.

Appendix A

Something Extra

svensk: Extra Material som Bilaga

A.1 Just for testing KTH colors

You have selected to optimize for digital output

- Primary color
  - kth-blue actually Deep sea
  - kth-blue80
- Secondary colors
  - kth-lightblue actually Stratosphere
  - kth-lightred actually Fluorescence
  - kth-lightred80
  - kth-lightgreen actually Front-lawn
  - kth-coolgray actually Office
  - kth-coolgray80

black

For DIVA