Technische Universität Berlin

Information Systems Engineering
Fakultät IV
Einsteinufer 17
10587 Berlin
https://www.tu.berlin/ise

Master Thesis

Design, Implementation and Evaluation of Zero Knowledge Rollups using the Zokrates Toolbox

De Tomasi Andrea

Matriculation Number: 1234567
01.01.2010

Supervised by
Prof. Dr. Stefan Tai
Prof. Dr. 2nd supervisor

Assistant Supervisor

fill with assistant supervisor

Hereby I declare that I wrote this thesis myself with the help of no more than the mentioned literature and auxiliary means.

Berlin, 01.01.2050

. . . . . .

(Signature) [your name]

Abstract

This template is intended to give an introduction of how to write diploma and master thesis at the chair 'Architektur der Vermittlungsknoten' of the Technische Universität Berlin. Please don't use the term 'Technical University' in your thesis because this is a proper name.

On the one hand this PDF should give a guidance to people who will soon start to write their thesis. The overall structure is explained by examples. On the other hand this text is provided as a collection of LaTeX files that can be used as a template for a new thesis. Feel free to edit the design.

It is highly recommended to write your thesis with LaTeX. I prefer to use Miktex in combination with TeXnicCenter (both freeware) but you can use any other LaTeX software as well. For managing the references I use the open-source tool jabref. For diagrams and graphs I tend to use MS Visio with PDF plugin. Images look much better when saved as vector images. For logos and 'external' images use JPG or PNG. In your thesis you should try to explain as much as possible with the help of images.

The abstract is the most important part of your thesis. Take your time to write it as good as possible. Abstract should have no more than one page. It is normal to rewrite the abstract again and again, so probaly you won't write the final abstract before the last week of due-date. Before submitting your thesis you should give at least the abstract, the introduction and the conclusion to a native english speaker. It is likely that almost no one will read your thesis as a whole but most people will read the abstract, the introduction and the conclusion.

Start with some introductionary lines, followed by some words why your topic is relevant and why your solution is needed concluding with 'what I have done'. Don't use too many buzzwords. The abstract may also be read by people who are not familiar with your topic.

Summary

Da die meisten Leuten an der TU deutsch als Muttersprache haben, empfiehlt es sich, das Abstract zusätzlich auch in deutsch zu schreiben. Man kann es auch nur auf deutsch schreiben und anschließend einem Englisch-Muttersprachler zur Übersetzung geben.

Contents

List of Figures xi

List of Tables xiii

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Objective . . . 1
- 1.3 Scope . . . 1
- 1.4 Outline . . . 1

2 Fundamentals and Related Work 3
- 2.1 Technologies . . . 3
  - 2.1.1 Technology A . . . 3
  - 2.1.2 Technology B . . . 3
  - 2.1.3 Comparison of Technologies . . . 4
- 2.2 Standardization . . . 4
  - 2.2.1 Internet Engineering Task Force . . . 4
  - 2.2.2 International Telecommunication Union . . . 4
  - 2.2.3 3GPP . . . 4
  - 2.2.4 Open Mobile Alliance . . . 4
- 2.3 Concurrent Approaches . . . 4

3 Requirements 5
- 3.1 Overview . . . 5
- 3.2 Technical Requirements . . . 5
  - 3.2.1 Sub-component A . . . 5
  - 3.2.2 Sub-component B . . . 5
- 3.3 Social Requirements . . . 5

4 Concept 7
- 4.1 Sub-component A . . . 7
- 4.2 Sub-component B . . . 7
- 4.3 Proposed API . . . 7
- 4.4 Layer X . . . 8
- 4.5 Interworking of X and Y . . . 8
- 4.6 Interface Specification . . . 8

5 Implementation **9**
5.1 Completing the Scale Tezos Blockchain with Zk-Rollups project 9
5.1.1 Storage 9
5.1.2 Balance and Nonce Merkle Tree 10
5.1.3 Rollup execution 10
5.1.4 Registration and Deregistration 11
5.1.5 Deregistration 11
5.2 Project Structure 11
5.3 Important Implementation Aspects 12
5.4 Graphical User Interface 12
5.5 Documentation 13

6 Evaluation **15**
6.1 Test Environment 15
6.2 Scalability 15
6.3 Usability 15
6.4 Performance Measurements 15

7 Conclusion **17**
7.1 Summary 17
7.2 Dissemination 17
7.3 Problems Encountered 17
7.4 Outlook 17

List of Acronyms **21**

Bibliography **23**

Annex **25**
talk to your supervisor if this is needed

List of Figures

1.1 Component X 2

4.1 Alice and Bob 7

5.1 Project Structure 11

List of Tables

2.1 Comparison of technologies 4

1 Introduction

This chapter should have about 4-8 pages and at least one image, describing your topic and your concept. Usually the introduction chapter is separated into subsections like 'motivation', 'objective', 'scope' and 'outline'.

1.1 Motivation

Start describing the situation as it is today or as it has been during the last years. 'Over the last few years there has been a tendency... In recent years...'. The introduction should make people aware of the problem that you are trying to solve with your concept, respectively implementation. Don't start with 'In my thesis I will implement X'.

1.2 Objective

What kind of problem do you adress? Which issues do you try to solve? What solution do you propose? What is your goal? 'This thesis describes an approach to combining X and Y... The aim of this work is to...'

1.3 Scope

Here you should describe what you will do and also what you will not do. Explain a little more specific than in the objective section. 'I will implement X on the platforms Y and Z based on technology A and B.'

Conclude this subsection with an image describing 'the big picture'. How does your solution fit into a larger environment? You may also add another image with the overall structure of your component.

'Figure 1.1 shows Component X as part of ...'

1.4 Outline

The 'structure' or 'outline' section gives a brief introduction into the main chapters of your work. Write 2-5 lines about each chapter. Usually diploma thesis are separated into 6-8 main chapters.

This example thesis is separated into 7 chapters.

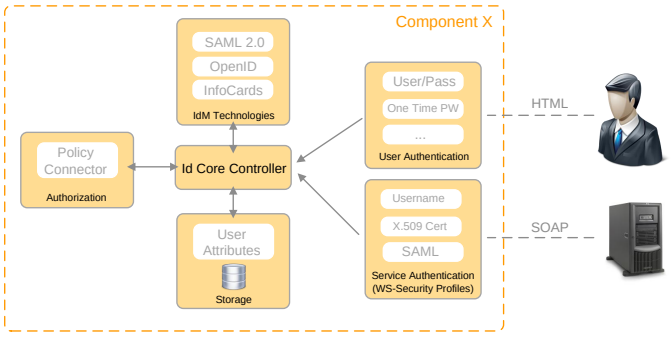

Figure 1.1: Component X

Chapter 2 is usually termed 'Related Work', 'State of the Art' or 'Fundamentals'. Here you will describe relevant technologies and standards related to your topic. What did other scientists propose regarding your topic? This chapter makes about 20-30 percent of the complete thesis.

Chapter 3 analyzes the requirements for your component. This chapter will have 5-10 pages.

Chapter 4 is usually termed 'Concept', 'Design' or 'Model'. Here you describe your approach, give a high-level description to the architectural structure and to the single components that your solution consists of. Use structured images and UML diagrams for explanation. This chapter will have a volume of 20-30 percent of your thesis.

Chapter 5 describes the implementation part of your work. Don't explain every code detail but emphasize important aspects of your implementation. This chapter will have a volume of 15-20 percent of your thesis.

Chapter 6 is usually termed 'Evaluation' or 'Validation'. How did you test it? In which environment? How does it scale? Measurements, tests, screenshots. This chapter will have a volume of 10-15 percent of your thesis.

Chapter 7 summarizes the thesis, describes the problems that occurred and gives an outlook about future work. Should have about 4-6 pages.

2 Fundamentals and Related Work

This section is intended to give an introduction about relevant terms, technologies and standards in the field of X. You do not have to explain common technologies such as HTML or XML.

2.1 Technologies

This section describes relevant technologies, starting with X followed by Y, concluding with Z.

2.1.1 Technology A

It's always a good idea to explain a technology or a system with a citation of a prominent source, such as a widely accepted technical book or a famous person or organization.

Exmple: Tim-Berners-Lee describes the "WorldWideWeb" as follows:
"The WorldWideWeb (W3) is a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents." [BL]

You can also cite different claims about the same term.
According to Bill Gates *"Windows 7 is the best operating system that has ever been released"* [Gat] (no real quote) In opposite Steve Jobs claims Leopard to be *"the one and only operating system"* [Job]

If the topic you are talking about can be grouped into different categories you can start with a classification. Example: According to Tim Berners-Lee XYZ can be classified into three different groups, depending on foobar [BL]:

- Mobile X
- Fixed X
- Combined X

2.1.2 Technology B

For internal references use the 'ref' tag of LaTeX. Technology B is similar to Technology A as described in section 2.1.1.

2.1.3 Comparison of Technologies

Name	Vendor	Release Year	Platform
A	Microsoft	2000	Windows
B	Yahoo!	2003	Windows, Mac OS
C	Apple	2005	Mac OS
D	Google	2005	Windows, Linux, Mac OS

Table 2.1: Comparison of technologies

2.2 Standardization

This sections outlines standardization approaches regarding X.

2.2.1 Internet Engineering Task Force

The IETF defines SIP as '...' [RSC+02]

2.2.2 International Telecommunication Union

Lorem Ipsum...

2.2.3 3GPP

Lorem Ipsum...

2.2.4 Open Mobile Alliance

Lorem Ipsum...

2.3 Concurrent Approaches

There are lots of people who tried to implement Component X. The most relevant are ...

3 Requirements

This section determines the requirements necessary for X. This includes the functional aspects, namely Y and Z, and the non functional aspects such as A and B.

3.1 Overview

In this chapter you will describe the requirements for your component. Try to group the requirements into subsections such as 'technical requirements', 'functional requirements', 'social requirements' or something like this. If your component consist of different partial components you can also group the requirements for the corresponding parts.

Explain the source of the requirements.

Example: The requirements for an X have been widely investigated by Organization Y.

In his paper about Z, Mister X outlines the following requirements for a Component X.

3.2 Technical Requirements

The following subsection outlines the technical requirements to Component X.

3.2.1 Sub-component A

Interoperability
Lorem Ipsum...

Scalability
Lorem Ipsum...

3.2.2 Sub-component B

Lorem Ipsum...

3.3 Social Requirements

Component X must compete with Y. Hence, it is required to provide an excellent usability. This includes ...

4 Concept

This chapter introduces the architectural design of Component X. The component consists of subcomponent A, B and C.

In the end of this chapter you should write a specification for your solution, including interfaces, protocols and parameters.

4.1 Sub-component A

The concept chapter provides a high-level explanation of your solution. Try to explain the overall structure with a picture. You can also use UML sequence diagrams for explanation.

Figure 4.1 illustrates the situation between Alice and Bob. (sequence diagram from www.websequencediagrams.com)

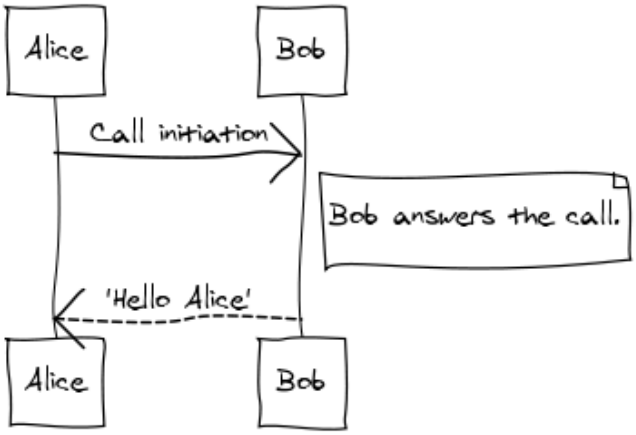

Figure 4.1: Alice and Bob

4.2 Sub-component B

Lorem Ipsum...

4.3 Proposed API

Lorem Ipsum...

4.4 Layer X

Lorem Ipsum...

4.5 Interworking of X and Y

Lorem Ipsum...

4.6 Interface Specification

Lorem Ipsum...

5 Implementation

This chapter describes the implementation of component X. Three systems were chosen as reference implementations: a desktop version for Windows and Linux PCs, a Windows Mobile version for Pocket PCs and a mobile version based on Android.

5.1 Completing the Scale Tezos Blockchain with Zk-Rollups project

This section explains the implementation of the remaining features of the *Scale Tezos Blockchain using Zk-Rollups* project. Since some other already-implemented features have been modified, this section also explains the changes made to those features to integrate the new changes made

5.1.1 Storage

Originally the storage has been designed to hold a limited amount of users, thus using a normal Map. It was composed by:

- A Map of accounts indexed by a natural number containing:
  - A public key;
  - A balance.
- A list of 8 scalar values representing the Merkle Tree root of the accounts public keys;
- A list of 8 scalar values representing the Merkle Tree root of the accounts balances.

Having as objective the system to be scalable the storage has been modified to support a growing number of users using a Big Map. This is a key difference because the Big Map is lazily deserialized[1], not having to use gas during a contract call to deserialize all the accounts Map. The storage is now composed by:

- A Big Map of accounts indexed by a natural number containing:
  - A public key;
  - A balance;
  - A nonce.

[1] `https://tezos.gitlab.io/michelson-reference/#type-big_map`

- A list of 8 scalar values representing the Merkle Tree root of the accounts public keys;
- A list of 8 scalar values representing the Merkle Tree root of the accounts balances concatenated to their respective nonces.

The nonce mechanism and the Merkle Tree roots are explained in the section 5.1.2.

5.1.2 Balance and Nonce Merkle Tree

To avoid the possible attack of having an user sending repeatedly the same transaction the nonce mechanism is used: the nonce is a propriety stored in the account Big Map, saved on Layer 1. Every time an user wants to submit a transaction the new transaction must include a nonce incremented by one. This allows to check if the transaction has been already executed.

To prove inside the Zokrates environment that we're using the updated version of balances and nonces a merkle tree of the lists must be computed. This merkle tree should be computed for both the balances and nonces lists passed as parameters. This adds complexity because we have to compute two full binary trees. The workaround to this problem is to concatenate and hash the balances and nonces for each user and compute a single merkle tree of them. The concatenation of balances and nonces is done by hashing the concatenation of the balances and nonces themselves with sha256. This returns a 32 bytes hash that is used as the input of the merkle tree leaf for an user. This approach adds only the complexity of the concatenation function that performs a number of hashes equal to the number of accounts, but still having proven that the balances and nonces are updated.

5.1.3 Rollup execution

The rollup execution has been changed to include the calculation of the new nonces of the users when calculating the new balances. A precondition that must be respected is that Zokrates receives the transaction ordered by nonce for a user. It can receive transactions from different users but the transactions of a single user must be ordered by nonce. The algorithm that checks the transaction nonces and calculates the new nonces is the following:

- Create a new array X copying the old nonce array;
- Iterate over the transactions:
-
  - Check in X if the transaction nonce for the sender is equal to the nonce of the user added one;
  - If it is, increment the nonce of the user in the array X and continue;
  - If it is not, fail execution.

This allows to receive multiple transactions from the same user while avoiding a duplicate transaction to be applied.

No relevant overhead is added to the complexity of the rollup execution because the nonce check is done in the same loop of the balance calculation and accessing directly to the array, it is still O(n).

5.1.4 Registration and Deregistration

This section describes the registration and deregistration process of a user. The main idea is to insert and remove users from the storage, leaving empty fake users in case of a deregistration. This is made in order to preserve the position of the users inside the storage.

To register a new user, two Merkle Trees must be recalculated with the new user's public key, balance and nonce. This computation is done inside a Zokrates program that returns the new root hashes of the Merkle Trees. The new root hashes are then passed to the contract that updates the storage with the new root hashes. The Zokrates program expects the exact position where to put the new data of the new user. This position is calculated from the external manager that can contact an rpc and compute the first empty slot inside the Big Map of the storage. The register process checks if at the specified position the account has an empty key, balance and nonce set to 0. After the initial checks it sets the nonce at the value of 1, the balance at the value of 0 and the public key at the value of the public key of the user.

5.1.5 Deregistration

The deregistration process is similar to the registration process described at 5.1.4. The

5.2 Project Structure

The implementation is separated into 2 distinguished eclipse projects as depicted in figure 5.1.

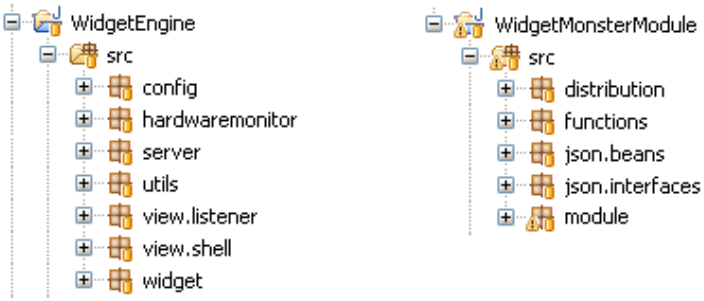

Figure 5.1: Project Structure

The following listing briefly describes the single packages of both projects in alphabetical order to give an overview of the implementation:

config
Lorem Ipsum...

server
Lorem Ipsum...

utils
Lorem Ipsum...

5.3 Important Implementation Aspects

Do not explain every class in detail. Give a short introduction about the modules or the eclipse projects. If you want to explain relevant code snippets use the 'lstlisting' tag of LaTeX. Put only short snippets into your thesis. Long listing should be part of the annex.

Listing 5.1: JSON String Code Snippet

```
{
        id: 1,
        method: "myInstance.getGroup",
        params: ["Teammates", 2, true]
}

{
        id: 2,
        result: [
                  "groupDesc":"These are my teammates",
                  {
                        "javaClass":"src.package.MemberClass",
                        "memberName": "Bob",
                  }
                ]
}
```

You can also compare different approaches. Example: Since the implementation based on X failed I choosed to implement the same aspect based on Y. The new approach resulted in a much faster ...

5.4 Graphical User Interface

Lorem Ipsum...

5.5 Documentation

Lorem Ipsum...

6 Evaluation

In this chapter the implementation of Component X is evaluated. An example instance was created for every service. The following chapter validates the component implemented in the previous chapter against the requirements.

Put some screenshots in this section! Map the requirements with your proposed solution. Compare it with related work. Why is your solution better than a concurrent approach from another organization?

6.1 Test Environment

Fraunhofer Institute FOKUS' Open IMS Playground was used as a test environment for the telecommunication services. The IMS Playground ...

6.2 Scalability

Lorem Ipsum

6.3 Usability

Lorem Ipsum

6.4 Performance Measurements

Lorem Ipsum

7 Conclusion

The final chapter summarizes the thesis. The first subsection outlines the main ideas behind Component X and recapitulates the work steps. Issues that remained unsolved are then described. Finally the potential of the proposed solution and future work is surveyed in an outlook.

7.1 Summary

Explain what you did during the last 6 month on 1 or 2 pages!

The work done can be summarized into the following work steps

- Analysis of available technologies
- Selection of 3 relevant services for implementation
- Design and implementation of X on Windows
- Design and implementation of X on mobile devices
- Documentation based on X
- Evaluation of the proposed solution

7.2 Dissemination

Who uses your component or who will use it? Industry projects, EU projects, open source...? Is it integrated into a larger environment? Did you publish any papers?

7.3 Problems Encountered

Summarize the main problems. How did you solve them? Why didn't you solve them?

7.4 Outlook

Future work will enhance Component X with new services and features that can be used ...

talk to your supervisor if this is needed

List of Acronyms

3GPP	3rd Generation Partnership Project
AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
AS	Application Server
CSCF	Call Session Control Function
CSS	Cascading Stylesheets
DHTML	Dynamic HTML
DOM	Document Object Model
FOKUS	Fraunhofer Institut fuer offene Kommunikationssysteme
GUI	Graphical User Interface
GPS	Global Positioning System
GSM	Global System for Mobile Communication
HTML	Hypertext Markup Language
HSS	Home Subscriber Server
HTTP	Hypertext Transfer Protocol
I-CSCF	Interrogating-Call Session Control Function
IETF	Internet Engineering Task Force
IM	Instant Messaging
IMS	IP Multimedia Subsystem
IP	Internet Protocol
J2ME	Java Micro Edition
JDK	Java Developer Kit
JRE	Java Runtime Environment
JSON	JavaScript Object Notation
JSR	Java Specification Request
JVM	Java Virtual Machine
NGN	Next Generation Network
OMA	Open Mobile Alliance
P-CSCF	Proxy-Call Session Control Function
PDA	Personal Digital Assistant
PEEM	Policy Evaluation, Enforcement and Management
QoS	Quality of Service
S-CSCF	Serving-Call Session Control Function
SDK	Software Developer Kit
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SMS	Short Message Service

SMSC	Short Message Service Center
SOAP	Simple Object Access Protocol
SWF	Shockwave Flash
SWT	Standard Widget Toolkit
TCP	Transmission Control Protocol
Telco API	Telecommunication API
TLS	Transport Layer Security
UMTS	Universal Mobile Telecommunication System
URI	Uniform Resource Identifier
VoIP	Voice over Internet Protocol
W3C	World Wide Web Consortium
WSDL	Web Service Description Language
XCAP	XML Configuration Access Protocol
XDMS	XML Document Management Server
XML	Extensible Markup Language

Bibliography

[BL] Berners-Lee, Tim: *WWW Book (no real book, just an example).*

[Gat] Gates, Bill: *no real citate.*

[Job] Jobs, Steve: *no real citate.*

[Joh03] Johnston, Alan B.: *SIP, understanding the Session Initiation Protocol, Second Edition.* Artech House Publishers, 2003. ISBN: 1580536557.

[RSC+02] Rosenberg, J., H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley and E. Schooler: *SIP: Session Initiation Protocol.* RFC 3261 (Proposed Standard), June 2002. Updated by RFCs 3265, 3853, 4320, 4916, 5393.

Annex

```
<?xml version="1.0" encoding="UTF-8"?>
<widget>
         <debug>off</debug>
         <window name="myWindow" title="Hello Widget" visible="true">
                 <height>120</height>
                 <width>320</width>
                 <image src="Resources/orangebg.png">
                        <name>orangebg</name>
                        <hOffset>0</hOffset>
                        <vOffset>0</vOffset>
                </image>
                 <text>
                         <name>myText</name>
                         <data>Hello Widget</data>
                         <color>#000000</color>
                         <size>20</size>
                         <vOffset>50</vOffset>
                         <hOffset>120</hOffset>
                 </text>
        </window>
</widget>
```

Listing 1: Sourcecode Listing

```
INVITE sip:bob@network.org SIP/2.0
Via: SIP/2.0/UDP 100.101.102.103:5060;branch=z9hG4bKmp17a
Max-Forwards: 70
To: Bob <sip:bob@network.org>
From: Alice <sip:alice@ims-network.org>;tag=42
Call-ID: 10@100.101.102.103
CSeq: 1 INVITE
Subject: How are you?
Contact: <sip:xyz@network.org>
Content-Type: application/sdp
Content-Length: 159
v=0
o=alice 2890844526 2890844526 IN IP4 100.101.102.103
s=Phone Call
t=0 0
c=IN IP4 100.101.102.103
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

SIP/2.0 200 OK
Via: SIP/2.0/UDP proxy.network.org:5060;branch=z9hG4bK83842.1
;received=100.101.102.105
Via: SIP/2.0/UDP 100.101.102.103:5060;branch=z9hG4bKmp17a
To: Bob <sip:bob@network.org>;tag=314159
From: Alice <sip:alice@network.org>;tag=42
Call-ID: 10@100.101.102.103
CSeq: 1 INVITE
Contact: <sip:foo@network.org>
Content-Type: application/sdp
Content-Length: 159
v=0
o=bob 2890844526 2890844526 IN IP4 200.201.202.203
s=Phone Call
c=IN IP4 200.201.202.203
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Listing 2: SIP request and response packet[Joh03]