Självständigt arbete på grundnivå

Independent degree project - first cycle

Datateknik

Computer Engineering

Hashing algorithms
A comparison for blockchains in Internet of things

Karl Dahlin
Abstract

In today’s society blockchains and the Internet of things are two very discussed subjects this has led to thoughts about combining them by using a blockchain in Internet of things. This objective of this study has been to solve the problem which hashing algorithms is the best for a blockchain used in an Internet of things network. This have been done by first selecting a few hashing algorithms and setting up a scenario where a blockchain can be used in an Internet of things network. After that I specified what to compare, speed, input length and output length. The study has been conducted with the aid of literary studies about the hashing algorithms and a program that implements the algorithms and tests their speed. The study has shown that out of the selected hashing algorithms MD5, SHA-256, SHA3-256 with the conditions specified for the study that the hashing algorithm SHA3-256 is the best option if speed is not of the utmost importance in the scenario since it is from a newer standard and do not have a max input length. If speed is the very important in other words if SHA3-256 is to slow then SHA-256 would be best for the Internet of things network.

Keywords: Internet of things, Blockchain, Hash, Hashing algorithms, MD5, SHA, C++, Matlab.
# Table of Contents

Abstract ........................................................................................................................................ iii

Table of Contents ............................................................................................................................ iv

1 Introduction ................................................................................................................................... 1
  1.1 Background and problem motivation ....................................................................................... 1
  1.2 Overall aim ............................................................................................................................... 2
  1.3 Concrete and verifiable goals .................................................................................................. 2
  1.4 Scope ....................................................................................................................................... 3
  1.5 Outline .................................................................................................................................... 3

2 Theory .......................................................................................................................................... 4
  2.1 Internet of things ..................................................................................................................... 4
  2.2 Blockchains ............................................................................................................................ 4
  2.3 Hash ........................................................................................................................................ 4
  2.4 Hashing algorithms .................................................................................................................. 5
  2.5 Related work ............................................................................................................................ 5

3 Methodology ................................................................................................................................ 7

4 Choice of solution ....................................................................................................................... 9
  4.1 Hashing algorithms .................................................................................................................. 9
  4.2 Scenarios ................................................................................................................................ 10
  4.3 Chosen solution ....................................................................................................................... 11

5 Implementation .......................................................................................................................... 12
  5.1 Hardware ................................................................................................................................ 12
  5.2 Hashing algorithm ................................................................................................................... 13
  5.3 Measurement setup .................................................................................................................. 13
  5.4 Presentation of data ................................................................................................................ 13

6 Results ......................................................................................................................................... 15
  6.1 Features ................................................................................................................................... 15
  6.2 Results from tests ..................................................................................................................... 16
  6.3 Analysis of the result ............................................................................................................... 21

7 Conclusions ................................................................................................................................ 22
  7.1 Ethical discussion ..................................................................................................................... 23
  7.2 Future work ............................................................................................................................... 24

References ....................................................................................................................................... 25

Appendix A: Source Code ............................................................................................................... 27
1 Introduction

Today we continually develop new technology, this is especially true concerning data handling and computer science. This development has led to a lot of new concepts, ideas and technologies. Two of these are the Internet of things and blockchains, these two things have become very popular with the general development of technology in the everyday life[1] and the rise of cryptocurrencies[2] .

The Internet of things is a concept that states that everything, such as household appliances, accessories and vehicles that have embedded electronics should be connected to a larger net so that they can exchange data. The Internet of things have become a very popular subject in recent years and due to that there is a lot of work and research being done in the field of Internet of things.

A blockchain is an immutable ledger since once data or a so called block have been added to the chain it cannot be changed. The blocks are generated by using different methods on the data that changes it in to a predetermined length, the changed data is stored and then the previous blocks changed data is added to the newly changed data and that creates the block. Since it includes the previous blocks changed data it is chained together and that is the reason for the name blockchain, a block can also contain additional information such as a timestamp. Probably the most know use of a blockchain today is the cryptocurrency Bitcoin. Bitcoin uses a blockchain just because it is an immutable ledger, the Bitcoin blockchain is a public blockchain and that means that pretty much anybody can access the blockchain.

1.1 Background and problem motivation

In recent years the blockchain technology have had a spike in popularity and a lot of people have been working with blockchains and using them. Due to blockchains rise in popularity a lot of hashing algorithms that can be used for blockchains have been created.

With the increase in popularity of both Internet of things and blockchains it has become interesting to implement blockchain technology in Internet of things. But due to the sometimes limited
computing power of an Internet of things component, all hashing algorithms might not be usable in such a blockchain.

In the survey of the algorithms the computing power required and response time plays an important part because the purpose is to compare hashing algorithms for blockchains that will be used by Internet of things devices and they may not have the same computing power that a regular PC or laptop have.

1.2 Overall aim

The study’s overall aim is to decide the best hashing algorithms for blockchains in an Internet of things scenario from a preselected number of algorithms. To decide which algorithm is the best a comparison needs to be completed. The comparison is to be done by taking a couple of things in to account. The benefits and drawbacks, the difference in hashing method and how it effects response time, effectiveness and how demanding the algorithm is, in other words can you run it on an Internet of things device.

Therefore, the problem I will aim to solve in this thesis is to compare and decide which hashing algorithms is the best for a blockchain used in an Internet of things network.

1.3 Concrete and verifiable goals

The survey’s objective is the following:

1. Determine three hashing algorithms to compare.

2. Setup different scenarios where a blockchain is usable in Internet of things.

3. Decide how to implement the algorithms and specify what to measure and compare.

4. Implement the algorithms and the method of comparison.

5. Run the program and collect information.

6. Analyze the gathered information from the comparison and draw a conclusion based on what has been decided during goal number three.
1.4 **Scope**

The study has its focus on hashing algorithms for blockchains used in an Internet of things scenario and focus mainly on the time aspect of the algorithms. I will only check a couple of hashing algorithms, not all the existing algorithms. A thing I will not test in this study is the security of the algorithms.

The survey’s conclusion should however be generally valid for most types of scenarios where blockchains are used since the hashing algorithm is run and compared on multiple kinds of hardware.

1.5 **Outline**

In chapter 2 previously done work that is relevant to the project and other relevant theory will be mentioned. Chapter 3 describes the method that was used during the project. Chapter 4 presents possible algorithms and scenarios for the study and describes which of those are chosen for this study. Chapter 5 describes how the tests of the study was implemented. In chapter 6 the results of the study are presented. In chapter 7 the conclusions regarding the study are drawn, the ethical aspects are discussed and suggestions of future work are presented.
2 Theory

In this chapter information about subjects relevant to the study will be presented and explained briefly.

2.1 Internet of things

The Internet of things [3] (IoT) is the concept that everything in our daily life that have an on and off switch should be connected and able to communicate with each other with minimal human intervention. This includes everything from cellphones, coffee makers, headphones and lamps.


2.2 Blockchains

There are two different kinds of blockchains public and private. A public blockchain is open for everyone and to access a private one you need to be granted access. In each block of the blockchain the previous blocks hash is included.

A blockchain is like ledger, take Bitcoins blockchain for example it is a public blockchain that is a ledger of all Bitcoin transactions and it is growing constantly since miners keep adding new blocks to it to record new transactions. Each user has a copy of the blockchain, it is downloaded automatically when a user joins the network. The blockchain then has all the transactions from the first to the most recent one.[5] Before a block is added to the blockchain it must be verified with a thing called proof-of-work which requires each miner to verify each transaction and block addition. Proof-of-work works by setting a condition for the hashed value and if the condition is not met the block will not be added so the users machine does a small change and try again. Bitcoins proof-of-work requires the blocks hash to be less than a set target.[6]

2.3 Hash

Hash is an abstraction of data to a different representation it is don’t by using a hashing algorithm or a hash function. Hash can be used for multiple things. For example if you use a cryptographic hashing
algorithm you can use it to analyze if two blocks are identical by comparing their hash.[7] Or it can be used as it is used in blockchains where every block have the previous blocks hash in it.

2.4 Hashing algorithms

A hashing algorithm is an algorithm that takes data of varying length and converts it to a hash of a specific length, this can be done in different ways and using different operations. There are different kinds of hashing algorithms for example multiplicative and cryptographic. When hashing algorithms is mentioned in this study it means the cryptographic kind.

A cryptographic hashing algorithm have some basic security requirements. They should be collision resistant, second preimage resistant and preimage resistant. Collision resistant means that it should be hard to find two messages $M_1$ and $M_2$ where $M_1$ and $M_2$ are different such that $\text{hash}(M_1)$ gives the same hash as $\text{hash}(M_2)$. Second preimage resistant means that for a given message $M_1$ it should be hard to find a message $M_2$ where $M_1$ and $M_2$ are different such that $\text{hash}(M_1)$ gives the same hash as $\text{hash}(M_2)$. Preimage resistant means that for a given hash value $H$ it should be hard to find a message $M$ such that $\text{hash}(M)$ is the same as $H$.[8] Some examples of operations used in cryptographic hashing algorithms are OR, AND, XOR and NOT.[9]

2.5 Related work

The study “A Comparative Analysis of SHA and MD5 Algorithm” by Gupta, Piyush, and Sandeep Kumar is a study that compares MD5, SHA-1, SHA-256, SHA-384 and SHA-512. It presents things like pseudocode for MD5 and SHA, security aspects, similarities, time taken for each algorithm and what hash different strings hashed to with each of the algorithms. The study’s conclusion is that SHA is more secure then MD5 but that MD5 is faster on 32 bit machines.

The study differs from what I have done since it goes more in depth on what the hashing algorithms do, the similarities and how secure the hashing algorithms are. Another difference is that it does not include any SHA3 algorithm.[10]

The study “Security Evaluation of SHA-224, SHA-512/224, and SHA-512/256.” By Dobruićć, Christoph, Maria Eichlseder, and Florian Mendel are a study that compares the security aspects of three of the
truncated variants in SHA2. It explains security goals and three different kinds of attacks on hashing algorithms, generic attacks, Preimage attacks and collision attacks. Then they draw conclusions about how to best attack each algorithm with each kind of attack. [8]

The similarities between this study and mine is that both compare hashing algorithms and have algorithms from SHA2. The difference is that this study only focuses on the security aspect, only focuses of the truncated variants of SHA2 and it describes the best way to attack the different algorithms and not just present a pure fact result as my study does.
3 Methodology

The work I will do during the study is literary studies of different internet sources, coding to implement and compare the hashing algorithms, evaluations of the gathered data and finally I will draw a conclusion from the gathered data.

To solve the first goal which is “Determine three hashing algorithms to compare.” I will conduct a literary study on the internet on a couple of different hashing algorithms by using search services like Google and Google scholar searching with search phrases like “Hashing algorithm” and “Hashing algorithms for blockchains”. Then from that data I will narrow it down further to three different algorithms that I will compare in this study.

To solve the second goal which is “Setup different scenarios where a blockchain is usable in Internet of things.” I will conduct a literary study of currently existing Internet of things networks by using search services like Google and Google scholar searching with search phrases like “Internet of things and blockchains” and from that try to come up with a couple of scenarios where a blockchain will be of use in Internet of things and pick one of them to use in this study.

To solve the third goal which is “Decide how to implement the algorithms and specify what to measure and compare.” I will try to come up with the best way to implement the algorithms on different types of hardware by checking already existing code and libraries for the selected algorithms. I will also specify what is relevant and measurable between the algorithms and how best to measure it by conducting a literature study on what is measurable in the hashing algorithms.

To solve the fourth goal which is “Implement the algorithm and the method of comparison.” I will follow the steps setup during the third goal by using the text editor Visual Studio Code to write the code.

To solve the fifth goal which is ”Run the comparison and collect information.” I will run the code I created when I completed the fourth goal on a couple of different hardware’s like a laptop and Raspberry pi and then save the data and display it in a readable way.
To solve the sixth goal which is “Analyze the gathered information from the comparison and draw a conclusion based on what has been decided during goal number three.” I will analyze the gathered data from the fifth goal in an objective way and draw a conclusion about which hashing algorithm is the best for the chosen scenario from the second goal based on the goal set up during the solving of the third goal.
4 Choice of solution

There are a lot of different existing hashing algorithms and scenarios where a blockchain might be usable for an Internet of things network in the following chapter I will mention a couple of hashing algorithms and scenarios and then chose what to focus on in this study.

4.1 Hashing algorithms

The hashing algorithms in this study were designed to be cryptographic hashing algorithms.

MD5 stands for “Message Digest 5” and is a hashing algorithm that takes an input of arbitrary length and returns a 128-bit hash using the operators AND, OR, XOR, NOT, left rotation and modulo-$2^{32}$ addition. MD5 is derived from MD4 as an improvement by Ronald Rivest 1992.[11] In 1996 research was published that described a weakness in MD5, further research in 2004 demonstrated the practical ability to generate collision and in 2005 the ability to generate colliding x.509 certificates were demonstrated. And in 2008 Carnegie Mellon University Software Engineering Institute declared it “cryptographically broken and unsuitable for further use”.[12]

SHA1 stands for “Secure hashing algorithm 1” is a hashing algorithm that takes an input less then $2^{64}$ and returns a 160-bit hash using the operators AND, OR, XOR, NOT, left shift [13] and modulo-$2^{32}$ addition. [14] SHA1 is based on similar principle as MD4.[13] In 2017 Google published an article in which they showed a practical example of a generated collision using SHA1 [15] which resulted in SHA1 officially becoming a broken hashing algorithm.

SHA2 stands for “Secure hashing algorithm 2” is a group of hashing functions which contains two main algorithms and four truncated variations, one variation on one of them and three on the other. The main algorithms are SHA-256 and SHA-512 and the truncated variations are SHA-224, SHA-384, SHA-512/224 and SHA-512/256, where SHA-224 are the truncated variant of SHA-256 and the other three are truncated variants of SHA-512.[8] The output size in bits for each of the algorithms are the number their name end on so SHA-256 have a 256-bit output and
SHA-512/224 have a 224-bit output. The operators used are AND, OR, XOR, NOT, rotate right, shift right and modulo-2^x addition, where x for SHA-256 and SHA-224 is 32 and for the rest it is 64. The input size of the algorithms is for SHA-256 and SHA-224 is less than 2^{64} and for the rest it is less than 2^{128}.

SHA3 stands for “Secure hashing algorithm 3” and around 2004 there were a lot of attacks on hashing algorithms and questions about the security of SHA1 and SHA2 were raised. [16] This led to a competition in the community to develop a new hashing standard. There were some requirements on the new standard two of these were the output and tunable parameters. The output should be able to be 224, 256, 284 and 512 bits and tunable parameters means that you can exchange performance for speed. The competition started in 2008 and in 2012 it ended with keccak winning[16] and in doing so became the base for SHA3. There are six SHA3 functions SHA3-224, SHA3-256, SHA3-384, SHA3-512 and two extendable-output function SHAKE128 and SHAKE256. The SHA3 functions can be implemented as alternatives to SHA2. The input size of these algorithms is unlimited.[17] The operators used in SHA3 are AND, XOR, Rot and Not.

4.2 Scenarios

There are a lot of scenarios where a blockchain can be used in an Internet of things network here are some examples of scenarios.

An example of a scenario where a blockchain could be used in Internet of things is that it could be used to create a common ledger that links together multiple vending machines and thus making it easier to get an overall picture of what sells and maybe even how much there still are in each vending machine.

Another Example of a scenario where a blockchain could be used in Internet of things is for a kind of log when it comes to shipping and transportation so you just add whatever you send to the blockchain and due to that it will be very easy to follow if and where a packet is lost since a blockchain is immutable. This blockchain could also be made public so people could follow their order in the same way they can today but it would probably require someone to create a user friendly user interface for it.
Another Example for a scenario is for a factory to want to add every produced item to a corresponding blockchain. The conditions for a scenario like this would be very different depending on what kind of factory is used in the scenario and what it produces and with what speed.

4.3 **Chosen solution**

One of the hashing algorithms I have chosen to include in this study is the hashing algorithm MD5. The reason I chose MD5 despite it being broken is because it is often used as a comparison when discuss or compare hashing algorithms so I can be used as a reference. Another of the hashing algorithms I have chosen to compare in this study is the hashing algorithm SHA-256 and the reason for that is because it is used in one of the most well-known blockchains there is today, the Bitcoin blockchain. The last hashing algorithm I have chosen to study and compare is the hashing algorithm SHA3-256 and the reason for this is to try to decide what and how big the difference in performance between the newer family of SHA the SHA3 family and the older SHA family SHA2.

The scenario I chose was the scenario with the vending machines and due to that I will hash a combination of numbers and letters during my tests since that is most likely what a vending machine would have to hash if it were to implement a blockchain system.
5 Implementation

Figure 1 shows the planned overall design of the study where a program is run on a device. The program sends text to a hashing algorithm, the program will also take the time that the hashing algorithm requires to hash the text by saving a start time right before the hashing algorithm starts hashing the text then setting a stop time right after it is finished hashing the text, then it will calculate the time required by the hashing algorithm to hash the text then save that value to a file. Then repeat the time measurement and hashing until the desired number of time measurements are gathered. The values are read with Matlab and with the help of Matlab graphs showing the result are made for complete source code see appendix A.

![Figure 1: Overview of the measurement setup for time measurement.](image)

5.1 Hardware

The measurement tests will be run on different hardware to give a better overview of how well the algorithms run on hardware with different performance. The test will be done on a laptop and a Raspberry Pi.

The laptop that I will run the test on is an Asus from the X554L series, it is a X554LJ-XO587H with an Intel Core i5-5200U up to 2.7 GHz and the operating system Windows 10.

The Raspberry Pi that I will run is a Raspberry pi 1 with an ARMv6-compatible processor rev 7 (v61) and the operating system Raspbian.

Both the laptop and the Raspberry Pi uses the c++ compiler called g++ to compile the code so that it can be run.
5.2 Hashing algorithm

The hashing algorithms will be used in a program I will write myself and will be used in the following way I will create a function that sends a text string to the hashing algorithm where the text string is hashed with the hashing algorithm.

The hashing algorithms is taken from an external C++ library [18] that is just the files in a folder so no additional installation of anything is needed this is to enable and make it easier to move the entire program to different hardware.

5.3 Measurement setup

The time will be measured by running a program that runs the hashing algorithm and times it. The time will be taken from when the hashing algorithm has received the data until it has been hashed and that time will be saved to a file and this will be repeated until a set amount of measurement have been taken.

To take the time I will use the “high_resolution_clock” and its functions from the library chrono in C++ and by using it save the time in an appropriate and readable way.

Output text will be taken from the program by simply printing out the hashed string and from here I can also confirm the output length.

5.4 Presentation of data

To present the gathered data I will write a Matlab script that reads the data from a file and calculates the mean and standard deviation using the build-in Matlab functions “mean” and “std”. The script will also plot the values using the built-in plot functions in Matlab such as “plot” and “bar” to plot a line plot over all the measured time values and a bar plot of the standard deviations and means of the algorithms for each kind of hardware. The script will also a statistical method to remove outliers that occurred due to different reasons not related to the hashing algorithm using Matlabs “isoutlier” function and after removing the outliers it will plot the same graphs again but then with a more accurate result.

The Matlab functions “mean” and “std” simply calculated the mean and standard deviation and the function “isoutlier” checks if a value is an outlier by using Median Absolute Deviation (MAD) and if it is an outlier
it returns true also known as a 1 and if it is not an outlier it returns false also known as a 0.
6 Results

In this chapter the features of the written program is explained and the result from the tests are presented. There is also a brief discussion about the results.

6.1 Features

To be able to measure the time each hashing algorithm takes I wrote two programs in C++ with a slight variation between them. The program works such as you select from four functions and each of the first three functions does the time measurement a different amount of times the first does is 10 000 000 times, the second 1 000 000 times and the third 100 000 times. You also need to select which hashing algorithms you want to run and then the program saves it to a file with a corresponding name for example “MD5_result_100000.txt” or “SHA256_result_1000000”. The fourth function is a print function which prints the plain text and the different hashes from the functions. If you want to change what is hashed or printed that can be done by changing the source code and there are a couple of example strings commented away in there that can easily be changed, in figure 2 you can see how this looks in the source code.

![Figure 2: Screenshot from code demonstrating the strings.](image)

The only difference between the two programs is that one saves the time in nanoseconds and one in microseconds this is because of the difference in time required by the laptop and the Raspberry Pi.

Figure 3 shows how the terminal looks after the program has been run twice first it just prints the plain text and what each of the functions hashed it to and after that it shows how it looks when the program is run 100 000 times with the hashing algorithm SHA-3-256
Other than the program I wrote a Matlab script that reads from files, calculates the mean and standard deviation, plots a line plot that shows all the time points and then plot a bar plot with the mean as bars and the standard deviation is shown as a so called errorbar to indicate the most common variation of the values. After that the Matlab script checks the gathered data for outliers and removes them then it plots the graphs anew with a more accurate result since the outliers have been removed.

The Matlab script is written for 100 000 values but can easily be modified so it fits the other sizes the program can produce this can be done by changing the name of the file to the filename corresponding to the number of values you want to run it on. A note on that is that every variable name and title on the figures will still say 100 000 if you do not change them aswell.

### 6.2 Results from tests

Figure 4 shows the mean and standard deviation of the time measurements for each of the hashing algorithms from the test run on the laptop displayed as a bar plot with bars indicating the standard deviation, with one bar per hashing algorithm.
Figure 4: Bar plot of mean and standard deviation from laptop test.

Figure 5 shows the mean and standard deviation of the time measurements for each of the hashing algorithms from the test run on the Raspberry pi displayed as a bar plot with bars indicating the standard deviation, with one bar per hashing algorithm.

Figure 5: Bar plot of mean and standard deviation from Raspberry pi test.

Figure 6 shows each individual time measurement displayed in a line plot for each of the hashing algorithms run on the laptop. There is one line plot per hashing algorithm.
Figure 6: Line plot time for each iteration of time measurement from laptop test.

Figure 7 shows each individual time measurement displayed in a line plot for each of the hashing algorithms run on the Raspberry pi. There is one line plot per hashing algorithm.

Figure 7: Line plot time for each iteration of time measurement from Raspberry pi test.

Figure 8 shows the mean and standard deviation of the laptop test as a bar plot after outliers have been removed, there is one bar per hashing algorithm.
Figure 8: Bar plot of mean and standard deviation from laptop test after the outliers have been removed.

Figure 9 shows the mean and standard deviation of the Raspberry pi test as a bar plot after outliers have been removed, there is one bar per hashing algorithm.

Figure 10 shows each individual time measurement displayed in a line plot for each of the hashing algorithms run on the laptop after the outliers have been removed. There is one line plot per hashing algorithm.
Figure 10: Line plot of laptop tests after outliers have been removed.

Figure 11 shows each individual time measurement displayed in a line plot for each of the hashing algorithms run on the Raspberry pi after the outliers have been removed. There is one line plot per hashing algorithm.

Figure 11: Line plot of Raspberry pi tests after outliers have been removed.

Figure 12 is a screenshot that shows most the variables from the workspace in Matlab after the script has been run in it values like standard deviations, means and how many time point are left after removing the outliers.
6.3 Analysis of the result

There were a lot of outliers as could be seen on figure 6 and figure 7, they really raised the standard deviation and some outliers were quite significantly bigger and this was probably due to that something else was running at the same time on the hardware but I have not proved that it was due to this but they have been determined as outliers using Matlab.

One thing that one can see by analyzing the data is that Matlab’s built-in function for outlier detection might not be optimal when it comes to the data from the laptop tests see figure 10 since it resulted in a straight line plot, but more tests are required to be sure.
7 Conclusions

Firstly I solved the goal “Determine three hashing algorithms to compare.” and I did so by doing a literary study using Google scholar with search words and phrases like “Hashing algorithm” and “Hashing algorithms for blockchains”. I decided to use the following hashing algorithms MD5, SHA-256 and SHA3-256 for a more detailed explanation see 4.3.

Secondly I solved the goal “Setup different scenarios where a blockchain is usable in Internet of things.” And I did so by conducting a literary study using Google scholar and the regular Google search engine. The scenarios I came up with where using it for vending machines, in transportation business to create a log and ... . These scenarios are explained in more detail in the chapter 4.2.

Thirdly I solved the goal “Decide how to implement the algorithms and specify what to measure and compare.” And I did so by searching the internet for C++ library with the algorithms that I decided upon in chapter 4.3. So I decided to use an C++ library that were files in a folder that did not require any additional installation. And I decided to compare the time, input and output.

Fourthly I solved the goal “Implement the algorithms and the method of comparison.” And I did so by writing a program in C++ that hashes a string using the different hashing algorithms the program can also print the hashed string as well as take time as described in the beginning of chapter 6.

Fifthly I solved the goal “Run the program and collect information.” And I solved this by running the program with the function for 100 000 time measurements with all three hashing algorithms which resulted in three result files with 100 000 values in each of them.

Sixthly I solved the goal “Analyze the gathered information from the comparison and draw a conclusion based on what has been decided during goal number three.” And I did this by looking at the graphs made by my Matlab script and comparing the graphs and thinking about what is the most important according to my scenario and from there I drew the conclusion that SHA3-256 is the best hashing algorithm
in my scenario due to the fact that in my scenario the difference in speed between SHA-256 and SHA3-256 is so small it can be ignored and due to that even if SHA-256’s max message size if very large you do not know what will happen in the future so you would want to avoid as many limitations as possible and because SHA3 where developed to have more security then SHA2. since MD5 were included in the study as a reference it was not even considered even if it was so much faster than the other algorithms.

And finally I came to a conclusion concerning the main problem of this study “compare and decide which hashing algorithms is the best for a blockchain used in an Internet of things network.” And since SHA3-256 is not a lot slower then SHA-256, is a from a newer standard that is supposed to be more secure and have less limitations the SHA-256 I drew the conclusion that SHA3-256 is the best option to use for a blockchain in an Internet of things network. An exception to this is if SHA3-256 is too slow for the Internet of things network where the blockchain shall be used then SHA-256 is a better choice since it is still viable.

7.1 Ethnical discussion

There is not much of an ethical aspect of this study since it is mostly focused on the time and evaluating what algorithm is the best for a blockchain in a certain scenario. But the ethical aspect of two subjects associated with this study are huge, the subjects being blockchains and hashing algorithms.

One part to discuss about blockchains is the use of proof of work since just the three biggest cryptocurrencies using proof of work consumes about the same amount of electricity as Kuwait.[19]

One part of the discussion about hashing algorithms is should we use these hashing algorithms. We think they are secure but we cannot be sure since there is a small possibility that someone in the world can knows a way to break the algorithms but just have not made the method for breaking them public. Another part is, is it reasonable to use hashing algorithms to set up a system now that might break when hardware with more computing power is available.
7.2 Future work

There are a lot of possibilities for future work on this study.

An example of future work would be to try to get measurements without outliers by changing things. One change that might help is to change the priority of the program by using the “nice” command on the Raspberry Pi and by using the task manager on the laptop.

Another example of future work could include more conditions in the study and look on more parts of the algorithms. In other words, expand on what is compared for example if and how much the message length effect the speed of the algorithms or how secure the algorithms are against attacks, in other words the algorithms security.

Another example of future work is to compare a wider variety of hashing algorithms to cover most of the existing ones, that would give a bigger picture since this study only focused on three algorithms.

Another example of future work is to implement the hashing algorithms in an Internet of things network with a blockchain and test if it works and measure how well the algorithms works in an actual scenario since there are a lot more variables when you look at an entire Internet of things network.
References


Published sep 2017, Retrieved 2018-06-21


https://www.usenix.org/legacy/events/hotos03/tech/full_papers/henson/henson.pdf


Hashing algorithms – A comparison for blockchains in Internet of things
Karl Dahlin

Retrived 2018-06-01

[12] Chad R Dougherty, “Vulnerability Note VU#836068 - MD5 vulnerable to collision attacks”
https://www.kb.cert.org/vuls/id/836068
Published 2009-01-21, Retrived 2018-06-03

https://www.rfc-editor.org/info/rfc3174

Published august 2015

https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html
Published 2017-02-23, Retrived 2018-06-04


[18] Stephan Brumme, “Portable C++ hashing library”
http://create.stephan-brumme.com/hash-library/
Published 2014-02-05, updated 2014-06-14

[19] Bigglet, “Proof-of-Work is draining the grid”
https://blog.xtrabytes.global/technology/proof-of-work/
Published 2018-01-14, Retreived 2018-06-05
Appendix A: Source Code

https://drive.google.com/drive/folders/1nt-bP9JX8-UVNAmdBsSBM-wOyBNOvayL?usp=sharing