940 Blanshard Street
Victoria, British Columbia
V8W 3E6
Susan Fiddler
Co-op Coordinator
Faculty of Engineering
University of Victoria
P.O. Box 1700
Victoria, B.C.
V8W 2Y2

Dear Susan,

Please accept the accompanying Work Term Report entitled "Determining uses of JIRA and Confluence at OFI IBM."

This report is the result of work completed at the SOME GENEriC ORGANIZATION. During my first work
term as a University of Victoria student, I used charts and tables to display information about issue, complied documentation for critical applications in a wiki and researched add-ons to extend functionality. In the course of work, I gained exposure to a technical environment, and learned how software can integrate together.

Through the course of the term, I was given the opportunity to learn much agile software development, testing applications, and software products. I feel that this knowledge will be helpful in future work terms, and in my career.

I would like to thank my manager, MISTER MAN, for his patience and good judgement, as well as the RANDOM FOLK who were always willing to help.
Sincerely,

David Li

Table of Contents

List of Figures

List of Tables

Listings

Summary

Glossary

Dapp Decentralized Application have backend code running on a decentralized peer-to-peer network and not controlled by a single entity. 1

smart contract computer program that directly controls transfer of digital currencies or assets under predefined conditions and used to automatic transactions on the blockchain. These transactions are trackable and irreversible. 1

Table 1 Timeline of Cryptocurrency

2009	Bitcoin
1968	Xerox Palo Alto Research Centre envisage the ’Dynabook
1971	Busicom ’Handy-LE’ Calculator
1973	First mobile handset invented by Martin Cooper
1978	Parker Bros. Merlin Computer Toy
1981	Osborne 1 Portable Computer
1982	Grid Compass 1100 Clamshell Laptop
1983	TRS-80 Model 100 Portable PC
1984	Psion Organiser Handheld Computer
1991	Psion Series 3 Minicomputer

1. Introduction

1.1. History of Cryptocurrency

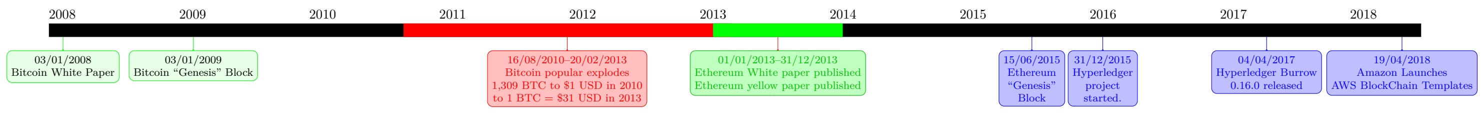

Figure 1: An example of server-blockchain architecture in a DAPP.

In 2008 bitcoin white paper [2] described a way to solve the double spending problem without a centralized body using blockchain. Released in July 30, 2015, Ethereum, an open-source platform based on blockchain technology, distinguishes itself from bitcoin through faster transactions, unlimited processing capability for smart contracts, and its network is optimized to supports Decentralized Application(DApp) [1].

1.2. Decentralized Applications

A blockchain is a digitized, decentralized, public ledger of all cryptocurrency transactions. To access websites on the Ethereum blockchain and use dapps a specialized browser is needed, or a plugin like metamask. As shown in Figure 2 dapp [1], a user's transactions on the application is publicly broadcasting to the blockchain. Implementing architecture for blockchain applications [2]adds an third layer to the standard client-server architecture, however, through the use of interfaces such as the JSON RPC and/or cloud hosting services [3]databases can be query publicly available data on the blockchain.

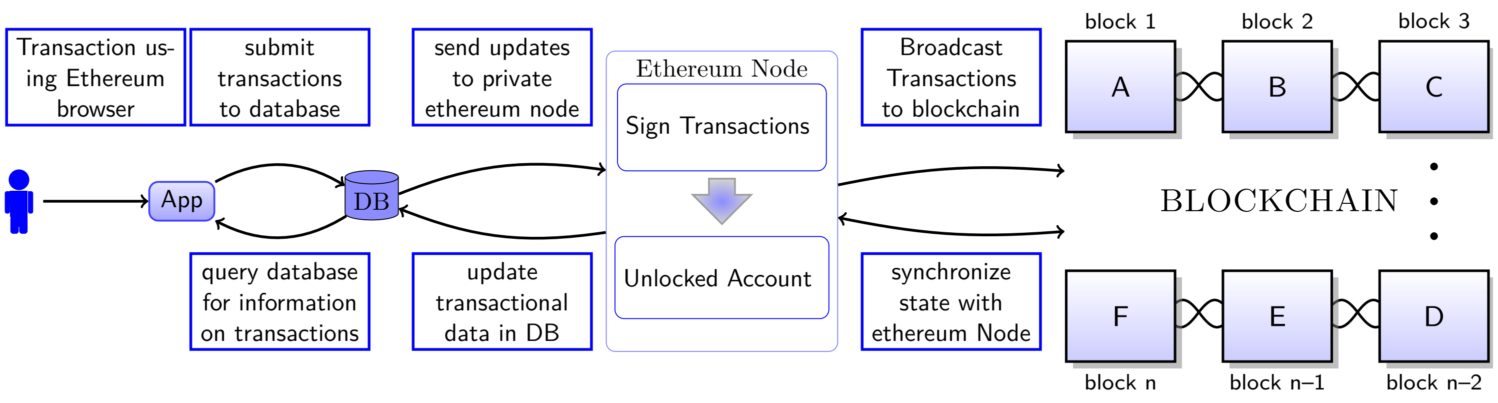
Transaction using Ethereum browser
submit transactions to database
send updates to private ethereum node
Ethereum Node
Sign Transactions
Unlocked Account
Broadcast Transactions to blockchain
block 1
block 2
block 3
A
B
C
App
DB
BLOCKCHAIN
query database for information on transactions
update transactional data in DB
synchronize state with ethereum Node
F
E
D
block n
block n–1
block n–2

Figure 2: An example of server-blockchain architecture in a DAPP.

Public and Private Keys **In a blockchain** system, any key holder can use their private key to sign a piece of data. This results in a signature. In a Dapp, this can be used for: Recovering the public key (ethereum account address) of the Author Verify if the raw data is the same as the one signed by Author using the public key. Verify whether the message is the same as the one signed by Author

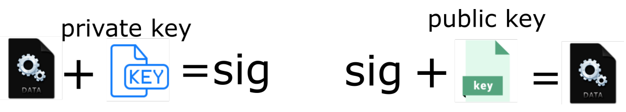

Figure 3: Illustrate how public and private keys are used to verify signatures

[1]Although, server-blockchain architecture with an abstraction layer resemble traditional applications, other approaches are available such as offline signing with a public node, and client-blockchain in serverless apps [1] and leveraging cloud infrastructure.
[2]Amazon recently started offering blockchain on AWS. [1]
[3]**Signing Transactions**: usning the the ethereum node, use its JSON RPC interface from the application to perform all blockchain operations.

1.3. Smart Contracts

2. Discussion

3. Conclusion

4. Recommendations

5. References

Cited References

[1] *Ethereum White Paper*. [Online] Available: https://github.com/ethereum/wiki/wiki/White-Paper. Accessed April 25, 2018.

[2] *Bitcoin White Paper*. [Online] Available: https://bitcoin.org/bitcoin.pdf. Accessed April 25, 2018.

General References

[1] *Full Stack Hello World Voting Ethereum Dapp TutorialPart 1*. URL: https://medium.com/@mvmurthy/full-stack-hello-world-voting-ethereum-dapp-tutorial-part-1-40d2d0d807c2.

[2] mjhm. *Hello World Dapp*. https://github.com/mjhm/hello_world_dapp. 2018.

A. Code Listings

Listing 1: Caesar Cipher for CSC 111

```
/*
 * Author:    David Li
 * UVicID:    V00818631
 * Date:      Oct 69, 2014
 * Lecture:   Assignment 7
 * File name: V00818631A7P3.c
 * Description: Reading, writing, and encoding files
 */

#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
```

```
#include <ctype.h>
#include <string.h>
#include <time.h>

#define INPUTFILENAME ("A7_2014_P3_TestingSherlock.txt")
#define OUTPUTFILENAME ("A7_2014_P3_SherlockEncoded.txt")
#define MAXSWAP (4)



void processFile(FILE* ifp, FILE* ofp, int seed) {
   char line[25];
   char word[25];
   printf("Begin process file\n");
    // your code goes here
   int rn1,rn2;
   int j;
   char tmp;

   while(!feof(ifp)) {

      fscanf(ifp,"%s",line);
      strcpy(word,line);

      if(strlen(line) > 3) {
         for(j=0; j<MAXSWAP; j++) {
            rn1 = rand() % (strlen(line) -2);
            rn2 = rand() % (strlen(line) -2);
            tmp = line[rn1 + 1];
            line[rn1 + 1] = line[rn2 + 1];
            line[rn2 + 1] = tmp;
         }
         if(strcmp(word,line) == 0) {
            tmp = line[1];
            line[1] = line[2];
            line[2] = tmp;
         }
      }

      if(!feof(ifp))
         fprintf(ofp,"%s ",line);

   }
   printf("End process file\n");
} /* ProcessFile */
```

```c
int main(void) {
   printf("Welcome to Sherlock Holmes\n\n");
   unsigned int seed = (unsigned int)time(NULL);
   srand(seed);
   FILE *ifp;
   FILE *ofp;
   ifp = fopen(INPUTFILENAME, "r");
   if (ifp == NULL) {
      printf("Cannot open input file %s\n", INPUTFILENAME);
      exit(EXIT_FAILURE);
   } /*if*/
   ofp = fopen(OUTPUTFILENAME, "w");
   if (ofp == NULL) {
      printf("Cannot create output file %s\n", OUTPUTFILENAME);
      exit(EXIT_FAILURE);
   } /*if*/
   processFile(ifp, ofp,seed);
   fclose(ofp);
   fclose(ifp);
   printf("\nWe encoded Sherlock Holmes\n");
   return EXIT_SUCCESS;
} /*main*/
```