La confiance dans les Blockchain passe par une cryptographie adaptée

Jean-Jacques Quisquater
jjq@uclouvain.be --- jjq@mit.edu

UCL – Louvain-la-Neuve
MIT-CSAIL
Académie Royale de Belgique

Confiance …

- Confiance basée sur la *cryptographie*,
- *Protocoles* (cryptographiques) prouvés,
- *Langages* corrects, bridés, avec des balises d'alerte, pour décrire les contrats.

28th Chaos Communication Congress

Behind Enemy Lines

The future of cryptology: which 3 letters algorithm(s) could be our Titanic?

RMS Olympic, RMS Titanic, HMHS Britannic vs Discrete Logarithm, Integer factorization, Conjectured hard problems

The lessons and best practices of the titanic will be extracted. Are we ready?

This will be a co-presentation (Jean-Jacques Quisquater / David Samyde) and occasional friendly exchange, with point and counter-point of different contrasting views on the impact of solving integer factorization and some other difficult problem in cryptography.

The idea is to perform a provocative comparison between the 'unbreakable' RSA algorithm and the unsinkable Titanic.

Receiving his RSA Conference Lifetime Achievement Award, Rivest said that it has not been demonstrated mathematically that factorization into primes is difficult. So "Factoring could turn out to be easy," and according to him "maybe someone here will find the method".

Since 1994 and Shor's algorithm, the danger of quantum computer is known: breaking RSA in polynomial time. Factoring large numbers is conjectured to be computationally infeasible on classic non quantum computers. No efficient algorithm is known and the research in the last 30 years did not show enormous progress.

Iceberg existence is predicted but not shown yet.

SPEAKERS

- Jean-Jacques Quisquater
- Renaud Devaliere

SCHEDULE

Day	Day 3 - 2011-12-29
Room	Saal 2
Start time	20:30
Duration	01:00

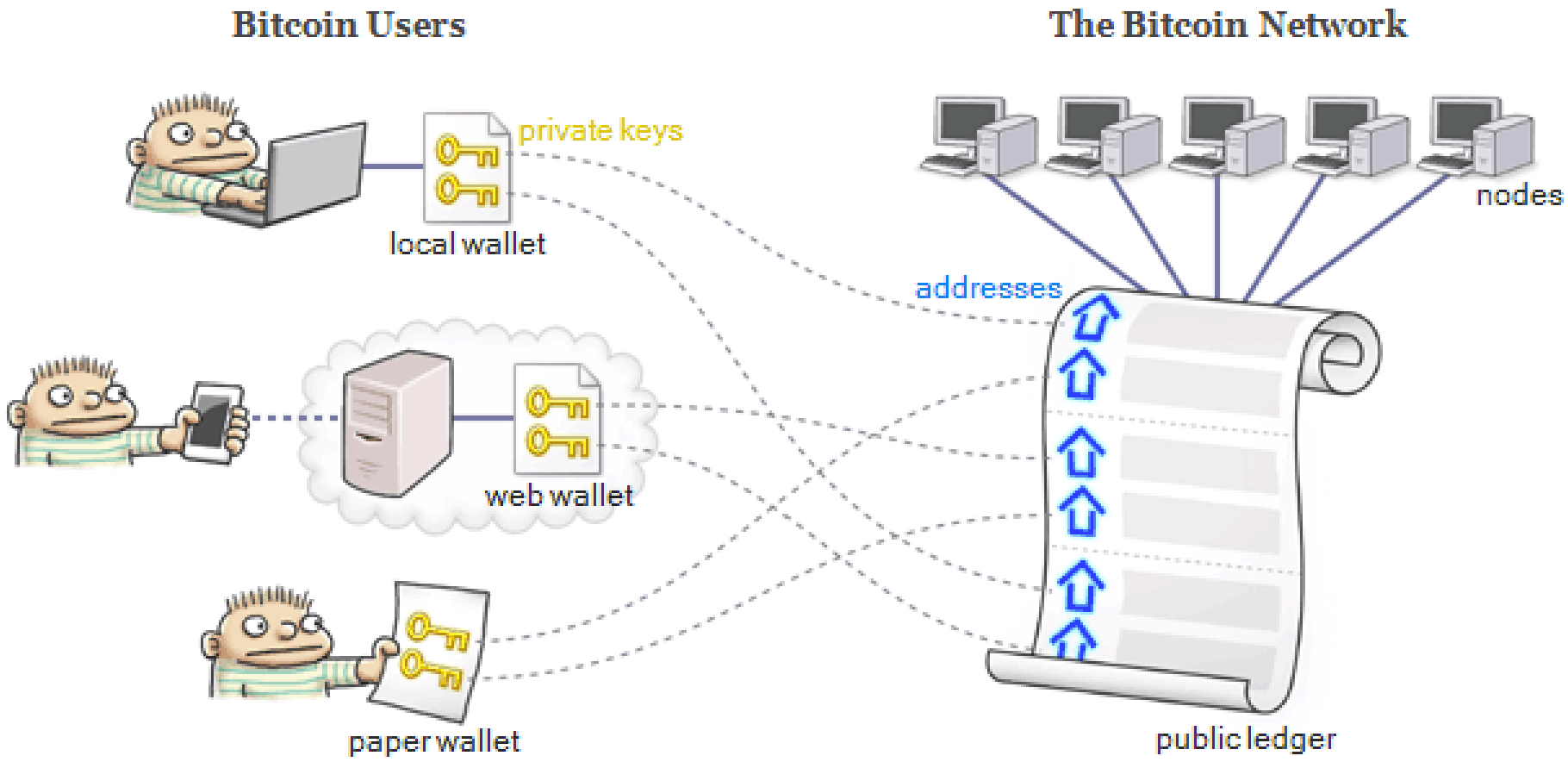
INFO

ID	4710
Event type	Lecture
Track	Hacking
Language used for presentation	English

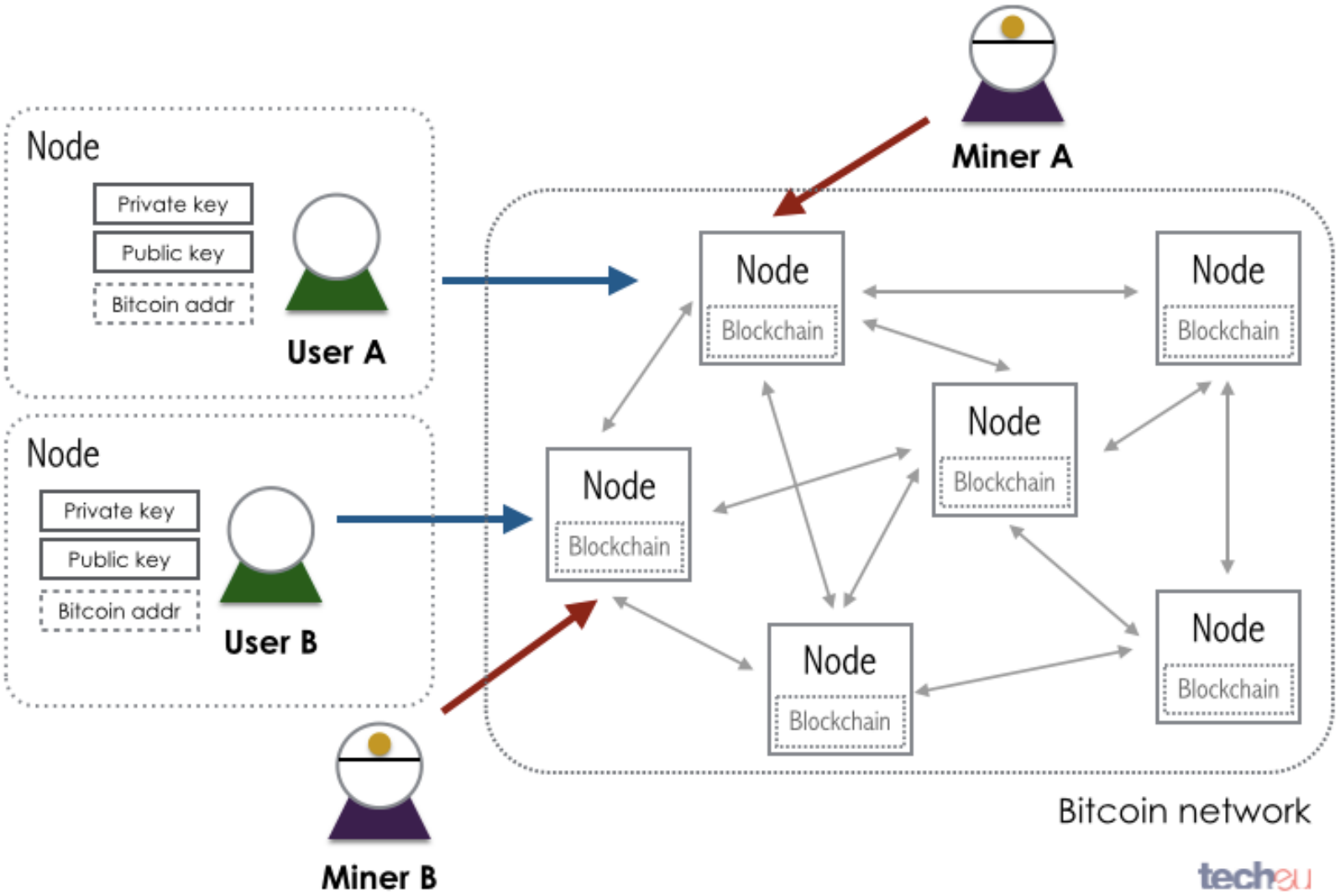
FEEDBACK

Did you attend this event? Give Feedback

Public ledger et blockchain



Blockchain pour bitcoin



Introduction

- In 2001, the Bank of Japan published a very interesting report about timestamping and trusted third-party.
- Analysis and conclusions were oriented towards distributed systems
- For sure the author(s) read it.

IMES DISCUSSION PAPER SERIES

The Security Evaluation of Time Stamping Schemes: The Present Situation and Studies

Masashi UNE

Discussion Paper No. 2001-E-18

IMES

INSTITUTE FOR MONETARY AND ECONOMIC STUDIES

BANK OF JAPAN

C.P.O BOX 203 TOKYO
100-8630 JAPAN

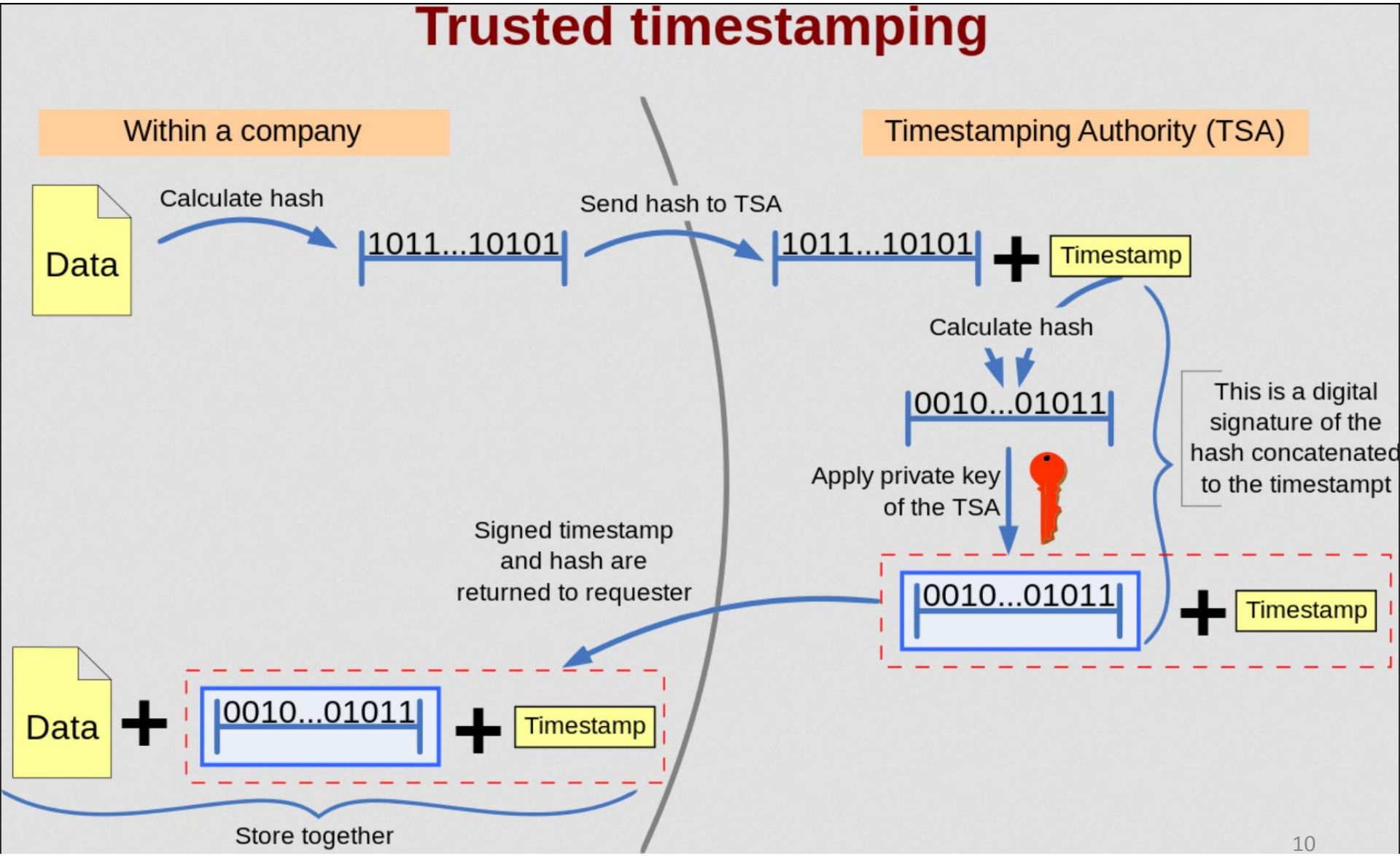
Timestamping

- A trusted digital timestamp gives you strong “legal” evidence that the contents of your work existed at a **point-in-time** and have not changed since that time. The procedures maintain complete privacy of your documents themselves.

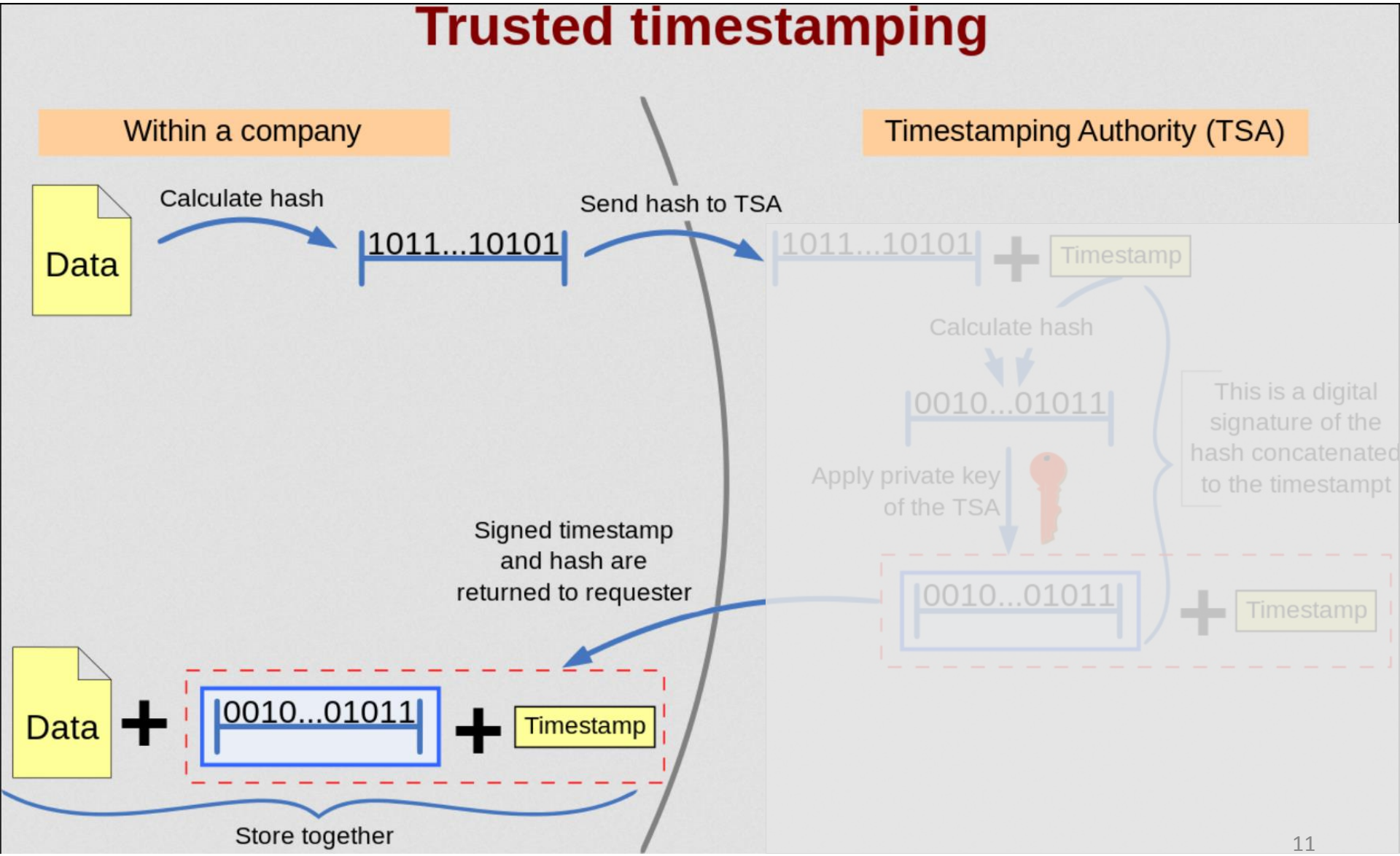
Timestamping

- Stuart Haber with Stornetta, 1990,
- surety.com, still exists,
- No success,
- why?
- starlab (interstrust): in 2000,

Trusted timestamping



Trusted timestamping



Timestamping authority: TSA

- Need of trust,
- Centralized,
- Isolated timestamp: no chaining,
- Other solutions?

Timestamping and notarisation

- This report of 2001 compares 7 systems, including mine, TIMESEC

- identifies 3 schemes:
  - Simple,
  - Chained,
  - Distributed.

completely and therefore find it hard to manipulate a time stamp. However, just as in the case of the linking scheme, a distributed scheme system is more complicated than a simple scheme. For example, the time signature distributed system (Takura et al. [1998]) belongs to this category, and Ansper et al. [2001] proposed a scheme possessing the characteristics of both the linking and distributed schemes. The main strengths and limitations of these schemes are summarized in Table 1.

Table 1 Main Strengths and Limitations of Three Schemes

Schemes	Strengths	Limitations
simple scheme	The system is relatively simple.	It is necessary to assume that the issuer is the trusted third party.
linking scheme	The assumption that the issuer is the trusted third party is rendered unnecessary, for example, by the periodical publication of a part of a chain of time stamps.	The system is relatively complicated because additional operations for linking all time stamps are needed.
distributed scheme	The assumption that the issuers are the trusted third parties is rendered unnecessary by sharing the secret data among multiple issuers.	The system is relatively complicated because multiple issuers generate a time stamp cooperatively.

The classification described above is also adopted in standardization activities relating to a time stamping service. A working draft of ISO/IEC 18014 (Time stamping services, ISO/IEC [2001]) includes the following two types of scheme: "mechanisms producing independent tokens" and "mechanisms producing linked tokens." These correspond to the simple and linking schemes, respectively. On the other hand, ISO/IEC 13888 (Non-repudiation, ISO/IEC [1997]) and IETF PKIX

Who are these authors?

- Une, Masashi, and Tsutomu Matsumoto,

- The « author » of bitcoin is:
  - Satoshi Nakamoto,

Who are these authors?

- Une, Masa**shi**, and Tsutomu Matsu**moto**,
- The author of bitcoin is:
  - Sato**shi** Naka**moto**,
- Curious coincidence !
- Better, **Satoshi** is a reference:
  - Takura, Akira, Satoshi Ono and Shozo Naito

Who is Masashi Une?

- Majored in experimental economy!
- research subjects: cryptography linked to financial services,
- Related to cryptographic systems of distributed chaining and trusted!

Masashi Une's Home Page

- I am Masashi Une, an invited research scientist of RCIS of AIST.
  RCIS: Research Center for Information Security
  AIST: National Institute of Advanced Industrial Science and Technology
- The address of RCIS is as follows:
  Akihabara Daibiru Room 1102, 1-18-13 Sotokanda, Chiyoda-ku, Tokyo 101-0021, Japan
- My email address is masashi-une (``at'' mark) aist Dot go Dot jp.

My History

- March 1994. I graduated from Socio-Economic Plannning, University of Tsukuba. (B.S.)
  - I majored in Economics, especially, Experimental Economics.
- April 1994. I joined Bank of Japan.
- September 1996. I started to follow the research trend of information security relevant to the financial sector at Institute for Monetary and Economic Studies (IMES) of Bank of Japan. My survey papers (mainly written in Japanese) are available at Paper List of CITECS (Center for Information TEChnology Studies).
- March 2003. I graduated from Graduate School of Engineering, Yokohama National University. (Ph.D.)
  - The title of my doctoral thesis is ``A study on Security Evaluation of Time Stamping Schemes.''
- May 2006. I started a research on information security techniques at RCIS of AIST.

My Research Interest and Selected Papers

I have much interest in cryptography and information security related to financial services. Especially, I have been studying how to evaluate the security of biometric authentication systems. My recent results are as follows:

- Masashi Une and Yuko Tamura, Liveness Detection in Biometric Authentication Systems, Kinyu Kenkyu 24 (S-2), Institute for Monetary and Economic Studies, Bank of Japan, 2005. (in Japanese)
- Naohiko Watanabe, Rie Shigetomi, Masashi Une, Akira Otsuka, and Hideki Imai, Universal Wolves in a Matching Algorithm with Finger Vein Patterns, Proceedings of CSS 2006, Oct. 2006, pp.621-626. (in Japanese)
- Masashi Une, Akira Otsuka, and Hideki Imai, Wolf Attack Probability: A New Security Measure in Biometrics-Based Authentication Systems, Proceedings of SCIS 2007, Jan. 2007. (in Japanese)
- Rie Kawakami, Rie Shigetomi, Kazuki Yoshizoe, Masashi Une, Akira Otsuka, and Hideki Imai, A Theoretical Study on Wolves in a Minutiae Matching Algorithm, Proceedings of SCIS 2007, Jan. 2007. (in Japanese)
- Naohiko Watanabe, Rie Shigetomi, Kazuki Yoshizoe, Masashi Une, Akira Otsuka, and Hideki Imai, Universal Wolves in a Matching Algorithm with Finger Vein Patterns -A Study on the Feature Extraction Process- Proceedings of SCIS 2007, Jan. 2007. (in Japanese)

Bitcoin: paper

Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

1. Introduction

Commerce on the Internet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust based model. Completely non-reversible transactions are not really possible, since financial institutions cannot avoid mediating disputes. The cost of mediation increases transaction costs, limiting the minimum practical transaction size and cutting off the possibility for small casual transactions, and there is a broader cost in the loss of ability to make non-reversible payments for non-reversible services. With the possibility of reversal, the need for trust spreads. Merchants must be wary of their customers, hassling them for more information than they would otherwise need. A certain percentage of fraud is accepted as unavoidable. These costs and payment uncertainties can be avoided in person by using physical currency, but no mechanism exists to make payments over a communications channel without a trusted party.

What is needed is an electronic payment system based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third party. Transactions that are computationally impractical to reverse would protect sellers from fraud, and routine escrow mechanisms could easily be implemented to protect buyers. In this paper, we propose a solution to the double-spending problem using a peer-to-peer distributed timestamp server to generate computational proof of the chronological order of transactions. The system is secure as long as honest nodes collectively control more CPU power than any cooperating group of attacker nodes.

references:

References

[1] W. Dai, "b-money," http://www.weidai.com/bmoney.txt, 1998.

[2] H. Massias, X.S. Avila, and J.-J. Quisquater, "Design of a secure timestamping service with minimal trust requirements," In *20th Symposium on Information Theory in the Benelux*, May 1999.

[3] S. Haber, W.S. Stornetta, "How to time-stamp a digital document," In *Journal of Cryptology*, vol 3, no 2, pages 99-111, 1991.

[4] D. Bayer, S. Haber, W.S. Stornetta, "Improving the efficiency and reliability of digital time-stamping," In *Sequences II: Methods in Communication, Security and Computer Science*, pages 329-334, 1993.

[5] S. Haber, W.S. Stornetta, "Secure names for bit-strings," In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 28-35, April 1997.

[6] A. Back, "Hashcash - a denial of service counter-measure," http://www.hashcash.org/papers/hashcash.pdf, 2002.

[7] R.C. Merkle, "Protocols for public key cryptosystems," In *Proc. 1980 Symposium on Security and Privacy*, IEEE Computer Society, pages 122-133, April 1980.

[8] W. Feller, "An introduction to probability theory and its applications," 1957.

references and tools:

References

[1] W. Dai, "b-money," http://www.weidai.com/bmoney.txt, 1998.

[2] H. Massias, X.S. Avila, and J.-J. Quisquater, "Design of a secure timestamping service with minimal trust requirements," In *20th Symposium on Information Theory in the Benelux*, May 1999.

[3] S. Haber, W.S. Stornetta, "How to time-stamp a digital document," In *Journal of Cryptology*, vol 3, no 2, pages 99-111, 1991.

[4] D. Bayer, S. Haber, W.S. Stornetta, "Improving the efficiency and reliability of digital time-stamping," In *Sequences II: Methods in Communication, Security and Computer Science*, pages 329-334, 1993.

[5] S. Haber, W.S. Stornetta, "Secure names for bit-strings," In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 28-35, April 1997.

[6] A. Back, "Hashcash - a denial of service counter-measure," http://www.hashcash.org/papers/hashcash.pdf, 2002.

[7] R.C. Merkle, "Protocols for public key cryptosystems," In *Proc. 1980 Symposium on Security and Privacy*, IEEE Computer Society, pages 122-133, April 1980.

[8] W. Feller, "An introduction to probability theory and its applications," 1957.

Timestamping

Proof of work

Merkle

Bitcoin: software

- The cryptographic primitives were OK for the year 2009 (publication),
- The paper gives remarks about possible evolution of cryptographic security, and many events were happening till today,
- See at the end of this talk …

Basic concepts

- Timestamping,
- Trusted third-party,

- Fonction de hachage,
- Block,
- Chain,
- Merkle tree,
- Digital signature,
- Public verification of signature,

- Distributed trusted third-party,
- Anonymity, tracability,
- Proof of work.

http://eprint.iacr.org/2016/167

On Bitcoin Security in the Presence of Broken Crypto Primitives

February 19, 2016

Ilias Giechaskiel
University of Oxford
Oxford, United Kingdom
ilias.giechaskiel@cs.ox.ac.uk

Cas Cremers
University of Oxford
Oxford, United Kingdom
cas.cremers@cs.ox.ac.uk

Kasper B. Rasmussen
University of Oxford
Oxford, United Kingdom
kasper.rasmussen@cs.ox.ac.uk

Abstract

Digital currencies like Bitcoin rely on cryptographic primitives to operate. However, past experience shows that cryptographic primitives do not last forever: increased computational power and advanced cryptanalysis cause primitives to break frequently, and motivate the development of new ones. It is therefore crucial for maintaining trust in a crypto currency to anticipate such breakage.

We present the first systematic analysis of the effect of broken primitives on Bitcoin. We identify the core cryptographic building blocks and analyze the various ways in which they can break, and the subsequent effect on the main Bitcoin security guarantees. Our analysis reveals a wide range of possible effects depending on the primitive and type of breakage, ranging from minor privacy violations to a complete breakdown of the currency.

Our results lead to several observations on, and suggestions for, the Bitcoin migration plans in case of broken cryptographic primitives.

not fully explained. Moreover, the subsequent steps after a contingency are hand-wavy and incomplete, e.g., "once the plans themselves are well-accepted, code implementing the plans can be written and tested in case the code is ever required" [11]. To the best of our knowledge, no adequate mechanism has been built into Bitcoin, and no plans for partial breakage (or weakening of a primitive) have been considered.

In practice, the situation is not black-and-white. Instead of abruptly breaking completely, cryptographic primitives usually break gradually. With hash functions, for example, it is common that first a single collision is found. This is then later generalized to multiple collisions, and only later do arbitrary collisions become feasible to compute. In parallel, the complexity of attacks (such as collisions) decreases to less-than-brute-force, and computational power increases. Finally, quantum computing will make some attacks easier, e.g., Grover's pre-image attack [23], or Shor's algorithm for discrete log computation [45].

Even if such attacks are years away from being practical, it is crucial to anticipate the impact of broken primi-

Cryptographic breakages

Breakage	Effect
SHA256	
Collisions	Steal coins
Second pre-image	Double spend
Pre-image	Complete failure
Bounded pre-image	All of the above
RIPEMD160	
Any of the above	Repudiate payments
ECDSA	
Selective forgery	Steal coins, Send fake alerts
Integrity break	Claim payment not received
Repudiation	Send fake alerts

Table 4: Effects of concrete primitive breakage on the current version of Bitcoin.

	Signature Property		
Hash Property	**Selective forgery**	**Integrity break**	**Repudiation**
Address Hash (H_A)			
Collision	Repudiate transaction	-	Change existing payment†
Second pre-image	Steal all coins	-	Change existing payment
Pre-image	Steal all coins	-	-
Bounded pre-image	All of the above	-	Change existing payment
Main Hash (H_M)			
Collision	Steal coins	Steal coins†	-
Second pre-image	Steal coins	Double spend†	-
Pre-image	-	-	-
Bounded pre-image	Steal coins	All of the above	-

† Achieving this requires a slight modification of the definitions. See text for details.

Table 3: The effects of a multi-breakage: broken signature scheme in combination with a break in H_A or H_M.

Breakage	Effect
Selective forgery	Steal coins from public key
Integrity break	Claim payment not received
Repudiation	-

Table 2: Effects of a break in the signature scheme.

Breakage	Address Hash (H_A)	Main Hash (H_M)
Collision	Repudiate payment	Destroy coins
Second pre-image	Repudiate payment	Double spend and steal coins
Pre-image	Uncover address	Complete failure of the blockchain ($2n$ calls)
Bounded pre-image	All of the above	Complete failure of the blockchain (n calls)

Table 1: Summary of the effects on Bitcoin for different types of breakage in the two hash functions used.

NSA and NIST (the future of crypto)

NATIONAL SECURITY AGENCY CENTRAL SECURITY SERVICE

Defending Our Nation. Securing The Future.

HOME | ABOUT NSA | ACADEMIA | BUSINESS | CAREERS | INFORMATION ASSURANCE | RESEARCH | PUBLIC INFORMATION | CIVIL LIBERTIES

Information Assurance

- About IA at NSA
- IA Client and Partner Support
- IA News
- IA Events
- IA Mitigation Guidance
- IA Academic Outreach
- IA Business and Research
- IA Programs
  - Commercial Solutions for Classified Program
  - Global Information Grid
  - High Assurance Platform
  - Inline Media Encryptor
  - **Suite B Cryptography**
  - NSA Mobility Program
  - National Security Cyber Assistance Program
- IA Careers
- Contact Information

Home > Information Assurance > Programs > NSA Suite B Cryptography

SEARCH

Cryptography Today

In the current global environment, rapid and secure information sharing is important to protect our Nation, its citizens and its interests. Strong cryptographic algorithms and secure protocol standards are vital tools that contribute to our national security and help address the ubiquitous need for secure, interoperable communications.

Currently, Suite B cryptographic algorithms are specified by the National Institute of Standards and Technology (NIST) and are used by NSA's Information Assurance Directorate in solutions approved for protecting classified and unclassified National Security Systems (NSS). Below, we announce preliminary plans for transitioning to quantum resistant algorithms.

Background

IAD will initiate a transition to quantum resistant algorithms in the not too distant future. Based on experience in deploying Suite B, we have determined to start planning and communicating early about the upcoming transition to quantum resistant algorithms. Our ultimate goal is to provide cost effective security against a potential quantum computer. We are working with partners across the USG, vendors, and standards bodies to ensure there is a clear plan for getting a new suite of algorithms that are developed in an open and transparent manner that will form the foundation of our next Suite of cryptographic algorithms.

Until this new suite is developed and products are available implementing the quantum resistant suite, we will rely on current algorithms. For those partners and vendors that have not yet made the transition to Suite B elliptic curve algorithms, we recommend not making a significant expenditure to do so at this point but instead to prepare for the upcoming quantum resistant algorithm transition.

Algorithm	Function	Specification	Parameters
Advanced Encryption Standard (AES)	Symmetric block cipher used for information protection	FIPS Pub 197	Use 256 bit keys to protect up to TOP SECRET
Elliptic Curve Diffie-Hellman (ECDH) Key Exchange	Asymmetric algorithm used for key establishment	NIST SP 800-56A	Use Curve P-384 to protect up to TOP SECRET.
Elliptic Curve Digital Signature Algorithm (ECDSA)	Asymmetric algorithm used for digital signatures	FIPS Pub 186-4	Use Curve P-384 to protect up to TOP SECRET.
Secure Hash Algorithm (SHA)	Algorithm used for computing a condensed representation of information	FIPS Pub 180-4	Use SHA-384 to protect up to TOP SECRET.
Diffie-Hellman (DH) Key Exchange	Asymmetric algorithm used for key establishment	IETF RFC 3526	Minimum 3072-bit modulus to protect up to TOP SECRET
RSA	Asymmetric algorithm used for key establishment	NIST SP 800-56B rev 1	Minimum 3072-bit modulus to protect up to TOP SECRET
RSA	Asymmetric algorithm used for digital signatures	FIPS PUB 186-4	Minimum 3072 bit-modulus to protect up to TOP SECRET.

A CNSS Advisory Memo is or will soon be available on the CNSS website. This CNSS Advisory Memo will serve as the official interim guidance to NSS customers until a revision to *CNSSP-15, National Information Assurance Policy on the Use of Public Standards for Secure Sharing of Information Among National Security Systems*, is published codifying the increased near-term algorithm flexibility described above.

Algorithm	Function	Specification	Parameters
Advanced Encryption Standard (AES)	Symmetric block cipher used for information protection	FIPS Pub 197	Use 256 bit keys to protect up to TOP SECRET
Elliptic Curve Diffie-Hellman (ECDH) Key Exchange	Asymmetric algorithm used for key establishment	NIST SP 800-56A	Use Curve P-384 to protect up to TOP SECRET.
Elliptic Curve Digital Signature Algorithm (ECDSA)	Asymmetric algorithm used for digital signatures	FIPS Pub 186-4	Use Curve P-384 to protect up to TOP SECRET.
Secure Hash Algorithm (SHA)	Algorithm used for computing a condensed representation of information	FIPS Pub 180-4	Use SHA-384 to protect up to TOP SECRET.
Diffie-Hellman (DH) Key Exchange	Asymmetric algorithm used for key establishment	IETF RFC 3526	Minimum 3072-bit modulus to protect up to TOP SECRET
RSA	Asymmetric algorithm used for key establishment	NIST SP 800-56B rev 1	Minimum 3072-bit modulus to protect up to TOP SECRET
RSA	Asymmetric algorithm used for digital signatures	FIPS PUB 186-4	Minimum 3072 bit-modulus to protect up to TOP SECRET.

A CNSS Advisory Memo is or will soon be available on the CNSS website. This CNSS Advisory Memo will serve as the official interim guidance to NSS customers until a revision to *CNSSP-15, National Information Assurance Policy on the Use of Public Standards for Secure Sharing of Information Among National Security Systems*, is published codifying the increased near-term algorithm flexibility described above.

secp256k1

SHA-256

Algorithm	Function	Specification	Parameters
Advanced Encryption Standard (AES)	Symmetric block cipher used for information protection	FIPS Pub 197	Use 256 bit keys to protect up to TOP SECRET
Elliptic Curve Diffie-Hellman (ECDH) Key Exchange	Asymmetric algorithm used for key establishment	NIST SP 800-56A	Use Curve P-384 to protect up to TOP SECRET.
Elliptic Curve Digital Signature Algorithm (ECDSA)	Asymmetric algorithm used for digital signatures	FIPS Pub 186-4	Use Curve P-384 to protect up to TOP SECRET.
Secure Hash Algorithm (SHA)	Algorithm used for computing a condensed representation of information	FIPS Pub 180-4	Use SHA-384 to protect up to TOP SECRET.
Diffie-Hellman (DH) Key Exchange	Asymmetric algorithm used for key establishment	IETF RFC 3526	Minimum 3072-bit modulus to protect up to TOP SECRET
RSA	Asymmetric algorithm used for key establishment	NIST SP 800-56B rev 1	Minimum 3072-bit modulus to protect up to TOP SECRET
RSA	Asymmetric algorithm used for digital signatures	FIPS PUB 186-4	Minimum 3072 bit-modulus to protect up to TOP SECRET.

A CNSS Advisory Memo is or will soon be available on the CNSS website. This CNSS Advisory Memo will serve as the official interim guidance to NSS customers until a revision to *CNSSP-15, National Information Assurance Policy on the Use of Public Standards for Secure Sharing of Information Among National Security Systems*, is published codifying the increased near-term algorithm flexibility described above.

Smart contracts

- Ethereum (DAO),
- Be careful.

Making Smart Contracts Smarter

Loi Luu, Duc-Hiep Chu
National University of Singapore
{loiluu, hiepcd}@comp.nus.edu.sg

Hrishi Olickel
Yale-NUS College
hrishi.olickel@yale-nus.edu.sg

Prateek Saxena
National University of Singapore
prateeks@comp.nus.edu.sg

Aquinas Hobor
Yale-NUS College
& National University of Singapore
hobor@comp.nus.edu.sg

ABSTRACT

Cryptocurrencies record transactions in a decentralized data structure called a blockchain. Two of the most popular cryptocurrencies, Bitcoin and Ethereum, support the feature to encode rules or scripts for processing transactions. This feature has evolved to give practical shape to the ideas of smart contracts, or full-fledged programs that are run on blockchains. Recently, Ethereum's smart contract system has seen steady adoption, supporting tens of thousands of contracts, holding tens of millions dollars worth of virtual coins.

In this paper, we investigate the security of running Ethereum smart contracts in an open distributed network like those of cryptocurrencies. We introduce several new security problems in which an adversary can manipulate smart contract execution to gain profit. These bugs suggest subtle gaps in the understanding of the distributed semantics of the underlying platform. As a refinement, we propose ways to enhance the operational semantics of Ethereum to make contracts less vulnerable. For developers writing contracts for the existing Ethereum system, we build a symbolic execution tool called OYENTE to find potential security bugs. Among 19,366 existing Ethereum contracts, OYENTE flags 8,519 of them as vulnerable. We discuss the severity of attacks for several case studies which have source code available and confirm the attacks (which target only our accounts) in the main Ethereum network.

1. INTRODUCTION

Decentralized cryptocurrencies have gained considerable interest and adoption since Bitcoin was introduced in 2009 [1]. At a high level, cryptocurrencies are administered publicly by users in their network without relying on any trusted parties. Users in a cryptocurrency network run a consensus protocol to maintain and secure a shared ledger of data (the *blockchain*). Blockchain technology was initially used for peer-to-peer Bitcoin payments [1], but more recently, it has been used more broadly [2–4]. One prominent new use for blockchain technology is to enable *smart contracts*.

A smart contract is a program that runs on the blockchain and has its correct execution enforced by the consensus protocol [5]. A contract can encode any set of rules represented in its programming language—for instance, a contract can execute transfers when certain events happen (e.g. payment of security deposits in an escrow system). Accordingly, smart contracts can implement a wide range of applications, including financial instruments (*e.g.*, sub-currencies, financial derivatives, savings wallets, wills) and self-enforcing or autonomous governance applications (*e.g.*, outsourced computation [6], decentralized gambling [7]).

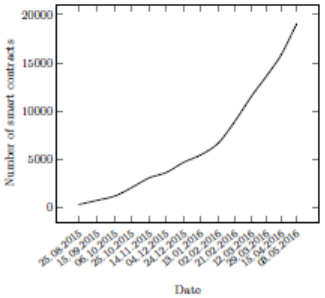


Figure 1: Number of smart contracts in Ethereum has increased rapidly.

A smart contract is identified by an address (a 160-bit identifier) and its code resides on the blockchain. Users invoke a smart contract in present cryptocurrencies by sending transactions to the contract address. Specifically, if a new transaction is accepted by the blockchain and has a contract address as the recipient, then all participants on the mining network execute the contract code with the current state of the blockchain and the transaction payloads as inputs. The network then agrees on the output and the next state of the contract by a consensus protocol. Ethereum, a more recent cryptocurrency, is a prominent Turing-complete smart contract platform [2]. Unlike Bitcoin, Ethereum supports stateful contracts in which values can persist on the blockchain to be used in multiple invocations. In the last six months alone, roughly 15,000 smart contracts have been deployed in the Ethereum network, suggesting a steady growth in the usage of the platform (see Figure 1). As Ethereum receives more public exposure and other similar projects like Rootstock [8] and CounterParty [9] emerge on top of the Bitcoin blockchain, we expect the number of smart contracts to grow.

Security problems in smart contracts. Smart contracts can handle large numbers of virtual coins worth hundreds of dollars apiece, easily making financial incentives high enough to attract adversaries. Unlike traditional distributed application platforms, smart contract platforms such as Ethereum operate in open (or permissionless) networks into which arbitrary participants can join. Thus, their execution is vulner-

Cryptology ePrint Archive: Report 2016/633

Making Smart Contracts Smarter

Loi Luu and Duc-Hiep Chu and Hrishi Olickel and Prateek Saxena and Aquinas Hobor

Abstract: Cryptocurrencies record transactions in a decentralized data structure called a blockchain. Two of the most popular cryptocurrencies, Bitcoin and Ethereum, support the feature to encode rules or scripts for processing transactions. This feature has evolved to give practical shape to the ideas of smart contracts, or full-fledged programs that are run on blockchains. Recently, Ethereum's smart contract system has seen steady adoption, supporting tens of thousands of contracts, holding tens of millions dollars worth of virtual coins.

In this paper, we investigate the security of running Ethereum smart contracts in an open distributed network like those of cryptocurrencies. We introduce several new security problems in which an adversary can manipulate smart contract execution to gain profit. These bugs suggest subtle gaps in the understanding of the distributed semantics of the underlying platform. As a refinement, we propose ways to enhance the operational semantics of Ethereum to make contracts less vulnerable. For developers writing contracts for the existing Ethereum system, we build a symbolic execution tool called Oyente to find potential security bugs. Among 19, 366 existing Ethereum contracts, Oyente flags 8, 519 of them as vulnerable. We discuss the severity of attacks for several case studies which have source code available and confirm the attacks (which target only our accounts) in the main Ethereum network.

Category / Keywords: Ethereum, Cryptocurrencies, blockchains, smart contracts

Date: received 18 Jun 2016, last revised 20 Jun 2016

Contact author: loiluu at comp nus edu sg

Available format(s): PDF | BibTeX Citation

Version: 20160621:154028 (All versions of this report)

Short URL: ia.cr/2016/633

Enhancing Accountability and Trust in Distributed Ledgers*

Maurice Herlihy
Brown University and Oracle Labs
mph@cs.brown.edu

Mark Moir
Oracle Labs
mark.moir@oracle.com

June 27, 2016

Abstract

Permisionless decentralized ledgers ("blockchains") such as the one underlying the cryptocurrency Bitcoin allow anonymous participants to maintain the ledger, while avoiding control or "censorship" by any single entity. In contrast, *permissioned* decentralized ledgers exploit real-world trust and accountability, allowing only explicitly authorized parties to maintain the ledger. Permissioned ledgers support more flexible governance and a wider choice of consensus mechanisms.

Both kinds of decentralized ledgers may be susceptible to manipulation by participants who favor some transactions over others. The real-world accountability underlying permissioned ledgers provides an opportunity to impose fairness constraints that can be enforced by penalizing violators after-the-fact. To date, however, this opportunity has not been fully exploited, unnecessarily leaving participants latitude to manipulate outcomes undetectably.

This paper draws attention to this issue, and proposes design principles to make such manipulation more difficult, as well as specific mechanisms to make it easier to detect when violations occur.

1 Introduction

A *blockchain* is a data structure used to implement tamper-resistant distributed ledgers. Multiple *nodes* follow a common protocol in which transactions from *clients* are packaged into *blocks*, and nodes use a consensus protocol to agree on successive blocks. Each block's *header* contains a cryptographic hash of the previous block's header, making it difficult to tamper with the ledger. Bitcoin [20] is the best-known blockchain-based distributed ledger today.

In *permissionless* implementations, such as Bitcoin, any node willing to follow the protocol can participate, and anybody can generate addresses that can receive bitcoins, and can propose transactions that transfer bitcoins from any address for which they have the associated private key. By contrast, in *permissioned* implementations, the sets of participating nodes are controlled by an authority, perhaps one organization, perhaps a consortium.

A permissionless implementation makes sense for applications such as Bitcoin, which seek to ensure that nobody can control who can participate, a property often called *censorship resistance*. By contrast, permissioned implementations explicitly permit some forms of censorship: for example, permitting compliance with "know your customer" regulations that exclude known money-launderers from financial markets. Moreover, permissioned implementations can often provide more effective governance: for example, by providing an orderly procedure for updating the ledger protocol [11].

Here, we focus on one more important difference: permissioned ledgers can hold participants *accountable* for misbehavior in ways that permissionless implementations cannot. Many distributed ledgers would benefit from *fairness* guarantees. For example, one client's proposed transactions should not be systematically delayed longer than others', or one client's transactions should not be systematically scheduled just

A nous de jouer !

- Nous pouvons vous aider …