

AI GUUDE WIE ! I AM WILLIE



B. SC. IN MATH, CURRENTLY M. SC. IN MATH AND
DISTRIBUTED LEDGER DEVELOPER @ MIDAS TECHNOLOGIES AG
IN THE CRYPTO-SCENE SINCE 2017
HEARD FROM BITCOIN IN 2017



BLOCKCHAIN@LAUBENHEIMER.EU
@IEM_WLL



ash.finance



EINSTIEG

Bitte stehen Sie auf und lassen die folgenden 4 Zitate auf sich wirken. Nach 2-4 Minuten bleiben Sie bei einem Zitat stehen und unterhalten sich mit Ihren Kommilitonen über die Thematik, welche Gedanken löst es bei Ihnen aus?

1. „80% der weltweit zur Verfügung stehenden Rechenkapazität wird zur Berechnung von Zufallszahlen benutzt.“
2. Ex-NSA Chef: „Wir töten aufgrund von Metadaten!“
3. „Sicherheit ist eine demokratische Entscheidung!“
4. „Ganze Berufsgruppen durch Smart Contracts/ schlaue digitale Programme vor dem Aus?“

Anschließend teilt eine oder zwei Personen den anderen mit, was in den Gesprächen aufkam.

INDIVIDUELLE ERWARTUNGEN

Was soll heute passieren/
Was möchte ich mitnehmen?

Was soll heute nicht passieren/
Worauf habe ich keine Lust?

MATHE EXKURS – KRYPTOGRAPHIE UND BLOCKCHAIN

1. Einführung in die Kryptographie

1. Zufallszahlen und Grundlagen

1. Pseudo-zufallszahlen
2. Teilen mit Rest (modulo)
3. Linearer Kongruenzgenerator
4. Ausblick

2. Kryptographie

1. Geschichte und Relevanz
2. Kryptographische Verfahren
3. Signatur
4. Ausblick

2. Einführung in die Blockchain-Technologie

1. Bitcoin

1. Geschichte
2. Innovation
3. Technisch

2. Ethereum

1. Geschichte
2. Innovation
3. Technisch
4. Hands On

3. Ausblick

EINFÜHRUNG IN DIE KRYPTOGRAPHIE | ZUFALLSZAHLEN UND GRUNDLAGEN

1. PSEUDO-ZUFALLSZAHLEN

Warum sind Zufallszahlen überhaupt wichtig?

Angeblich werden **50-80% der gesamten Rechenkapazität** auf unserem Planeten zum Erzeugen von Zufallszahlen verwendet. (Hauptbestandteil komplexer Simulationen)

Generatoren von Zufallszahlen:

1. Physikalische (Hardware-)Generatoren:
 1. Radioaktiver Zerfall
 2. Thermisches Rauschen
 3. Elektrische Schwankungen
2. Software Generatoren:
 1. Algorithmen (Erzeugung mathematischer Zufallszahlen)
 2. **Pseudo-Zufallszahlen** (weniger Anforderungen, die der gewünschten Anwendung genügen)

Vorgehen:

- 1955: Buch mit 1 Millionen Zufallszahlen
- 1995: CD mit ca. 4,8 Milliarden Zufallszahlen
- Heutzutage: **ad-hoc Generierung**

Forderungen:

- Gleichverteilung (auf $[0,N]$ oder $[0,1]$)
- Unvorhersagbarkeit (Konstruktionsmechanismus komplex)
- Reproduzierbarkeit (für Fehlersuche, Vergleich von Simulationen, Wiederholen von Key-generierung, ec.)
- Effizienz (geringer Speicherbedarf)

EINFÜHRUNG IN DIE KRYPTOGRAPHIE | ZUFALLSZAHLEN UND GRUNDLAGEN

1. PSEUDO-ZUFALLSZAHLEN

Eine der ersten Methoden zur Erzeugung von Zufallszahlen wurde 1940 von Metropolis von Neumann Entwickelt.

Middle-Square-Methode

1. Wähle eine vierstellige Zahl
2. Quadriere diese Zahl
3. Falls die erzeugte Zahl keine 8 Zahlen hat, füge von links Nullen hinzu, bis 8 Zahlen erreicht sind
4. Wähle die mittleren 4 Zahlen als neue Zahl

Aufgabe:

1. Benutze die M-S-M mit drei Durchläufen und **beliebigem Startwert**
2. Benutze die M-S-M mit **Startwert 8441** und acht Durchläufen

Probleme der Middle-Square-Methode:

1. Periodizität (unendlicher Kreislauf)
2. Die Iteration Konvergiert (7182, 5811, ..., 0012, 0001, 0000)

EINFÜHRUNG IN DIE KRYPTOGRAPHIE | ZUFALLSZAHLEN UND GRUNDLAGEN

2. TEILEN MIT REST (MODULO)

Normales Teilen:

$$4:2 = \frac{4}{2} = 2 \quad \text{Wie oft passt die 2 in die 4?}$$

$$5:2 = \frac{5}{2} = 2,5 \quad \text{Wie oft passt die 2 in die 5?}$$

ODER

Teilen mit Rest:

$$5:2 = 2 \text{ R } 1 \quad \text{Wie oft passt die 2 **ganz** in die 5?}$$

Und was bleibt übrig? (REST)

$$50:3 = 16 \text{ R } 2 \quad \text{Wie oft passt die 3 **ganz** in die 50?}$$

Aufgaben:

1. $5:3 = ?$
2. $6:3 = ?$
3. $14:5 = ?$
4. $17:8 = ?$
5. $17:9 = ?$

Modulo ist eine Schreibweise für teilen mit Rest:

$$4 \bmod 2 = 0$$

$$5 \bmod 2 = 1$$

$$14 \bmod 5 = 4$$

...

Eine Zahl heißt Primzahl, wenn sie keinen ganzzahligen Teiler außer der 1 und sich selber hat. (Durch „nichts“ teilbar.)

Beispiel:

Die Teiler von 6 sind 2 und 3.

Die Teiler von 8 sind 4 und 3.

Die Teiler von 7 sind ... ?

Zwei Zahlen x und y sind Teilerfremd, wenn sie **keinen gemeinsamen Teiler** haben.

Aufgaben: 6 und 9, 10 und 6, 15 und 14, 9 und 7

EINFÜHRUNG IN DIE KRYPTOGRAPHIE | ZUFALLSZAHLEN UND GRUNDLAGEN

3. LINEARER KONGRUENZGENERATOR

Rechenvorschrift: $x_{n+1} = a * x_n + b \bmod m$

Ziel: Die Variablen x_0 (auch Seed/Startwert genannt), a und b sollten so gewählt werden, dass die erzeugten Zufallszahlen möglichst gleichverteilt in $\{0, \dots, m - 1\}$ liegen. Der Seed garantiert hier die Reproduzierbarkeit.

Beispiel: $a=1, b=1, m=10$ und $x_0=5$: $X=\{6,7,8,9,0,1,2,3,4,5,6 \dots\}$ hat maximale Periode.
 $a=7, b=7, m=10$ und $x_0=7$: $X=\{6,9,0,7,6, \dots\}$ hat Periode 4.

Satz von Knuth:

LCG ist periodisch mit maximaler Länge, genau dann wenn:

1. b und m sind teilerfremd
2. $p \mid (a-1)$ **für alle** Primteiler p von m („ p teilt $(a-1)$ “)
3. falls $4 \mid m$ dann auch $4 \mid (a-1)$ („falls 4 Teiler von m ist dann muss 4 auch $(a-1)$ teilen“)

Aufgabe: Erzeugen Sie einen Linearen Kongruenzgenerator und beweisen Sie, dass dieser maximale Periode hat.

- Wie kriegt man Zufallszahlen auf $[0,1]$ verteilt? (Normalisieren: $y_i = \frac{x_i}{m}$)
- Gütetests für Zufallszahl-Generatoren
- Quasi-Zufallszahlen (Zufall nicht so wichtig, Fokus auf der Gleichverteilung)
 - Van der Corput Folge
 - Halton Folge
- Generierung Standard normalverteilter Zufallszahlen (Stichwort Eulersche Glockenkurve)
 - Box-Muller-Verfahren



Fragen?

Ideen?

Anmerkungen?

Arbeitsauftrag

WO und WARUM ist Kryptographie von Bedeutung / wichtig?

EINFÜHRUNG IN DIE KRYPTOGRAPHIE | KRYPTOGRAPHIE

1. GESCHICHTE UND RELEVANZ

Definition von Wikipedia

Kryptographie bzw. **Kryptografie** (altgriechisch κρυπτός *kryptós*, deutsch ‚verborgen‘, ‚geheim‘ und γράφειν *gráphein*, deutsch ‚schreiben‘) ist ursprünglich die Wissenschaft der Verschlüsselung von Informationen. Heute befasst sie sich auch allgemein mit dem Thema Informationssicherheit, also der Konzeption, Definition und Konstruktion von Informationssystemen, die widerstandsfähig gegen Manipulation und unbefugtes Lesen sind.

Schon die Ägypter benutzten verschiedenste Verschlüsselungsmethoden. (drittes Jahrtausend v. Chr.)

Alan Turing schuf die Grundlagen für den heutigen modernen Computer. Er entschlüsselte, die damals als unknackbar geltende Verschlüsselungstechnik Enigma, und verkürzte somit vermutlich die Dauer des zweiten Weltkrieges. (um 1940 n. Chr.)

Ex-NSA Chef Hayden: „Wir töten Menschen aufgrund von Metadaten“ (2014 n. Chr.)

Ziele der Kryptographie [Bearbeiten | Quelltext bearbeiten]

Die moderne Kryptographie hat vier Hauptziele zum Schutz von Datenbeständen, Nachrichten und/oder Übertragungskanälen:^[3]

1. **Vertraulichkeit**/Zugriffsschutz: Nur dazu berechtigte Personen sollen in der Lage sein, die Daten oder die Nachricht zu lesen oder Informationen über ihren Inhalt zu erlangen.
2. **Integrität**/Änderungsschutz: Die Daten müssen nachweislich vollständig und unverändert sein.
3. **Authentizität**/Fälschungsschutz: Der Urheber der Daten oder der Absender der Nachricht soll eindeutig identifizierbar sein, und seine Urheberschaft sollte nachprüfbar sein.
4. **Verbindlichkeit**/Nichtabstreitbarkeit: Der Urheber der Daten oder Absender einer Nachricht soll nicht in der Lage sein, seine Urheberschaft zu bestreiten, d. h., sie sollte sich gegenüber Dritten nachweisen lassen.

Kryptographische Verfahren und Systeme dienen nicht notwendigerweise gleichzeitig allen der hier aufgelisteten Ziele.

EINFÜHRUNG IN DIE KRYPTOGRAPHIE | KRYPTOGRAPHIE

2. KRYPTOGRAPHISCHE VERFAHREN

Es gibt zwei Arten von Verschlüsselungsmethoden

Symmetrische Verschlüsselung

- Ein Schlüssel für beide Teilnehmer
- Dieser Schlüssel muss auf einem sicheren Weg ausgetauscht werden
- Ver- und Entschlüsselung werden mit dem gleichen Schlüssel gemacht

Asymmetrische Verschlüsselung

- Ein Schlüssel zum verschlüsseln (öffentlicher Schlüssel)
- Ein Schlüssel zum entschlüsseln (privater Schlüssel)
- Der öffentliche Schlüssel wird veröffentlicht, da er nur zum verschlüsseln benutzt werden kann. Also ist kein direkter Schlüsselaustausch zwischen den Teilnehmern nötig

EINFÜHRUNG IN DIE KRYPTOGRAPHIE | KRYPTOGRAPHIE

2.1 KRYPTOGRAPHISCHE VERFAHREN – RSA

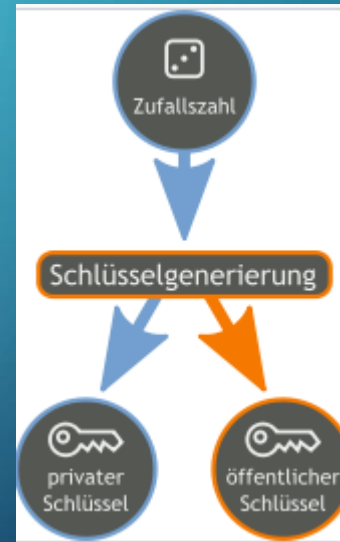
RSA ist eine weit verbreitete Verschlüsselungsmethode.

Das Verfahren:

1. Wähle zwei möglichst große Primzahlen p und q
2. Berechne $N = p * q$ (auch RSA-Modul genannt)
3. Berechne $\phi(N) = (p-1)*(q-1)$ und wähle $1 < e < \phi(N)$, sodass e teilerfremd zu $\phi(N)$ ist.
4. Der private Schlüssel (d, N) ist das multiplikative Inverse d von e , bezüglich modulo $\phi(N)$.
Das heißt, d muss die Gleichung $(d * e \bmod \phi(N)) = 1$ erfüllen. Dafür benutzt man den erweiterten euklidischen Algorithmus.

Öffentlicher Schlüssel:
 (e, N)

Privater Schlüssel:
 (d, N)

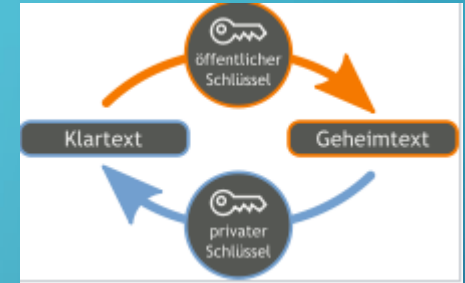
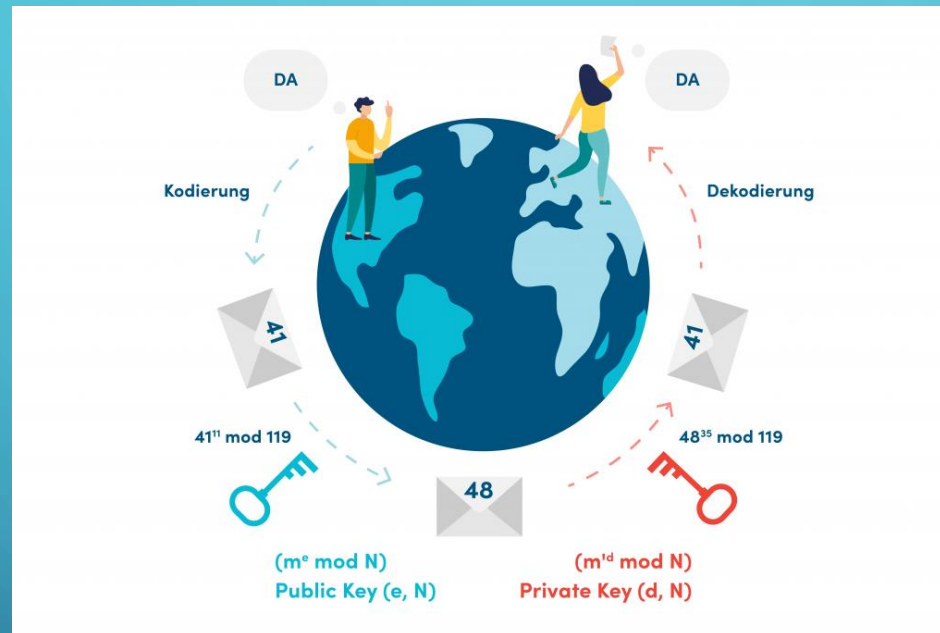


Merke: $\phi(N)$ ist nur dem Schlüsselerzeuger bekannt. Die Sicherheit dieses Kryptosystems beruht auf dem mathematischen Problem, dass es nicht möglich ist in konstanter Zeit eine natürliche Zahl in seine Primfaktoren zu zerlegen! Denn N ist ja öffentlich bekannt...

EINFÜHRUNG IN DIE KRYPTOGRAPHIE | KRYPTOGRAPHIE

2.1 KRYPTOGRAPHISCHE VERFAHREN – RSA

Ein Beispiel: <https://cryptomonday.de/wie-funktioniert-verschluesselung/>



Und wie sieht das jetzt in echt aus? → <https://gnupg.org/download/index.html>

EINFÜHRUNG IN DIE KRYPTOGRAPHIE | KRYPTOGRAPHIE

3. SIGNATUR – HASH FUNKTION

Was ist eine Hash Funktion?

Eine Hashfunktion kann man auch als „Einwegfunktion“ bezeichnen. Man gibt etwas rein und kriegt etwas heraus. Allerdings kann man niemals durch das, was herauskommt, auf das, was man hereingetan hat, kommen.

Mathematischer: $h(x) = y$, aber es gibt kein $h^{-1}(y) = x$. (Keine Umkehrfunktion)

Kennen wir y , dann ist das Problem, irgendein x zu finden, sodass $h(x)=y$ gilt, nicht lösbar.

Außerdem sollte eine Hashfunktion Kollisionsresistent sein. Das heißt, es ist fast unmöglich zwei Inputs zu finden, sodass die Outputs identisch sind. ($h(a)=h(b)$, $a \neq b$ praktisch nicht möglich)

Beispiele mit python. → https://www.onlinegdb.com/online_python_compiler

EINFÜHRUNG IN DIE KRYPTOGRAPHIE | KRYPTOGRAPHIE

3. SIGNATUR – SCHNORR SIGNATUR - SCHLÜSSELERZEUGUNG

- Sei **q** eine große Primzahl und **p** eine noch größere mit $(p \bmod q) = 1$ (In der Praxis wählt man meist $q \approx 2^{160}$ und $p \approx 2^{1024}$)
- Sei **g** ein Erzeuger der zyklischen Untergruppe G der Ordnung q in der ganzzahligen, multiplikativen Gruppe mod p. (Es ist einfach einen solchen Erzeuger zu finden, indem man eine zufällige ganze Zahl mit $((p-1)/q)$ potenziert und diese mod p rechnet.)
- Sei **H** eine Hash-Funktion, die Werte der Menge $\{0, 1, \dots, q-1\}$ annimmt.
- Zum Vorbereiten der Signatur wählt Alice zuerst ein geheimes, ganzzahliges x; mit $0 < x < q$. Dies ist ihr **geheimer Schlüssel**. Dann berechnet sie $y = g^x \pmod{p}$, welches ihr **öffentlicher Schlüssel** ist.

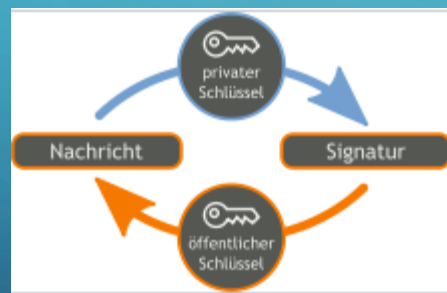
Zusammengefasst:

1. Es werden wieder mithilfe zweier Primzahlen ein öffentlicher Schlüssel x und ein geheimer Schlüssel y erzeugt.
2. Das zugrunde liegende mathematische Problem bei der Schnorr Signatur ist der diskrete Logarithmus.

EINFÜHRUNG IN DIE KRYPTOGRAPHIE | KRYPTOGRAPHIE

3. SIGNATUR – SCHNORR SIGNATUR - DIE SIGNATUR

1. Um eine Nachricht zu signieren, wählt Alice zuerst ein **zufälliges** $0 < k < q$.
Dabei muss k für jede Nachricht neu gewählt werden!
2. Sie berechnet $r = g^k \pmod{p}$ und berechnet dann $h = H(m, r)$
3. Zuletzt berechnet Alice $s = (k + h * x \pmod{q})$
4. Die Schnorr Signatur ist **(h, s)**
5. Um die Signatur zu verifizieren berechnet Bob $r' = g^s * y^{-h}$ und prüft, ob $H(m, r') = h$ gilt.
6. Öffentlich bekannt sind g und y , außerdem kennt Bob h und s von der Signatur. Somit kann er die Überprüfung auch durchführen.



EINFÜHRUNG IN DIE KRYPTOGRAPHIE | KRYPTOGRAPHIE

4. AUSBLICK

- Multisignaturen: Mehrere Teilnehmer erzeugen gemeinsame Signaturen (auch mit Schnorr Sigs möglich)
- Elliptic-Curve Kryptographie: Anderes zugrundeliegende mathematisches Problem
- Zero Knowledge Proofs (Monero)
- Noch viel, viel mehr „langweilige“ Mathematik 😊

A top-down view of a white ceramic coffee cup filled with a frothy, light brown beverage. In the center of the foam is a large, white, stylized Bitcoin symbol (₿). The cup is placed on a dark brown wooden surface with a visible grain pattern. The text "Endlich Mittagspause 😊" is written in white, bold, sans-serif font across the top left of the cup's rim.

Endlich Mittagspause 😊

Fragen?

Ideen?

Anmerkungen?

EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | BITCOIN

1. GESCHICHTE

Bitcoin

Bitcoin is a cryptocurrency developed by Satoshi Nakamoto in 2009. Bitcoin is used as a digital payment system. Rather than use traditional currency (USD, YEN, EURO, etc.) individuals may trade in, or even mine Bitcoin. It is a peer-to-peer system, and transactions may take place between users directly.

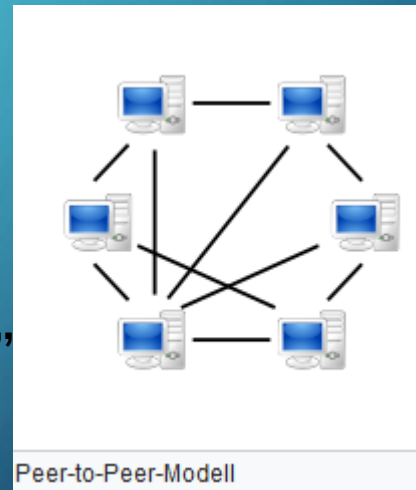


Als Reaktion auf die globale Finanzkrise im Jahre 2008 veröffentlichte eine bis heute unbekannte Person oder Gruppe unter dem Synonym „Satoshi Nakamoto“ das Bitcoin-Whitepaper anfangs 2009.

“We have proposed a system for electronic transactions without relying on trust.”

Dieser Satz aus dem Whitepaper spiegelt das Ziel dieser Arbeit sehr gut wieder. Man hatte ein System für elektronische Transaktionen entwickelt, ohne gegenseitiges Vertrauen zu benötigen.

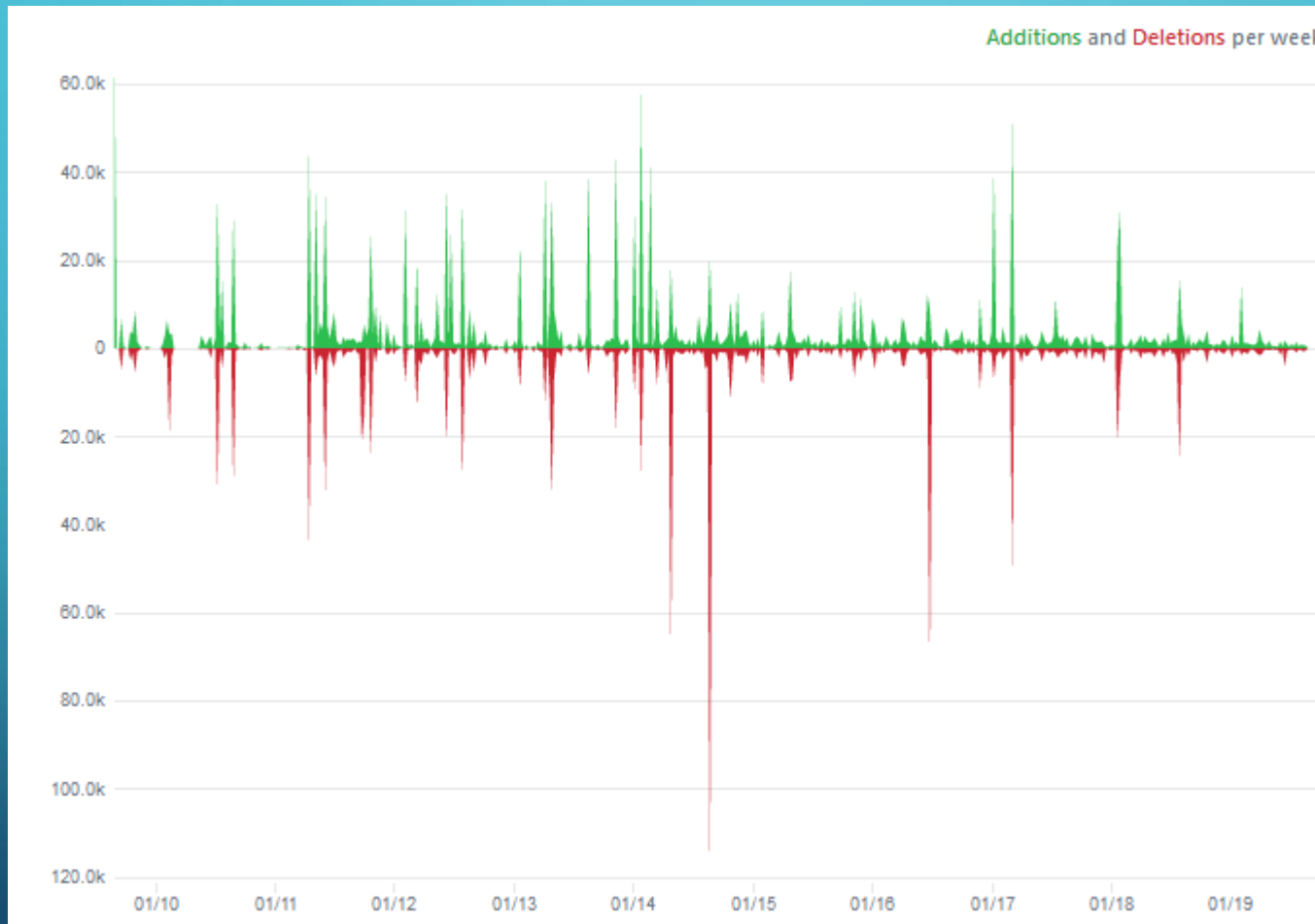
Nach der Veröffentlichung des Codes auf Github, blieb dieses Projekt noch sehr lange unentdeckt und nur eine kleine Gruppe digitaler Pioniere begriff den Wert und das Potential dieses Protokolls. Nachdem es aufgrund der Funktionalität auf Schwarzmärkten und im Gaming-bereich benutzt wurde kam es in den Jahren 2016-2018 in der Weltbevölkerung an.



EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | BITCOIN

1. GESCHICHTE

Bitcoin code frequency




EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | BITCOIN

1. GESCHICHTE

1. Issue:

JSON-RPC support for mobile devices ("ultra-lightweight" clients) #1


 Closed gavinandresen opened this issue on 19 Dec 2010 · 3 comments

2010

Ca. 2 Jahre später
geschlossen

1. Bug:

Mac UI issues #16

 Closed gavinandresen opened this issue on 29 Dec 2010 · 3 comments

2010

Ca. 1 Jahr später
geschlossen


Remove IsFromMe() call from bool CtxMemPool::accept()
#2178

 Closed SergioDemianLerner opened this issue on 14 Jan 2013 · 22 comments

2013

Ca. 10 Tage
später geschlossen


hard crash when validateaddress RPC endpoint is
requested #6963

 Closed jllopp opened this issue on 6 Nov 2015 · 30 comments

2015

Ca. 4 Tage später
geschlossen

v0.14.0 shutdown hangs #9950

 Closed Ceelean opened this issue on 8 Mar 2017 · 12 comments

2017

Ca. 1 Tag später
geschlossen

EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | BITCOIN

2. INNOVATION

- Größtes Open Source Community Projekt
- Ein System für elektronische Transaktionen, das kein Vertrauen der Transaktionspartner benötigt
- Unbekannter Gründer/Erfinder (Mythos)
- Neue Art von Datenbank und dezentrale Sicherung der Daten
- Sicherheit ist eine demokratische Entscheidung

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.

Größtes Open Source Community Projekt

| Unternehmen | „likes“ | Forks | Entwickler |
|-------------|---------|---------|------------|
| Bitcoin | ~40.000 | ~24.000 | 643 |
| Google | ~17.700 | ~5.000 | 27 |
| Microsoft | ~80.000 | ~12.000 | ~1.000 |
| NASA | ~8.000 | ~700 | 71 |

Einziges Projekt mit knapp 200 Milliarden Marktkapitalisierung, bei dem JEDE ENTSCHEIDUNG öffentlich ist.

EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | BITCOIN

3. TECHNISCH



Grundlagen für neue Nutzer

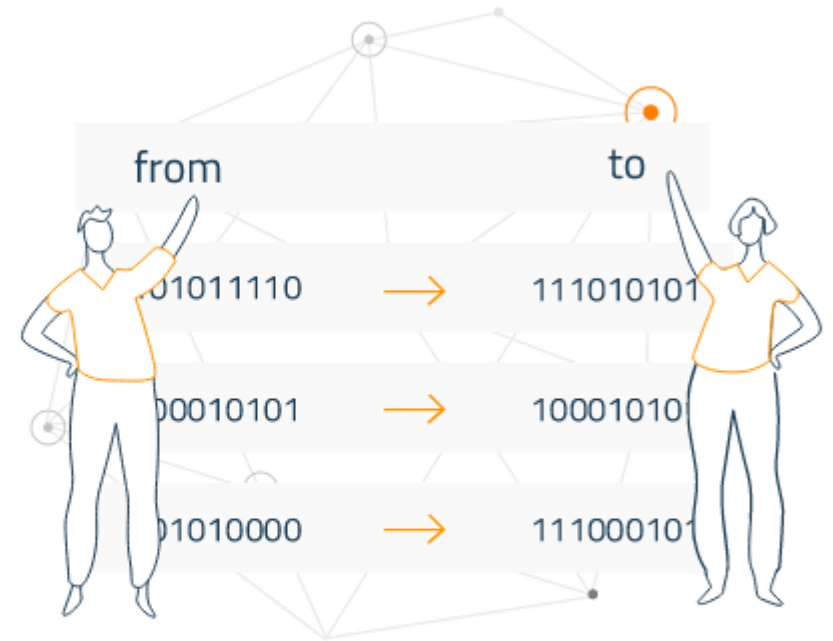
Als neuer Nutzer können Sie mit Bitcoin **loslegen**, ohne die technischen Details zu verstehen. Sobald Sie eine Wallet auf Ihrem Computer oder Smartphone installiert haben, generiert diese Ihre erste Bitcoin-Adresse und Sie können weitere erstellen, sobald welche benötigt werden. Sie können Ihre Bitcoin-Adressen an Ihre Freunde weitergeben, so dass diese Geld an Sie senden können, oder umgekehrt. Tatsächlich ist das vergleichbar mit der Funktionsweise von E-Mails, außer dass Bitcoin-Adressen nur einmal verwendet werden sollten.

EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | BITCOIN

3. TECHNISCH

Kontostände - Blockchain

Die Blockchain ist ein **gemeinsam genutztes öffentliches Buchungssystem**, auf dem das gesamte Bitcoin-Netzwerk basiert. Alle bestätigten Transaktionen werden in der Blockchain gespeichert. Auf diese Art können Bitcoin-Wallets den Kontostand berechnen und neue Transaktionen können nur ausgeführt werden, wenn die Bitcoins dem Sender tatsächlich gehören. Die Integrität und die chronologische Reihenfolge der Blockchain werden durch **Kryptographie** sichergestellt.



EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | BITCOIN

3. TECHNISCH - TRANSAKTIONEN



Transaktionen - private Schlüssel

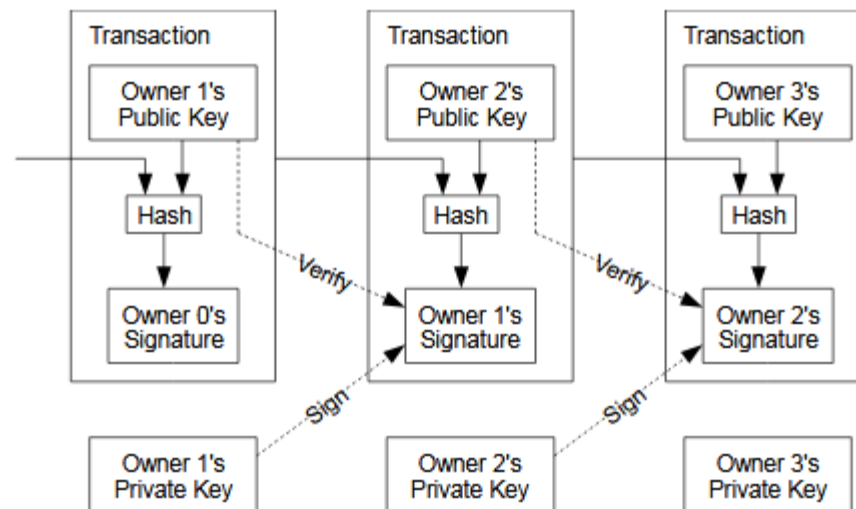
Eine Transaktion ist **der Transfer eines Betrages zwischen Bitcoin-Wallets**, der in die Blockchain eingetragen wird. Bitcoin-Wallets enthalten einen geheimen Datenblock der *privater Schlüssel* oder "Seed" genannt wird. Er wird verwendet, um Transaktionen zu signieren, wodurch der mathematische Beweis erbracht wird, dass sie vom Eigentümer der Wallet kommen. Die *Signatur* verhindert auch, dass Transaktionen nach dem Senden von jemandem modifiziert werden können. Alle Transaktionen werden über das Netzwerk verbreitet und innerhalb von 10-20 Minuten beginnt die Bestätigung durch das Netzwerk mit Hilfe eines Prozesses der *Mining* genannt wird.

EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | BITCOIN

3. TECHNISCH - TRANSAKTIONEN

2. Transactions

We define an electronic coin as a chain of digital signatures. Each owner transfers the coin to the next by digitally signing a hash of the previous transaction and the public key of the next owner and adding these to the end of the coin. A payee can verify the signatures to verify the chain of ownership.

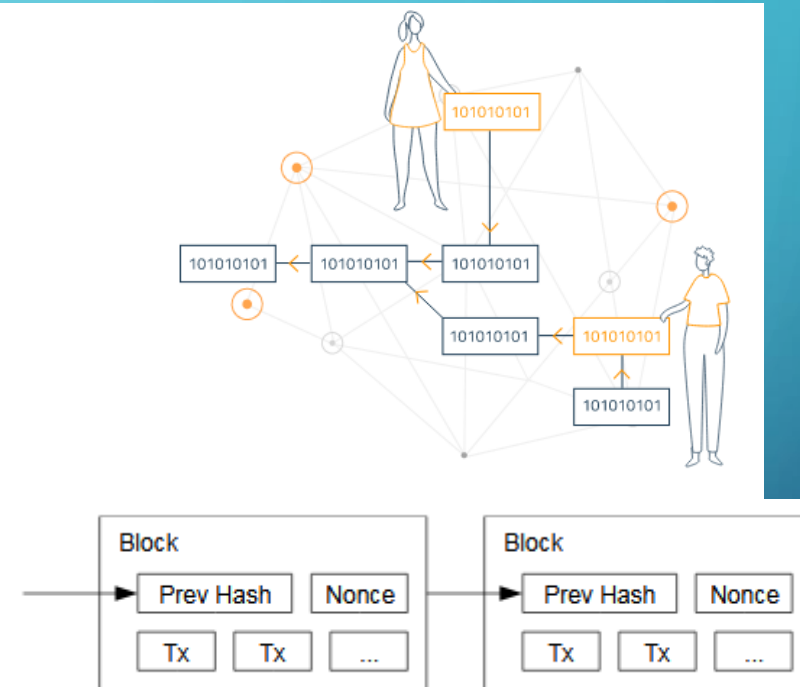


EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | BITCOIN

3. TECHNISCH - MINING

Verarbeitung - Mining

Mining ist ein **verteiltes Konsens-System**, das verwendet wird, um ausstehende Transaktionen durch deren Aufnahme in die Blockchain zu **bestätigen**. Mining erzwingt eine chronologische Reihenfolge der Blockchain, schützt die Neutralität des Netzwerks und sorgt dafür, dass sich die verschiedenen Computer über den Status des Systems einig sind. Um bestätigt zu werden, müssen Transaktionen in einen **Block** eingefügt werden. Dieser muss sehr strengen kryptographischen Regeln genügen, die durch das Netzwerk verifiziert werden. Diese Regeln verhindern, dass vorangegangene Blöcke modifiziert werden können, denn eine Änderung würde alle darauffolgende Blöcke ungültig machen. Das Mining ist auch eine Art Lotterie mit starker Konkurrenz, die verhindert, dass jemand einfach neue aufeinanderfolgende Blöcke zu der Blockchain hinzufügt. Auf diese Weise kann niemand kontrollieren, was in die Blockchain aufgenommen wird, oder Teile der Blockchain so modifizieren, dass eigene Ausgaben rückgängig gemacht werden.



5. Network

The steps to run the network are as follows:

- 1) New transactions are broadcast to all nodes.
- 2) Each node collects new transactions into a block.
- 3) Each node works on finding a difficult proof-of-work for its block.
- 4) When a node finds a proof-of-work, it broadcasts the block to all nodes.
- 5) Nodes accept the block only if all transactions in it are valid and not already spent.
- 6) Nodes express their acceptance of the block by working on creating the next block in the chain, using the hash of the accepted block as the previous hash.

Gemeinsamer Arbeitsauftrag 😊

Alles zusammen in eine Bitcoin-Grafik bringen



Fragen?

Ideen?

Anmerkungen?

1. GESCHICHTE

1. Veröffentlichung des Whitepapers (hat im Gegensatz zu Bitcoin -9 Seiten- 236 Seiten) Ende 2013 durch Vitalik Buterin (Mitwirkender am Bitcoin Protokoll und Co-founder des Bitcoin Magazin)
2. Crowdfunding durch Bitcoin im tausch gegen Ether (von Gavin Wood organisiert)
3. Mitte 2015 startet das Ethereum-Netzwerk
 - Ca. ein halbes Jahr später eine Marktkapitalisierung von 500 Millionen USD
 - 2 Wochen später ca. 1 Milliarde USD Marktkapitalisierung

Insgesamt zweit „schwerste“ Kryptowährung nach Marktkapitalisierung (hinter Bitcoin).



- Größte Innovation: Turing vollständig → Eine Maschine oder Programmiersprache, die jedes Problem, das mit code gelöst werden kann, lösen kann, ist Turing vollständig. (z.B. For-, Foreach-, While-Schleifen, Where, ec.) (sehr vereinfacht!)
- Zusammengefasst: Beliebige komplexe dezentrale Programme (dApps) auf der Ethereum Blockchain
- Daraus resultierend eine der bekanntesten Innovationen: **Ganz einfaches Erstellen neuer „Kryptowährungen“**. (<https://www.youtube.com/watch?v=ac1P3GXkFxc>)

Aber wie funktioniert das?

1. Smart Contract schreiben
2. Ins Netzwerk deployen (schicken) – mit Transaktionskosten
3. Jede Codeausführung kostet (Gas)

Ethereum Einheiten:

- Größte Einheit: Ether
- Kleinste Einheit: Wei
- 1 Ether entspricht 10^{18} Wei
- Gas Preis entspricht in der Regel Gwei
- 1 Ether entspricht 10^9 Gwei (Giga Wei)



EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | ETHEREUM

3. TECHNISCH – ERC20 TOKEN STANDARD

ERC: Token standard #20

Closed

frozeman opened this issue on Nov 19, 2015 · 362 comments

- 11/19/2015 – das Geburtsdatum der berühmten Issue #20 im Ethereum Repository
- ERC ist die kurze Version für Ethereum Issues, die kommentiert/diskutiert werden soll – Ethereum Request for Comments

ERC: Token standard ERC editor-needs-to-review

#20 by frozeman was closed on Apr 24, 2017

362

Ethereum Request for Comments (ERCs) and Ethereum Standards (ESDs)

#16 by ethers was closed on Nov 19, 2015

8

- Das 20-te Problem hat über 350 Kommentare und wurde von Fabian Vogelsteller nach knapp 2 Jahren Diskussion und Arbeit am 29.ten September 2017 geschlossen.

frozeman commented on Sep 29, 2017

Member

The final standard can be found here: <https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20-token-standard.md>

EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | ETHEREUM

3. TECHNISCH - ERC20 TOKEN STANDARD

- Der ursprüngliche Grund war die Standardisierung von Tokens zur einfach Integrierung in Apps/Anwendungen
- Börsen und Anwendungen müssen so nur einmal die Standardfunktionen implementieren und alle ERC20 Token funktionieren sofort
- Die Standardisierung der Tokens war einer der Gründe für den exponentiellen Wachstum an neuen Kryptowährungen in 2018
- Bis heute werden Kosten und Zeit durch einmalige Arbeit, die über tausend mal benutzt werden kann, gespart
- Der ERC20 Token Standard ist die „Killer-Application“ auf Ethereum

EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | ETHEREUM

3. TECHNISCH - ERC20 TOKEN STANDARD

```
/**
 * See https://github.com/ethereum/EIPs/issues/20
 */
contract ERC20 {
    // Triggered when tokens are transferred.
    event Transfer(address indexed _from, address indexed _to, uint256 _value);

    // Triggered whenever approve(address _spender, uint256 _value) is called.
    event Approval(address indexed _owner, address indexed _spender, uint256 _value);

    // Get the total token supply
    function totalSupply() constant returns (uint256 totalSupply);

    // Get the account balance of another account with address _owner
    function balanceOf(address _owner) constant returns (uint256 balance);

    // Send _value amount of tokens to address _to
    function transfer(address _to, uint256 _value) returns (bool success);

    // Send _value amount of tokens from address _from to address _to
    function transferFrom(address _from, address _to, uint256 _value) returns (bool success);

    // Allow _spender to withdraw from your account, multiple times, up to the _value amount. If this
    // function is called again it overwrites the current allowance with _value.
    function approve(address _spender, uint256 _value) returns (bool success);

    // Returns the amount which _spender is still allowed to withdraw from _owner
    function allowance(address _owner, address _spender) constant returns (uint256 remaining);
}
```

Was passiert wenn Sie mir einen ERC20 Token schicken?

1. Sie führen die Transaktion `transfer(address _to, uint256 _value)` von dem ERC20-Token Contract aus.
2. Der Contract subtrahiert den Wert `_value` intern von dem Guthaben, dass mit Ihrer Adresse verknüpft ist und addiert den Wert intern zu meiner Adresse.
3. Dann wird ein Event getriggert, sonst nichts...

```
contract SunToken is ERC20, Owned {
    mapping (address => uint256) public balances;
    mapping (address => mapping (address => uint256)) public allowance;
    uint public price; // price per token.
    uint256 supply;
```

```
function transfer(address _to, uint256 _value) No0x(_to) ValidBalance(msg.sender, _to, _value)
returns (bool success) {
    balances[msg.sender] -= _value;           // Subtract from the sender
    balances[_to] += _value;                 // Add the same to the recipient
    Transfer(msg.sender, _to, _value);       // Notify anyone listening that this transfer took place
    return true;
}
```

Wurde also ein Token transferiert?

Nein!

Alles passiert lediglich in dem Smart Contract von dem Token. Getreu dem Motto:

„Was im Smart Contract passiert, bleibt im Smart Contract...“

EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | ETHEREUM

3. TECHNISCH – HANDS ON

Download „Cipher-browser“:



IOS



Android

- Generieren Sie ein neues Wallet
- Gehen Sie zu den Einstellungen und wählen unter Netzwerk KOVAN aus
- Gehen Sie zurück zu dem Wallet und tippen auf receive

EINFÜHRUNG IN DIE BLOCKCHAIN-TECHNOLOGIE | ETHEREUM

3. TECHNISCH – HANDS ON

1. Klicke Add Token -> Add custom Token
2. Scan diesen QR-Code für die Adresse
3. Nennen Sie ihn Fun Token
4. Symbol = FUN
5. Decimals = 0



- Jetzt kriegen Sie ein paar Ether von mir, senden Sie diese oder einen Bruchteil an den Contract
- Senden Sie sich untereinander die neuen Fun Token zu

REMIX?
LOTTERIE?

VIELEN DANK FÜR IHRE AUFMERKSAMKEIT!

Fragen?

Ideen?

Anmerkungen?

QUELLEN

<https://de.wikipedia.org/wiki/Kryptographie>

<https://cryptomonday.de/wie-funktioniert-verschluesselung/>

https://de.wikipedia.org/wiki/Asymmetrisches_Kryptosystem

<https://github.com/topics/bitcoin>

<https://de.wikipedia.org/wiki/Peer-to-Peer>

<https://bitcoin.org/de/wie-es-funktioniert>

<https://bitcoin.org/bitcoin.pdf>

<https://de.wikipedia.org/wiki/Ethereum>

N. Koblitz and A. J. Menezes, Another Look at Provable Security, July 4, 2004

T. Gerstner, Monte Carlo Simulationen, 2010