Technische Universität Braunschweig

Institut für Betriebssysteme und Rechnerverbund

SEP Distributed Systems

IBRCoin: Eine ressourcenschonende Alternative zu Bitcoin

Signe Rüsch, Nico Weichbrodt

April 6, 2018

Table of Contents

Organisatorisches

- Wöchentliche Treffen
  - Ihr präsentiert den aktuellen Status (Probleme, Lösungen, Plan, . . .)
    ⇒ Terminplanung heute!
- SELPstorganisation
  - Wir (Betreuer) sind eure Kunden, ihr die Entwickler
  - Ihr organisiert euch in der Gruppe selbst
  - Jeder soll ungefähr gleich viel beitragen
  - **Jeder auf allen Gebieten!**
    (Einer programmiert der andere schreibt Text geht **nicht!**)

Organisatorisches (2)

- Mailingliste: `sepds@ibr.cs.tu-bs.de`
- Technischer Support:
  - **Signe Rüsch**, Raum 116
    sepds@ibr.cs.tu-bs.de
  - **Nico Weichbrodt**, Raum 116
    sepds@ibr.cs.tu-bs.de
- Dokumente:
  - **Jana Rehbein**
    sepds@ibr.cs.tu-bs.de
- Redmine:

  `https://sep.isf.cs.tu-bs.de/redmine/projects/18-ibr_ds_0`

Organisatorisches (3)

- Abgaben der Dokumente
  - in LaTeX, Vorlagen kommen vom ISF
  - Pre-Abgabe Review von Jana
    1. ins SVN hochladen
    2. über Mailingliste Review einfordern
  - ISF
    - **NUR** im Redmine
    - **Jeder** ist verantwortlich
  - Immer und **NUR** ins SVN hochladen
    Nicht Email, Dropbox, GoogleDrive, USB Sticks, Github, ...
  - „Aber git/hg/... ist soviel besser als SVN ... “
    → beschwert euch beim ISF

Das Angebot

- Eure Zusammenfassung der Aufgabenstellung,
  so wie ihr sie verstanden habt.
- Inklusive Projektablauf, Rahmenbedingungen, Richtlinien,
  Projektorganisation.

- Abgabe am 18.04.2018, spätestens 23:59:59 CEST bei uns per Mail.

Dokumente

- ISF-Deadlines der Abgaben mittwochs
- interne Deadlines:
  - 1. Möglichkeit: eine Woche vorher, Feedback bis Freitag
  - 2. Möglichkeit: bis Montag Morgen, Feedback bis Dienstag Abend
- wenn bis Montag Morgens keine Anfrage eingegangen ist, gibt es auch kein Feedback!
- Fragen zu Dokumenten können jederzeit gestellt werden

Dokumente - zeitlicher Ablauf

ISF	intern	Dokument
18.04.	**11.04.**	Angebot
09.05.	**02.05.**	Pflichtenheft & Abnahmetestspezifikation
30.05.	**23.05.**	Fachentwurf
20.06.	**13.06.**	Technischer Entwurf
04.07.	**27.06.**	Testdokumenation
12.07.		TDSE

Organisatorisches – Termin

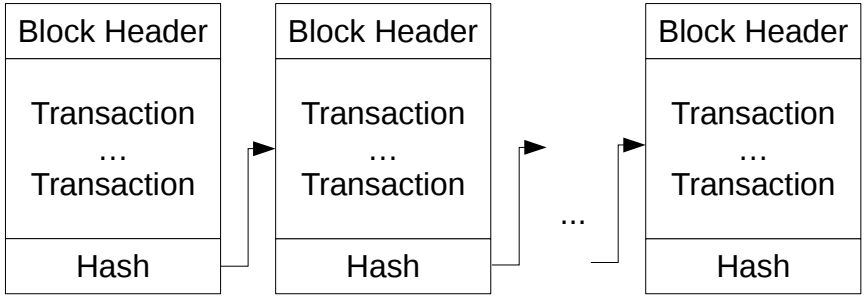
	Mo	Di	Mi	Do	Fr
10:00 - 10:30					
10:30 - 11:00					
11:00 - 11:30					
11:30 - 12:00					
12:00 - 12:30					
12:30 - 13:00					
13:00 - 13:30					
13:30 - 14:00					
14:00 - 14:30					
14:30 - 15:00					
15:00 - 15:30					
15:30 - 16:00					
16:00 - 16:30					
16:30 - 17:00					
17:00 - 17:30					

Table of Contents

Themenvorstellung

What is a blockchain?

- Blöcke mit Transaktionen
- Jeder Block beinhaltet Hash vom vorherigen
- Streng geordnete Nachrichten
- Unmodifizierbare Blöcke
- Oft Krypto-Währungen, z. B. Bitcoin



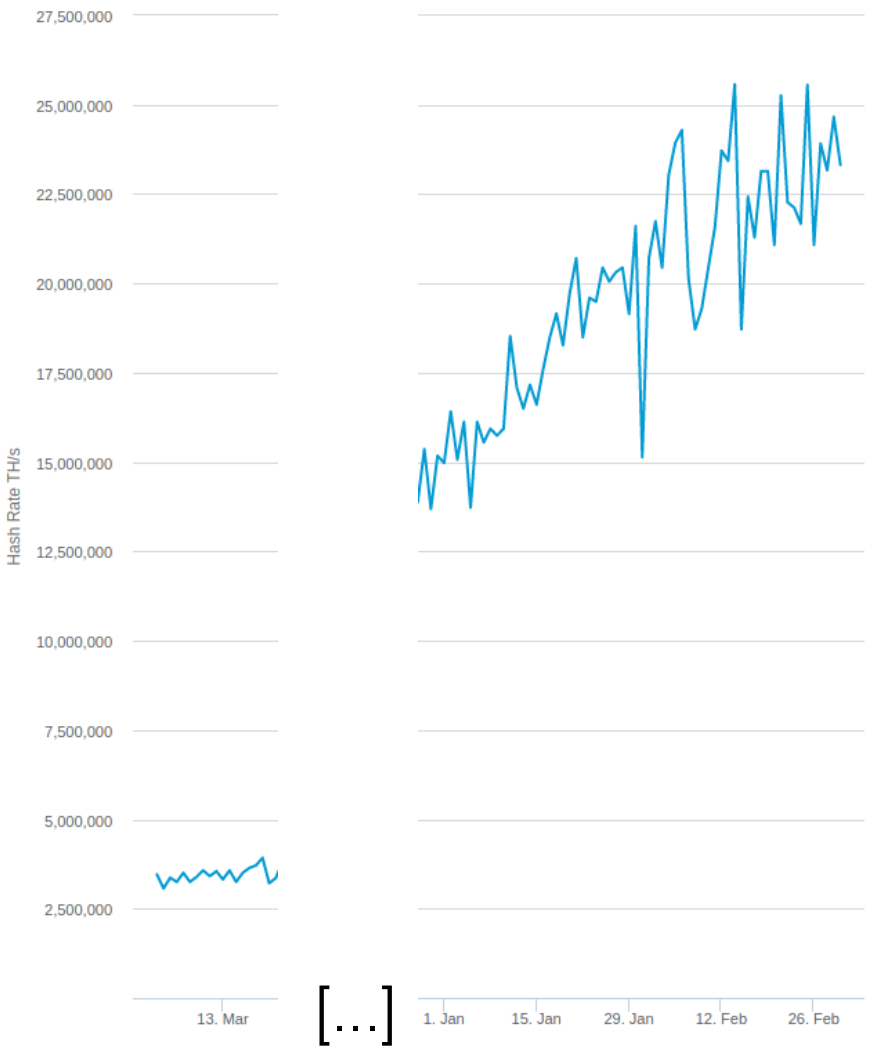
[Bessani et al., 2017]

Themenvorstellung

Proof-of-Work Mining

- Bitcoin Mining hat größeren Energieverbrauch als Irland
- Eine Stunde Confirmation Time
- Zentralisierung durch spezielle Hardware

→ Alternativen



Themenvorstellung

Alternativen

- Proof-of-Stake: Währungsanteil
- Proof-of-Burn
- Protokolle mit Byzantinischer Fehlertoleranz
- ...

Themenvorstellung

Alternativen

- Proof-of-Stake: Währungsanteil
- Proof-of-Burn
- Protokolle mit Byzantinischer Fehlertoleranz
- ...

→ **Proof-of-Luck**

Themenvorstellung

Proof-of-Luck

- Jeder Teilnehmer erstellt einen Block und eine Zufallszahl
- Teilnehmer warten mit Veröffentlichung, Dauer basiert auf Zahl
  - Größere Zahlen = mehr „Luck“ = schneller veröffentlicht
- Wenn ein neuer Block mit mehr „Luck“ empfangen wird, wird eigener verworfen
- Effizient und praktikabel durch Intel SGX:
  - wird in abgesicherten Bereich im Arbeitsspeicher ausgeführt
  - verhindert Sybil Angriffe und Zentralisierung
- Mehr Details im Paper: Milutinovic et al., „Proof of Luck: An Efficient Blockchain Consensus Protocol“, SysTEX '16, `https://eprint.iacr.org/2017/249.pdf`

Table of Contents

SGX in a Nutshell

- Generelle Idee: Vertraue niemanden außer dir selbst
  - Nicht dem OS, anderer Software, ...
- Wie erreicht man das?

SGX in a Nutshell

- Generelle Idee: Vertraue niemanden außer dir selbst
  - Nicht dem OS, anderer Software, ...
- Wie erreicht man das?
- Intel SGX: Isolation von Programmteilen in Enklaven
- Enklaven sind ...
  - ... Teil einer normalen Anwendung
  - ... nicht manipulierbar (auf Code Ebene)
  - ... verifizierbar

SGX in a Nutshell

- Enklavencode kann nicht direkt angesprungen werden
- Betreten/Verlassen durch spezielle Instruktionen
- Normale Anwendungen können Enklaven Speicher nicht auslesen
  - Enklaven können aber auf allen Speicher zugreifen
- Schutz vor physischem Zugriff: Enklaven Speicher ist verschlüsselt
- Ich kann mir verifizieren lassen, dass meine unveränderte Enklave läuft (Attestation)
- Technische Details zu SGX in der SGX Programming Reference

Arbeiten mit Enklaven

- Intel SGX SDK für Windows / Linux
- `https://01.org`
- Intel bietet Beispiel Enklaven an
- SDK erzeugt automatisch Wrapper für Enklaven Beitritt/Verlassen
- SDK Details in der SGX Developer Reference

Fachbegriffe:

- Calls in die Enklave rein: ECall
- Calls aus der Enklave nach außen: OCall
- (Un)trusted Runtime System: urts / trts
- Trusted Standard C(++) Library: tstdc (tstdcxx)

Aufgabe: My 1st Enclave

1. SDK installieren
   `https://download.01.org/intel-sgx/linux-1.9/`
   Unsere Rechner haben v1.9 installiert
2. SampleEnclave kompilieren, ausführen & verstehen
3. Abwandlungen/Eigene Funktionen
   - Key-Value-Store
   - Hash erstellen
   - Verschlüsseln
   - ...

Bis zum nächsten Treffen baut jeder mal eine Enklave.