Blockchain Engineering

Lightning (Blitz)

Dr. Lars Brünjes

GEFÖRDERT VOM
Bundesministerium
für Bildung
und Forschung

MODULARES INNOVATIVES
NETZWERK FÜR DURCHLÄSSIGKEIT

AUFSTIEG DURCH
BILDUNG
OFFENE HOCHSCHULEN

17. Oktober 2019

- Problem
  - Bei bestehenden Kryptowährungen muss jede Transaktion auf allen Knoten gespeichert werden.
  - Der Konsens-Algorithmus schränkt die Rate ein, mit der neue Blöcke erzeugt werden können. Außerdem sind Blöcke in ihrer Größe beschränkt.
  - Dies führt dazu, dass die Transaktions-Rate von Kryptowährungen im Vergleich zu zentralisierten Netzen wie Visa eher bescheiden ist (um Größenordnungen geringer).

- Problem
  - Bei bestehenden Kryptowährungen muss jede Transaktion auf allen Knoten gespeichert werden.
  - Der Konsens-Algorithmus schränkt die Rate ein, mit der neue Blöcke erzeugt werden können. Außerdem sind Blöcke in ihrer Größe beschränkt.
  - Dies führt dazu, dass die Transaktions-Rate von Kryptowährungen im Vergleich zu zentralisierten Netzen wie Visa eher bescheiden ist (um Größenordnungen geringer).
- Lightning
  - Das Lightning- (Blitz-) Netz will dieses Problem lösen.
  - Es ist zunächst für Bitcoin geplant, aber die Idee ist so generisch, dass sie prinzipiell für viele Kryptowährungen anwendbar ist.
  - Die Grundidee ist, die Knoten zu entlasten, indem parallele “Seitenkanäle” einen Großteil der Zahlungen abwickeln.

- Die Grundidee von Lightning ist, dass Parteien, die einander Zahlungen senden wollen, einen Payment Channel (Zahlungskanal) einrichten, der unabhängig vom Bitcoin-Netz ist.
- Über diesen Kanal können sehr schnell und günstig beliebig viele Zahlungen getätigt werden.
- Im Normalfall, wenn beide Parteien sich an die Regeln halten, müssen nur *zwei* Transaktionen an die Bitcoin-Blockchain gesendet werden, eine zur Eröffnung des Kanals und eine zum Schließen des Kanals.
- Trotzdem sind alle Zahlungen über den Kanal sicher und werden von der Blockchain garantiert.

- Die Grundidee von Lightning ist, dass Parteien, die einander Zahlungen senden wollen, einen Payment Channel (Zahlungskanal) einrichten, der unabhängig vom Bitcoin-Netz ist.
- Über diesen Kanal können sehr schnell und günstig beliebig viele Zahlungen getätigt werden.
- Im Normalfall, wenn beide Parteien sich an die Regeln halten, müssen nur *zwei* Transaktionen an die Bitcoin-Blockchain gesendet werden, eine zur Eröffnung des Kanals und eine zum Schließen des Kanals.
- Trotzdem sind alle Zahlungen über den Kanal sicher und werden von der Blockchain garantiert.
- Nachdem wir verstanden haben, wie ein solcher Kanal zwischen zwei Parteien funktioniert, werden wir sehen, wie das System erweitert werden kann, um Zahlungen zwischen Parteien zu erlauben, die keinen direkten Kanal besitzen.

- Nehmen wir an, Alice und Bob wollen einen Zahlungskanal zwischen sich einrichten, weil sie sich häufig Bitcoin überweisen.
- Sie eröffnen den Kanal, indem sie beide einen bestimmten Betrag, zum Beispiel jeder 5 ₿, auf der Blockchain deponieren (falls Alice erwartet, in Zukunft mehr an Bob zu zahlen als von ihm zu erhalten, könnte sie auch mehr als Bob deponieren).
- Dabei gehen sie wie folgt vor:

Einrichten eines Kanals

- Nehmen wir an, Alice und Bob wollen einen Zahlungskanal zwischen sich einrichten, weil sie sich häufig Bitcoin überweisen.
- Sie eröffnen den Kanal, indem sie beide einen bestimmten Betrag, zum Beispiel jeder 5 ₿, auf der Blockchain deponieren (falls Alice erwartet, in Zukunft mehr an Bob zu zahlen als von ihm zu erhalten, könnte sie auch mehr als Bob deponieren).
- Dabei gehen sie wie folgt vor:

1. Sie richten eine 2-von-2-Multisig-Adresse ein und bereiten ein Transaktion vor, die ihre Anteile an diese Adresse schickt, unterschreiben die Transaktion aber noch nicht.

- Nehmen wir an, Alice und Bob wollen einen Zahlungskanal zwischen sich einrichten, weil sie sich häufig Bitcoin überweisen.
- Sie eröffnen den Kanal, indem sie beide einen bestimmten Betrag, zum Beispiel jeder 5 ₿, auf der Blockchain deponieren (falls Alice erwartet, in Zukunft mehr an Bob zu zahlen als von ihm zu erhalten, könnte sie auch mehr als Bob deponieren).
- Dabei gehen sie wie folgt vor:

1. Sie richten eine 2-von-2-Multisig-Adresse ein und bereiten ein Transaktion vor, die ihre Anteile an diese Adresse schickt, unterschreiben die Transaktion aber noch nicht.
2. Jeder wählt eine zufällige Zahl, ein Geheimnis, und schickt dessen Hash an den anderen.

- Nehmen wir an, Alice und Bob wollen einen Zahlungskanal zwischen sich einrichten, weil sie sich häufig Bitcoin überweisen.
- Sie eröffnen den Kanal, indem sie beide einen bestimmten Betrag, zum Beispiel jeder 5 ₿, auf der Blockchain deponieren (falls Alice erwartet, in Zukunft mehr an Bob zu zahlen als von ihm zu erhalten, könnte sie auch mehr als Bob deponieren).
- Dabei gehen sie wie folgt vor:

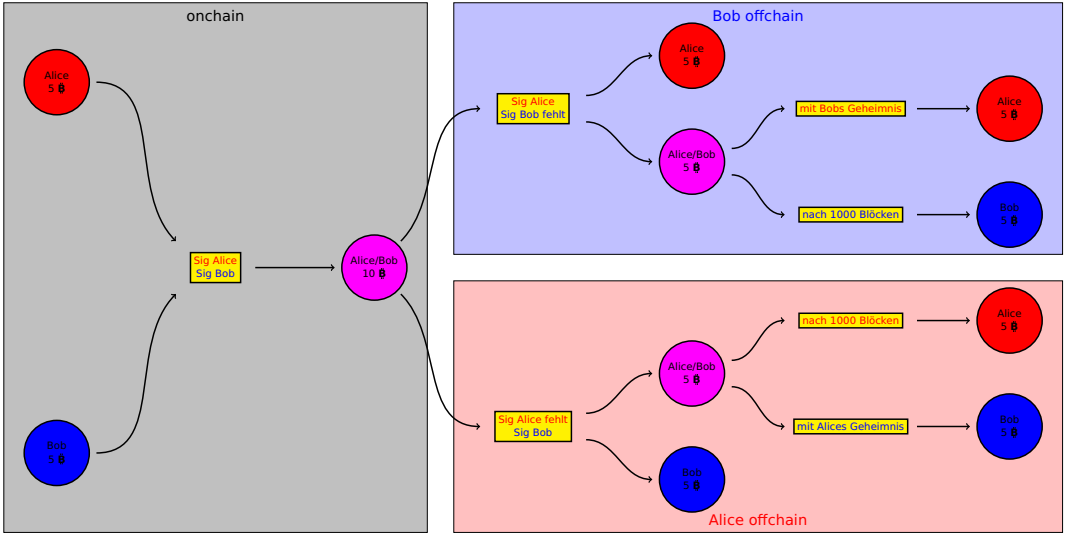
1. Sie richten eine 2-von-2-Multisig-Adresse ein und bereiten ein Transaktion vor, die ihre Anteile an diese Adresse schickt, unterschreiben die Transaktion aber noch nicht.
2. Jeder wählt eine zufällige Zahl, ein Geheimnis, und schickt dessen Hash an den anderen.
3. Alice erzeugt, unterschreibt und gibt Bob eine Transaktion mit der Multisig-Adresse als Input und zwei Outputs — ihr Anteil an sich selbst, Bobs Anteil an eine neue Multisig-Adresse, die *entweder* nach 1000 Blöcken von Bob *oder* sofort von Alice, sofern sie Bobs Geheimnis kennt, geleert werden kann.
4. Bob erzeugt, unterschreibt und gibt Alice eine Transaktion mit der Multisig-Adresse als Input und zwei Outputs — sein Anteil an ihn selbst, Alices Anteil an eine neue Multisig-Adresse, die *entweder* nach 1000 Blöcken von Alice *oder* sofort von Bob, sofern er Alices Geheimnis kennt, geleert werden kann.

- Nehmen wir an, Alice und Bob wollen einen Zahlungskanal zwischen sich einrichten, weil sie sich häufig Bitcoin überweisen.
- Sie eröffnen den Kanal, indem sie beide einen bestimmten Betrag, zum Beispiel jeder 5 ₿, auf der Blockchain deponieren (falls Alice erwartet, in Zukunft mehr an Bob zu zahlen als von ihm zu erhalten, könnte sie auch mehr als Bob deponieren).
- Dabei gehen sie wie folgt vor:

1. Sie richten eine 2-von-2-Multisig-Adresse ein und bereiten ein Transaktion vor, die ihre Anteile an diese Adresse schickt, unterschreiben die Transaktion aber noch nicht.
2. Jeder wählt eine zufällige Zahl, ein Geheimnis, und schickt dessen Hash an den anderen.
3. Alice erzeugt, unterschreibt und gibt Bob eine Transaktion mit der Multisig-Adresse als Input und zwei Outputs — ihr Anteil an sich selbst, Bobs Anteil an eine neue Multisig-Adresse, die *entweder* nach 1000 Blöcken von Bob *oder* sofort von Alice, sofern sie Bobs Geheimnis kennt, geleert werden kann.
4. Bob erzeugt, unterschreibt und gibt Alice eine Transaktion mit der Multisig-Adresse als Input und zwei Outputs — sein Anteil an ihn selbst, Alices Anteil an eine neue Multisig-Adresse, die *entweder* nach 1000 Blöcken von Alice *oder* sofort von Bob, sofern er Alices Geheimnis kennt, geleert werden kann.
5. Alice und Bob tauschen die Transaktionen aus Schritten 3 und 4 aus.

- Nehmen wir an, Alice und Bob wollen einen Zahlungskanal zwischen sich einrichten, weil sie sich häufig Bitcoin überweisen.
- Sie eröffnen den Kanal, indem sie beide einen bestimmten Betrag, zum Beispiel jeder 5 ₿, auf der Blockchain deponieren (falls Alice erwartet, in Zukunft mehr an Bob zu zahlen als von ihm zu erhalten, könnte sie auch mehr als Bob deponieren).
- Dabei gehen sie wie folgt vor:

1. Sie richten eine 2-von-2-Multisig-Adresse ein und bereiten ein Transaktion vor, die ihre Anteile an diese Adresse schickt, unterschreiben die Transaktion aber noch nicht.
2. Jeder wählt eine zufällige Zahl, ein Geheimnis, und schickt dessen Hash an den anderen.
3. Alice erzeugt, unterschreibt und gibt Bob eine Transaktion mit der Multisig-Adresse als Input und zwei Outputs — ihr Anteil an sich selbst, Bobs Anteil an eine neue Multisig-Adresse, die *entweder* nach 1000 Blöcken von Bob *oder* sofort von Alice, sofern sie Bobs Geheimnis kennt, geleert werden kann.
4. Bob erzeugt, unterschreibt und gibt Alice eine Transaktion mit der Multisig-Adresse als Input und zwei Outputs — sein Anteil an ihn selbst, Alices Anteil an eine neue Multisig-Adresse, die *entweder* nach 1000 Blöcken von Alice *oder* sofort von Bob, sofern er Alices Geheimnis kennt, geleert werden kann.
5. Alice und Bob tauschen die Transaktionen aus Schritten 3 und 4 aus.
6. Alice und Bob unterschreiben die Transaktion aus Schritt-1 und schicken sie an die Blockchain.


onchain
Alice
5 ₿
Bob
5 ₿
Sig Alice
Sig Bob
Alice/Bob
10 ₿
Bob offchain
Sig Alice
Sig Bob fehlt
Alice
5 ₿
Alice/Bob
5 ₿
mit Bobs Geheimnis
Alice
5 ₿
nach 1000 Blöcken
Bob
5 ₿
Alice offchain
Sig Alice fehlt
Sig Bob
Alice/Bob
5 ₿
nach 1000 Blöcken
Alice
5 ₿
mit Alices Geheimnis
Bob
5 ₿
Bob
5 ₿

- Dieses komplizierte Arrangement garantiert, dass Alice und Bob wieder an ihr Geld kommen können, egal, was passiert.
- Unter normalen Umständen, wenn sich beide einig sind, können sie eine normale 2-von-2-Multisig-Transaktion an die Blockchain schicken, die von beiden unterschrieben ist und beiden ihren Anteil zurück zahlt.

- Dieses komplizierte Arrangement garantiert, dass Alice und Bob wieder an ihr Geld kommen können, egal, was passiert.
- Unter normalen Umständen, wenn sich beide einig sind, können sie eine normale 2-von-2-Multisig-Transaktion an die Blockchain schicken, die von beiden unterschrieben ist und beiden ihren Anteil zurück zahlt.

- Wenn Alice ohne Bobs Hilfe ihren Anteil zurück bekommen möchte, unterschreibt sie die Transaktion, die Bob ihr gegeben hat, und schickt diese an die Blockchain.
  - Bob bekommt seinen Anteil unmittelbar zurück.
  - Alice muss 1000 Blöcke warten, bis sie an ihren Anteil kommt.
  - Falls Bob in der Zwischenzeit an Alices Geheimnis kommt, kann er sich Alices Anteil holen. (Wir werden später sehen, wozu das gut ist.)

- Dieses komplizierte Arrangement garantiert, dass Alice und Bob wieder an ihr Geld kommen können, egal, was passiert.
- Unter normalen Umständen, wenn sich beide einig sind, können sie eine normale 2-von-2-Multisig-Transaktion an die Blockchain schicken, die von beiden unterschrieben ist und beiden ihren Anteil zurück zahlt.

- Wenn Bob ohne Alices Hilfe seinen Anteil zurück bekommen möchte, unterschreibt er die Transaktion, die Alice ihm gegeben hat, und schickt diese an die Blockchain.
  - Alice bekommt ihren Anteil unmittelbar zurück.
  - Bob muss 1000 Blöcke warten, bis er an seinen Anteil kommt.
  - Falls Alice in der Zwischenzeit an Bob Geheimnis kommt, kann sie sich Bobs Anteil holen. (Wir werden später sehen, wozu das gut ist.)

- Nach all diesem Aufwand wollen Alice und Bob ihren nagelneuen Zahlungskanal benutzen, um Zahlungen zu tätigen!
- Wenn Alice 1 ฿ an Bob schicken will, gehen die beiden wie folgt vor:

- Nach all diesem Aufwand wollen Alice und Bob ihren nagelneuen Zahlungskanal benutzen, um Zahlungen zu tätigen!
- Wenn Alice 1 ฿ an Bob schicken will, gehen die beiden wie folgt vor:

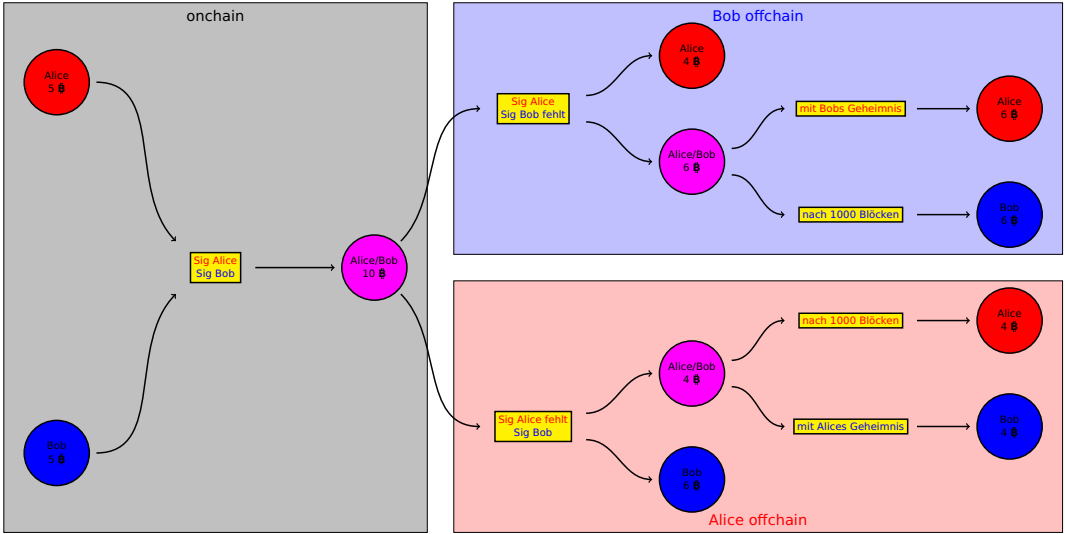
1. Jeder wählt ein neues Geheimnis und schickt dessen Hash an den anderen.

- Nach all diesem Aufwand wollen Alice und Bob ihren nagelneuen Zahlungskanal benutzen, um Zahlungen zu tätigen!
- Wenn Alice 1 ₿ an Bob schicken will, gehen die beiden wie folgt vor:

1. Jeder wählt ein neues Geheimnis und schickt dessen Hash an den anderen.
2. Alice erzeugt, unterschreibt und gibt Bob eine Transaktion mit der Multisig-Adresse als Input und zwei Outputs — 4 ₿ an sich selbst, 6 ₿ an eine neue Multisig-Adresse, die *entweder* nach 1000 Blöcken von Bob *oder* sofort von Alice, sofern sie Bobs *neues* Geheimnis kennt, geleert werden kann.
3. Bob erzeugt, unterschreibt und gibt Alice eine Transaktion mit der Multisig-Adresse als Input und zwei Outputs — 6 ₿ an sich selbst, 4 ₿ an eine neue Multisig-Adresse, die *entweder* nach 1000 Blöcken von Alice *oder* sofort von Bob, sofern er Alices *neues* Geheimnis kennt, geleert werden kann.

- Nach all diesem Aufwand wollen Alice und Bob ihren nagelneuen Zahlungskanal benutzen, um Zahlungen zu tätigen!
- Wenn Alice 1 ฿ an Bob schicken will, gehen die beiden wie folgt vor:

1. Jeder wählt ein neues Geheimnis und schickt dessen Hash an den anderen.
2. Alice erzeugt, unterschreibt und gibt Bob eine Transaktion mit der Multisig-Adresse als Input und zwei Outputs — 4 ฿ an sich selbst, 6 ฿ an eine neue Multisig-Adresse, die *entweder* nach 1000 Blöcken von Bob *oder* sofort von Alice, sofern sie Bobs *neues* Geheimnis kennt, geleert werden kann.
3. Bob erzeugt, unterschreibt und gibt Alice eine Transaktion mit der Multisig-Adresse als Input und zwei Outputs — 6 ฿ an sich selbst, 4 ฿ an eine neue Multisig-Adresse, die *entweder* nach 1000 Blöcken von Alice *oder* sofort von Bob, sofern er Alices *neues* Geheimnis kennt, geleert werden kann.
4. Die beiden tauschen ihre alten Geheimnisse aus.


onchain
Alice
5 ₿
Bob
5 ₿
Sig Alice
Sig Bob
Alice/Bob
10 ₿
Bob offchain
Sig Alice
Sig Bob fehlt
Alice
4 ₿
Alice/Bob
6 ₿
mit Bobs Geheimnis
Alice
6 ₿
nach 1000 Blöcken
Bob
6 ₿
Alice offchain
Sig Alice fehlt
Sig Bob
Alice/Bob
4 ₿
Bob
6 ₿
nach 1000 Blöcken
Alice
4 ₿
mit Alices Geheimnis
Bob
4 ₿

- Unter normalen Umständen, wenn sich beide einig sind, können sie später eine normale 2-von-2-Multisig-Transaktion an die Blockchain schicken, die von beiden unterschrieben ist und Alice 4 ฿ und Bob 6 ฿ auszahlt.

- Unter normalen Umständen, wenn sich beide einig sind, können sie später eine normale 2-von-2-Multisig-Transaktion an die Blockchain schicken, die von beiden unterschrieben ist und Alice 4 ฿ und Bob 6 ฿ auszahlt.

- Wie zuvor kann Alice ohne Bobs Hilfe ihre 4 ฿ zurück bekommen, wenn sie die neue Transaktion, die sie von Bob bekommen hat, an die Blockchain schickt.
- Falls sie versucht, Bobs alte Transaktion einzulösen, muss sie 1000 Blöcke auf ihre 5 ฿ warten, aber Bob kennt jetzt ihr altes Geheimnis und kann vorher alles für sich selbst nehmen. Bobs alte Transaktion ist daher jetzt wertlos für Alice.

- Unter normalen Umständen, wenn sich beide einig sind, können sie später eine normale 2-von-2-Multisig-Transaktion an die Blockchain schicken, die von beiden unterschrieben ist und Alice 4 ฿ und Bob 6 ฿ auszahlt.

- Wie zuvor kann Bob ohne Alices Hilfe seine 6 ฿ zurück bekommen, wenn er die neue Transaktion, die er von Alice bekommen hat, an die Blockchain schickt.
- Er hat kein Interesse daran, Alices alte Transaktion einzulösen, weil diese ihm nur 5 ฿ anstelle von 6 ฿ geben würde. Darüberhinaus könnte Alice in diesem Fall alles für sich selbst nehmen, da sie jetzt Bobs altes Geheimnis kennt. Alices alte Transaktion ist daher jetzt wertlos für Bob.

- Auf diese Weise können Alice und Bob einander beliebig viele Zahlungen schicken (sofern die Billanz den auf der Blockchain ursprünglich deponierten Betrag nicht überschreitet).
- Solange sich beide an die Regeln halten, ist keine dieser Transaktionen auf der Blockchain sichtbar. Die Kommunikation zwischen Alice und Bob findet parallel zur Blockchain statt, ist “blitzschnell” und (so gut wie) umsonst. Die Zahlungen sind also viel schneller und viel günstiger als gewöhnliche Bitcoin Transaktionen.
- Zu jedem Zeitpunkt ist Alices und Bobs Geld sicher. Sie können jederzeit die aktuelle Transaktion des anderen an die Blockchain schicken, um ihr Geld (nach 1000 Blöcken) zurück zu bekommen.
- Wenn sie sich einig sind, den Kanal später schließen zu wollen, können sie dies mit einer gewöhnlichen 2-von-2-Multisig-Transaktion tun. Im Normalfall sind also nur zwei “echte” Bitcoin-Transaktionen nötig, eine, um den Kanal zu eröffnen, und eine, um ihn am Ende wieder zu schließen.

Zahlungen ohne direkten Kanal

- Lightning ermöglicht auch Zahlungen an Personen, mit denen man keinen direkten Zahlungskanal unterhält.

Zahlungen ohne direkten Kanal

- Lightning ermöglicht auch Zahlungen an Personen, mit denen man keinen direkten Zahlungskanal unterhält.
- Nehmen wir an, Alice will 1 ₿ an Charlie senden, ohne einen Zahlungskanal mit ihm einzurichten. Nehmen wir weiter an, dass Bob und Charlie einen Zahlungskanal besitzen.

Zahlungen ohne direkten Kanal

- Lightning ermöglicht auch Zahlungen an Personen, mit denen man keinen direkten Zahlungskanal unterhält.
- Nehmen wir an, Alice will 1 ₿ an Charlie senden, ohne einen Zahlungskanal mit ihm einzurichten. Nehmen wir weiter an, dass Bob und Charlie einen Zahlungskanal besitzen.
- Die Idee ist, die Kanäle Alice-Bob und Bob-Charlie zu benutzen, um Alice ihre Zahlung an Charlie tätigen zu lassen.

- Lightning ermöglicht auch Zahlungen an Personen, mit denen man keinen direkten Zahlungskanal unterhält.
- Nehmen wir an, Alice will 1 ₿ an Charlie senden, ohne einen Zahlungskanal mit ihm einzurichten. Nehmen wir weiter an, dass Bob und Charlie einen Zahlungskanal besitzen.
- Die Idee ist, die Kanäle Alice-Bob und Bob-Charlie zu benutzen, um Alice ihre Zahlung an Charlie tätigen zu lassen.
- Alice, Bob und Charlie gehen dabei wie folgt vor:

- Lightning ermöglicht auch Zahlungen an Personen, mit denen man keinen direkten Zahlungskanal unterhält.
- Nehmen wir an, Alice will 1 ₿ an Charlie senden, ohne einen Zahlungskanal mit ihm einzurichten. Nehmen wir weiter an, dass Bob und Charlie einen Zahlungskanal besitzen.
- Die Idee ist, die Kanäle Alice-Bob und Bob-Charlie zu benutzen, um Alice ihre Zahlung an Charlie tätigen zu lassen.
- Alice, Bob und Charlie gehen dabei wie folgt vor:

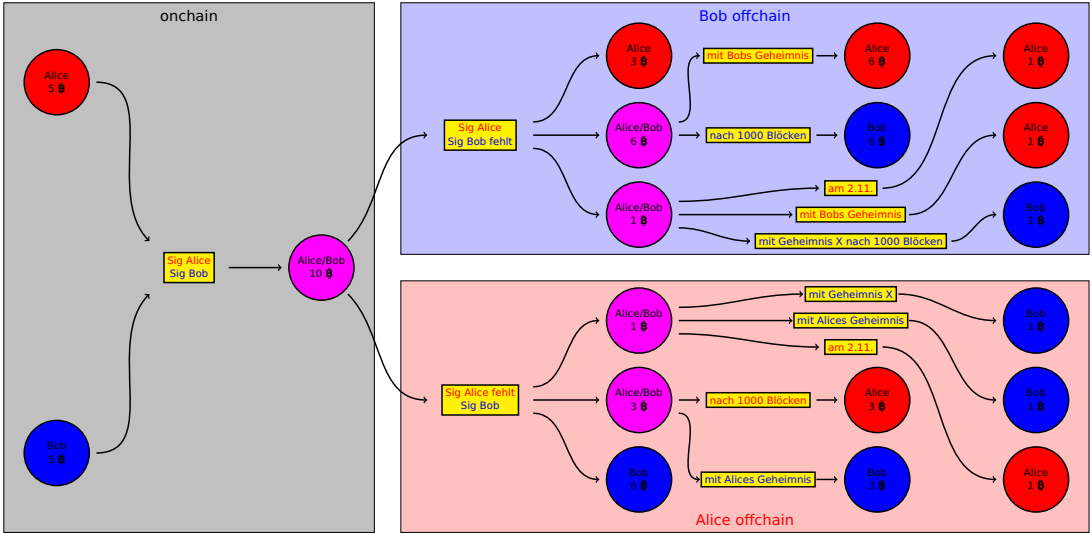
1. Alice bittet Charlie, ein Geheimnis X zu erzeugen und ihr dessen Hash zu schicken.

- Lightning ermöglicht auch Zahlungen an Personen, mit denen man keinen direkten Zahlungskanal unterhält.
- Nehmen wir an, Alice will 1 ₿ an Charlie senden, ohne einen Zahlungskanal mit ihm einzurichten. Nehmen wir weiter an, dass Bob und Charlie einen Zahlungskanal besitzen.
- Die Idee ist, die Kanäle Alice-Bob und Bob-Charlie zu benutzen, um Alice ihre Zahlung an Charlie tätigen zu lassen.
- Alice, Bob und Charlie gehen dabei wie folgt vor:

1. Alice bittet Charlie, ein Geheimnis X zu erzeugen und ihr dessen Hash zu schicken.
2. Bob benutzt seinen Kanal mit Charlie, um Charlie 1 ₿ im Austausch für Geheimnis X zu bezahlen.

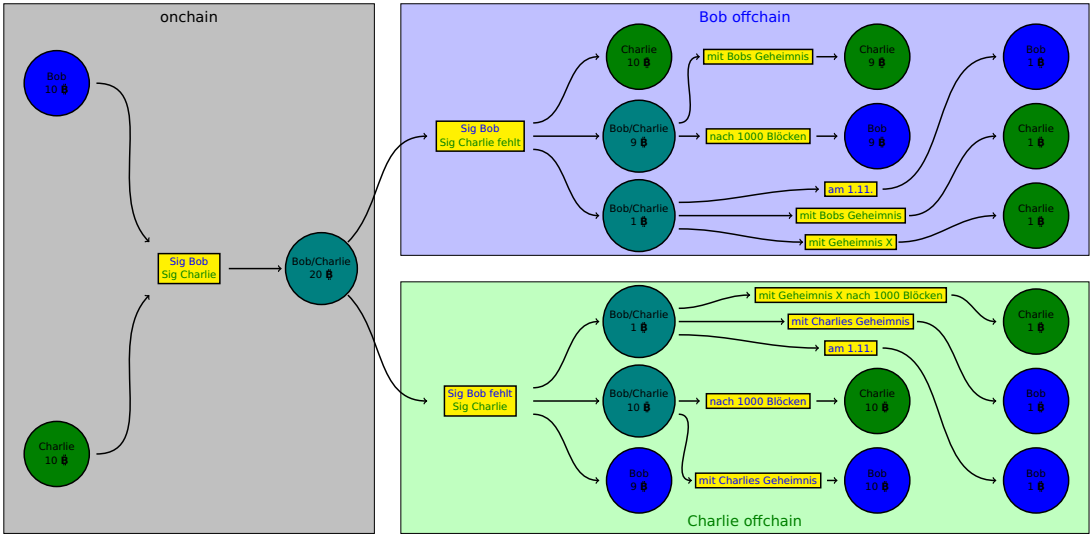
- Lightning ermöglicht auch Zahlungen an Personen, mit denen man keinen direkten Zahlungskanal unterhält.
- Nehmen wir an, Alice will 1 ₿ an Charlie senden, ohne einen Zahlungskanal mit ihm einzurichten. Nehmen wir weiter an, dass Bob und Charlie einen Zahlungskanal besitzen.
- Die Idee ist, die Kanäle Alice-Bob und Bob-Charlie zu benutzen, um Alice ihre Zahlung an Charlie tätigen zu lassen.
- Alice, Bob und Charlie gehen dabei wie folgt vor:

1. Alice bittet Charlie, ein Geheimnis X zu erzeugen und ihr dessen Hash zu schicken.
2. Bob benutzt seinen Kanal mit Charlie, um Charlie 1 ₿ im Austausch für Geheimnis X zu bezahlen.
3. Alice benutzt ihren Kanal mit Bob, um Bob 1 ₿ im Austausch für Geheimnis X zu bezahlen.
4. In Schritten 2 und 3 werden — ähnlich zum Vorgehen bei direkten Zahlungen — spezielle Hash Time-Locked Contracts (HTLCs) benutzt. Diese benutzen absolute, keine relativen Zeitschlösser.


onchain
Alice 5 ₿
Bob 5 ₿
Sig Alice Sig Bob
Alice/Bob 10 ₿
Bob offchain
Sig Alice Sig Bob fehlt
Alice 3 ₿
Alice/Bob 6 ₿
Alice/Bob 1 ₿
mit Bobs Geheimnis
nach 1000 Blöcken
am 2.11.
mit Bobs Geheimnis
mit Geheimnis X nach 1000 Blöcken
Alice 6 ₿
Bob 6 ₿
Alice 1 ₿
Alice 1 ₿
Bob 1 ₿
Alice offchain
Sig Alice fehlt Sig Bob
Alice/Bob 1 ₿
Alice/Bob 3 ₿
Bob 6 ₿
mit Geheimnis X
mit Alices Geheimnis
am 2.11.
nach 1000 Blöcken
mit Alices Geheimnis
Alice 3 ₿
Bob 3 ₿
Bob 1 ₿
Bob 1 ₿
Alice 1 ₿

- Die Teile, die von der Zahlung von 1 ₿ nicht betroffen sind, sind wie zuvor.
- Für die Zahlung von 1 ₿ wird in beiden Transaktionen eine neue Multisig-Adresse eingerichtet, die auf drei verschiedene Arten geleert werden kann:
  - Wenn Bob Geheimnis X kennt und unterschreibt, bekommt er das Geld. Allerdings muss er zusätzlich 1000 Blöcke warten, wenn er derjenige ist, der den Kanal schließt. Wenn er diese Option wählt, ist Geheimnis X öffentlich sichtbar auf der Blockchain.
  - Derjenige, der den Kanal nicht schließt, kann an das Geld, wenn er das Geheimnis des anderen kennt. Wie zuvor dient dies dazu, veraltete Transaktionen unbrauchbar zu machen.
  - Unabhängig davon, wer den Kanal schließt, kann Alice am 2. November ihr Geld zurück bekommen.

Hash Time-Locked Contracts — Kanal von Bob und Charlie



- Die Situation ist analog zu der zwischen Alice und Bob.
- Allerdings ist der Termin, zu dem Bob seine 1 ฿ zurück bekommen kann, vor dem Termin, zu dem Alice *ihre* 1 ฿ zurück bekommen kann, damit Bob Zeit hat, sich das Geld von Alice zu sichern, sobald Charlie Geheimnis X offenbart hat.

- Die Situation ist analog zu der zwischen Alice und Bob.
- Allerdings ist der Termin, zu dem Bob seine 1 ₿ zurück bekommen kann, vor dem Termin, zu dem Alice *ihre* 1 ₿ zurück bekommen kann, damit Bob Zeit hat, sich das Geld von Alice zu sichern, sobald Charlie Geheimnis X offenbart hat.

Bemerkung

Natürlich sind auf gleiche Weise auch Transaktionen über mehr als eine Zwischenstation möglich. Es muss nur darauf geachtet werden, die Termine der Zeitschlösser so zu staffeln, dass sie Parteien weiter hinten in der Kette genug Zeit geben, zu reagieren.

- Die Situation ist analog zu der zwischen Alice und Bob.
- Allerdings ist der Termin, zu dem Bob seine 1 ฿ zurück bekommen kann, vor dem Termin, zu dem Alice *ihre* 1 ฿ zurück bekommen kann, damit Bob Zeit hat, sich das Geld von Alice zu sichern, sobald Charlie Geheimnis X offenbart hat.

Bemerkung

Wie bei der direkten Zahlung über einen Zahlungskanal müssen auch hier Transaktionen im Normalfall nie an die Blockchain geschickt werden. Solange sich alle Parteien an die Regeln halten, geschieht alles offline.

- Skalierung ist eine der größten Herausforderungen in der Blockchain-Technologie: Zentralisierte Netze wie Visa sind um Größenordnungen schneller als derzeitige Blockchain-Systeme.
- Bitcoin Lightning ist eine mögliche Lösung dieses Problems, und die Idee ist allgemein genug, um auf viele Blockchains übertragbar zu sein.
- Bei Lightning richten Parteien bidirektionale Zahlungskanäle zwischen sich ein und können dann miteinander und über “Kanalketten” auch mit anderen Transaktionen austauschen.
- Solange sich alle an die Regeln halten, sind nur zwei Bitcoin-Transaktionen pro Kanal nötig, eine zur Eröffnung des Kanals und eine, um ihn wieder zu schließen. Alle anderen Transaktionen können schnell und günstig “offchain” durchgeführt werden.
- Bitcoin garantiert die Sicherheit des Systems: Bei Regelverletzungen verliert keine ehrliche Partei ihr Geld.

Hinweis

Diese Publikation wurde im Rahmen des vom Bundesministerium für Bildung und Forschung (BMBF) geförderten Bund- Länder- Wettbewerbs “Aufstieg durch Bildung: offene Hochschulen” erstellt. Die in dieser Publikation dargelegten Ergebnisse und Interpretationen liegen in der alleinigen Verantwortung der Autor/innen.