

<b>Vorwort</b> .....	<b>17</b>
<b>Vorwort zur deutschen Ausgabe</b> .....	<b>25</b>

---

## Teil I Das Lightning-Netzwerk verstehen

<b>1 Einführung</b> .....	<b>31</b>
Grundlegende Konzepte des Lightning-Netzwerks .....	32
Vertrauen in dezentralisierten Netzwerken .....	33
Fairness ohne zentrale Autorität .....	34
Vertrauenswürdige Protokolle ohne Intermediäre .....	35
Ein Fairness-Protokoll in Aktion .....	36
Sicherheits-Primitive als Grundbausteine .....	37
Beispiel für Fairness-Protokolle .....	38
Motivation für das Lightning-Netzwerk .....	39
Blockchains skalieren .....	40
Die das Lightning-Netzwerk definierenden Eigenschaften .....	42
Anwendungsfälle, Nutzer und ihre Geschichten .....	42
Fazit .....	43
<b>2 Erste Schritte</b> .....	<b>45</b>
Alice' erste Lightning-Wallet .....	45
Lightning-Nodes .....	46
Lightning-Explorer .....	46
Lightning-Wallets .....	47
Testnet-Bitcoin .....	50
Balance zwischen Komplexität und Kontrolle .....	51
Download und Installation einer Lightning-Wallet .....	52
Eine neue Wallet anlegen .....	53
Verantwortung für die Schlüsselverwahrung .....	53
Mnemonische Wörter .....	53
Die mnemonische Phrase sichern .....	54

Bitcoin auf die Wallet laden . . . . .	55
Bitcoin beschaffen . . . . .	55
Bitcoin empfangen . . . . .	55
Von Bitcoin zum Lightning-Netzwerk . . . . .	59
Lightning-Netzwerk-Kanäle . . . . .	59
Einen Lightning-Kanal öffnen . . . . .	61
Eine Tasse Kaffee über das Lightning-Netzwerk kaufen . . . . .	63
Bobs Café . . . . .	63
Eine Lightning-Rechnung . . . . .	64
Fazit . . . . .	66
<b>3 Wie das Lightning-Netzwerk funktioniert . . . . .</b>	<b>67</b>
Was ist ein Zahlungskanal? . . . . .	68
Grundlagen eines Zahlungskanals . . . . .	68
Zahlungen über Kanäle routen . . . . .	69
Zahlungskanäle . . . . .	70
Multisignaturadressen . . . . .	71
Funding-Transaktion . . . . .	71
Commitment-Transaktionen . . . . .	73
Betrug mit vorherigem Zustand . . . . .	74
Ankündigung des Kanals . . . . .	77
Den Kanal schließen . . . . .	78
Rechnungen . . . . .	82
Zahlungshash und Preimage . . . . .	83
Zusätzliche Metadaten . . . . .	84
Zustellung der Zahlung . . . . .	85
Das Peer-to-Peer-Gossip-Protokoll . . . . .	85
Wegfindung und Routing . . . . .	86
Quellbasierte Wegfindung . . . . .	87
Onion-Routing . . . . .	88
Algorithmus zur Zahlungsverweiterung . . . . .	90
Verschlüsselung der Peer-to-Peer-Kommunikation . . . . .	91
Überlegungen zum Vertrauen . . . . .	92
Vergleich mit Bitcoin . . . . .	93
Adressen versus Rechnungen, Transaktionen versus Zahlungen . . . . .	93
Outputs wählen versus Pfad finden . . . . .	94
Change-Output bei Bitcoin versus kein Wechselgeld bei Lightning . . . . .	94
Mining-Gebühren versus Routing-Gebühren . . . . .	95
Gebühren basierend auf Traffic versus angekündigte Gebühren . . . . .	95
Öffentliche Bitcoin-Transaktionen versus private Lightning-Zahlungen . . . . .	96
Warten auf Bestätigungen versus sofortiger Zahlungseingang . . . . .	96

Senden beliebiger Summen versus Kapazitätsbeschränkungen . . . .	97
Anreize für Großbeträge versus Kleinbeträge . . . . .	97
Blockchain als Kassenbuch versus Blockchain als	
Gerichtssystem . . . . .	98
Offline versus online, asynchron versus synchron . . . . .	98
Satoshis versus Millisatoshis . . . . .	99
Gemeinsamkeiten von Bitcoin und Lightning . . . . .	99
Monetäre Einheit . . . . .	99
Unumkehrbarkeit und Finalität von Zahlungen . . . . .	99
Vertrauen und Gegenparteirisiko . . . . .	100
Genehmigungsfreier Betrieb . . . . .	100
Open Source und Open System . . . . .	100
Fazit . . . . .	100
<b>4 Lightning-Node-Software . . . . .</b>	<b>101</b>
Die Lightning-Entwicklungsumgebung . . . . .	102
Die Kommandozeile nutzen . . . . .	102
Das Buch-Repository herunterladen . . . . .	103
Docker-Container . . . . .	103
Bitcoin Core und Regtest . . . . .	105
Den Bitcoin-Core-Container erzeugen . . . . .	106
Das c-lightning-Lightning-Node-Projekt . . . . .	109
c-lightning als Docker-Container . . . . .	109
Ein Docker-Netzwerk einrichten . . . . .	110
Ausführen der bitcoind- und c-lightning-Container . . . . .	110
c-lightning aus dem Quellcode installieren . . . . .	112
Vorab benötigte Bibliotheken und Pakete installieren . . . . .	112
Den c-lightning-Quellcode kopieren . . . . .	113
Den c-lightning-Quellcode kompilieren . . . . .	113
Das Lightning-Netzwerk-Daemon-Node-Projekt . . . . .	115
Der LND-Docker-Container . . . . .	116
Die bitcoind- und LND-Container ausführen . . . . .	117
LND aus dem Quellcode installieren . . . . .	118
Den LND-Quellcode kopieren . . . . .	119
Den LND-Quellcode kompilieren . . . . .	119
Das Eclair-Lightning-Node-Projekt . . . . .	120
Der Eclair-Docker-Container . . . . .	120
Die bitcoind- und Eclair-Container ausführen . . . . .	121
Eclair aus dem Quellcode installieren . . . . .	123
Den Eclair-Quellcode installieren . . . . .	123
Den Eclair-Quellcode kompilieren . . . . .	124

Ein Netzwerk diverser Lightning-Nodes aufbauen. . . . .	124
docker-compose zur Orchestrierung von Docker-Containern nutzen . . . . .	125
docker-compose-Konfiguration . . . . .	125
Das Demo-Lightning-Netzwerk starten. . . . .	126
Kanäle öffnen und eine Zahlung routen . . . . .	127
Fazit . . . . .	129
<b>5 Eine Lightning-Netzwerk-Node betreiben. . . . .</b>	<b>131</b>
Eine Plattform wählen . . . . .	132
Warum ist Zuverlässigkeit für den Betrieb einer Lightning-Node wichtig? . . . . .	132
Hardware für Lightning-Nodes . . . . .	133
Betrieb in der Cloud. . . . .	133
Eine Node zu Hause betreiben. . . . .	134
Welche Hardware wird zum Betrieb einer Lightning-Node benötigt? . . . . .	135
Serverkonfiguration in der Cloud wechseln. . . . .	136
Einen Installer oder Helfer nutzen . . . . .	137
RaspiBlitz . . . . .	137
Mynode . . . . .	138
Umbrel . . . . .	138
BTCPay Server . . . . .	140
Bitcoin-Node oder leichtgewichtige Lightning-Node . . . . .	140
Wahl des Betriebssystems . . . . .	141
Wahl der Lightning-Node-Implementierung . . . . .	142
Eine Bitcoin- oder Lightning-Node installieren . . . . .	143
Hintergrunddienste . . . . .	143
Prozessisolation . . . . .	144
Node starten. . . . .	144
Node-Konfiguration. . . . .	146
Netzwerkkonfiguration . . . . .	147
Die Sicherheit Ihrer Node . . . . .	151
Betriebssystemsicherheit . . . . .	152
Node-Zugriff . . . . .	153
Node- und Kanal-Backups . . . . .	154
Hot-Wallet-Risiko . . . . .	156
Sweeping von Guthaben . . . . .	156
Uptime und Verfügbarkeit einer Lightning-Node . . . . .	159
Fehler tolerieren und die Dinge automatisieren. . . . .	159
Die Node-Verfügbarkeit überwachen . . . . .	160
Watchtower . . . . .	161

Kanalmanagement . . . . .	162
Ausgehende Kanäle öffnen . . . . .	162
Eingehende Liquidität beschaffen . . . . .	166
Kanäle schließen . . . . .	167
Kanäle wieder ausgleichen . . . . .	167
Routing-Gebühren . . . . .	168
Node-Management . . . . .	169
Ride The Lightning . . . . .	170
lndmon . . . . .	170
ThunderHub . . . . .	170
Fazit . . . . .	171

---

## Teil II Das Lightning-Netzwerk im Detail

<b>6 Architektur des Lightning-Netzwerks . . . . .</b>	<b>175</b>
Die Lightning-Netzwerk-Protokoll-Suite . . . . .	175
Lightning im Detail . . . . .	176
<b>7 Zahlungskanäle . . . . .</b>	<b>179</b>
Eine andere Art, das Bitcoin-System zu nutzen . . . . .	180
Bitcoin-Eigentümerschaft und -Kontrolle . . . . .	181
Diversität der (unabhängigen) Eigentümerschaft und Multisig . . . . .	181
Gemeinsamer Besitz ohne unabhängige Kontrolle . . . . .	182
»Gespernte« und nicht einzulösende Bitcoin verhindern . . . . .	182
Einen Zahlungskanal aufbauen . . . . .	182
Private und öffentliche Node-Schlüssel . . . . .	183
Node-Netzwerkadresse . . . . .	183
Node-Kennungen . . . . .	183
Nodes als direkte Peers verbinden . . . . .	184
Den Kanal aufbauen . . . . .	184
Peer-Protokoll für das Kanalmanagement . . . . .	185
Nachrichtenfluss beim Aufbau des Kanals . . . . .	185
Die Funding-Transaktion . . . . .	188
Eine Multisignaturadresse generieren . . . . .	189
Die Funding-Transaktion konstruieren . . . . .	189
Signierte Transaktionen halten, aber nicht veröffentlichen . . . . .	190
Refunding vor Funding . . . . .	190
Die vorsignierte Refund-Transaktion konstruieren . . . . .	190
Transaktionen ohne Veröffentlichung verketteten . . . . .	191
(Ver-)Formbarkeit von Transaktionen (Segregated Witness) . . . . .	192
Die Funding-Transaktion veröffentlichen . . . . .	194

Zahlungen über den Kanal senden . . . . .	195
Das Guthaben aufteilen . . . . .	195
Konkurrierende Commitments . . . . .	196
Mit alten Commitment-Transaktionen betrügen. . . . .	197
Alte Commitment-Transaktionen widerrufen. . . . .	197
Asymmetrische Commitment-Transaktionen . . . . .	198
Verzögerte Auszahlung von to_self (Timelock). . . . .	199
Widerrufsschlüssel. . . . .	200
Die Commitment-Transaktion . . . . .	201
Den Kanalzustand verändern . . . . .	203
Die commitment_signed-Nachricht. . . . .	204
Die revoke_and_ack-Nachricht . . . . .	204
Widerruf und neuer Commit. . . . .	205
Betrug und Sanktionierung in der Praxis . . . . .	205
Die Kanalreserve: das Spiel am Laufen halten . . . . .	208
Den Kanal schließen (kooperatives Schließen) . . . . .	209
Die shutdown-Nachricht . . . . .	209
Die closing_signed-Nachricht . . . . .	210
Die Kooperatives-Schließen-Transaktion. . . . .	211
Fazit . . . . .	212
<b>8 Routing in einem Netzwerk aus Zahlungskanälen . . . . .</b>	<b>213</b>
Routing einer Zahlung . . . . .	213
Routing versus Wegfindung . . . . .	215
Ein Netzwerk von Zahlungskanälen aufbauen . . . . .	215
Ein reales Beispiel für das Routing . . . . .	216
Fairness-Protokoll. . . . .	222
Implementierung atomarer vertrauensfreier Multihop-	
Zahlungen . . . . .	222
Ein erneuter Blick auf das Spenden-Beispiel. . . . .	223
On-Chain- versus Off-Chain-Abwicklung von HTLCs . . . . .	224
Hash Time-Locked Contracts. . . . .	225
HTLCs in Bitcoin-Skript . . . . .	226
Zahlungs-Preimage und Hashverifikation . . . . .	227
HTLCs von Alice zu Dina propagieren . . . . .	227
Rückpropagation des Secrets. . . . .	228
Signaturbindung: Diebstahl von HTLCs verhindern . . . . .	230
Hashoptimierung . . . . .	232
Kooperations- und Timeout-Fehler . . . . .	233
Kürzere Timelocks . . . . .	234
Fazit . . . . .	235

<b>9 Kanalbetrieb und Zahlungsverweiterung</b> .....	<b>237</b>
Lokal (einzelner Kanal) versus geroutet (mehrere Kanäle) .....	238
Zahlungen weiterleiten und Commitments aktualisieren mit HTLCs ...	238
Nachrichtenfluss für HTLC und Commitment .....	239
Zahlungen mit HTLCs weiterleiten .....	239
Einen HTLC hinzufügen .....	239
Die update_add-HTLC-Nachricht .....	240
HTLC in Commitment-Transaktionen .....	241
Neues Commitment mit HTLC-Output .....	242
Alice' Commit .....	242
Bob bestätigt das neue Commitment und widerruft das alte .....	244
Bobs Commit .....	246
Mehrere HTLCs .....	248
Den HTLC einhalten .....	249
HTLC-Propagation .....	249
Dina erfüllt den HTLC mit Chan .....	249
Bob verrechnet den HTLC mit Alice .....	250
Einen HTLC aufgrund eines Fehlers oder Verfallsdatums entfernen ...	253
Eine lokale Zahlung vornehmen .....	254
Fazit .....	255
<b>10 Onion-Routing</b> .....	<b>257</b>
Ein anschauliches Beispiel des Onion-Routings .....	258
Einen Pfad wählen .....	258
Die Schichten aufbauen .....	259
Die Schichten abschälen .....	261
Einführung in das Onion-Routing von HTLCs .....	262
Alice wählt den Pfad .....	262
Alice konstruiert die Nutzdaten .....	264
Schlüsselgenerierung .....	267
Die Onion-Schichten verpacken .....	271
Onions fester Länge .....	271
Die Onion verpacken (Zusammenfassung) .....	272
Dinas Hop-Nutzdaten verpacken .....	273
Chans Hop-Nutzdaten verpacken .....	277
Bobs Hop-Nutzdaten verpacken .....	278
Das finale Onion-Paket .....	279
Die Onion senden .....	280
Die update_add_htlc-Nachricht .....	280
Alice sendet die Onion an Bob .....	280
Bob überprüft die Onion .....	281
Bob generiert Füllbytes .....	281

Bob »entschleiert« seine Hop-Nutzdaten. . . . .	282
Bob extrahiert den äußeren HMAC für den nächsten Hop. . . . .	283
Bob entfernt seine Nutzdaten und verschiebt die Onion-Daten nach links. . . . .	283
Bob konstruiert das neue Onion-Paket . . . . .	284
Bob verifiziert die HTLC-Details . . . . .	284
Bob sendet update_add_htlc an Chan. . . . .	284
Chan leitet die Onion weiter . . . . .	285
Dina empfängt die finalen Nutzdaten . . . . .	286
Fehler zurückgeben. . . . .	286
Fehlermeldungen . . . . .	287
Spontane Zahlungen per keysend. . . . .	289
Benutzerdefinierte Onion-TLV-Records . . . . .	290
keysend-Zahlungen senden und empfangen . . . . .	290
Keysend und benutzerdefinierte Records in Lightning-Anwendungen . . . . .	291
Fazit . . . . .	291
<b>11 Gossip und der Kanal-Graph. . . . .</b>	<b>293</b>
Peers entdecken . . . . .	296
P2P-Bootstrapping . . . . .	296
DNS-Bootstrapping . . . . .	297
SRV-Query-Optionen . . . . .	300
Der Kanal-Graph. . . . .	301
Ein gerichteter Graph. . . . .	301
Gossip-Protokoll-Nachrichten . . . . .	302
Die node_announcement-Nachricht . . . . .	303
Die channel_announcement-Nachricht. . . . .	305
Die channel_update-Nachricht . . . . .	309
Fortlaufende Pflege des Kanal-Graphen . . . . .	310
Fazit . . . . .	310
<b>12 Wegfindung und Zustellung der Zahlung . . . . .</b>	<b>311</b>
Wegfindung in der Lightning-Protokoll-Suite . . . . .	311
Wo ist das BOLT? . . . . .	312
Wegfindung: Welches Problem lösen wir? . . . . .	312
Wahl des besten Pfads . . . . .	313
Wegfindung in Mathematik und Informatik . . . . .	313
Kapazität, Guthaben, Liquidität . . . . .	314
Unsicherheit der Guthaben . . . . .	314
Komplexität der Wegfindung . . . . .	315
Die Dinge einfach halten . . . . .	316
Wegfindung und Zahlungszustellung. . . . .	316



Aufbau des Kanal-Graphen . . . . .	317
Liquidität: Unsicherheit und Wahrscheinlichkeit . . . . .	320
Gebühren und andere Kanalmetriken . . . . .	322
Pfadcandidaten finden. . . . .	323
Zustellung der Zahlung (Versuch-und-Irrtum-Schleife) . . . . .	324
Erster Versuch (Pfad #1) . . . . .	324
Zweiter Versuch (Pfad #4) . . . . .	325
Multipart-Zahlungen . . . . .	326
MPP nutzen. . . . .	327
Versuch und Irrtum über mehrere »Runden«. . . . .	328
Fazit. . . . .	330
<b>13 Wire-Protokoll: Framing und Erweiterbarkeit . . . . .</b>	<b>331</b>
Messaging-Schicht in der Lightning-Protokoll-Suite . . . . .	331
Wire-Framing . . . . .	332
High-Level-Wire-Framing. . . . .	332
Typcodierung . . . . .	333
Type-Length-Value-Nachrichtenerweiterungen . . . . .	334
Das Protocol-Buffers-Nachrichtenformat . . . . .	334
Vor- und Rückwärtskompatibilität. . . . .	334
Type-Length-Value-Format . . . . .	335
BigSize-Integercodierung . . . . .	336
Beschränkungen der TLV-Codierung. . . . .	336
Kanonische TLV-Codierung. . . . .	336
Feature-Bits und Erweiterbarkeit des Protokolls. . . . .	337
Feature-Bits als Mechanismus zur Erkennung von Upgrades. . . . .	337
TLV für Vor- und Rückwärtskompatibilität . . . . .	339
Taxonomie des Upgrade-Mechanismus . . . . .	339
Updates auf Ebene der Kanalkonstruktion. . . . .	341
Fazit. . . . .	341
<b>14 Lightnings verschlüsselter Nachrichtentransport . . . . .</b>	<b>343</b>
Verschlüsselter Transport in der Lightning-Protokoll-Suite . . . . .	343
Einführung. . . . .	344
Der Kanal-Graph als dezentralisierte Public-Key-Infrastruktur. . . . .	344
Warum nicht TLS?. . . . .	345
The Noise-Protokoll-Framework. . . . .	345
Lightnings verschlüsselter Transport im Detail. . . . .	346
Noise_XK: Noise-Handshake des Lightning-Netzwerks . . . . .	346
Handshake-Notation und Protokollfluss . . . . .	347
Übersicht auf hohem Niveau . . . . .	347
Handshake in drei Akten . . . . .	348
Fazit. . . . .	358

<b>15</b>	<b>Lightning-Zahlungsanforderungen</b> . . . . .	<b>359</b>
	Rechnungen in der Lightning-Protokoll-Suite . . . . .	359
	Einführung . . . . .	359
	Lightning-Rechnungen versus Bitcoin-Adressen . . . . .	360
	BOLT #11: Serialisierung und Interpretation von Lightning-Rechnungen . . . . .	361
	Rechnungscodierung in der Praxis . . . . .	361
	Das visuell lesbare Präfix . . . . .	361
	bech32 und das Datensegment . . . . .	362
	Fazit . . . . .	364
<b>16</b>	<b>Sicherheit und Privatsphäre im Lightning-Netzwerk</b> . . . . .	<b>365</b>
	Warum ist Privatsphäre wichtig? . . . . .	365
	Privatsphäre definieren . . . . .	365
	Evaluierung der Privatsphäre . . . . .	366
	Anonyme Menge . . . . .	367
	Privatsphäre: Unterschiede zwischen Lightning-Netzwerk und Bitcoin . . . . .	368
	Angriffe auf Lightning . . . . .	370
	Zahlungsbeträge beobachten . . . . .	370
	Sender und Empfänger verknüpfen . . . . .	370
	Kanalguthaben aufdecken (Probing) . . . . .	372
	Denial of Service . . . . .	374
	Commitment-Jamming . . . . .	376
	Einfrieren der Kanalliquidität . . . . .	376
	Schichtenübergreifende Deanonymisierung . . . . .	376
	Clustering von On-Chain-Bitcoin-Entitäten . . . . .	377
	Clustering von Off-Chain-Lightning-Nodes . . . . .	378
	Schichtenübergreifende Verknüpfung: Lightning-Nodes und Bitcoin-Entitäten . . . . .	379
	Lightning-Graph . . . . .	379
	Wie sieht der Lightning-Graph wirklich aus? . . . . .	380
	Zentralisierung im Lightning-Netzwerk . . . . .	382
	Ökonomische Anreize und Graph-Struktur . . . . .	383
	Praktischer Rat zum Schutz der Privatsphäre . . . . .	383
	Unangekündigte Kanäle . . . . .	383
	Routing-Erwägungen . . . . .	384
	Kanäle akzeptieren . . . . .	385
	Fazit . . . . .	386
	Quellenangaben und Literaturhinweise . . . . .	386

<b>17 Fazit</b> .....	<b>389</b>
Dezentralisierte und asynchrone Innovation .....	389
Innovationen im Bitcoin-Protokoll und bei Bitcoin-Skript .....	390
Innovationen des Lightning-Protokolls .....	390
TLV-Erweiterbarkeit .....	391
Konstruktion von Zahlungskanälen .....	391
Opt-in-Ende-zu-Ende-Features .....	391
Lightning-Anwendungen (LApps) .....	392
Auf die Plätze, fertig, los! .....	393
<b>Anhang A Bitcoin-Grundlagen</b> .....	<b>395</b>
<b>Anhang B Docker: Grundlegende Installation und Nutzung</b> .....	<b>415</b>
<b>Anhang C Nachrichten des Wire-Protokolls</b> .....	<b>419</b>
<b>Anhang D Quellen und Lizenzinformationen</b> .....	<b>437</b>
<b>Glossar</b> .....	<b>439</b>
<b>Index</b> .....	<b>457</b>