BRAINS Publishing

LIGHTNING NETWORK

PLATBY BUDOUCNOSTI

Autor: MICHAL NOVÁK
Předmluva: DOMINIK STROUKAL

Platby budoucnosti

BRAIINS

Česká firma působící na globální úrovni, která svými produkty posouvá bitcoin dopředu. Od operačního systému Braiins OS+ pro ASIC minery, nástroje Farm Proxy k agregaci jejich těžebního výkonu přes Braiins Pool – první těžební pool na světě – až po protokol Stratum V2, který zásadním způsobem vylepšuje infrastrukturu těžby bitcoinu.

PROJEKTY BRAIINS

BRAIINS OS+

FARM Proxy

STRATUM V2

BRAIINS POOL

Formerly **Slush Pool**

Tato kniha by nevznikla bez finanční podpory české a slovenské bitcoinové komunity. Předně chceme poděkovat největším patronům. První mezi nimi je **Tomáš Greif**, jemuž je věnována následující strana. Rozhodl se ji přenechat svým dcerám, které na svém obrázku znázornily zářnou budoucnost LN plateb :)

Druhý je tajemný podporovatel, který svůj příspěvek na knihu doplnil věnováním „From beyond, Hal Finney“. Jeho pseudonymita tak dokonale zapadá do konceptu bitcoinu: štědrým finančním příspěvkem pomohl knize na svět, zůstal skryt za svou přezdívkou, a jeho odkaz bude žít na stránkách fyzických výtisků knihy. Když se nám přece jenom podařilo se s ním spojit, ukázalo se, že se jmenuje **Jirka**.
Jeho přáním bylo, aby se v knize objevil neoficiální slogan bitcoinu ***Vires in Numeris***, protože v číslech je skutečně síla.

Třetím z patronů, který podobně jako předchozí dva mecenáši přispěl ke vzniku knihy nemalou částkou, je **Honza Přibyslavský**.

Vedle těchto velikánů knihu podpořily další stovky lidí z české i slovenské komunity a všem jim děkujeme.

Všichni jsme Satoshi!


KIKI & TERI

OBSAH

PŘEDMLUVA

Wow. Stáli jsme s Alexem Pilařem na Chaincampu 2022 v Ostravě a řeč přišla na knihu, kterou právě teď držíte v rukou a od které vás zdržuji. Wow, říkali jsme oba. Wow. Já se cítil tak trochu jako nekritický nadšenec, ale Alex mě přesvědčil, že jsem nic nepřehlédnul. To, co se právě chystáte přečíst, je extrémně dobré. Poučené, jednoduché, ale zároveň kompletní.

Jistě, není to pro každého. Spousta lidí říká, že chce pochopit Lightning Network (anebo vlastně Bitcoin jako takový), ale vlastně tím jenom oznamuje, že chce jen vědět, jak si něco koupit. Ne tomu rozumět. Vy tomu ale chcete rozumět, a proto jste tady. Proto jste otevřeli tuto knihu. A na lepší adrese nemůžete být.

Skutečně jsem se bál. Těch textů o Bitcoinu vzniklo za poslední roky tolik, že bylo těžké oddělovat zrna od plev a náhodou najít poklad. Přiznám se, že moje očekávání, když jsem v draftu knihy viděl zmínku o CBDC, byla extrémně malá. Sám se tomu tématu věnuju akademicky a je to natolik okrajová záležitost, že bych nikdy neměl nikomu za zlé, že danému tématu nerozumí. Jenže po otevření této knihy jsem žasl. Autor ví, o čem píše. Má nastudováno, rozumí tomu. Získal si mě natolik, abych začal znovu od první strany a nepřestal.

Nepíšu to jen jako chválu následujících řádků, ale hlavně jako obhajobu toho, že podobný text tu musel vzniknout, a pochvalu toho, že se těchto řádků ujal zrovna Michal Novák. Píše s lehkou rukou, ale zároveň nevynechá žádný detail. A považte, že v tom nejdůležitějším, co se dnes ve světě kryptoměn děje. Potřebujeme někoho, kdo jen nemluví a nepíše, ale používá to. A to používání je potřeba podtrhnout.

Nikdy jsme více než teď nepotřebovali, aby tu byl někdo, kdo ukazuje skutečné a nejlepší možné chování současné bitcoinové sítě. Na Bitcoin se ze všech stran valí kritika, tu za energetickou náročnost, tu za pomalost a drahé transakce, jindy za to, že nestíhá inovovat vedle jiných kryptoměn. Každá jedna stránka této knihy vás přesvědčí o opaku.

Kniha mě nadchla pro Bitcoin tak, jak jsem to naposledy cítil, když jsem poprvé vyzkoušel hardwarovou peněženku Trezor, což už je hodně let zpátky. O trošku lepší pocit jsem měl snad jen při čtení prvních informací o Bitcoinu před více než deseti lety.

Sám jsem byl vůči Lightning Network skeptik a pořád mám několik otazníků. Ale ty má i Michal Novák a není nijak přehnaně nadšený, respektive z textu to zbytečně nesálá. Je to popis reality. Je to návod, je to průvodce. Je to průvodce, jakého bych si přál, abych napsal sám, ale sám bych na to neměl.

Jste na dobré adrese. Užijte si čtení.

Dominik Stroukal
Metropolitní univerzita Praha
1. října 2022

ÚVOD

Jak již sám název knihy napovídá, Lightning Network se může stát eventuální budoucností plateb. I když je tato technologie stále na počátku a určitě ještě zabere nějaký čas, než dospěje do takového stavu, aby byla plnohodnotně použitelná pro běžnou veřejnost, už nyní je jasné, že může kompletně změnit to, jak na platby mezi dvěma stranami dnes pohlížíme, a přinést nové možnosti, které se současnými technologiemi nebyly možné.

A co že to vůbec Lightning Network je? Jedná se o decentralizovanou platební síť, která umožňuje rychle, levně a relativně anonymně platit za zboží a služby pomocí bitcoinu. Pro její využívání vám postačí peněženka ve formě mobilní aplikace, kterou pouze naskenujete příslušný QR kód u prodejce, a platba je během pár vteřin s poplatkem v úrovni haléřů dokončena. V poslední době se začínají rozvíjet i platby pomocí bezkontaktních karet s využitím technologie NFC. Jedná se tedy o velmi efektivní způsob používání bitcoinů jako prostředku směny. Při osobním studiu Lightning Network jsem však narazil na to, že neexistuje ucelený a komplexní popis této technologie v českém jazyce. Případný zájemce si musí jednotlivé informace doslova „vyzobávat" z různých článků, videí, přednášek a podobně. Tato kniha by právě to měla případným čtenářům usnadnit – přinést jim kompletní přehled o fungování Lightning Network jak po technické, tak po praktické stránce.

Rozhodl jsem se knihu rozdělit na dvě části – základní a pokročilou. Základní část, kterou pokrývají první čtyři kapitoly, se zaměřuje na vysvětlení této technologie takovým způsobem, aby ji pochopil i začátečník, který s Lightning Network zatím nemá žádné zkušenosti. Jsou k tomu využity různé analogie, které pomohou pochopit složitější části i netechnickým lidem. Naopak další kapitoly v pokročilé části jdou již více do hloubky a jsou určené primárně pro zkušenější čtenáře, kteří mají s Lightningem již nějaké zkušenosti a chtějí této technologii více porozumět, dále ji rozvíjet či ji využít ve svých aplikacích.

Jedna poznámka na závěr – Bitcoin je naprosto skvělá technologie, ale má jednu zásadní nevýhodu. Zabere velmi mnoho času (z mých zkušeností to jsou desítky či spíše stovky hodin), než ji člověk, který se s ní zatím nikdy nesetkal, pochopí a zjistí, co přináší a jak funguje se všemi souvislostmi. Z tohoto důvodu by bylo dobré, aby čtenáři měli alespoň základní povědomí o principech, smyslu a fungování Bitcoinu před samotným čtením této knihy. Pokud máte k Bitcoinu vysloveně negativní postoj, nevěříte mu a nevidíte v něm smysl, tak se obávám, že pro vás bude velmi těžké Lightning Network ocenit.

ZÁKLADNÍ PRINCIPY FUNGOVÁNÍ BITCOINU

Mít alespoň základní povědomí o fungování Bitcoinu je pro pochopení Lightning Network nutností. Pokud bychom chtěli Bitcoin probrat komplexně – tedy od historie peněz a finančních systémů, přes aktuální fungování světových ekonomik, účelu centrálních i komerčních bank, až po jeho detailní technickou specifikaci – zabralo by to stovky stran textu. My se ale v této kapitole zaměříme pouze na naprosté základy fungování Bitcoinu, jejichž znalost je nutná pro pochopení Lightning Network. Protože se zatím nacházíme v základní části knihy, většinu principů se zde budu snažit vysvětlovat lidsky a zjednodušeně. Pokud máte hluboké znalosti Bitcoinu, s klidem můžete tuto kapitolu přeskočit. Naopak, jestli si chcete svoje znalosti prohloubit nad rámec tohoto textu (což zajisté není na škodu), existuje dnes na toto téma řada knížek, videí, přednášek a článků, které rozebírají jednotlivá témata související s Bitcoinem více do hloubky. Základní povědomí o Bitcoinu je ale jakousi prerekvizitou, berte tedy tuto kapitolu spíše jako opakování a doplnění vašich znalostí.

Účty

Jestliže chcete Bitcoin používat (tedy ho vlastnit, přijímat a odesílat), musíte mít něco, co by se dalo nazvat účtem. Jedná se o dvojici – **adresa** a **klíč**. Adresa je veřejná informace (podobně jako e-mailová adresa, telefonní číslo nebo číslo bankovního účtu), kterou můžete sdílet ostatním. Naopak klíč (často zvaný soukromý či privátní) je tajná informace, kterou **nesmíte nikdy nikomu prozradit** – jako heslo do internetového bankovnictví, PIN k platební kartě apod. Příklad:

Účet	Adresa	Klíč
č. 1	1HJvVePCWLQ7xT2NEPpXifVh1CdzHrJwfx	8272029382726...9917371890
č. 2	3GKzR29LdyXg8Vao8Mni4MkyrTKXyeDbfN	3648194618304...2728193601
č. 3	bc1q7cez6lxmxkeftayurvallj4j0qznwylh8l5vnt	1827103912821...1737818719

Takovýchto účtů můžete mít několik. Ptáte se proč? Je to z důvodu zachování většího soukromí, kdy je běžnou praxí, že si pro každou transakci vytvoříte novou adresu, podobně jako kdybyste pro každý e-mail vytvořili novou schránku. Adres je extrémně

velké množství, takže i kdyby Bitcoin používali všichni lidé na planetě a pro každou transakci si vytvořili novou adresu, nikdy je za éru lidstva nevyčerpáme všechny.

Co je to vůbec ta zmíněná adresa? Nám bude zatím stačit informace, že je to řetězec písmen a číslic. Něco jako náhodně vygenerované uživatelské jméno. A klíč? Jedná se o číslo, které kdybychom zapsali v desítkové soustavě, tak má 78 číslic. I když to na první pohled možná tak nevypadá, vězte, že celkové množství klíčů je velmi podobné nejpřesnějším odhadům počtu atomů v celém vesmíru. Šance, že by někdo náhodou trefil váš klíč, je tedy tak extrémně malá, že ji lze prohlásit za nulovou.

Oproti analogii s elektronickou poštou (e-mailová adresa a heslo pro přístup do schránky) je zde jeden rozdíl. Adresu ani klíč si nemůžete změnit – jsou totiž matematicky provázané. Jedinou možností je vygenerovat si další dvojici. A nebojte, všechny tyto adresy a klíče (kterých můžete mít časem desítky i stovky) si nemusíte pamatovat. O to se postará tzv. bitcoinová peněženka, která vám nové účty generuje a pamatuje si všechny, jež jste kdy použili. Může být ve formě programu na stolní počítač, jako mobilní aplikace nebo speciální hardwarové zařízení. Vám poté stačí se do peněženky přihlásit a ta součtem zůstatků přes všechny adresy zjistí, kolik bitcoinů vlastníte.

> **Satoshi**
>
> Pokud pracujete s Bitcoinem, nemusíte vždy uvažovat v celých bitcoinech. Vzhledem k jeho ceně to pro běžné částky začíná být vcelku nepraktické. Bitcoin se dále dělí na tzv. satoshi (podle anonymního tvůrce Bitcoinu), podobně jako se koruna dělí na haléře. Jeden bitcoin lze rozdělit na 100 000 000 satoshi, zkráceně také sat. Pokud dlužíte kamarádovi za pivo, bude jednodušší uvažovat tak, že mu musíte poslat 5 000 satoshi než 0,000 05 bitcoinu.

Odesílání bitcoinů

Představme si nyní, že již nějaké bitcoiny vlastníme. Dejme tomu, že jsme velmi bohatí a máme 20 bitcoinů (dále také BTC). Rozhodneme se, že si koupíme nové auto a zaplatíme za něj prodejci 3 BTC – jak bude tato transakce probíhat? Nejprve si otevřeme naši peněženku a vytvoříme transakci, ve které budeme specifikovat, kolik bitcoinů (v našem případě 3) chceme odeslat, a na jakou adresu – tu nám sdělí prodejce. Nemusíme ji opisovat písmeno po písmenu, můžeme například použít i QR kód. Jakmile máme transakci připravenou, zkontrolujeme, že vše souhlasí,

a odešleme. Každá peněženka je nějakým způsobem napojena na **bitcoinový uzel**, kterému tuto transakci předá.

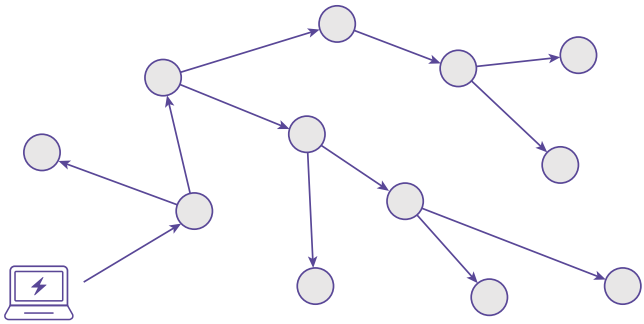
Bitcoinový uzel (též fullnode)

Jedná se o počítač, na kterém běží speciální software – Bitcoin. Kdokoliv si může tento program zdarma stáhnout na internetu, spustit a provozovat tak svůj vlastní uzel. Není potřeba žádné registrace ani speciálního povolení. Tyto programy spolu navzájem napřímo komunikují a dohromady tvoří bitcoinovou síť. Všechny uzly jsou si rovny, není zde žádná centrální autorita ani nikdo, kdo by síť řídil jako celek. Takovýchto uzlů jsou aktuálně desetitisíce a jsou distribuované po celém světě. I kdyby nastal výpadek většiny těchto uzlů (což je kvůli jejich decentralizované povaze málo pravděpodobné), bitcoinovou síť to nijak na funkčnosti neomezí. Tyto uzly zaručují funkčnost a bezpečnost celého bitcoinového protokolu. Provozovat vlastní uzel ale není nutné.

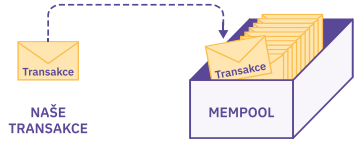
Co udělá první uzel, který takovouto transakci dostane? Ověří ji. Mimo jiné zkontroluje:

- Zda na naší adrese vlastníme alespoň tolik bitcoinů, kolik chceme odeslat.
- Zda vlastníme platný klíč k naší adrese a můžeme tedy s bitcoiny nakládat.
- Zda již tyto bitcoiny nebyly v minulosti utraceny.
- A také, jestli je naše transakce validní – tedy zda má správnou strukturu atd.

Pokud transakce nebude platná, tak ji uzel zahodí. Jestliže s ní je vše v pořádku, což je i náš případ, odešle ji všem dalším uzlům, se kterými je spojen. Tyto uzly provedou naprosto stejnou validaci veškerých parametrů a odešlou ji dále.



Funguje to podobně jako známé řetězové e-maily. Při předpokladu, že každý uzel je spojen například s 15 dalšími, tak v prvním kole se zpráva dostane z našeho PC k prvnímu uzlu. Ten ji odešle 15 sousedům. Každý z těchto sousedů ji odešle svým 15 sousedům a nyní o této transakci ví již $15^2 = 225$ uzlů. Takto bychom mohli pokračovat: $15^3 = 3\,375$, $15^4 = 50\,625$ atd. Díky tomuto síťovému efektu se zpráva rozšíří celou sítí velmi rychle, zpravidla během několika vteřin. Co s ní **každý uzel** udělá po ověření? Zjednodušeně řečeno si ji dočasně hodí do „košíku" (který se nazývá **mempool** – zkrácenina z memory pool) s dalšími transakcemi, které takto přijal.



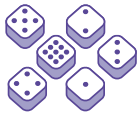
Poplatky

Zpočátku jsme vlastnili 20 bitcoinů, 3 jsme odeslali prodejci automobilů – kolik bitcoinů nám v peněžence zůstalo? Kdybyste takovýto příklad dali žákovi prvního stupně základní školy, tak by vám pravděpodobně odpověděl 17. Realita je však trochu jiná, v peněžence nám zbude například něco okolo 16,999 992 90 BTC. Co se stalo s našimi 710 satoshi? Ty posloužily jako transakční poplatek. Ten nebude vždy přesně 710 satoshi, ale v čase se mění – detaily si vysvětlíme později.

Těžba

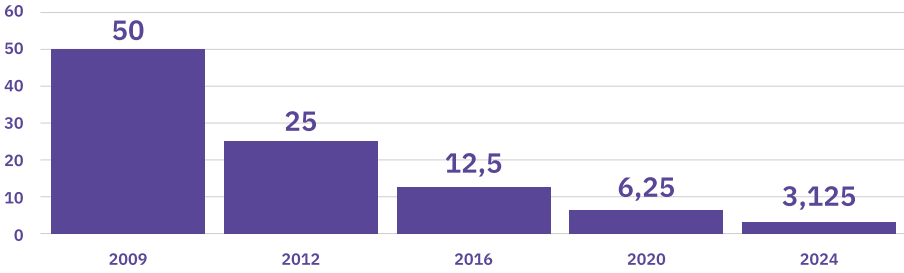
Co je to vlastně ta těžba bitcoinu, o které se všude mluví? Pokud bych chtěl být stručný, tak bych ji mohl definovat jako opakované počítání hashe hlavičky kandidujícího bloku, kdy neustále měníme nonce, časovou známku nebo kořen merkelova stromu tak, aby výsledná hodnota hashe byla nižší, než je požadováno aktuální obtížností. Tak to bychom měli, můžeme na další kapitolu. Ne, teď vážně – pokud této definici nerozumíte, což tak trochu předpokládám, zkusme si těžbu popsat nějak srozumitelněji a zjednodušeně.

Těžba je proces, při kterém určitou činnost opakuji neustále dokola, než splním předem dané podmínky. Jakmile tyto podmínky splním, což je velmi náročné, tak jsem vytěžil bitcoin. Zároveň ale platí, že verifikace splnění těchto podmínek je triviální.



Představme si to třeba tak, že nám někdo dá do velké misky 100 hracích šestistěnných kostek. My touto miskou několikrát zatřepeme, otočíme ji vzhůru dnem a vysypeme je na zem. Pokud **na všech 100 kostkách padla šestka**, tak jsme splnili dané podmínky, vyhráli, a tudíž vytěžili bitcoin. Je vám asi jasné, že tento proces není triviální a při takovémto počtu kostek ho budete muset opakovat opravdu hodněkrát. Po celém světě existuje mnoho těžařů, kteří takto neustále hází kostkami. Mají na to speciální zařízení, které dokáže provést miliony hodů za vteřinu, i tak se ale nelze k cíli dostat jinak než neustálým opakováním jednotlivých hodů. Naopak ověřit, že někdo vyhrál, je vcelku triviální – stačí se podívat na jeho kostky, jestli je na všech šestka.

Těžaři mezi sebou takto soupeří – co se stane v případě, že jeden vyhraje? Jako odměnu získá nové bitcoiny. Při vzniku Bitcoinu v roce 2009 to bylo 50 bitcoinů za „výhru“ a tato odměna se přibližně každé 4 roky snižuje na polovinu. Díky tomu víme, jaký bude celkový počet vytěžených bitcoinů (necelých 21 milionů) a jakým tempem se budou dostávat do oběhu. Zbývající části posledního bitcoinu se takto vytěží přibližně v roce 2140.

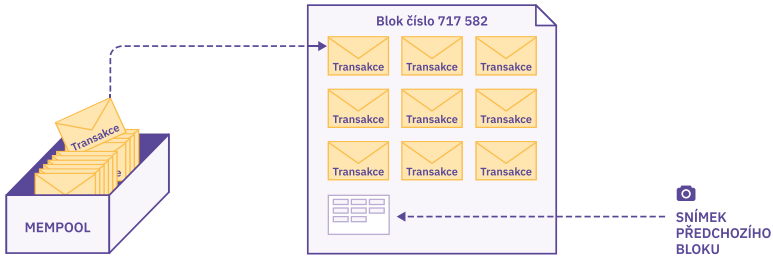


Ke skutečnosti, že se někomu podaří všemi kostkami hodit šestku, dochází přibližně jednou za 10 minut. Jak je to zaručeno? Vždyť je to přece naprosto náhodný proces. Co když bude házet kostkami najednou dvojnásobně více lidí? Tak by čistě teoreticky k výhře mělo docházet častěji. A obráceně, pokud budu házet na celém světě sám, případně jen s několika kamarády, tak je dost pravděpodobné, že se nám nepodaří mít na všech 100 kostkách šestku nikdy, nebo to bude trvat několik let. V Bitcoinu ale existuje tzv. **úprava obtížnosti**. Ta spočívá v tom, že se dynamicky přibližně každých 14 dní mění počet kostek tak, aby někdo hodil samé šestky v průměru jednou za 10 minut. Čím více lidí hází (těží), tím více kostek je ve hře. Naopak s ubývajícími těžaři ubývají i kostky. Tato vlastnost zajišťuje, že se nové bitcoiny budou do oběhu uvolňovat stále stejně rychle, na rozdíl třeba od zlata, kde s rostoucím počtem těžebních firem roste i množství vytěženého kovu.

Samotná těžba ale není jen o získávání nových bitcoinů. Aby těžař odměnu ve formě nových bitcoinů získal, musí sestavit tzv. **blok**. Co je to ten blok? Můžeme si ho představit jako papír o velikosti A4 (tudíž má omezenou velikost), který obsahuje:

- Číslo – pořadí bloku (začíná od 0 a s každým dalším blokem se zvyšuje).
- Vybrané transakce.
- Fotku předchozího bloku.

Pamatujete si ještě náš košík, kam jsme si dočasně ukládali veškeré transakce, které jsme přijali? A vzpomínáte si na to, že každá transakce musí obsahovat poplatek? Těžař, který vyhraje v hodu kostkami, se podívá do svého košíku a seřadí si všechny transakce sestupně od té, která platí nejvyšší poplatek, až po tu, která platí nejméně. Vezme tu s nejvyšším poplatkem ze seznamu a nalepí ji na A4. Následně vezme druhou, třetí a tak dále, dokud na A4 zbývá ještě nějaké místo. Jakmile je papír už téměř plný, nalepí tam na poslední volné místo fotku (otisk) celého předchozího bloku, který má pořadové číslo o 1 menší.



Těžař právě vytvořil blok a za odměnu si může přivlastnit nově vytěžené bitcoiny. Co s takto vytěženým blokem udělá? To samé jako s transakcí – tedy odešle uzlům, se kterými je spojen, a ty jej **rozešlou dále do celé sítě**. Každý uzel po cestě blok kontroluje, zda je platný:

- Zda je jeho struktura v pořádku.
- Zda je každá transakce v něm obsažená platná.
- Zda obsahuje důkaz o tom, že těžař hodil samé šestky.
- Zda si těžař nepřivlastnil více bitcoinů, než je aktuální odměna za vytěžení bloku.
- Zda obsahuje fotku odkazující na předchozí blok a další náležitosti.

Jakmile by cokoliv nebylo v pořádku, uzel takový blok zahodí a nebude jej distribuovat dále do sítě. To je důvod, proč se v tomto systému nevyplácí podvádět – těžař vynaložil veliké úsilí, aby vytěžil blok (hodil na všech kostkách šestku), a kdyby se pokusil o podvod tím, že by do bloku zahrnul neplatnou transakci nebo si chtěl

třeba přivlastnit více bitcoinů, než mu náleží, bitcoinová síť by takový blok odmítla a on by nedostal odměnu. Tudíž by spálil veliké množství energie na těžbu ve formě házení kostek a nic by z toho neměl. Tomuto systému se říká také **proof-of-work** neboli důkaz o vykonané praci.

Poplatky podruhé

Když už nyní víme, jak vzniká blok, vraťme se na chvilku ještě zpět k poplatkům. Jak jsme se dozvěděli v předchozí kapitole, těžař získal za vytěžení bloku nové bitcoiny. To ale není jediná jeho odměna. Dále si vzal poplatky ze všech transakcí, které do bloku zahrnul. To je důvod, proč si je předtím řadil sestupně od těch, které obsahují poplatky nejvyšší. Aktuálně (rok 2022) je odměna za vytěžení bloku 6,25 BTC a poplatky mohou v průměru představovat třeba dalších 0,25 BTC (toto je velmi variabilní, jak si popíšeme níže). Celková odměna pro těžaře je tedy v tomto případě 6,5 BTC za vytěžení bloku.

Jak velké poplatky pro transakci zvolit? To záleží na aktuálním vytížení bitcoinové sítě. Snadno si to lze představit jako zastávku, kde lidé čekají na autobus. Ten přijíždí v průměru každých 10 minut (interval vytěžení bloku) a má 60 míst (paralela s omezeným množstvím místa v jednom bloku). Řidič autobusu přijede na zastávku, seřadí si čekající lidi podle těch, kteří jsou ochotni zaplatit nejvíce, a prvním 60 dovolí nastoupit. Zbytek musí čekat na další spoj. Takto se to pořád opakuje. I když budete aktuálně ve frontě třeba dvacátí, nemáte jistotu, že se do autobusu dostanete. Ten sice přijíždí v průměru každých 10 minut, ale protože těžba je náhodný proces, tak se může stát, že přijede až za 20 minut. Mezitím na zastávku přijde dalších 50 lidí, kteří budou ochotni zaplatit více než vy, a tak vás předběhnou. Občas se může stát, že na zastávce bude čekat při příjezdu autobusu méně lidí, než je kapacita autobusu – v tomto případě jedou všichni. Pokud bych to měl shrnout – abyste se dostali do autobusu, musíte v době jeho příjezdu být mezi prvními 60 nejlépe platícími.

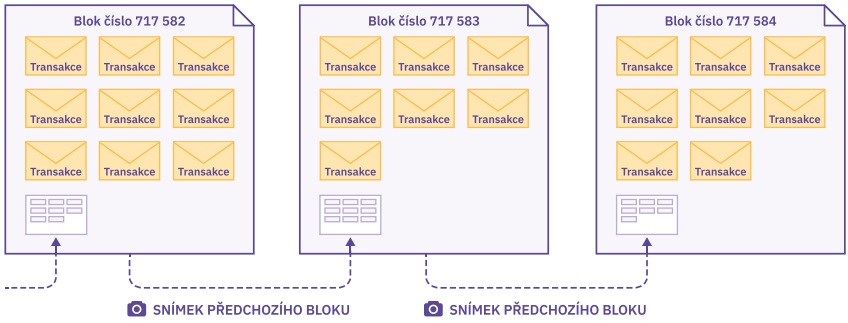
Při odesílání transakcí se tedy vyplatí sledovat aktuální vytížení sítě (abych neplatil zbytečně velké poplatky anebo příliš malé, které by způsobily, že budu čekat na zahrnutí transakce i několik hodin). Zároveň může být vhodné, pokud to situace dovolí, odesílat bitcoiny v průběhu víkendu, kdy zpravidla bývají poplatky nižší. Dnešní moderní peněženky automaticky navrhují ideální výši poplatků za vás.

Pozornější z vás možná napadlo, co se stane po roce 2140, kdy budou všechny bitcoiny vytěženy, a těžař už nebude dostávat žádné nové. V tomto případě se jeho odměna bude skládat pouze z transakčních poplatků. Pokud bude Bitcoin v tu dobu ještě pořád používaný, je velmi pravděpodobné, že tyto poplatky budou velmi vysoké.

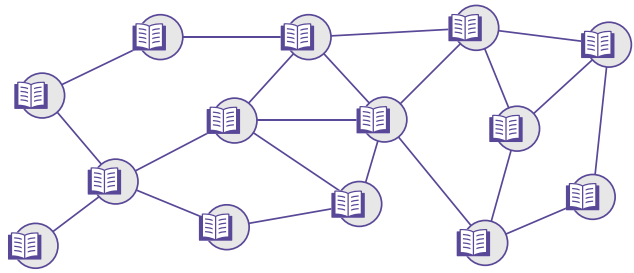
Blockchain

V předchozí kapitole jsme se dozvěděli, že těžař vytěžený blok distribuuje celou sítí ke všem ostatním uzlům. Ty si tento blok uloží do struktury, které říkáme blockchain, česky řetězec bloků. Mnoho lidí tvrdí, že blockchain je revoluční technologie, jež vyřeší veškeré problémy lidstva, hladomor i globální oteplování a měla by se proto nasadit naprosto všude. Realita je však trochu jinde. Blockchain není nic jiného, než tzv. **decentralizovaná účetní kniha**, která obsahuje všechny transakce, jež kdy byly pomocí této sítě odeslány. Ano, takže i transakce za dvojitý hamburger, který jste si v roce 2014 koupili za bitcoiny, bude navždy uložena v blockchainu.

Blockchain se skládá z jednotlivých bloků. První blok s pořadovým číslem 0 byl vytvořen tvůrcem Bitcoinu, Satoshi Nakamotem. Jednotlivé bloky na sebe navazují pomocí hashů (představme si je jako fotky) toho předešlého, což tvoří něco jako řetězec. Každý uzel, který nový blok obdrží a uzná, že je platný, si ho přidá na konec svého řetězce. Zároveň smaže transakce v něm obsažené ze svého „košíku čekajících transakcí“, mempoolu.



Výsledkem je, že každý uzel v síti má zcela identický blockchain. Zjednodušeně řečeno se jedná o knihu, kam si zakládám jednotlivé papíry A4 (bloky) s transakcemi. Tato účetní kniha je v úplně stejné kopii uložena na všech uzlech po celém světě.



Právě v decentralizaci tkví hlavní přínos blockchainu. Tuto síť nikdo centrálně neovládá, nikdo si nemůže bitcoiny vytvořit „ze vzduchu“, ani podvádět (ostatní uzly by to detekovaly a takovouto transakci nepřijaly), ani nelze bitcoinovou síť nijak vypnout. Stačí totiž, aby existoval jeden jediný uzel s aktuálním blockchainem, a po znovuzprovoznění sítě by se tato účetní kniha začala distribuovat všem ostatním uzlům.

Kdybych měl stručně shrnout, jak celý systém funguje, tak bitcoinová peněženka vytvoří transakci, podepíše ji naším klíčem a rozešle do sítě, kde si ji každý uzel ověří a uloží do dočasného úložiště – mempoolu (košíku). Zároveň mezi sebou soupeří těžaři o to, komu se podaří vytěžit nový blok (hodit samé šestky). Ten, kdo uspěje, zapíše určitý počet transakcí s nejvyšším poplatkem z mempoolu do bloku a tyto transakce z mempoolu odstraní. Za odměnu získá nové bitcoiny a všechny transakční poplatky. Tento nový blok opět odešle do sítě a každý uzel si ho ověří a uloží. V tuto chvíli celá síť ví o naší transakci a ta je považována za dokončenou.

> **On-chain transakce**
>
> Tento typ transakce, který jsme si popsali v odstavci výše, se nazývá on-chain transakcí, transakcí zapsanou do blockchainu, občas také transakcí na první vrstvě. Existují ještě jiné typy transakcí, které si vysvětlíme v kapitole o Lightning Network. Tam se budu občas odkazovat na tento pojem, proto kdykoliv bude u transakce pojem „on-chain“, znamená to výše popsaný způsob se zápisem do blockchainu, čekáním na vytěžení bloku a povinnými poplatky těžařům.

Úprava minulosti

Že nemůže podvádět ten, kdo odesílá transakci, ani těžař, jsme si již vysvětlili – síť by podvodnou transakci či blok odmítla. Je ale možné upravit blockchain? Co když někdy v minulosti proběhla platba, zapsala se do blockchainu a já bych ji nyní chtěl odvolat? Například jsem zaplatil 3 BTC za nové auto, odjel s ním domů a teď bych chtěl tuto transakci stornovat a bitcoiny si ponechat.

Tomuto typu podvodu brání právě ty fotky, které jsou v každém bloku. Reálně to nejsou fotky, ale výstupy hashovací funkce, pro zjednodušení ale zůstaneme u fotek. Dejme tomu, že číslo aktuálního bloku je 700. V bloku 690, tedy před deseti bloky, což odpovídá necelým 2 hodinám, jsem provedl transakci, kterou bych chtěl nyní vymazat.

Když to ale provedu – nastane problém. Fotka tohoto právě upraveného bloku (690) se nachází v následujícím bloku (691). Protože jsem změnil jeho obsah, musím změnit i fotku tohoto změněného bloku v následujícím bloku. Dobře, do bloku 691 vložím novou fotku bloku 690. Ale tím jsem vlastně upravil blok 691 a nesedí mi fotka v následujícím s číslem 692. A tak to půjde pořád dokola. Kvůli změně jedné malé hodnoty v jakémkoliv bloku musím změnit všechny bloky na něj navazující, protože jsou provázány fotkami.

Pokud si to upravím takto sám, bude mi to zcela k ničemu. Potřebuji, aby tímto způsobem upravenou historii přijala celá síť. Síť jako celek totiž rozhoduje o tom, co je pravda. Co ale každý uzel u jakéhokoliv přijatého bloku kontroluje? Jednou z věcí je důkaz o práci, tedy zda se mi podařilo na všech kostkách hodit šestku. Musel bych mít tudíž extrémní štěstí a v této těžařské loterii vyhrát 10x za sebou. Dejme tomu, že by se mi to povedlo. Takto upravené bloky rozešlu všem sousedům. Ti nyní ale budou mít jak tu „správnou" původní verzi blockchainu, tak i tu moji upravenou. Kterou přijmou za platnou? V bitcoinovém protokolu platí, že platný blockchain je ten, na kterém bylo vykonáno nejvíce práce – v našem zjednodušeném případě je to **ten delší**.

Pokud bych tedy chtěl přepsat historii, musel bych vytěžit všechny bloky od čísla 690 do 700 znovu. Protože je to velmi časově náročné (obzvlášť, když to budu dělat sám), tak než se mi to povede, zbytek sítě může být klidně u čísla 750. A delší řetězec vyhrává. To znamená, že jedinou možností podvodu je celou síť dohnat, a ještě navíc předběhnout. Teoreticky to možné je (říká se tomu 51% útok), ale při představě, že po celém světě jsou tisíce těžařů se speciálními zařízeními za miliony dolarů, kteří dokážou házet kostkami extrémně rychle, a proti nim stojím já, který se je snaží všechny předběhnout, tak vám asi dojde, jakou mám šanci na úspěch.

Uchovávání bitcoinů

Mnoho lidí si mylně myslí, že bitcoiny lze nějak „uchovávat u sebe". Že je to něco jako soubor, který vlastním, a pokud je chci někomu poslat, tak mu jej přepošlu. Tak to ale není. Bitcoiny jako takové jsou čistě virtuální a nejsou tudíž uloženy nikde. Pouze je v blockchainu (který je v identické kopii na všech uzlech po celém světě) uložena informace, kolik bitcoinů se nachází na které adrese spolu s informacemi o všech proběhlých transakcích.

Pouze **vlastník klíče** k dané adrese je majitel a může je odeslat, nikdo jiný. Vlastnit bitcoiny tedy znamená vlastnit klíče. Toto je velmi důležité si uvědomit. Pokud se totiž kdokoliv dostane k vaším klíčům, o své bitcoiny přijdete. Neexistuje nic jako telefonní číslo na podporu, kam byste zavolali, a mohli požádat o vrácení svých

bitcoinů. V tomto případě jste plně zodpovědní za své činy, a tak je více než důležité si tyto klíče bezpečně uchovat. Právě na tuto problematiku se nyní podíváme.

Mnoho lidí si kupuje bitcoiny na burzách či směnárnách. Jedná se o webové služby, kde po registraci a většinou i vaší identifikaci můžete nakupovat a prodávat jednotlivé kryptoměny za koruny, eura či dolary. Dále vám burzy umožňují své bitcoiny na jejich platformě i uchovávat. Toto je ale velmi nebezpečné. V takovém případě totiž žádné klíče nevlastníte. Klíče vlastní burza a s největší pravděpodobností ani nemáte svoji vlastní adresu. Mají pouze v interní databázi poznamenáno, kolik bitcoinů z jejich celkového obnosu je vašich. Existuje zde velké riziko, že takováto burza zkrachuje, bude vykradena anebo vlastníci ze dne na den „zavřou krám“ a se všemi prostředky zmizí. Všechny tyto případy se v historii staly nesčetněkrát, takže se říká, že není otázkou, jestli se to stane i vám, jde jen o to kdy. Držet větší obnos na burzách je **vždy špatné**. Bitcoin vám dává oproti jiným aktivům možnost jej vlastnit přímo prostřednictvím svých klíčů bez jakéhokoliv prostředníka. Využijte toho, a jakmile to bude možné, odešlete si bitcoiny z burzy do své vlastní peněženky.

Vlastnit bitcoiny ve své peněžence, což může být program na počítači či mobilní aplikace, je lepší, než je mít na burze. Pořád zde ale existuje značné riziko. Případní útočníci jsou totiž finančně motivováni vám bitcoiny ukrást. Stačí jakákoliv nepozornost, neaktualizovaná verze programu či otevření nebezpečné přílohy e-mailu, a jakmile se kdokoliv dostane do vašeho zařízení, nic mu většinou již nebrání si vaše klíče opsat a bitcoiny vám ukrást. Ani toto tudíž není ideální varianta.

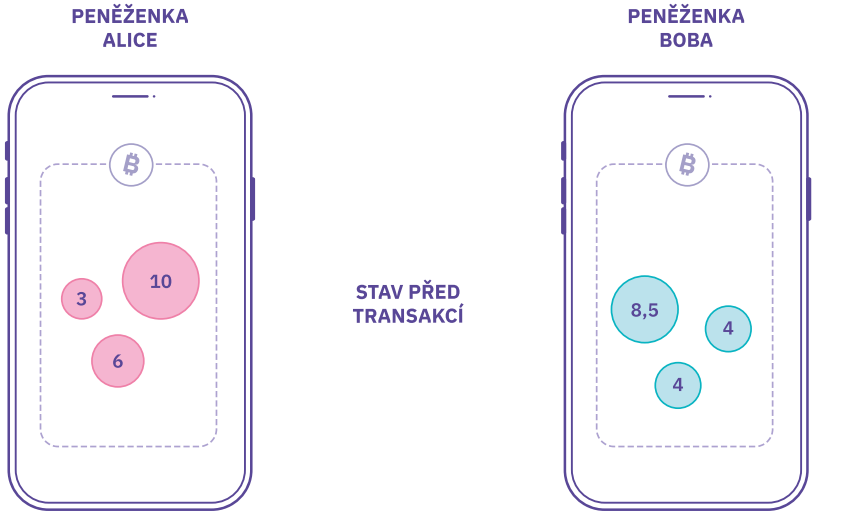
Nejbezpečnějším způsobem, jak uchovávat svoje bitcoiny, je speciální zařízení, tzv. hardwarová peněženka. Jedná se většinou o malé zařízení velikosti USB flash disku, které slouží k **ukládání vašich klíčů**. Výrobce této peněženky vám dodá i potřebný program, pomocí kterého můžete své bitcoiny spravovat – odesílat, přijímat, sledovat jejich množství a transakce. Mimo to, že jsou tyto hardwarové peněženky většinou velmi dobře zabezpečené, hlavní výhoda tkví v tom, že se klíče na nich uložené nikdy nedostanou do vašeho počítače. Pokud chcete odeslat transakci, tak ji váš počítač vygeneruje, odešle ji na hardwarovou peněženku (jediné místo, kde jsou klíče uloženy), tam dojde k podpisu a takto podepsaná transakce se již šíří do bitcoinové sítě. I kdybych měl jako útočník plný přístup k vašemu počítači, nemohu vám klíče odcizit. Investice do takovéto peněženky není nikterak nákladná a důrazně ji doporučuji.

Celou tuto kapitolu zde máme proto, že v rámci Lightning Network platí trochu jiná pravidla ohledně uchovávání bitcoinů a bezpečnosti. Je ale nutné rozumět tomu, jak správně uchovávat bitcoiny na první vrstvě. Jak je tomu u Lightningu, si povíme v příslušné kapitole.

UTXO – mince

Posledním technickým termínem, který si probereme, je UTXO, což je zkratka pro Unspent Transaction Output, tedy zatím neutracený transakční výstup. V běžném světě, kde zatím ještě používáme hotovost, máme mimo jiné i mince, které jsou principiálně s UTXO podobné. České mince mají různou hodnotu – 1, 2, 5, 10, 20 a 50 Kč. Ve světě Bitcoinu je to podobné, jen tyto mince mohou mít libovolnou hodnotu. Mohu mít např. minci s hodnotou 3,15 BTC a druhou, která bude reprezentovat 0,86 BTC. Zůstatek na našich peněženkách (přesněji na našich adresách) je právě z takovýchto mincí složený.

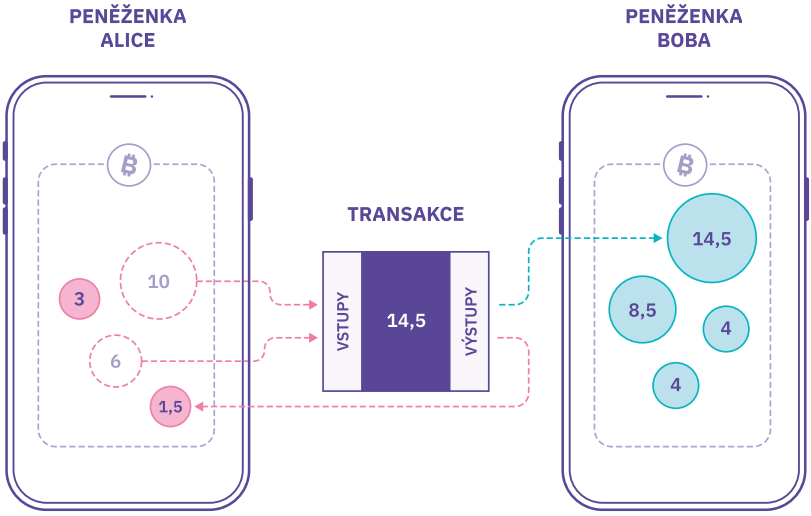
Pokud mi někdo do prázdné peněženky pošle 10 BTC, druhý den 6 BTC a třetí den 3 BTC, budu vlastnit tři mince/UTXO přesně o těchto hodnotách. Ze svých celkových 19 BTC budu chtít odeslat kamarádce Janě 14,5 BTC. Jak to udělám? Naprosto stejně jako s mincemi v běžném světě. Podívám se do peněženky a zjistím, že minci o této hodnotě nemám a musím tedy platbu složit z více mincí. Jako ideální kandidát se nabízí 10 BTC a 6 BTC.



Vstupy a výstupy transakce

Každá transakce má jeden nebo více vstupů (moje UTXO/mince, které chci utratit) a stejně také jeden nebo více výstupů (tyto mince/UTXO budou patřit příjemci transakce). Počet vstupů a výstupů ve většině případů automaticky sestavuje peněženka a pro běžného uživatele je toto skryto.

Tyto dvě mince tedy vložím jako vstup transakce – tím zaniknou. Výstupem budou dvě nové mince, jedna o hodnotě 14,5 BTC pro Janu a druhá s hodnotou 1,5 BTC, kterou pošlu sám sobě.



Proč to zde řešíme, když je tento interní mechanismus pro většinu uživatelů skryt? Protože to má co do činění s výší poplatků za běžnou transakci. A právě levnější poplatky jsou jednou z několika výhod Lightning Network, jak se dozvíme později.

Z běžného světa jsme zvyklí, že pokud má nějaká platba poplatek, většinou si její zprostředkovatel bere určitá procenta z obnosu, např. 1 %. Čím větší částku tedy chceme zaplatit, tím více prostředník dostane. V Bitcoinu to ale tak neplatí. Jedním z nejcennějších artiklů je zde místo v bloku. Jeho maximální velikost je 4 MB v závislosti na typu transakcí v něm obsažených. Pokud tedy někdo vytvoří extrémně velkou transakci a zabral by třeba polovinu bloku, měl by za to odpovídajícím způsobem zaplatit. Poplatky v Bitcoinu tedy nejsou závislé na částce, ale na **velikosti transakce**. Pokud budu posílat 100 BTC, tak zaplatím stejnou částku, jako kdybych odesílal 1 BTC.

A jak že se vlastně ta velikost počítá? Zjednodušeně řečeno by se dalo říct, že čím více vstupů a výstupů transakce obsahuje, tím zabere více místa, a tudíž je za ni nutné zaplatit větší poplatek. Pokud se vrátíme k analogii s mincemi, tak by to znamenalo, že by obchodník zvážil náš měšec s mincemi a podle jeho váhy bychom zaplatili poplatek. Neřešil by obsah měšce, pouze jeho váhu. Transakce, která se skládá z mnoha malých vstupů, je tedy dražší, než pokud by vstup byla jedna obrovská mince.

Závěrem by se tedy dalo říct, že velikost poplatku za on-chain transakci závisí na tom, kolik vstupů jsem musel použít k jejímu vytvoření a na kolik výstupů bitcoiny posílám, jinak řečeno – kolik nových mincí vytvářím. A dále závisí také na tom, jak dlouho jsem ochoten čekat na její zařazení do bloku (paralela s autobusovou zastávkou – čím větší nastavím poplatek, tím dříve bude moje transakce zapsána do blockchainu).

Shrnutí

Bitcoin je alternativní finanční systém. Jsou to svobodné a decentralizované peníze, které neovládá žádný stát ani centrální banka. Všechny bitcoinové uzly v síti si jsou naprosto rovny – není důležité, z jaké jste země, kolik bitcoinů vlastníte, ani jaké je vaše postavení ve společnosti. Kdokoliv na světě se může k této síti přidat a být jejím členem.

Počet bitcoinů je pevně zafixován na hodnotě 21 milionů – známý je i přesný způsob uvolňování nových. U Bitcoinu nemusíte důvěřovat několika centrálním bankéřům, že neznehodnotí vaše úspory „tiskem“ dalších peněz. Stačí vám důvěra v matematiku a počítačový kód. Takový kód, který bez závažnějších problémů a výpadků funguje jako švýcarské hodinky již od roku 2009. Jedná se o finanční systém, kterému mnoho kritiků proklamuje konec, pád jeho hodnoty, a tvrdí, že se jedná o bublinu. Přesto tržní kapitalizace Bitcoinu neustále roste a stejně s ním i adopce mezi běžnými lidmi a institucemi.

Bitcoin nezná hranice jednotlivých zemí ani kontinentů. Je jedno, jestli odesíláte transakci kamarádce přes ulici nebo známému do Austrálie. Bitcoin neřeší, jaká měna se používá v které zemi, a směny mezi nimi. Pokud bitcoiny uchováváte správně, nikdo vám je nemůže sebrat ani zabavit, i kdyby to byl nejkrutější diktátor. Navíc jednotlivé transakce nelze na úrovni protokolu blokovat. O držení bitcoinů se staráte vy sami, nemusíte to přenechávat bankám. Vy jste vaše vlastní banka.

Bitcoinovou síť kvůli decentralizaci nelze zničit. Než se někomu povede vyřadit jeden uzel z provozu, zastoupí jeho místo několik dalších. Síť je navíc elastická, a tak se podobným pokusům o zákaz bude bránit svou případnou úpravou. Ke skladování libovolně vysoké hodnoty vám stačí malé elektronické zařízení – nemusíte řešit složité uskladnění jako v případě zlata. Pro ověření pravosti vám stačí počítačový program. Síť sama o sobě je navržena takovým způsobem, aby nemohlo docházet k podvodům.

Pevně daný počet bitcoinů oproti tradičním měnám zaručuje v dlouhodobém měřítku, že vaše úspory nebudou ztrácet kupní sílu. I přes velikou energetickou náročnost těžby, která v důsledku zajišťuje bezpečnost celého systému, je zde díky konkurenčnímu tlaku mezi jednotlivými těžaři o získání co nejlevnější energie rostoucí poptávka po obnovitelných a volných zdrojích energie, které méně zatěžují životní prostředí. Zároveň se čím dál více objevují případy, kdy se těžba přesouvá na odlehlá místa a využívá energii, která by jinak přišla vniveč.

Abych byl férový, tak je potřeba poznamenat, že i Bitcoin má nějaká drobná negativa a problémy, kterým bude v budoucnu čelit. Můžeme mezi ně zařadit hrozící státní regulace nebo například rozpočet na zabezpečení sítě v době, kdy by měly převládat odměny pro těžaře ve formě transakčních poplatků. Až budoucnost tedy ukáže, jak moc úspěšný Bitcoin bude. Nejedná se ale o žádný rigidní systém stvořený geniálním vynálezcem, který by se již nijak nevyvíjel. Naopak – vše je nyní v rukou samotných uživatelů a vývojářů, kteří systém neustále vylepšují.

Bitcoin je svoboda. Bitcoin je budoucnost.

ÚVOD DO LIGHTNING NETWORK

Škálovatelnost Bitcoinu

Platit bitcoinem na první vrstvě (on-chain) za produkty každodenní potřeby jako je oběd, parkovné či vstupenka do kina **není zrovna výhodné**. Jednak se v období vysokého využití sítě poplatky za transakci mohou rovnat ceně kupovaného produktu, navíc je zde nutnost čekat na zařazení transakce do bloku (desítky minut). Představa, že si koupím na koncertu pivo, budu čekat u stánku desítky minut na potvrzení a zároveň zaplatím několik korun navíc jako poplatek těžařům, není zrovna lákavá. Obchodník se sice může rozhodnout, že mi kupovanou věc předá ihned, jakmile se tato transakce začne šířit sítí, a nebude čekat na její zařazení do bloku – toto ale není bezpečné, jelikož riskuje tzv. double-spend.

> **Double-spend**
>
> Jedná se o situaci, kdy jsou jedny a ty samé bitcoiny na první pohled utraceny vícekrát. K této situaci můžete dojít třeba tak, že já jako útočník odešlu transakci, jejímž příjemcem je stánkař prodávající pivo a v té samé chvíli vytvořím a odešlu druhou transakci, která bude utrácet ty samé bitcoiny, ale tentokrát bude příjemcem kamarád. Obě transakce se budou šířit sítí nezávisle a je možné, že polovina uzlů se dozví první o té pro stánkaře, ke druhé polovině uzlů dorazí dříve ta pro kamaráda. Toto je očekávaný stav a finálním rozhodčím je těžař, který vytěží blok a jednu z těchto dvou transakcí do něj zapíše. Tím ji potvrdí a zneplatní druhou z nich v celé síti. Pokud se k těžaři dostane dříve transakce pro kamaráda, tak mi stánkař prodal pivo a nezískal žádné bitcoiny. Je tedy vždy nutné počkat na zahrnutí transakce do bloku (samotná hloubka závisí na obnosu transakce).

Dva hlavní problémy používání bitcoinu (on-chain) pro každodenní platby bychom tedy měli – je to pomalé a drahé. V závislosti na podmínkách sítě zvládne Bitcoin zpracovat aktuálně maximálně okolo 10 transakcí za vteřinu, reálně to je však většinou o něco méně. Pro srovnání Visa tvrdí, že její síť zvládne zpracovat 1 700 transakcí za stejné období, Mastercard dokonce 5 000. Jak je na první pohled vidět, čísla těchto společností jsou řádově úplně někde jinde. Co s tím? Nejčastěji lidi napadají tato 2 řešení:

- **Zvětšíme samotný blok**. Aktuálně je maximální velikost jednoho bloku 4 MB. Kdyby byl blok dvojnásobně veliký, zvládl by až 20 transakcí za vteřinu. Kdyby byl desetinásobný, tak 100 transakcí a tak dále. Pokud bychom chtěli zpracovávat

6700 transakcí za vteřinu (stejně jako Visa a Mastercard dohromady), jeden blok by musel mít velikost přibližně 1,1 GB. Tento lineární způsob škálování ale není řešením. Proč? Vedlo by to totiž k extrémní centralizaci bitcoinové sítě. Aktuálně jsou v provozu tisíce bitcoinových uzlů, které ověřují veškeré transakce a zabezpečují samotnou síť. Zároveň platí, že takovýto uzel můžete provozovat doma na minipočítači za nižší jednotky tisíc korun a postačí vám k tomu disk o velikosti stovek GB. V případě takto extrémně velkých bloků by ale bitcoinovou síť mohli provozovat pouze velcí hráči v datacentrech, kteří budou mít dostatečné diskové kapacity pro ukládání celého blockchainu, odpovídající výpočetní výkon pro ověřování všech transakcí a vysokorychlostní internet schopný přijímat a odesílat gigabytové bloky. Celá síť by byla více centralizovaná, a proto méně bezpečná. Zvětšení bloku v řádu nižších jednotek tedy problém nevyřeší, naopak extrémní zvětšení narušuje bezpečnost. Toto tedy není správná cesta pro škálování.

- **Zrychlíme těžení bloků**. Aktuálně je nový blok vytěžen přibližně každých 10 minut. Kdybychom upravili parametry sítě tak, aby byl vytěžen například každou minutu, zvládli bychom zpracovat desetkrát více transakcí. Případně by mohl nový blok být vytěžen každou vteřinu. Toto řešení je ale velmi podobné předchozímu. Čím častěji budou vznikat nové bloky, tím větší bude potřebné místo pro uložení kompletního blockchainu, a tudíž porostou HW nároky na provozování uzlu – veškerá negativa ohledně centralizace z předchozího bodu tudíž platí i zde. Navíc tady bude problém i s distribucí nových bloků. Ty by totiž pravděpodobně vznikaly rychleji, než by se stačily ty předchozí rozesílat mezi všechny uzly po celém světě, a často by se stávalo, že by těžař těžil na bloku, který není poslední. Toto by vedlo k velmi častým rozdvojením blockchainu a nuceným reorganizacím, což je opět cesta, kterou se vydat nechceme.

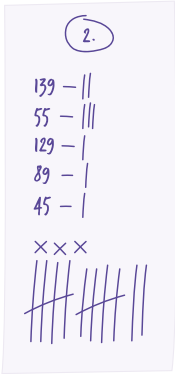
Existují samozřejmě i další možnosti řešení škálovatelnosti, které jsou využívány v jiných kryptoměnách. Jejich podrobný popis by však byl na samostatnou knihu. Zaměřme se tedy na to, jak je problém škálovatelnosti řešen v Lightning Network.

Modelový příklad

Než se pustíme do vysvětlování technických detailů Lightningu, podíváme se společně na jeden modelový příklad z běžného života, který je velmi podobný principům fungování Lightning Network.

Představte si, že se živíte jako farmář a prodáváte mléko, vejce a maso ze svojí farmy. Bydlíte na vesnici a máte tam oblíbenou hospůdku, kam chodíte o víkendech s rodinou na oběd a občas si tam zajdete s kamarády na pivo. Běžně to funguje

tak, že vám hospodský dá na stůl lístek, kam vám píše veškerou útratu – jak jídlo, tak pití. Činí tak z toho důvodu, že vyžadovat platbu za každé jedno pivo by nebylo příliš efektivní. Jakmile s konzumací skončíte, hospodský sečte vaši útratu z lístku a vy vše zaplatíte jednou transakcí, ať již hotovostní, nebo pomocí platební karty.



Protože je hospodský váš dlouholetý kamarád, bude to pro vás mít další výhody. Jednak vás nechá konzumovat ve své restauraci tzv. „na sekyru“, tedy nebude po vás chtít zaplatit účet ihned, ale protože vám důvěřuje a zná vás osobně, tak vám jej uschová na příště a spokojí se s platbou například 1x za 14 dní či až bude částka za jednotlivé útraty velmi vysoká. Druhou výhodou bude, že se společně dohodnete na tom, že vejce a mléko z vaší farmy, které od vás hospodský pravidelně nakupuje, budete odečítat z vašeho lístku. Pokud by byly vaše útraty v restauraci a hospodského nákupy vyvážené, dalo by se takto fungovat bez finálního vypořádání docela dlouhou dobu.

Problémem je zajisté to, že neustálé přepisování jednoho a toho samého lístku může být po delší době docela nepřehledné. Zároveň do tohoto způsobu fungování vstupuje **vzájemná důvěra**, kdy musíte mít jistotu, že druhý nebude při úpravě lístku podvádět ani zničehonic nezmizí, jakmile by byl stav pro něj nevýhodný (dlužil by peníze). Toto tedy rozhodně není něco, co by v běžném světě mohlo fungovat pro masy lidí.

Cílem tohoto příkladu bylo ukázat základní princip Lightning Network, a to konkrétně, že si dvě strany mohou mezi sebou udržovat jakýsi soukromý účet a **jen upravovat jeho stav** – k finálnímu vypořádání může dojít až za nějakou dobu. To ušetří čas, protože přijímat platbu na terminálu nebo vracet zpět peníze v hotovosti je vždy pomalejší, než udělat další čárku na lístek. Také se tímto dají **minimalizovat poplatky**, které by pro položky v řádu jednotek korun a při platbě kartou nebyly výhodné. A pokud využíváme platební kartu, tak se také **zvyšuje anonymita**, jelikož banky a karetní společnosti vidí až konečné vypořádání, nikoliv jednotlivé transakce.

Základní princip Lightning Network

Položme si otázku – je nutné, aby veškeré transakce byly zapsané v blockchainu? Pokud si budu kupovat něco drahého (v řádech vyšších tisíců či milionů korun), tak zajisté ano. Jeho decentralizovaná povaha spolu s nemožností měnit historii mi zaručuje velmi vysokou bezpečnost a nepopiratelnost dané transakce. Co když si ale jen kupuji kávu? Co když platím za vstup na rozhlednu? Co když jsem si domů nechal dovézt pizzu? Je opravdu nutné, aby o této transakci věděl skrze zápis do blockchainu

naprosto každý, nebo by bylo pro tyto menší a každodenní platby vhodnější navrhnout jiný systém, který by spolupracoval s blockchainem, byl bezpečný a zároveň umožňoval rychlé a anonymní platby s minimálním poplatkem? Odpovědí je Lightning.

Multisig adresa

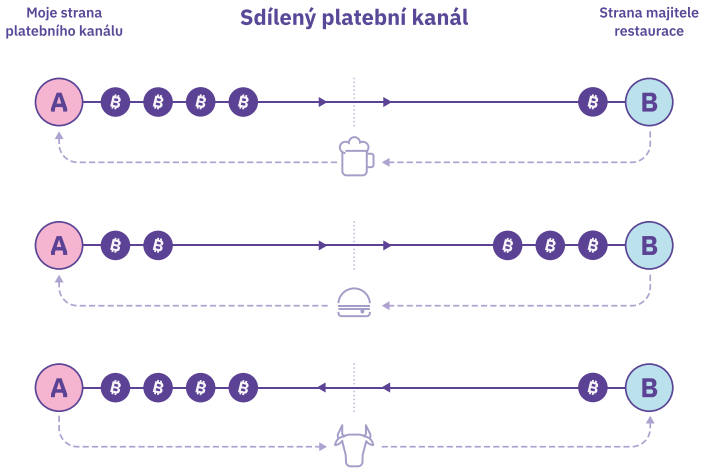
Ke klasické bitcoinové adrese existuje vždy právě jeden privátní klíč, jehož majitel může s bitcoiny libovolně nakládat. Existují ale i adresy, které mají těchto klíčů více, označme jejich počet symbolem N. Dále je vyžadováno, aby M privátních klíčů bylo použito pro nakládání s bitcoiny, kdy M musí být nižší či rovno N. Daná adresa se pak nazývá multisig M z N. Příkladem může být 4 z 5 – to nám říká, že k dané adrese existuje 5 privátních klíčů, ale jen 4 z nich jsou potřeba pro utracení bitcoinů. Představte si to jako truhlu, na které je 5 zámků, ale po otevření libovolných 4 se dostanete dovnitř. V rámci Lightning Network se využívají multisig adresy 2 ze 2. Existují tedy dva privátní klíče a oba jsou potřeba pro utracení bitcoinů. Pokud je každý klíč držen jinou osobou (což bude náš případ), je nutné, aby obě tyto osoby svým podpisem potvrdily utracení daných bitcoinů.

Základním stavebním kamenem Lightning Network je **platební kanál**. Tento pojem bude asi nejčastěji skloňovaným slovním spojením po zbytek knihy. O co se jedná? Platební kanál je **sdílený účet** mezi dvěma osobami (technicky řečeno se jedná o multisig adresu 2 ze 2). Tento platební kanál obsahuje určitý počet bitcoinů (tzv. kapacita či velikost kanálu), jejichž množství si libovolně určil ten, kdo kanál otevřel.

V mém konkrétním případě by dávalo smysl si otevřít kanál s majitelem restaurace, o kterém jsem psal v modelovém případě výše. Otevřu tedy svoji peněženku pro Lightning Network (což ve většině případů není nic jiného než aplikace pro mobilní telefon), vytvořím kanál s hospodským a uzamknu do něj 200 000 satoshi (nadále budu pro reálnost určovat hodnotu v satoshi, nikoliv v celých bitcoinech). Nyní jsem tedy vytvořil svůj první kanál – jedná se o klasickou on-chain transakci (odeslání 200 000 satoshi z mé adresy na multisig adresu), za kterou musím zaplatit poplatek podle vytížení sítě a počkat, až bude zařazena do bloku. Otevření kanálu může tedy několik desítek minut trvat (pokud se nepoužijí pokročilé techniky umožňující otevření kanálu ihned – o nich více v pokročilé části knihy).

Každý platební kanál má dvě strany, které určují, komu patří jaká část sdílených prostředků. Protože jsem v tomto případě inicioval vytvoření kanálu já a vložil do

něj 200 000 satoshi ze svého, veškeré bitcoiny patří mně. Nyní mohu provádět platby pomocí Lightning Network. V restauraci si mohu objednávat jídlo a pití, platit budu pouze přesunutím prostředků na druhou stranu. Zároveň platí, že tento kanál je obousměrný, tudíž hospodský si ode mě může koupit farmářské výrobky a satoshi se pouze přesunou zpět na moji stranu. Takovýchto plateb můžeme udělat neomezené množství.



Hlavní výhoda Lightning Network je, že aktuální stav kanálu si udržujeme pouze my dva a neodesílá se do bitcoinové sítě. V tomto případě se tedy neplatí **žádné poplatky**, platba je **instantní** – stačí pouze kontaktovat druhou stranu, což zabere maximálně vteřinu.

Po určité době, klidně po několika letech, se může stát, že se majitel restaurace odstěhuje pryč do jiného města a svoje podnikání v naší vesnici ukončí. V tomto případě mi již nebude nadále dávat smysl s ním mít otevřený platební kanál a domluvíme se na finálním vypořádání. To proběhne pomocí klasické on-chain transakce, kdy se jednotlivé prostředky sdíleného kanálu rozdělí mezi nás dva v poměru, který odpovídá aktuálnímu stavu.

Za celou životnost platebního kanálu jsme tedy mohli provést tisíce plateb, kdežto do blockchainu se odeslala pouze ta první, která vytvářela kanál, a ta poslední, která rozdělovala prostředky a kanál zavřela. Veškeré lightningové transakce pomocí platebního kanálu byly instantní a zdarma.

Zajisté vám došlo, že zde existuje určité omezení – pokud veškeré prostředky (v mém případě 200 000 satoshi) přesunu na druhou stranu, nemohu již nadále provádět žádné platby, jelikož na mé straně již nic není, nemám z čeho utrácet. Zároveň pokud otevřeme kanál a veškeré bitcoiny jsou na naší straně, hospodský nám tímto kanálem nemůže poslat žádnou platbu, protože nyní na své straně nemá nic on. Moderní Lightning peněženky se s těmito problémy vypořádají různými způsoby, více se o nich dočtete v samostatné kapitole.

Bezpečnost a podvody

Je Lightning spolehlivý způsob plateb? Nemohu o své prostředky přijít? Jak je to s bezpečností? Na danou problematiku se podíváme v této podkapitole. Celý bitcoinový ekosystém je založen na **trustless modelu**.

> **Trustless model**
>
> Trustless model je koncept, kdy nemohu nebo nechci nikomu věřit. V běžném světě lidé důvěřují například bankám, že se postarají o jejich úspory a že o ně nepřijdou. Ve světě Bitcoinu ale není žádná centrální autorita, komunikuje zde každý s každým jako s rovnocenným partnerem. A protože svět není růžový, čas od času se stane, že narazíte na podvodníka, který nebude hrát podle pravidel a bude se vás snažit okrást. V samotném Bitcoinu (i Lightningu) je toto řešeno v rámci protokolu samotného. Vždy je tedy vhodnější spolupracovat s čestnými uživateli – pokud ale narazíte na podvodníka, systém je navržen takovým způsobem, že nebude mít možnost vás jakkoliv okrást. Samozřejmě pokud se budete řídit určitými pravidly, které si rozebereme později.

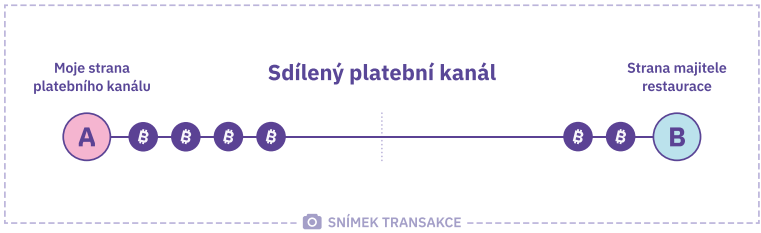
První věcí, kterou by takový podvodník mohl udělat, je si říct: *„Hmm, 70 % prostředků je na druhé straně, já vlastním jen zbývajících 30 % a to se mi nelíbí. O tom, jaký je reálný stav, stejně nikdo kromě nás dvou neví, tak zalžu a pokusím se uzavřít kanál s tím, že všem budu tvrdit, že na mé straně bylo 90 %.“* Odešle tedy do bitcoinové sítě transakci uzavírající kanál, kde 90 % prostředků připadne jemu. Co se stane? První uzel v síti takovouto **transakci zahodí**. Důvod? Pokouší se totiž odeslat prostředky z multisig adresy 2 ze 2. Aby to mohl udělat, potřebuje podpisy obou stran. Svůj podpis má, druhou stranu ale k tomuto podvodu přesvědčí jen těžko. Transakce s chybějícím podpisem je tedy neplatná, a tudíž se nebude propagovat sítí ani nebude nikdy zapsána do blockchainu.

Dobře, takto by to tedy nešlo. Podvodník na to zkusí jít jinak: *„Když se kouknu na aktuální stav, opět většina prostředků připadá druhé straně. Ale ještě před dvěma měsíci to bylo naopak, to jsem v jednu dobu vlastnil 85 % bitcoinů z kanálu já. Co kdybych dělal, že je toto aktuální stav a že následující transakce nikdy neproběhly?“* Stejným způsobem, jako je uvedeno výše, tedy odešle do sítě transakci uzavírající kanál, která ale bude reflektovat stav z minulosti, jenž byl pro něj výhodnější. Tentokrát to tedy není úplně vymyšlený stav kanálů, pouze neaktuální. Jak se bránit tomuto podvodu? Abychom si na to mohli odpovědět, musíme se podívat, jak funguje samotný proces přesunu prostředků z jedné strany na druhou.

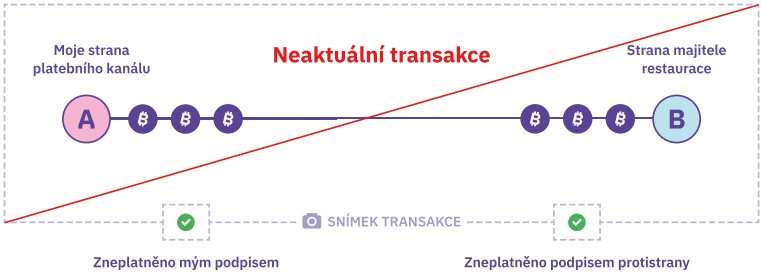
Kdybychom šli opravdu do hloubky, tak zjistíme, že tento proces není úplně triviální. Protože jsme ale v základní části knihy, zkusíme si to celé **zjednodušit**. Během přesunu prostředků musíme udělat dvě věci:

- Společně si potvrdit nový stav.
- Zneplatnit ten předchozí.

Jak si potvrdíme nový stav? Zkusme si to představit tak, že ho jednoduše vyfotíme a fotku si každý z nás uloží.



Zároveň vezmeme fotku předchozího stavu a na ni napíšeme NEAKTUÁLNÍ. Tuto zneplatněnou fotku navíc oba stvrdíme svým podpisem a budeme držet každý po jedné kopii.



Vytvořit novou fotku a zneplatnit tu předchozí je **nutné** udělat při **každé** úpravě stavu kanálu. K čemu nám to bude? Jakmile podvodník vyšle do bitcoinové sítě transakci, která kanál uzavírá, ale je z minulosti a nereflektuje aktuální stav, máme jasný důkaz o tom, že podvádí. Na samotné transakci, která uzavírá kanál a odesílá bitcoiny zpět do našich peněženek, je totiž jakýsi **časový zámek**, který definuje, za jaký čas bitcoiny doputují do naší peněženky. Toto nelze nijak obejít, musí se prostě počkat (například 1 den). A právě tohoto můžeme využít. Časový zámek je zde z toho důvodu, aby protistrana měla šanci zareagovat. To uděláme tím, že do bitcoinové sítě odešleme důkaz o podvodu – fotku se stavem, jenž se podvodník snaží uzavřít, na které ale bude napsáno NEAKTUÁLNÍ. Navíc tato fotka bude podepsána oběma stranami. Znamená to tedy, že musí existovat novější fotka a došlo zde k pokusu o podvod publikováním stavu z minulosti. V tomto případě **veškeré** bitcoiny z kanálu připadají **ihned** nám a podvodník nedostane vůbec nic. Podvádět se tedy obecně v drtivé většině případů nevyplácí – případný zisk nevyrovná riziko ztráty všech bitcoinů.

Co z toho vyplývá? Podvody se v naprosté majoritě případů nevyplatí, protože riskujete ztrátu veškerých prostředků. Na druhou stranu je zde povinnost provádět 2 úkony:

- **Být neustále online** a sledovat, zda se protistrana nepokusila uzavřít kanál ve stavu z minulosti.
- **Ukládat si veškeré zneplatněné stavy** z minulosti.

Standardně tyto 2 úkony za vás řeší samotná lightningová peněženka.

A jaký je poslední problém, který může nastat? Otevřete si s někým kanál a po určité době protistrana půjde offline a přestane odpovídat. Jelikož tato situace bude trvat týdny a s provozovatelem se vám nepodaří nijak spojit, nelze předpokládat, že došlo pouze ke krátkodobému výpadku. Jak nyní zajistit, že mi moje bitcoiny nezůstanou v kanálu uzamčeny navždy?

Zde opět využijeme našich „fotek“. Odešleme do bitcoinové sítě tu poslední, na které je aktuální stav kanálu, a budeme čekat 1 den, aby měla protistrana šanci zareagovat v případě, že bychom podváděli. Zde ale k žádné aktivitě nedojde a bitcoiny z naší strany kanálu se po 24 hodinách přesunou do naší peněženky. Tím zároveň dojde k uzavření kanálu.

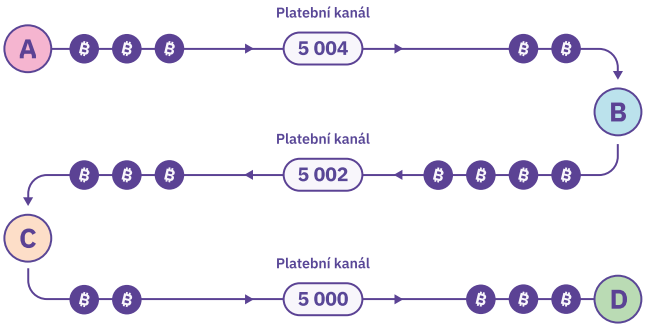
Jak je vidět na příkladech výše, i když svoje bitcoiny posílám na jakýsi sdílený účet zvaný platební kanál, můžu si být jistý, že když dodržím veškerá pravidla, tak o ně nepřijdu. Maximálně budu na jejich připsání zpět čekat 1 den a případně zaplatím poplatek za uzavření kanálu. Zde bych chtěl jen podotknout, že reálně se v Lightning Network

nepoužívají žádné fotky, ale využívá se kryptografie. Pokud by vás zajímalo více do hloubky, jak je toto konkrétně implementováno, rozebereme to v pokročilé části knihy.

Přesměrování plateb

Jak jsme si popsali v předchozí kapitole, pokud mám v úmyslu s určitou osobou provádět transakce pravidelně, vyplatí se mi s ní otevřít platební kanál. To může být případ hospodského z úvodu. Co když ale chci někomu pomocí Lightning Network zaplatit pouze jednorázově? Například si chci koupit zmrzlinu na koupališti nebo vstupenku do muzea. Kvůli jedné platbě se kanál otevírat nevyplatí, jelikož zaplatím on-chain poplatek za jeho otevření, provedu jednu platbu zdarma a poté bych ho uzavřel, za což opět zaplatím.

Princip Lightning Network byl navržen tak, aby si mohly platit i osoby, které spolu **nemají napřímo otevřený kanál**. Dejme tomu, že chci zaplatit Davidovi 5 000 satoshi, které mu dlužím. Nemám s ním ale otevřený platební kanál. Mohu však využít faktu, že mám otevřený kanál s Bobem, který má otevřený kanál s Cyrilem a ten má kanál s Davidem. Platba proběhne postupným přesouváním prostředků v jednotlivých kanálech – já odešlu 5 004 satoshi Bobovi, ten odešle 5 002 satoshi Cyrilovi a Cyril přesune 5 000 satoshi na stranu Davida.

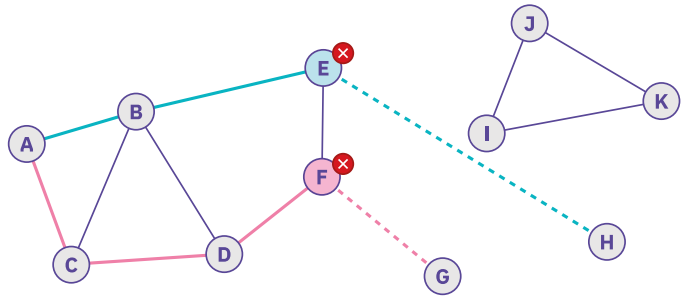


Prakticky jsem tedy odeslal prostředky Davidovi a Boba s Cyrilem využil pouze jako prostředníky. Určitě se ptáte, proč odesílám Bobovi o 4 satoshi navíc, a stejně tak, proč Bob odesílá Cyrilovi o 2 satoshi více. Jednoduchá odpověď – **poplatky**. Protože uzel uprostřed, který přesměroval naši platbu, musel přesunout v rámci kanálu určitý počet bitcoinů na druhou stranu (i když ten stejný počet mu přitekl jiným kanálem), účtuje si za to drobný poplatek. Více o poplatcích a jejich výši se dozvíte v samostatné kapitole.

Důležitou vlastností takovýchto plateb je, že buď proběhne celá (zaplatím Davidovi 5 000 satoshi), nebo neproběhne vůbec a prostředky mi zůstanou. Jinými slovy se **nemůže stát**, že si například Bob dané satoshi nechá a neodešle je dále. Na detaily, jak je tohoto dosaženo, se podíváme v pokročilé části knihy, prozatím to berme jako fakt.

Další významnou vlastností je **anonymita**. V příkladu výše Cyril netuší, že odesílá satoshi ode mě pro Davida. Má pouze informaci, že má přesunout 5 000 satoshi Davidovi, a získá za to stejnou částku od Boba a k tomu 2 satoshi navíc jako poplatek za vynaloženou práci. Cyril ale netuší, kolikátý je v tomto řetězci plateb. Z jeho pohledu platbu odeslal buď Bob, anebo někdo ještě před ním, a příjemcem je buď David, anebo někdo v řetězci za ním. Tato krásná vlastnost nám zaručuje, že v rámci Lightning Network nelze platby nijak cenzurovat, jelikož kromě částky samotné o nich nemáte žádné další informace.

Někdy se může stát, že takováto platba selže a lightningová peněženka vám zahlásí chybu. S postupným rozvojem samotné sítě a lepší implementací jednotlivých peněženek je to již méně časté, stoprocentně to ale vyloučit nelze. Co může být důvodem?



- Uzel A platí uzlu J – zde došlo k problému, že v lightningové síti neexistuje žádná cesta (série otevřených kanálů) mezi těmito dvěma uzly.
- Uzel A platí uzlu H – může se stát, že některý uzel po cestě (v našem případě E) nemá dostatečné prostředky v kanálu s uzlem H. Dejme tomu, že kapacita mezi E a H je 50 000 satoshi, vy ale chcete odeslat větší částku. V daném případě platba tyrkysovou cestou selže.
- Uzel A platí uzlu G – výjimečně se může stát, že jeden z uzlů přestane komunikovat (například těsně před provedením platby půjde offline). Růžová cesta tedy v takovémto případě také neprojde.

To, že jedna z cest selže, ještě neznamená, že celá transakce bude neúspěšná. Odesílající uzel vždy vypočítá několik cest, seřadí si je podle výhodnosti (celkové

poplatky, délka atd.) a zkouší jednu po druhé, dokud se mu platba nepovede. Samotná transakce selže pouze v případě, že by neprošla žádnou z cest.

Při běžném používání Lightning Network je takovýchto plateb (tedy mezi dvěma osobami, které spolu nemají otevřený kanál napřímo a využívají ostatní jako prostředníky) **naprostá většina**. Lightningová síť se ale neustále rozrůstá, co se týká počtu uzlů, kanálů i jejich velikosti, a vzhledem k chytrým řešením jednotlivých peněženek je selhání platby spíše výjimkou a do budoucna se to bude jistě jen zlepšovat.

Poplatky

Obecně v celém bitcoinovém ekosystému jsou dva druhy poplatků:

- **On-chain poplatky** placené těžařům za zařazení dané transakce do bloku.
- **Lightningové poplatky** placené mezilehlým uzlům za přesměrování platby k cíli.

On-chain poplatek musíme zaplatit za samotné otevření a uzavření kanálu, jelikož se jedná o klasické transakce se zápisem do blockchainu. Po **otevření kanálu** však již platíme **jen lightningové poplatky**. Pokud mám s někým otevřený kanál napřímo (např. se známým hospodským z předchozích kapitol), žádné lightningové poplatky nikomu neplatím, transakce jsou zdarma. Pokud tedy vím, že někomu budu často platit (anebo naopak od něj budu často přijímat platby), rozhodně se mi vyplatí s ním mít otevřený platební kanál.

Co již ale zdarma není, je platba, která je přesměrována přes několik prostředníků, jelikož s cílovou osobou nemám napřímo otevřený kanál. Peněženka musí nejprve takovouto cestu z jednotlivých kanálů sestavit a poté si každý uzel po cestě účtuje poplatek za přesměrování. Tento poplatek se skládá ze dvou složek:

- **Pevná část**
- **Variabilní část**

Pevná část, jak již název napovídá, je poplatek, který si uzel účtuje za přesměrování platby nezávisle na tom, jak velkou částku přeposílá. Velmi častá hodnota v lightningové síti je 1 satoshi. Naopak variabilní část je závislá na velikosti přesměrované částky. Průměrná hodnota je dnes okolo 0,01 %. Každý uzel si může obě tyto části vybíraných poplatků nastavit libovolně (existují dokonce uzly, které přesměrovávají platby naprosto zdarma). Jedná se o vysoce konkurenční

prostředí, protože pokud nastavíte poplatky příliš vysoké, tak si sice můžete teoreticky vydělat, ale s největší pravděpodobností většina peněženek vybere alternativní cestu, která bude levnější. Majitelé uzlů tedy mezi sebou takovouto cenovou politikou nepřímo soupeří.

Pokud bychom tedy s poplatky výše chtěli odeslat kamarádovi 5000 satoshi přes 1 prostředníka, stálo by nás to 1,5 satoshi (1 satoshi jako pevná část a 0,5 satoshi variabilní poplatek). Tabulka níže popisuje několik ukázkových případů, kde pro lepší přehlednost budou poplatky vyjádřeny jak v satoshi, tak v korunách, tak i v procentech. Jako kurz bitcoinu budu používat 50 000 dolarů. (*Pokud čtete tuto knihu několik let po jejím vydání, tak jsem zvědav, jak moc levná či drahá vám tato cena připadá.*) Budeme kalkulovat s nepatrně reálnější situací, kdy platba procházela přes 2 prostředníky a každý si účtoval výše zmiňovaný 1 satoshi jako pevnou část poplatku a 0,01 % jako variabilní.

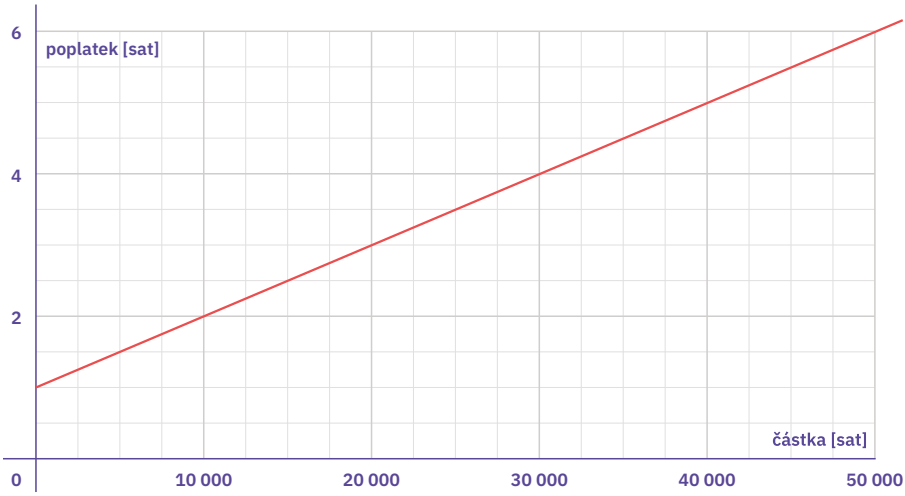
Částka [sat]	Částka [CZK]	Poplatek [sat]	Poplatek [CZK]	Poplatek [%]
1 000 sat	10 Kč	2,2 sat	0,02 Kč	0,22 %
5 000 sat	50 Kč	3 sat	0,03 Kč	0,06 %
30 000 sat	300 Kč	8 sat	0,08 Kč	0,03 %
100 000 sat	1000 Kč	22 sat	0,22 Kč	0,02 %
1 000 000 sat	10 000 Kč	202 sat	2,2 Kč	0,02 %

Jak je z tabulky patrné, lightningové poplatky jsou opravdu **velmi malé**. Berte však prosím výše uvedené hodnoty pouze jako příklad, reálně mohou být poplatky mírně rozdílné – jejich výše totiž závisí na zvolené cestě, kterou vaše peněženka vypočítá.

Pokud bychom chtěli odeslat stejné částky pomocí on-chain transakce (budu pro příklad počítat, že naše transakce má 1 vstup, 2 výstupy a těžařům jsme zaplatili 5 sat vB, tedy 5 satoshi za jeden virtuální bajt v bloku), poplatek by byl vždy 700 satoshi **nezávisle na odesílané částce**, jelikož při zápisu do blockchainu se platí pouze za místo, nikoliv za odesílanou hodnotu. Co to pro nás znamená? Pokud odesíláme menší částky, tak se jednoznačně vyplatí používat Lightning, naopak pro velmi vysoké částky bude vhodnější klasická on-chain transakce. Přesnou hodnotu, od které je vhodnější využívat transakci na 1. vrstvě, vám nesdělím, jelikož její výpočet je závislý na aktuálních poplatcích pro těžaře, vaší ochotou čekat na zápis do bloku a poplatcích pro lightningové uzly na cestě, kterou vaše peněženka zvolila. Obecně se dá ale říct, že pokud odesíláte více jak milion satoshi, je vhodné se zamyslet

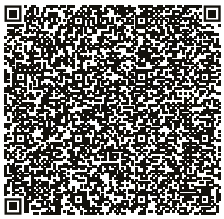
a spočítat si, zda by nebylo vhodnější použít on-chain transakci. Nejedná se zde pouze o poplatek, ale při takto vysokých částkách může být problém najít v lightningové síti sérii uzlů, které budou mít dostatečné prostředky pro přeposlání vaší platby k cíli.

Příklad velikosti a rozložení lightningových poplatků nám demonstruje následující graf:



PENĚŽENKY A PRAXE

Doposud jsme se zabývali teorií a zjednodušeně jsme si vysvětlili, jak Lightning funguje. Pokud se vám zdá celá tato technologie komplikovaná, tak věřte, že praktické placení pomocí Lightningu složité vůbec není. Zjednodušeně řečeno se jedná pouze o **skenování QR kódů**, nic víc. Asi každý z vás již platil přes QR kód u klasického internetového či mobilního bankovnictví. Místo opisování čísla účtu, variabilního symbolu a přesné částky pouze svým smartphonem naskenujete QR kód a všechny údaje se předvyplní. Na podobném principu funguje i Lightning, takže se není čeho obávat.



Existuje několik způsobů placení pomocí Lightning Network a ten zajisté nejpoužívanější využívá takzvaných **faktur**. Faktura není nic jiného než QR kód, ve kterém jsou uložené informace o příjemci a částce. Pokud tedy budu chtít přijmout od kamaráda 1 000 satoshi, tak si otevřu svoji mobilní peněženku, kliknu na *příjem* a volitelně specifikuji částku. Tento QR kód kamarádovi zobrazím (případně odešlu přes libovolný komunikátor) a on po jeho naskenování pouze potvrdí, že mi chce platbu odeslat. Ta mi během pár vteřin dorazí do peněženky, nic složitého. Dejte si však pozor na fakt, že klasické faktury mají omezenou platnost (většinou 1 hodina) a lze je zaplatit pouze jedenkrát, poté je nutné vytvořit fakturu novou. Dále se také můžete setkat s podobou faktury ve formátu textového řetězce začínajícího písmeny lnbc. Princip je úplně stejný, jedná se pouze o jinou interpretaci dat, protože někdy může být vhodnější odeslat takovýto textový řetězec než obrázek QR kódu.

Existují 2 typy peněženek, **custodia**l a **non-custodial**. Jaký je mezi nimi rozdíl? U non-custodial peněženek sami držíte vaše privátní klíče k vašim bitcoinům, kdežto u custodial řešení přenecháváte tuto činnost třetí straně. Představte si to na příkladu se zlatem, kdy non-custodial řešení znamená jít do obchodu, koupit si zlatou cihlu, přepravit ji domů a tam ji bezpečně uschovat. To vše znamená nějaké náklady a máte zde určitou zodpovědnost, kdežto u custodial řešení odešlete příkaz obchodníkovi se zlatem, ten cihlu koupí, uschová ji ve svém trezoru a vy dostanete pouze jakýsi certifikát, že máte u něj uloženo zlato o určité hodnotě. Jaké jsou výhody a nevýhody je pravděpodobně všem jasné. Co se týká Lightningu, tak tam custodial řešení fungují většinou dobře – ihned po nainstalování peněženky můžete začít přijímat i odesílat transakce a nic víc neřešíte. Máte zde však riziko toho, že kdyby jednoho krásného dne provozovatel vaší peněženky zmizel ze světa, ke svým prostředkům se nikdy nedostanete. Používání non-custodial řešení může být někdy o něco složitější, protože

čas od času je potřeba vytvořit platební kanál (za což se platí on-chain poplatky), a dále si musíte zajistit dostatečnou likviditu pro příjem. Co z toho plyne? Obecně jsou non-custodial peněženky **bezpečnější**. Pokud jste ale začátečník a zároveň neplánujete mít ve vaší peněžence uloženu velkou částku, jsou pro začátek custodial řešení přijatelná. Na non-custodial můžete přejít později, jakmile si Lightning více osaháte. Jen upozornění – toto doporučení platí pouze pro malé částky v rámci Lightning Network. Své on-chain bitcoiny **vždy** držte ideálně v **hardwarové peněžence**.

V následujících kapitolách se podíváme na 3 vybrané lightningové peněženky a stručně si je představíme. Jedna z nich je custodial a dvě non-custodial. Všechny jsou dostupné jak pro iOS, tak pro Android, takže nezáleží na typu vašeho telefonu. Na trhu existují i další peněženky, vyberte si tedy tu, která vám bude vyhovovat.

Než se však do přehledu peněženek pustíme, dovolte mi jedno malé upozornění. V dřívějších dobách byla obzvlášť mezi začátečníky vcelku oblíbená peněženka **BlueWallet**, která umožňovala spravovat jak on-chain bitcoin, tak i pracovat s Lightningem. Bohužel ke 30. dubnu roku 2023 byla její custodial část pro Lightning ukončena a zůstala možnost využití pouze s vlastním uzlem, čímž se stala pro většinu naprostých začátečníků nevhodnou. Její on-chain část však funguje beze změn. Vzhledem k jejímu rozšíření mi přišlo důležité na tuto změnu upozornit. Nyní ale pojďme k samotnému přehledu.

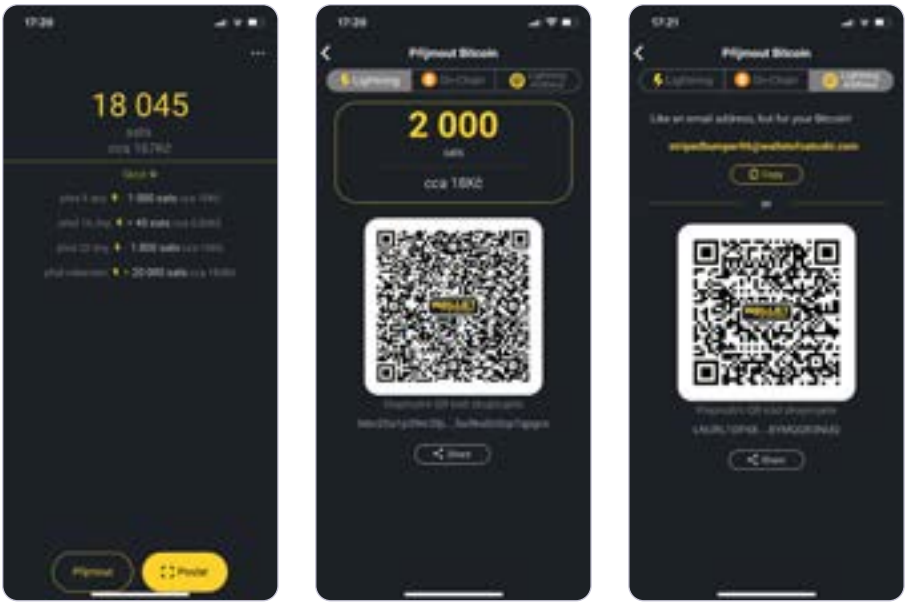
Wallet of Satoshi

Wallet of Satoshi je jednoduchá a uživatelsky přívětivá custodial lightningová peněženka. Doporučuji se do aplikace zaregistrovat a přihlásit, díky tomu se ke svým prostředkům dostanete i v případě, že ztratíte telefon. Ihned po spuštění můžete začít přijímat bitcoiny skrze Lightning. Po klepnutí na tlačítko přijmout máte na výběr tři možnosti. V prvním případě můžete jako obvykle vytvořit fakturu, to již známe. Druhou možností je příjem přes klasickou on-chain transakci (zde je poplatek 0,3 %), což vám vygeneruje bitcoinovou adresu. Poslední možností je tzv. Lightning Address. Tento koncept využívá technologie zvané LNURL, kterou do detailu rozebereme v pokročilé části knihy. Nyní nám bude stačit vědět, že v Lightningu lze odesílat prostředky i na adresu, která je ve formátu `jméno@doména.cz`. Nemusíte tedy složitě generovat faktury, stačí si tuto adresu uložit do kontaktů, a kdykoliv na ni odešlete nějaké prostředky, cílovému vlastníkovi se připíší do peněženky. Wallet of Satoshi vám takovouto adresu vygeneruje, aktuálně si však nemůžete jméno nikterak upravit.

Pro odesílání klikněte na tlačítko poslat a můžete naskenovat lightningovou fakturu, na kterou se ihned odešlou prostředky. Druhou možností je odeslání nějakých satoshi

na Lightning Address popsanou výše, případně je zde možnost i odeslat na on-chain adresu. Zde jsou ale účtovány velmi vysoké poplatky, proto doporučuji obecně tuto peněženku používat primárně na Lightning Network, k čemuž byla vytvořena.

Jedná se o custodial peněženku, je tedy vhodná pro menší částky a začátečníky.

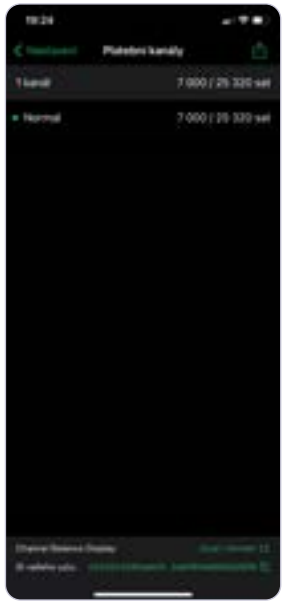


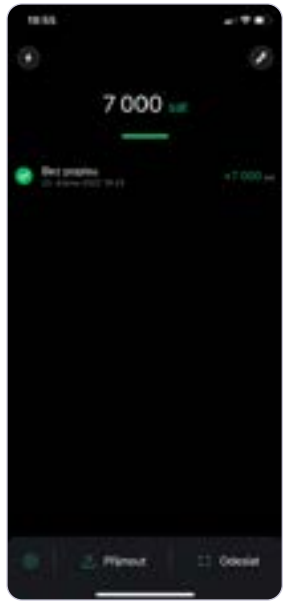
Phoenix

Phoenix je povedená a oblíbená peněženka od francouzských vývojářů, kteří si říkají ACINQ. Oproti předchozím peněženkám je tato non-custodial, což znamená větší kontrolu nad vašimi prostředky, ale zároveň i nějaké starosti navíc. Po založení nové peněženky je vám důrazně doporučeno si bezpečně zapsat 12 slov, která slouží k vygenerování vašich privátních klíčů. Oproti custodial řešením se tedy ke svým prostředkům dostanete i v případě, že by celá firma ACINQ včetně své infrastruktury jednoho dne zmizela ze světa.

Nenajdete zde žádný on-chain zůstatek, Phoenix slouží výhradně pro Lightning. Jedná se o lightweight implementaci kompletního lightningového uzlu, a tudíž je zpočátku nutné vytvořit alespoň jeden platební kanál. Pro první příjem je minimální hodnota 10 000 satoshi, z čehož se sebere 1 % jako poplatek (avšak minimálně 3 000 satoshi). Tento poplatek slouží k otevření privátního turbo kanálu (více o tomto názvosloví v pokročilé části knihy) ve směru od ACINQ uzlu

k vaší mobilní peněžence. Dokud máte nějaké prostředky pro příjem (lze vidět na prostředním screenshotu níže), je možné peněženkou přijímat a není potřeba vytvářet nové kanály, tedy platit tyto poplatky navíc. Jakmile byste ale vyčerpali všechny bitcoiny, které můžete přijmout, bude potřeba otevřít nový kanál a opět za to zaplatit. Původní kanál vám pořád zůstane a budete mít tedy dva. Z tohoto důvodu doporučuji napoprvé přijmout vyšší částku, jelikož tím vytvoříte kanál s větší kapacitou. Zároveň Vám ACINQ přidá nějaké prostředky na příjem ze svého. Přijímat lze i skrze on-chain, pouze kliknete na *přijmout* a poté *zobrazit bitcoinovou adresu*. Velikost poplatků je však stejná jako v případě výše. Phoenix je vhodná peněženka pro zkušenější uživatele a ty, kteří mají rádi věci pod svojí kontrolou.





Breez

Breez je další zástupce non-custodial peněženek. Pokud ji budete mít delší dobu vypnutou a některé funkcionality nebudou fungovat, tak si zkontrolujte, zda se vám v pravém horním rohu netočí kolečko symbolizující synchronizaci s bitcoinovou sítí. Většinou stačí jen pár desítek vteřin počkat.

Po technické stránce je velmi podobná peněžence Phoenix – opět si zde musíte nejprve vytvořit platební kanál. Tentokrát je ale poplatek přijatelnější – činí 0,4 % s minimem 2 000 satoshi. Tento poplatek je také nutno uhradit v případě, pokud po nějakém čase nemáte žádné prostředky pro příjem, což je vidět při

vytváření faktury (první screenshot). Stejně jako Phoenixu vám Breez při navazování kanálu ze svého uzlu přidá celkem slušnou příchozí likviditu ze svého. Breez ale nabízí navíc dvě zajímavé funkce.

Tou první jsou podcasty. Ty totiž můžete přímo z této aplikace poslouchat a v průběhu streamovat autorům satoshi (platí se libovolná částka za minutu). Pokud jste velmi spokojení s určitou částí, lze jim mimořádně odeslat větší jednorázovou částku tlačítkem *boost*.

Druhou zajímavostí je integrovaný platební terminál. Díky němu lze přijímat bitcoiny přes Lightning přímo do vaší mobilní peněženky například v obchodě. Lze si vydefinovat určité položky včetně ceny, případně je k dispozici klávesnice pro zadání konkrétní částky. Následně je vytvořena zákazníkovi faktura. Jedná se tedy o velmi jednoduchý způsob, jak přijímat Lightning ve svém obchodě například přes tablet.

BUDOUCNOST

Adopce Lightning Network

Lightning Network je platební síť, která zatím rozhodně není rozšířená mezi běžnými lidmi a znají ji spíše fanoušci Bitcoinu a ostatních kryptoměn. Pojďme se společně podívat na **čtyři podmínky**, které by musely nastat, aby se Lightning dostal mezi širokou veřejnost a mohlo se o něm uvažovat jako o alternativní metodě k dnešním platbám běžnými fiat penězi.

> **Fiat peníze**
>
> Alternativně také měna s nuceným oběhem, jsou peníze vytvořené mocí úřední, tedy státem. Jedná se o zákonné platidlo, kterým lze platit daně, splácet úvěry a nakupovat za něj zboží či služby. Příkladem je česká koruna, americký dolar nebo euro. Tyto měny již nějakou dobu nejsou kryty zlatem ani jinými komoditami, což umožňuje centrálním bankám a jiným institucím téměř neomezeně vytvářet peníze nové. Kvůli těmto faktům fiat peníze stále více neplní jednu z hlavních funkcí, a tou je uchování hodnoty. Běžným střadatelům se tak snižuje hodnota jejich naspořených peněz (inflace) a jsou často nuceni tyto peníze měnit za jiná aktiva – akcie, nemovitosti, komodity, kryptoměny, dluhopisy a jiné, aby uchovali jejich kupní sílu.

První podmínkou je nutná **větší adopce Bitcoinu** běžnými lidmi. V Lightning Network se jako aktivum používá bitcoin, čím více lidí tedy bude chápat výhody, které pro společnost přináší, tím více se rozšíří i povědomí o samotném Lightningu. Bitcoin se jako bezpečná a decentralizovaná platební síť za několik let své existence přetransformoval z neznámých internetových peněz pro IT nadšence přes spekulativní investiční aktivum až po alternativní peníze, o které se nyní zajímají banky, státy i instituce. Na druhou stranu je férové přiznat, že majorita lidí Bitcoinu stále nedůvěřuje a považuje ho za podvod, finanční bublinu nebo vysoce spekulativní aktivum, které slouží pouze kriminálníkům. Osobně ale vidím budoucnost v tomto ohledu velmi pozitivně. Proč? Pochopit principy a fundamentální výhody Bitcoinu trvá stovky hodin. Člověk musí mít povědomí o široké škále témat od historie peněz přes různé finanční revoluce až po dnešní fungování centrálních bank a úrokových sazeb, komerčních bank, principů inflace, vzniku peněz dluhem a podobně. Jinak řečeno, pokud člověk nechápe, jak fungují dnešní peníze (což mimochodem není vůbec jednoduché), nedokáže většinou ocenit přínosy Bitcoinu. Adopce tedy bude ještě nějakou dobu trvat. Na druhou stranu dobrou zprávou zajisté je, že

pokud člověk jednou Bitcoin nastuduje, už většinou není cesty zpět. Téměř každý bitcoinový fanoušek byl dříve v táboře těch, kterým se toto aktivum na první pohled zdálo přinejmenším pochybné a nedůvěřovali mu. Až investování určitého času do snahy pochopit Bitcoin je většinou vedlo k tomu, že plně porozuměli jeho výhodám a například si do něj průběžně spoří. Tímto si prošel téměř každý. Kolik ale znáte lidí, kteří by Bitcoinu věřili, fandili mu, znali jeho principy a v určitou chvíli by mu přestali důvěřovat a vrátili se zpět čistě k fiat penězům? Lidé se v zásadě dělí na dvě skupiny – na ty, kteří Bitcoinu rozumí a fandí mu, a ty, kteří jej ještě nestihli nastudovat. Čest výjimkám. Zároveň je potřeba se na věc dívat v horizontu několika let. Je naprosto pochopitelné, že moderní technologie (mezi které Bitcoin zajisté patří) jsou doménou spíše mladší části populace. Dnešní děti vyrůstají v digitální době a když si představíme strukturu obyvatel přibližně za 10 až 15 let, tak je velmi pravděpodobné, že používání různých digitálních aktiv obecně bude naprosto běžné. Zároveň se dnes o Bitcoin zajímají i samotné státy, některé z nich jej zavedly jako zákonné platidlo a jiné o tom uvažují. O Bitcoinu se diskutuje na půdě amerického Fedu (centrální banka), mluví o něm vcelku nekriticky americká ministryně financí Janet Yellenová a důsledky kryptoměn se probírají i na půdě Evropského parlamentu. Běžně se o něm píše v mainstreamových médiích a určitá část firem ho nakupuje do své finanční rozvahy. Bitcoin tu je a s největší pravděpodobností tu s námi i zůstane. Je pouze potřeba počkat na jeho větší adopci, ruku v ruce s tím totiž přijde i adopce Lightning Network.

Druhým faktem je nutnost **lepší technické vyspělosti** lightningové sítě. Aktuální stav je takový, že Lightning funguje, lze jím již platit ve vybraných obchodech či e-shopech, odesílat bitcoiny známým a kamarádům a platby v drtivé většině případů bez problémů prochází. Nalijeme si čistého vína – jeho technická vyspělost aktuálně není na té úrovni, aby mohl plně nahradit platební společnosti typu Mastercard či Visa. Čas od času nějaká platba neprojde, velmi výjimečně se může stát, že se transakce na delší dobu zasekne, občas bývá problém najít cestu k příjemci s dostatečnou likviditou a podobně. Oproti stavu před několika lety je to mnohonásobně lepší a použitelnější, pořád bych ale Lightning označoval za technologii, která je zatím neustále ve fázi vývoje. Dobrou zprávou je, že na řadu z těchto problémů již existují vymyšlená řešení, pouze se musí důkladně otestovat a implementovat. Určitě to zabere ještě několik let, než se vyladí veškeré technické drobnosti a síť bude po technické stránce použitelná pro masy lidí, rozhodně je ale vše na dobré cestě. Vývoj plně decentralizované platební sítě s takto vysokou propustností, výbornou bezpečností a anonymitou spolu s nízkými poplatky zabere nějaký čas, musíme být v tomto ohledu trpěliví. Pokud by vás zajímaly konkrétní novinky, které se pravděpodobně budou v rámci Lightning Network v budoucnosti implementovat, do detailu je rozebereme v pokročilé části knihy.

Třetí záležitostí, na které je nutné zapracovat, je **lepší uživatelská přívětivost** bitcoinového a lightningového ekosystému jako celku. I zde jsme ušli obrovský kus cesty, stále se ale najde mnoho lidí, pro které je obtížné si bitcoin koupit, správně uchovat a používat. Pokud se omezíme na Lightning a mobilní peněženky, což je s největší pravděpodobností jediné rozhraní, se kterým přijdou běžní uživatelé do styku, i zde máme obrovské rezervy. Na trhu je již větší množství mobilních peněženek, takže případný uživatel si může vybírat – u mnoha z nich je ale ještě potřeba zapracovat na uživatelské přívětivosti tím způsobem, aby ji zvládl používat opravdu každý. I ten, kdo toho o Lightningu po technické stránce moc netuší a chce jím pouze platit. Čas od času se totiž stane, že se uživateli zobrazí informace o nedostatečné příchozí likviditě, nutnosti vytvořit nový platební kanál, celkově práce s poplatky není vždy transparentní a podobně. Zkušení uživatelé Lightningu si s těmito problémy většinou poradí a zároveň si dokážou najít kvalitní peněženku, která má sama o sobě minimum případných chyb. To ale rozhodně neplatí pro běžné uživatele. Pokud vyvíjíme technologii, která by jednou mohla čistě teoreticky sloužit k běžným platbám pro masy lidí, tak musíme uživatele od těchto technických detailů odstínit. Nemělo by být nutností znát interní principy fungování Lightning Network, uživatel by správně nemusel rozumět ani pojmům jako platební kanál, kapacita, likvidita a rozdíly mezi on-chain a off-chain platbami. Běžný uživatel by měl mít ve své peněžence jeden zůstatek, velmi jednoduchou možnost dobití prostředků a dvě tlačítka – *přijmout* a *odeslat*. Samotné transakce musí fungovat minimálně v 99,99 % případů bez jakékoliv potřebné technické znalosti uživatele. Pouze tehdy lze o Lightningu uvažovat jako o alternativě k běžným platbám. Do té doby to bude technologie pro zkušenější uživatele, kteří dokážou výše uvedené problémy vyřešit. Je zkrátka nutné, aby vývojáři peněženek měli v hlavách známý slogan „It just works!“, který velmi rád používal Steve Jobs. Pokud totiž chceme nahradit fiat peníze Lightningem v oblasti prostředku směny, technologie musí být minimálně tak jednoduchá a spolehlivá, jako jsou dnešní platby přiložením telefonu či hodinek k platebnímu terminálu. I zde se podle mne jedná jen o otázku času a již nyní je vidět, že mnoho vývojářů na lepší uživatelské přívětivosti pracuje. Příkladem může být mobilní peněženka Muun, kde máte jeden zůstatek, a uživatel neřeší, zda platí on-chain nebo s využitím Lightningu. Pro zkušenější uživatele bude na trhu zajisté na výběr dostatek peněženek, kde budou mít mnoho pokročilých funkcí.

Poslední okolností, která by adopci Lightningu pomohla, je **selhání standardních fiat peněz v oblasti prostředku směny**. Dnes již téměř nikdo ekonomicky vzdělaný nepochybuje o tom, že běžné bankovky neplní dobře úlohu uchovatele hodnoty. Na druhou stranu jako prostředek směny zatím fungují (alespoň v České republice) výborně. I když celý platební systém je na pozadí velmi složitý, uživatelé si mohou digitálně a dnes již téměř instantně odesílat peníze skrze mobilní bankovnictví, případně platit bezkontaktně v obchodech. A obecně platí, že pokud něco funguje

dobře, lidé většinou nehledají alternativu. To se ale může změnit. V budoucnu by se totiž mohlo stát, že by vám banka, eventuálně stát, nějakou transakci zakázal, první náznaky se objevují již dnes. Případně pokud by nedošlo k samotnému zákazu, byly by po vás vyžadovány informace, kolik platíte, komu, proč a za co. Dle mého názoru by toto byl obrovský zásah do osobních svobod, protože každý občan by měl mít právo komukoliv jinému soukromě odeslat určité finanční prostředky, aniž by o tom musel informovat nadřazené instituce a čekat na případné schválení. Samozřejmě, pokud bude jednat v mezích zákona. Tato problematika je ale trochu složitější, podíváme se na ni proto v následující samostatné kapitole.

Aktuálně je Lightning Network v naprostém počátku své adopce. Pokud však dojde k širšímu povědomí o Bitcoinu mezi běžnými lidmi, vyřešíme drobné technické problémy, zlepšíme uživatelskou přívětivost a fiat peníze přestanou být soukromé a svobodné, tak jsem, co se týká širší adopce Lightningu, velký optimista.

Budoucnost fiat peněz

Peníze se neustále vyvíjí, to platí i pro ty vydávané státem. Pojďme se nyní společně podívat na to, jak by se mohly fiat peníze v následujících letech změnit, a co by to pro nás znamenalo. Na počátku bych chtěl upozornit, že následující text je můj čistě subjektivní názor, který není (a aktuálně ani být nemůže) podložen pevnými fakty. Budu se zde totiž snažit předpovědět budoucnost, která nemusí být úplně optimistická. Jedná se tedy o částečné „věštění z křišťálové koule“, které je podpořeno různými střípky informací pospojovaných dohromady spolu s trendy, které lze v globální ekonomice pozorovat. Rozhodně je možné, že se ve svém odhadu pletu, to nyní vyloučit nelze. Zároveň ale nejde následující text stoprocentně vyvrátit a prohlásit za nesmyslný, jelikož jaká bude realita, nám ukáže až budoucnost. Případů, kdy politici prohlásili určité informace za totálně vymyšlené a po krátké době se ukázaly jako pravdivé, je v historii lidstva velmi mnoho. Příkladem může být známý rozhlasový projev prezidenta Antonína Zápotockého 29. května roku 1953, ve kterém veřejnost ujišťoval, že k žádné měnové reformě nedojde a vše jsou jen fámy. Několik dní poté, konkrétně 1. června téhož roku, však k měnové reformě došlo, v jejím důsledku byly běžným lidem masivním způsobem znehodnoceny veškeré jejich úspory.

V následujícím textu budeme velmi často skloňovat zkratku **CBDC** (Central Bank Digital Currency) neboli digitální měnu vydávanou centrální bankou. Jedná se o nový koncept digitálních peněz, které by vydávala přímo centrální banka daného státu. Téměř všechny hlavní světové centrální banky se touto problematikou zabývají, včetně naší ČNB. Zpráva BIS (Bank for International Settlements, Banka

pro mezinárodní vypořádání) v roce 2019 provedla výzkum, ve kterém zjistila, že přes 80 % centrálních bank provádí alespoň teoretický výzkum CBDC, 10 % z nich je již ve fázi pilotního testování. Nejdále je v této oblasti ze států pravděpodobně Čína, kde se již v určitých provinciích CBDC testuje, dále také Švédsko a Bahamy.

Pojďme se nyní společně podívat na smyšlený příklad, jak by takové zavedení CBDC mohlo v praxi vypadat a jaké by to mělo důsledky. Aktuálně může mít občan peníze uložené buď ve formě hotovosti, což je z účetního pohledu pasivum centrální banky, nebo je mít v elektronické podobě na účtu u své komerční banky. CBDC by bylo jakýmsi průnikem, tedy digitálními penězi drženými u centrální banky, nazvěme jej například e-koruny. Zpočátku by rozhodně probíhala fáze testování, nelze tedy vyloučit tzv. vrtulníkové peníze pro skupinu obyvatel, která by se do tohoto experimentu zapojila. Dostat peníze za nic je na první pohled velmi lákavé, o zájemce by tedy určitě nebyla nouze a obliba CBDC by byla na vysoké úrovni. Spolu s tím by bylo prezentováno množství výhod, které tyto nové digitální peníze mohou přinést.

Jedním z hlavních cílů centrální banky je tvorba opatření a zásad měnové politiky daného státu, mimo jiné také udržování finanční stability. Čas od času se stane, že se ekonomika dostane do recese a že je potřeba její kola opět roztočit. Naopak v jiné době může být ekonomika přehřátá a je zapotřebí ji zchladit. Variabilní úročení CBDC by tak dávalo centrální bance větší flexibilitu a nové možnosti v oblasti monetární politiky. Dejme tomu, že by bylo zavedeno 15% úročení vkladů na digitálních účtech. K čemu by to vedlo? Hodně lidí by veškeré své úspory převedlo do CBDC a masivně by se snížilo utrácení, jelikož takovýto úrok by byl bezkonkurenční a mnoho lidí by omezilo spotřebu. Zhodnotit svoje úspory o 15 % za rok je těžký oříšek i pro zkušeného investora, zde by to navíc bylo bezrizikové. V opačném případě lze zavést ale i negativní úročení, které by naopak podporovalo útratu a čistě teoreticky dokázalo pomoct s přicházející recesí. Při negativním úročení (například -15 %) by totiž většina lidí začala ihned své CBDC utrácet, dokud ještě mají nějakou hodnotu, a kola ekonomiky by se roztočila. Tento nástroj by tedy přinesl nové a vcelku efektivní možnosti práce s monetární politikou pro centrální banky.

Další vlastností, kterou budou CBDC mít, je fakt, že budou tzv. programovatelné. To znamená, že jim lze přiřadit určité vlastnosti, jako je omezení platnosti nebo specifikace, za co je možné je utratit. Nemusíme chodit daleko do historie – při covidové krizi v roce 2021 prezident USA Joe Biden zavedl tzv. koronavirové balíčky, které mimo jiné zajišťovaly přímé platby Američanům v hodnotě několika desítek tisíc korun. Jinak řečeno, kvůli tomuto opatření na potlačení chudoby ve Spojených státech dostali lidé vcelku vysoké příspěvky „za nic“, klasický příklad vrtulníkových peněz. Problémem ale bylo, že se současnými schopnostmi peněz nebylo možné ovlivnit, za co je občané utratí. Řada z nich tak proto místo nákupu

zboží a pomoci lokální ekonomice zasažené celosvětovou pandemií takto získané peníze uložila anebo investovala. K přímé podpoře ekonomiky tak často nedošlo. U CDBC by ale šla omezit platnost těchto peněz. Jinak řečeno – každý obyvatel USA by dostal šek na 1400 dolarů, ale musel by jej utratit do půl roku, jinak by mu peníze zmizely. Zároveň by se dalo specifikovat, že tyto peníze se smí utratit pouze za určité přesně definované zboží, například podpořit kolabující restaurace. I když můžeme sáhodlouze filozofovat o tom, zda je tento přístup správný či nikoliv, je to jen krůček k tomu, aby podobné vlastnosti byly vynucovány v budoucnu globálně. Občané by poté byly nuceni své úspory průběžně utrácet a nikoliv spořit, čímž by se uměle oddalovala recese. Zároveň by mohly být zavedeny určité limity, například pokud byste utratili za aktuální měsíc za maso, benzín a letenky více, než je povoleno kvůli vaší uhlíkové stopě, a nákup dalších podobných aktiv by vám byl zamezen. Navíc při plné digitalizaci je možné vyžadovat důvod a ověření identity protistrany při sebemenší transakci, například při platbě kamarádovi či známému. Veškerá svoboda nakládání s vlastními penězi by byla minulostí. K čemu že je vůbec potřeba anonymity u podobných transakcí? Vzpomeňme si například na Kanadu v roce 2022 a protest řidičů kamionů proti povinným vakcinacím s názvem Freedom Convoy, který se později přeměnil na demonstrace proti vládě Justina Trudeaua. Důsledkem bylo doslova diktátorské využití zákona o nouzovém stavu, který dal vládě pravomoci zmrazit účty (tedy často veškerý majetek) lidem, kteří se demonstrace účastnili nebo jen účastníky finančně podpořili. Nechci zde řešit oprávněnost a smysluplnost tohoto protestu, pouze zde chci poukázat na fakt, že pokud nebude existovat svobodná a anonymní alternativa, může vás stát podpora jakékoliv protivládní demonstrace veškerý váš majetek a bude s vámi zacházeno jako s teroristou.

I samotná ČNB ve svých materiálech přiznává, že vznik CBDC byl do určité míry inspirován Bitcoinem. Mnoho lidí by tak mylně mohlo dojít k závěru, že Bitcoin a CBDC jsou velmi podobné. Opak je však pravdou. Kromě toho, že jsou obě aktiva digitální, tak ve své hlavní podstatě se jedná o naprosté protiklady. Bitcoin je decentralizovaná síť s pevně daným a dopředu známým počtem bitcoinů, nad kterou nemá žádná entita moc. Nezáleží na tom, jak jste bohatý, koho dokážete uplatit, jaké máte konexe ani kolik bitcoinů již vlastníte. Pravidla jsou dopředu daná, jasná, předvídatelná a nejslabší bitcoinový fullnode běžící v polorozpadlé chatce v Africe má stejné hlasovací právo při validování transakcí jako nejmodernější počítač v datacentru. Nad Bitcoinem nemá rozhodovací právo žádná organizace, stát ani instituce. Veškeré případné změny jsou důsledkem souhlasu většiny sítě. CBDC je přesným opakem. Zde má centrální banka, potažmo stát, plnou moc nad touto digitální měnou. Může určovat její množství, přidávat nové e-koruny do oběhu, určovat a průběžně upravovat její vlastnosti, případně lidem toto aktivum sebrat. Důležitou otázkou je také bezpečnost takto centralizovaného řešení před hackerskými útoky.

Pokud se podíváme do aktuálních materiálů o výzkumu CBDC, téměř všude se uvádí, že tyto digitální peníze by byly zpočátku směnitelné 1:1 s těmi běžnými, které dnes používáme. Měli bychom zde tak hotovost, CDBC a klasické účty u komerčních bank, vše vedle sebe. Pesimistické scénáře popsané výše by se tak daly obejít tím, že by člověk prostě a jednoduše CBDC nepoužíval. Musí ale mít tu možnost. Většina institucí ujišťuje, že v nejbližší době nedojde ke zrušení hotovosti (alespoň dokud o ní bude v rámci veřejnosti zájem), ale podobných ujištění jsme již slyšeli dost. Rozhodně je zapotřebí tuto situaci bedlivě monitorovat.

Připadá vám výše prezentovaný příklad velmi pesimistický a nereálný? Pesimistický určitě. CBDC mohou přinést i určitá pozitiva. Můžeme mezi ně zařadit garantované bezrizikové digitální oběživo, které se nevypaří, pokud zkrachuje vaše komerční banka, což by mohlo být důvodem k tomu, proč by lidé měli určitý obnos svých peněz uložený v podobě CBDC i bez pozitivního úročení. Dále také někteří lidé mohou vnímat centrální banku jako důvěryhodnější instituci a budou věřit spíše ČNB, že jim na základě zůstatku na účtu nebude volat pán z banky s nabídkou úvěru. V textu výše jsem se ale schválně zaměřil více na negativní vlastnosti a dotáhl je do extrému. Neříkám tedy, že se toto určitě stane, ale že **zde bude možnost podobná opatření technicky provést**. V takovém momentu by koncept Bitcoinu jako svobodného uchovatele hodnoty a Lightningu jako prostředku směny zvýšil radikálně svou důležitost pro společnost. A že je tento příklad nereálný? Ještě před pár lety by mi to celé připadalo jako konspirační teorie, proto níže přikládám několik odkazů na důvěryhodné zdroje, které z určité části výše uvedené obavy podporují.

Ředitel BIS o CBDC
Agustín Carstens, **generální ředitel Banky pro mezinárodní vypořádání** (jedná se o organizaci pro podporu mezinárodní měnové a finanční spolupráce, vlastně o takovou banku pro centrální banky), zde doslova říká: „*… u hotovosti například nevíme, kdo používá tuto konkrétní stodolarovou bankovku. Hlavní rozdíl u CDBC je ten, že centrální banka bude mít* ***absolutní kontrolu*** *nad pravidly a regulacemi peněz…*“
https://platbybudoucnosti.cz/bis-cbdc

MMF a budoucnost peněz
Mezinárodní měnový fond zde popisuje, jak by v budoucnu mohly záporné úrokové sazby (to znamená, že vám budou naspořené peníze doslova mizet pod rukama) pomoci v oživení ekonomiky v době různých krizí. Problém ale představuje hotovost, která nemá žádný úrok, ani negativní, ani pozitivní, takže stokoruna zůstane vždy stokorunou. V uvedené zprávě, která se zabývá novým elektronickým standardem peněz, jsou popsány způsoby, jak by bylo možné zavést negativní úroky i na hotovost, aby si do ní lidé neukládali svoje úspory, jakmile budou negativně úročeny jejich vklady

v digitální podobě. Navrhují sadu limitů, poplatků a omezení držení hotovosti – její samotné zrušení (jak někteří navrhují) prý zatím nutné není.
https://platbybudoucnosti.cz/mmf-penize

Bank of England a CBDC
Bank of England (centrální banka Velké Británie) vyzvala tamní ministry, aby rozhodli, zda by CBDC měly být „programovatelné", což by dávalo centrální bance kontrolu nad tím, jak peníze vlastník utratí. Cituji: *„... otevírá nové technologické možnosti, včetně programování: efektivně umožňuje jedné straně, jako je stát nebo zaměstnavatel, kontrolovat, jak příjemce utrácí peníze..."*
https://platbybudoucnosti.cz/boe-cbdc

Na závěr doslovná citace z webu České národní banky, konkrétně ze článku o CBDC.

> „CBDC by také mohly otevřít nové možnosti v oblasti měnové politiky. V hypotetickém případě, v němž by CBDC zcela nahradily hotovost, by umožnily překonání problému dolní hranice úrokových sazeb. Hotovost totiž představuje nulově úročenou alternativu k prostředkům na účtech úročených negativní sazbou, což limituje pokles měnověpolitických sazeb hlouběji do záporu. Centrální banky však deklarují, že hotovost budou podporovat, dokud po ní bude společenská poptávka (BIS 2020).
>
> Hampl a Havránek (2018) vidí v CBDC možnost, jak implementovat jiný teoretický nástroj nekonvenční měnové politiky, a sice přímou podporu spotřeby neboli Miltonem Friedmanem navrhované tzv. vrtulníkové peníze (helicopter drop of money). V recesi a v rámci svého mandátu cenové stability by centrální banka mohla připsat každému občanovi určitou sumu ‚digitální hotovosti' a zavést pobídky ke spotřebě spíše než k úspoře těchto prostředků. Využití CBDC by zároveň umožnilo aplikovat tento nástroj bez nutnosti koordinace s fiskální politikou, a nehrozilo by tak omezení nezávislosti centrální banky."
>
> Zdroj: https://platbybudoucnosti.cz/cnb-cbdc

Až budoucnost ukáže, jaká bude ohledně CBDC realita. Osobně doufám, že se výše popsaný scénář nenaplní. Každopádně je nutné situaci sledovat, protože podobné názory se ozývají stále častěji a brzy budou mít státy technologii, která by jim to technicky umožnila.

Přínos Lightning Network

Možná se vám stále Lightning jeví jako složitá a pochybná technologie, která není nikterak revoluční a nevidíte v ní do budoucna žádný přínos pro společnost. Částečně

váš názor chápu. Je ale zapotřebí se na celou věc podívat jinýma očima. Konkrétně Česká republika je země, ve které se z finančního hlediska žije velmi dobře. Každý občan zde může mít bankovní účet, platit lze hotovostí nebo elektronicky pouze přiložením mobilního telefonu, naše měna aktuálně netrpí hyperinflací a žijeme v relativně demokratické společnosti. To ale neplatí pro všechny státy na světě.

Když se vžijeme do role člověka, který žije v nějaké rozvojové zemi v Africe, případně ve Střední Americe, tak tam již všechno růžové být nemusí. Dle statistik Světové banky nemají 2 miliardy lidí na světě přístup k bankovnímu účtu. A právě pro rozvojový svět má Lightning zpočátku své adopce největší využití. Není totiž potřeba žádné banky, nepotřebujete žádné povolení, nezáleží na tom, jak moc diktátorský režim je ve vaší zemi nastolen. Bohatě vám postačí mobilní telefon s internetem a můžete anonymně a digitálně přijímat platby od lidí z celého světa. A to jak lokálně pomocí QR kódu na displeji telefonu, tak i přes internet. Důležitou roli v těchto zemích hrají také remitence, tedy odesílání finančních prostředků migranty ze země zaměstnání jejich rodinám do země původu. Díky Lightningu se dají na poplatcích ušetřit nemalé peníze, které rodinám zbudou.

Pojďme se nyní podívat na příklad. Já, jakožto autor této knihy, si chci vytvořit webové stránky, kde ji budu zdarma nabízet ke stažení. Na internetu si najdu mladého studenta z Turecka, který si při škole přivydělává tvorbou grafických šablon pro webové stránky. Při náhledu do jeho portfolia se mi jeho práce velmi zalíbí a rád ho podpořím. Přes videohovor se domluvíme na zadání. Za několik dní obdržím finální práci, kterou ještě během několika dalších iterací posuneme k naprosto skvělému výsledku, se kterým jsem maximálně spokojen.

Jakmile bych chtěl autorovi zaplatit běžným způsobem, musel bych vyřešit několik problémů:

- Já běžně operuji s českou korunou, autor grafiky s tureckou lirou. Budu tedy muset vyřešit směnu, která může být nevýhodná a její provedení zabere nějaký čas.

- Při takovéto platbě do zahraničí musím rozhodně počítat s velkými poplatky a tím, že než student v Turecku peníze obdrží, uběhne několik dní.

- Kvůli tomu, jaký je aktuálně v Turecku režim, zde hrozí riziko, že platba bude z nějakého důvodu pozastavena či zabavena.

- Příjemce navíc kvůli svému věku, omezení na straně místních zákonů anebo jen tím, že nebude z Turecka, ale z nějaké více rozvojové země, nemusí mít vůbec přístup k bankovnímu účtu.

- I kdyby platba nakonec proběhla, autor grafiky ji obdrží v turecké liře, jejíž inflace v době psaní této knihy dosahuje téměř 70 %, a tudíž je nevhodná pro dlouhodobé držení.

Oproti tomu zde máme Lightning Network. Stačilo by jednoduše vygenerovat QR kód, zaslat mi ho libovolným způsobem a po zaplacení by měl autor peníze během několika vteřin připsané ve své peněžence. Odpadla by zdlouhavá a nákladná směna mezi jednotlivými měnami, poplatky by byly v řádu haléřů, platba by byla naprosto instantní a nebylo by možné ji ze strany autoritativního režimu zabavit. Navíc není potřeba žádného bankovního účtu – postačuje mobilní aplikace a internet. Zároveň se nikdo, kromě nás dvou, nedozví detaily platby. Vzhledem k povaze plateb v Lightning Network by bylo možné provádět i částečné platby po každé iteraci, což by snížilo riziko podvodů na obou stranách. A já osobně bych raději z dlouhodobého hlediska držel bitcoin než krachující tureckou liru – směnu lze případně provést kdykoliv.

Někteří z vás mohou namítat, že podobného výsledku by se dalo docílit i s klasickou on-chain transakcí. A mají zajisté pravdu. Lightning nám zde ale přináší několik výhod:

- **Rychlost** – jednotky vteřin vs. desítky minut.
- **Poplatky** – jednotky halířů vs. jednotky až desítky korun v závislosti na aktuálním využití sítě.
- **Anonymitu** – pokud provedeme on-chain transakci, veškeré její detaily jsou navždy uložené v blockchainu, kde si je může kdokoliv přečíst a v některých případech i přiřadit ke konkrétním osobám.

Debata by ale neměla směřovat k tomu, zda jsou lepší on-chain transakce či Lightning Network. Obojí má svoje využití. Pro platby vysokých částek, kde není nutné instantní vypořádání jako je koupě bytu, pozemku či auta, je zajisté vhodnější použít on-chain transakci a počkat na vytěžení několika následujících bloků. Na druhou stranu, pro každodenní platby menších částek je Lightning Network vhodnější, jelikož umožňuje téměř neomezenou škálovatelnost. Oba tyto způsoby platby bitcoiny tedy mohou a zajisté budou fungovat vedle sebe.

> **Citace z nedávného výzkumu společnosti Arcane Research**
>
> „Lidé bez přístupu k bankovnímu účtu jsou obecně chudší a často mají zaměstnání s nízkým nebo dokonce nulovým příjmem. Tím, že jim umožníme finanční začlenění s využitím Bitcoinu a Lightning Network, vytvoříme pro tyto lidi nové možnosti, jak zlepšit jejich životy, což časem pravděpodobně přispěje k výraznému nárůstu jejich životní úrovně."
>
> Zdroj: https://platbybudoucnosti.cz/vyzkum-arcane

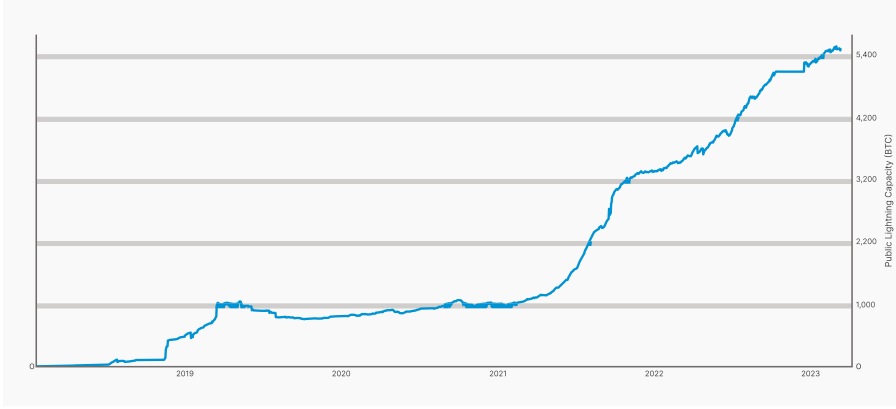
Řada lidí bude muset změnit myšlení. Po mnoho let jsme byli zvyklí, že máme k dispozici banky a další státní instituce, bez kterých by platby nemohly bezpečně probíhat. Nyní vznikly Bitcoin s Lightningem. Je už poté na každém z vás, zda více důvěřuje osobám, společnostem, bankám, institucím a politikům jako v případě fiat peněz, nebo matematice, kryptografii a počítačovému kódu jako v případě Bitcoinu. Tvrzení, že nelze bezpečně provádět platby bez bank, je podobné návrhu z 80. let minulého století, když se začal rozvíjet fax. Americká pošta tehdy přišla s návrhem, že se jedná o nový prémiový způsob odesílání zpráv, a požadovala, aby všechny faxové zprávy byly směrovány tomuto úřadu, který je před přeposláním zaregistruje a orazítkuje. Jak to ve výsledku dopadlo, dnes vidíme.

Často se také setkáte s názorem, že lidé nebudou ochotni svůj naspořený bitcoin utrácet, a proto Lightning Network nikdy nebude reálně fungovat. A i na tomto tvrzení je částečně něco pravdy. Zdravý rozum totiž velí, že je vhodné se prvně zbavit aktiva, které ztrácí na hodnotě (fiat peníze), a naopak neutrácet něco, co bude mít v budoucnu pravděpodobně větší hodnotu. Je to stejné, jako když si otevřete ledničku a jedno máslo prochází za týden, druhé za měsíc – po kterém sáhnete? Toto je validní argument, protože opravdu ekonomicky příliš nedává smysl převádět těžce naspořené bitcoiny ze své hardwarové peněženky na Lightning a utrácet je za produkty každodenní potřeby. Na celou věc se lze ale dívat jinak. Většina lidí dostává jedenkrát měsíčně mzdu od svého zaměstnavatele. Část těchto peněz utratí za povinné výdaje, jako může být splátka hypotéky, nájem, elektřina, voda, plyn atd. Druhá třetina padne na běžné každodenní výdaje – pivo v restauraci s kamarády, rodinná vstupenka na rozhlednu, zmrzlina pro dceru, parkovné a podobně. Pokud jim ještě část peněz zbude, tak ji někam odloží či investují, například do bitcoinu. Co lze na tomto postupu změnit, je to, že jistou část prostředků určených pro každodenní potřebu si směním na Lightning a budu jím platit. Dejme tomu, že mám na tuto běžnou útratu vyčleněno 5 000 Kč měsíčně, tak si 1 000 Kč směním na Lightning. Ve výsledku jsem tím nijak nezměnil rozložení svého rozpočtu a ani nemusím sahat do svých úspor. Pouze jsem 20 % prostředků každodenní útraty přeměnil do Lightningu a platím s využitím této technologie – využívám menších poplatků, vyšší anonymity a celkově pomáhám adopci.

Lightning Network, bez ohledu na to, zda uspěje či nikoliv, je zajisté revoluční technologie. Poprvé v historii lidstva tu máme možnost elektronicky, svobodně a anonymně platit ostatním lidem a nebýt závislí na žádné bance, instituci ani státu. U Lightningu nezáleží na hranicích země, na používané měně, národnosti, pohlaví, ani státním uspořádání. Všichni lidé si zde jsou rovni a mají přístup k opravdu svobodným penězům.

Na závěr se podívejme na graf historického součtu bitcoinů, které jsou aktuálně uzamčené v lightningových kanálech. Pokud by křivka pokračovala dále podobným tempem, o další adopci Lightningu bych strach neměl.

Kapacita lightningové sítě



PŘEDMLUVA K POKROČILÉ ČÁSTI KNIHY

Doposud jsme si Bitcoin a Lightning představovali zjednodušeně s využitím různých analogií (např. místo kryptografické hashovací funkce jsme používali fotky), aby jejich principy dokázali pochopit i čtenáři, kteří nemají hluboké technické znalosti. Základní část knihy byla tedy určena těmto lidem, případně těm, kterým stačí mít o Lightningu základní povědomí.

V pokročilé části knihy, která začíná touto stránkou, se již ponoříme více do hloubky. **První polovina** je zaměřena na **teorii** a do detailu se podíváme na principy fungování Lightning Network, představíme si celý lightningový stack, dozvíme se, jaké jsou možnosti plateb mimo základní faktury, popíšeme si různá rozšíření a nadstavby, podíváme se na určitá vylepšení Lightningu, která ještě nejsou implementována, a na závěr si probereme i různé útoky a zranitelnosti.

Předpokladem pro tuto část knihy je již pokročilejší znalost Bitcoinu a dále témat jako je kryptografie, matematika, síťové protokoly a obecně další navazující IT technologie. V tomto segmentu jsem se snažil předat maximum možných informací, ale zároveň nezahltit čtenáře hlubokými technickými detaily. Jinak řečeno, spousta principů se dá vysvětlit více způsoby – mohl bych tu například do detailu rozebírat jednotlivé síťové pakety, popisovat, na které konkrétní bajty se aplikuje operátor XOR s předchozí částí zprávy, jaké klíče slouží k podpisům, jaké pro šifrování a jakým způsobem se do detailu počítá HMAC odesílaných zpráv. To vše by se dalo ještě okořenit různými matematickými rovnicemi a důkazy použité kryptografie. I když věřím, že by si informace své publikum našly, tak jsem se rozhodl částečně od těchto technických detailů čtenáře odstínit a výše popsanou problematiku popsat slovně s využitím různých schémat a diagramů. Pokud by vás zajímal Lightning Network do úplných detailů, odkázal bych vás na specifikaci Lightningu zvanou BOLT (Basis of Lightning Technology).

V **druhé polovině** pokročilé části se podíváme na praktickou stránku, a to konkrétně provozování svého vlastního lightningového uzlu. Nejdříve si probereme, co všechno zahrnuje takový uzel provozovat, co to může uživateli přinést, a následně se zaměříme na technické detaily po hardwarové i softwarové stránce. Pokud očekáváte konkrétní návod příkaz po příkazu, jak si takový uzel zprovoznit, tak vás musím zklamat. Existují tisíce způsobů, jak takovýto uzel provozovat. My se zde spíše podíváme obecně na všechny tyto možnosti, protože každému bude vyhovovat jiná, podle jeho osobních preferencí. Po přečtení této části knihy byste tedy měli mít slušné povědomí o tom, co to obnáší provozovat lightningový uzel,

jaké jsou možnosti a na co vše si dát pozor. Následně se můžete rozhodnout pro svoji vlastní variantu. Zároveň kniha není vhodné médium pro popisování konkrétního návodu, jelikož technologie (a obzvlášť Lightning Network) se vyvíjí tak rychle, že by za velmi krátkou dobu byl takovýto konkrétní návod neaktuální.

Pokud by vám teoretická část (obzvlášť ke konci) přišla příliš složitá, zkuste přeskočit na praktickou část a k samotné teorii se můžete vrátit později.

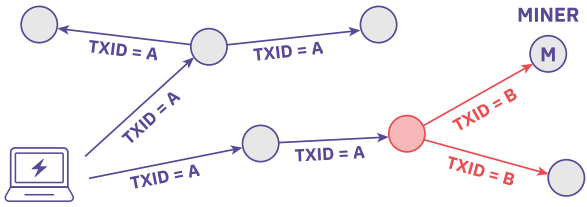
TEORIE LIGHTNING NETWORK

Transaction malleability a SegWit

Než se pustíme do samotných detailů Lightning Network, podíváme se na jeden problém, bez jehož vyřešení by bylo velmi obtížné Lightning naimplementovat. Jedná se o tzv. **transaction malleability**, který nám umožňuje změnit ID samotné transakce před jejím finálním uložením do bloku.

Každé transakci, kterou naše peněženka odešle do bitcoinové sítě, je přidělen unikátní identifikátor **TXID**, jenž je vytvořen jako výstup z hashovací funkce, jejímž vstupem je obsah transakce. TXID se dále využívá pro jednoznačnou identifikaci transakce – při vyhledávání v blockchain exploreru, monitorování dané platby peněženkou anebo odkazování se na konkrétní UTXO při vytváření transakce nové.

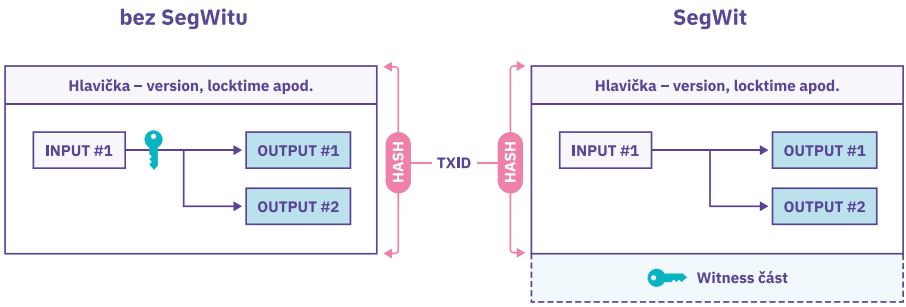
Problémem ale je, že jakýkoliv uzel bitcoinové sítě, který takovouto ještě nevytěženou transakci přijme (aby ji poslal dále), ji může **lehce pozměnit**, a to tím způsobem, že bude stále platná (bude odesílat stejné množství bitcoinů na stejnou adresu), ale její TXID bude **odlišné**. Jak toho dosáhne? Může využít faktu, že TXID je spočítáno jako hash celé transakce, kdežto kryptografický podpis (který by měl zabránit jakýmkoliv změnám) neobsahuje všechna data – primárně sám sebe a svoji délku, protože tyto hodnoty nejsou v době podpisu známy. Trik spočívá v tom, že mohu v transakci chytře upravit nějaké pole (jednoduše řečeno například změním délku podpisu ze 140 bajtů na 0140 bajtů, což jsou stejné hodnoty, ale vyprodukují jiný hash), a tím dosáhnu toho, že transakce bude stále platná, ale bude mít odlišné TXID, jelikož se změnil vstup hashovací funkce. Mimochodem burza Mt. Gox tvrdí, že právě transaction malleability byl důvod jejího vykradení – pravdu se však již asi nikdy nedozvíme.



Jak je vidět na obrázku výše – stačí jeden uzel, který pozmění TXID, a pokud budeme mít smůlu a takto upravená transakce se dostane k těžaři, který ji vytěží, úspěšně nám byl změněn identifikátor. Proč je to pro Lightning Network problém? Jak

bude do detailu vysvětleno v dalších kapitolách, při otevírání jakéhokoliv kanálu se vytváří dvě transakce – **funding** (otevírající), která slouží k definování toho, jak velkou kapacitu kanál bude mít, a poté **commitment** (finalizující), která se využívá ke spravedlivému rozdělení bitcoinů z kanálu v případě, že by např. jedna strana přestala spolupracovat. Funding transakce nesmí být do bitcoinové sítě odeslána dříve, než bude oběma stranami podepsána commitment transakce – jinak by zakládající člen kanálu riskoval ztrátu. A zde je kámen úrazu – v commitment transakci se jako vstup **odkazujeme na TXID funding transakce**, tudíž změna jejího ID by ji celou zneplatnila a nebylo by tedy bezpečné otevírat kanály. Existují sice teoretické možnosti, jak toto obejít, jsou však velmi komplikované, uživatelsky nepřívětivé a nikdy nebyly uvedeny v praxi.

Jaké je řešení? V roce 2017 byl aktivován upgrade Bitcoin protokolu s názvem **SegWit**, což je zkratka pro Segregated Witness, i když pro většinu lidí by byl pochopitelnější název třeba Separate Signature. Tento zpětně kompatibilní soft-fork přináší plno výhod jako více místa v bloku, menší poplatky, nový formát adres, ale řeší i transaction malleability, a to relativně jednoduchým způsobem. Před zavedením SegWitu byly unlock skripty (kód, který odemyká přístup k UTXO) ihned za každým vstupem a TXID se počítalo jako hash celé této transakce. Nově jsou všechny tyto skripty (jinak zvané také jako Witness) přesunuty na konec a TXID se počítá jako hash transakce bez těchto skriptů. Výsledkem tedy je, že jakákoliv jejich změna neovlivní výsledné TXID, a to zůstane od odeslání peněženkou až po vytěžení vždy stejné. Lze se na něj tedy spolehnout v navazujících transakcích, pokud ta předchozí ještě nebyla vytěžena. A díky tomu můžeme nyní v Lightning Network bezpečně otevírat kanály.



Historie

O principu platebních kanálů jako možnosti škálování blockchainu se na různých místech diskutovalo již téměř od samotného vzniku Bitcoinu – na nějakou další vrstvu

nad Bitcoinem odkazoval Hal Finney v e-mailové komunikaci se Satoshi Nakamotem již v roce 2008. Přesnější koncepci mu ale dali až Joseph Poon a Thaddeus Dryja v roce 2015 ve whitepaperu s názvem **The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments**. Ten detailně popisoval koncept bitcoinových smart kontraktů (HTLC) a obousměrných platebních kanálů, které společně umožňují bezpečné odesílání plateb skrze několik uzlů. To dovolovalo vznik decentralizované sítě pro rychlé, anonymní a levné platby využívající jako aktivum bitcoin.

Rok poté byla založena společnost Lightning Labs, která se dodnes zabývá rozvojem Lightningu a vývojem potřebných aplikací a služeb. Důležitý update Bitcoinu v roce 2017 s názvem SegWit víceméně odstartoval vývoj a používání Lightningu tak, jak ho známe dnes. Poté se začaly objevovat první reálně aplikovatelné implementace – Lightning Network Daemon, c-lightning a Eclair.

Mimochodem, i když dále v této knize budu Lightning Network popisovat jako nadstavbu nad Bitcoinem, lze ho provozovat nad jakýmkoliv blockchainem, který má požadované vlastnosti (popsané později). První lightningová transakce byla provedena nad Litecoinem v květnu 2017, přibližně 2 hodiny po aktivaci SegWitu. Autorem byl Christian Decker a využíval prvotní implementace c-lightning. Aktivací SegWitu na Bitcoinu přibližně o 3 měsíce později se ale vývoj přesunul primárně na tento blockchain.

Platební kanál

Platební kanál mezi dvěma stranami není nic jiného, než multisig adresa 2 ze 2 (existují k ní 2 privátní klíče a oba jsou nutné pro utracení bitcoinů) spolu se společnou domluvou, kolik prostředků připadá které straně. Kanály se navazují mezi jednotlivými uzly (počítače či servery, na kterých běží implementace Lightningu).

Než mohu s protistranou otevřít kanál, musím se s ní nejprve **síťově spojit**. K tomu potřebuji znát její ID, adresu a port. ID uzlu není nic jiného než veřejný klíč zapsaný v hexadecimálním formátu. Příklad ID uzlu:

03864ef025fde8fb587d989186ce6a4a186895ee44a926bfc370e2c366597a3f8f

Co se týká adresy, může se jednat o:

- IPv4
- IPv6
- Tor

Uzly mohou být současně dostupné například jak přes IPv4, tak i přes Tor – záleží na konfiguraci. Pro spojení si poté protistrana vybere ten komunikační protokol, který preferuje. Posledním potřebným parametrem je síťový port. Standardně se pro lightningovou P2P komunikaci používá TCP/9735, ale provozovatel uzlu si může tuto hodnotu libovolně změnit. Nepovinnou součástí identifikace uzlu je **alias** – jedná se o libovolný až 32 znaků dlouhý řetězec, který identifikuje váš uzel pro snadnější zapamatování. Alias je dobrovolný parametr, který si lze kdykoliv změnit a neexistuje žádná kontrola, zda ho již někdo nepoužívá.

Každý uzel je poté definován ve tvaru:

ID_uzlu@adresa:port

Například jeden z aktuálně největších a nejznámějších uzlů v lightningové síti s aliasem ACINQ má tuto identifikaci:

03864ef025fde8fb587d989186ce6a4a186895ee44a926bfc370e2c366597a3f8f@3.33.236.230:9735

Konstrukce kanálu

Jakmile jsme s protistranou spojeni, můžeme se pokusit si mezi sebou otevřít platební kanál. Pro potřeby následujícího textu budu používat dvě snad nejznámější osoby v IT světě – Alici a Boba. Při samotné konstrukci kanálu se odešle celkem 6 typů zpráv, které si nyní popíšeme podrobněji. Počítejme v našem případě, že Alice iniciuje otevření kanálu s Bobem.

První zprávou, kterou Alice Bobovi odešle, je `open_channel`. Udávají se tam parametry, s kterými chce Alice kanál otevřít, jako například nad jakým blockchainem, jaká bude velikost kanálu, kolik prostředků se přesune ihned po otevření k Bobovi (pokud vůbec nějaké), svůj veřejný klíč pro tento kanál, nastavení časových zámků apod. Pravděpodobně nejdůležitější a nejzajímavější hodnotou je právě cílová **velikost kanálu**.

Bob na to odpoví zprávou `accept_channel`, kde specifikuje své parametry on. V tomto případě by se mohlo stát, že Bob platební kanál nebude akceptovat – nejčastějším důvodem bývá, že má nastavenu minimální velikost platebního kanálu, která je vyšší než ta, s kterou plánuje kanál otevřít Alice. V našem případě ale k tomuto problému nedošlo, a tak můžeme pokračovat dál.

Nyní může Alice vytvořit tzv. **funding transakci**. Ta by šla do češtiny přeložit jako „otevírající transakce“, protože ale hodně termínů v Lightningu nemá český

ekvivalent a nejsem fanouškem přílišného počešťování, po zbytek knihy budu používat termín funding transakce. Pro její vytvoření jsou potřeba vstupy a výstupy. Výstupem bude multisig adresa 2 ze 2, kde se jako klíče použijí hodnoty, které si oba uzly vyměnily v předchozích zprávách. Pro utracení bitcoinů uložených na této adrese je tedy nutná **vzájemná kooperace Alice i Boba**, protože každý z nich drží jeden klíč. Jako vstup budou jednotlivé UTXO od Alice (protože ta otevírá kanál). V našem příkladě počítejme, že se Alice snaží vytvořit kanál o velikosti 1M satoshi (0,01 BTC). Je velmi nepravděpodobné, že by ve své peněžence měla přesně takto velké UTXO, bude si muset tedy poslat drobné zpět. Funding transakce by tedy mohla vypadat takto:

Funding transakce (vytvořená Alicí)			
Vstupy		Výstupy	
1,7M	Peněženka Alice	1M	Multisig adresa 2 ze 2 kontrolovaná Alicí i Bobem
		0,7M	„Drobné" zpět Alici

Alice ale nyní **nesmí** v žádném případě tuto funding transakci **odeslat do bitcoinové sítě**. Velmi snadno by se totiž mohlo stát, že by o svých 0,01 BTC navždy přišla. Bohatě stačí, že Bob půjde offline a nebude se s ním možné spojit. Protože Alice nezná jeho privátní klíč, nemůže bitcoiny získat zpět, ty by zůstaly takto „zaseknuté" v této multisig adrese napořád. Druhou variantou by bylo, že by si Bob uvědomil svoji pozici a vydíral Alici, ať mu dá něco za oplátku, že jí dovolí bitcoiny utratit. Bob totiž v tomto případě nic neriskuje, veškeré prostředky patří Alici. Prozatím si tedy Alice tuto transakci drží u sebe.

Nyní může Alice přistoupit k tvorbě druhé transakce, které říkáme **commitment** (finalizující). Jedná se o transakci, jež má jako vstup společný multisig výstup z nedávno vytvořené funding transakce, a rozděluje ho spravedlivě mezi Alici a Boba. Na začátku připadají všechny bitcoiny Alici, protože ona je tam vložila. Bobovi zatím z tohoto kanálu nepatří nic. Jak si můžete všimnout, commitment transakce utrácí bitcoiny z funding transakce, která ještě reálně není zapsaná v blockchainu, má ji pouze Alice. Toto je naprosto v pořádku, a právě zde využíváme SegWitu, který vyřešil transaction malleabillity. Na funding transakci se odkazujeme pomocí TXID (transakčního ID) a je tedy nutné, aby se nemohlo nikdy změnit. Aktuálně ale tato transakce není platná, protože utrácí prostředky z multisig adresy, ke které Alice nemá Bobův podpis.

Commitment transakce			
Vstupy		Výstupy	
1M	Multisig adresa 2 ze 2 z funding transakce	1M	Alice

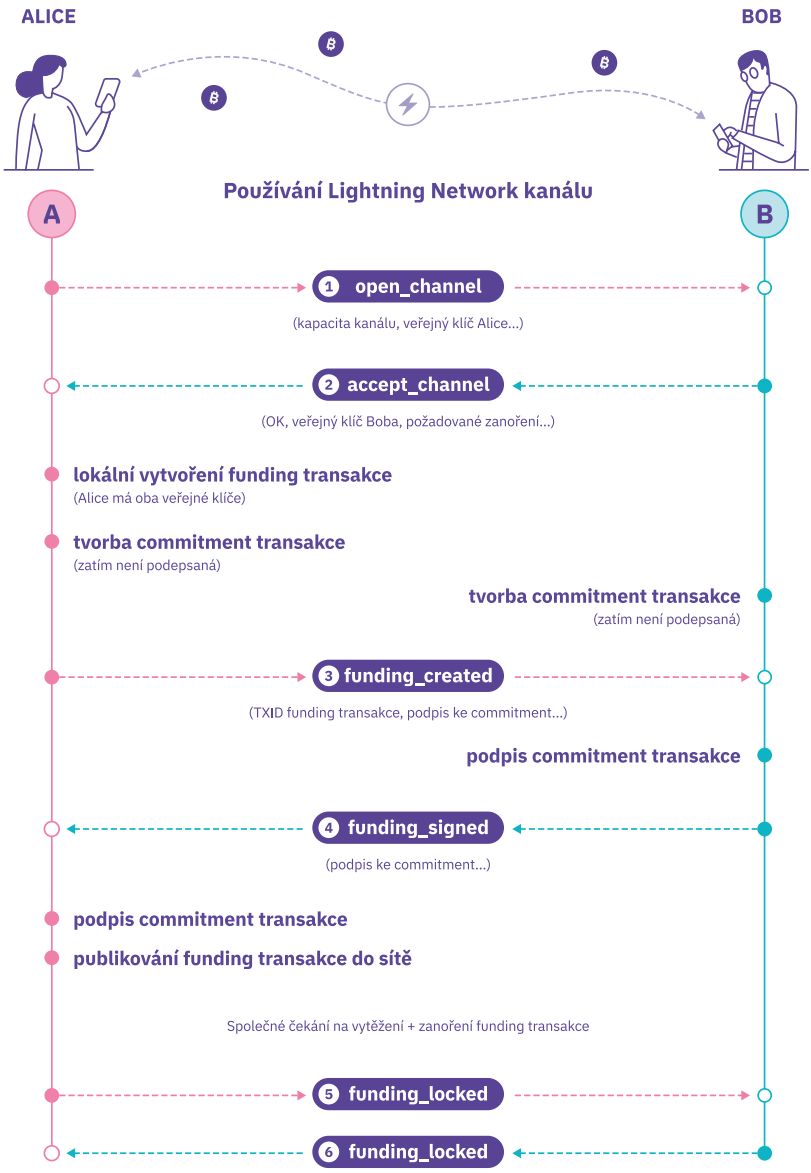
V tuto chvíli Alice odesílá Bobovi v pořadí třetí zprávu, které říkáme `funding_created`. Jak již název napovídá, Alice Bobovi sdělí TXID funding transakce spolu s tím, o jaký výstup se jedná (většinou výstup číslo 0 bude multisig adresa a číslo 1 budou bitcoiny zpět, ale může to být i jinak). Dále mu odešle svůj podpis k utracení odemčení funding transakce (přesněji řečeno k jejímu multisig výstupu). Bob tento podpis potřebuje, aby mohl podepsat vlastní commitment transakci – svůj podpis k ní má, ten od Alice právě získal.

Bob v tuto chvíli odpoví Alici zprávou `funding_signed`, ve které na oplátku odešle svůj podpis k funding transakci. V tuto chvíli si i Alice může dokončit svoji commitment transakci a **mají ji tak validní obě strany**. Důležité je si vždy tyto podpisy ověřit, to platí pro oba. Zároveň se v této zprávě domluví na unikátním ID kanálu (podobně jako existují ID uzlů).

> **ID kanálu**
>
> ID_kanálu = TXID_funding_transakce XOR index_výstupu

Teprve až nyní nastává ten okamžik, kdy je bezpečné odeslat funding transakci do bitcoinové sítě. Jakto? Alice totiž drží v ruce žolíka v podobě platné commitment transakce, která by v případě toho, že by Bob přestal komunikovat, odeslala veškeré prostředky jí zpátky (samozřejmě mimo on-chain poplatků). Bob toto ví, a tak se ani nemá smysl pokoušet o nějaký podvod. Alice tedy funding transakci odešle do bitcoinové sítě a společně čekají, až bude vytěžena a bude mít potřebný počet potvrzení. Poté si pouze společně vymění zprávy `funding_locked` a mohou kanál používat.

Princip konstrukce platebního kanálu znázorňuje následující diagram:


ALICE
BOB
Používání Lightning Network kanálu
A
B
1 open_channel
(kapacita kanálu, veřejný klíč Alice...)
2 accept_channel
(OK, veřejný klíč Boba, požadované zanoření...)
lokální vytvoření funding transakce
(Alice má oba veřejné klíče)
tvorba commitment transakce
(zatím není podepsaná)
tvorba commitment transakce
(zatím není podepsaná)
3 funding_created
(TXID funding transakce, podpis ke commitment...)
podpis commitment transakce
4 funding_signed
(podpis ke commitment...)
podpis commitment transakce
publikování funding transakce do sítě
Společné čekání na vytěžení + zanoření funding transakce
5 funding_locked
6 funding_locked

Posílání plateb kanálem

Kanál máme vytvořený, nyní pomocí něj zkusíme odeslat nějaké prostředky. V tuto chvíli je může poslat pouze Alice Bobovi, protože ona vlastní vše a Bob nemá nic, co by na její stranu kanálu mohl přesunout. Dejme tomu, že mu bude chtít Alice odeslat 100k satoshi.

Vzpomeňme si na naši commitment transakci, která rozděluje prostředky z kanálu mezi obě strany. Odeslání platby tedy není nic jiného než tvorba nové commitment transakce, která bude bitcoiny rozdělovat podle aktuálního stavu, tedy po platbě 100k satoshi bude tato částka patřit Bobovi, Alici zbude 900k.

Commitment transakce #1			
Vstupy		Výstupy	
1M	Multisig adresa 2 ze 2 z funding transakce	0,9M	Alice
		0,1M	Bob

Nyní lze provést další platbu, tentokrát odešle Alice Bobovi dalších 300k satoshi.

Commitment transakce #2			
Vstupy		Výstupy	
1M	Multisig adresa 2 ze 2 z funding transakce	0,6M	Alice
		0,4M	Bob

A do třetice nyní odešle Bob 200k zpět Alici.

Commitment transakce #3			
Vstupy		Výstupy	
1M	Multisig adresa 2 ze 2 z funding transakce	0,8M	Alice
		0,2M	Bob

Takovýchto transakcí můžeme za životnost kanálu provést neomezené množství a neplatíme za ně žádný poplatek. Vždy je ale nutné si transakci navzájem podepsat, aby byla platná. Tyto podepsané transakce zatím držíme u sebe a **neodesíláme** je do bitcoinové sítě. Takovýto konstrukt bez jakýchkoliv dalších podmínek má ale trhliny. Když se podíváme na naše commitment transakce z příkladu výše, tak jsou

všechny platné. Bob zjistí, že v té poslední mu přísluší 200k satoshi. Kdyby ale odeslal do bitcoinové sítě tu s pořadovým číslem 2, získal by o 200k satoshi více. Co mu v tom zabrání? Všechny tyto transakce utrácejí z jednoho vstupu (double-spend). Blockchain nám sice zajistí, že poté, co se jedna z nich vytěží, už žádná další nebude platná, ale bitcoinový blockchain neví, která z těch transakcí je aktuálně poslední. Budeme to tedy muset vyřešit jinak.

Zamezení podvodům

Abychom zabránili tomu, že kterákoliv strana může publikovat neaktuální commitment transakci, jež je pro ni výhodnější, využíváme zde tří faktů:

- Některé výstupy commitment transakce na sobě mají **časový zámek** (timelock), který definuje dobu, po níž nelze s těmito bitcoiny nakládat.
- Commitment transakce držená Alicí je **odlišná** od té, kterou drží Bob.
- Do celého procesu vstupují ještě **revokační klíče**, pomocí kterých můžeme předchozí commitment transakci zneplatnit.

Pojďme si nyní se znalostmi výše upravit naši commitment transakci. Doposud jsme používali u identifikace výstupů jména Alice a Bob. V Lightning Network sepoužívají termíny `to_local` a `to_remote`. Jak jsme si uvedli výše, každá strana bude mít odteď jinou commitment transakci – u té, co vlastní Alice, bude výstup `to_local` připadat jejím prostředkům, kdežto výstup `to_remote` bude Bobův. U Boba to bude přesně naopak. Výstup `to_local` (tedy to, co připadá mně) má na sobě vždy časový zámek (často zvaný `CSV_delay`), pro příklad 1 den, během kterého nemohu tyto bitcoiny utratit v nějaké další transakci.

Commitment transakce #3 (držená Alicí)			
Vstupy		Výstupy	
1M	Multisig adresa 2 ze 2 z funding transakce	0,8M	po 1 dnu → **`to_local`** (Alice)
		0,2M	**`to_remote`** (Bob)

Commitment transakce #3 (držená Bobem)			
Vstupy		Výstupy	
1M	Multisig adresa 2 ze 2 z funding transakce	0,8M	**`to_remote`** (Alice)
		0,2M	po 1 dnu → **`to_local`** (Bob)

Smysl těchto časových zámků je v tom, že pokud by se Bob pokusil o podvod a odeslal starou transakci #2, tak než si bude moct bitcoiny přisvojit, bude muset 1 den počkat. Toto dává Alici **čas na to zareagovat** a dokázat, že Bob podvádí. Pokud to prokáže, **veškeré** bitcoiny z tohoto kanálu připadnou jí bez ohledu na poslední stav. Tento mechanismus (nazývaný jako penalty transakce – do detailu probereme později) je zde z toho důvodu, aby se vůbec nevyplácelo se o nějaké podvody pokoušet.

Výstup `to_local` na sobě nemá pouze časový zámek, ale zároveň ještě revokační klíče, které slouží k tomu, abychom si předchozí commitment transakci zneplatnili.

Commitment transakce #3 (držená Alicí)			
Vstupy		Výstupy	
1M	Multisig adresa 2 ze 2 z funding transakce	0,8M	po 1 dnu → **`to_local`** (Alice) *nebo* ihned **`to_remote`** (Bob), pokud mám **`rev_key_B`**
		0,2M	**`to_remote`** (Bob)

Commitment transakce #3 (držená Bobem)			
Vstupy		Výstupy	
1M	Multisig adresa 2 ze 2 z funding transakce	0,8M	**`to_remote`** (Alice)
		0,2M	po 1 dnu → **`to_local`** (Bob) *nebo* ihned **`to_remote`** (Alice) pokud mám **`rev_key_A`**

Revokační klíče

Všem je jasné, že pokud chceme být fér, tak by se staré commitment transakce odesílat do bitcoinové sítě neměly. Jak tomu ale zabránit? Bob může říct, že si všechny předchozí transakce smazal – bude mu to Alice věřit? Těžko, potřebujeme lepší mechanismus. Trik spočívá v tom, že si při každé úpravě stavu kanálu (platbě) vyměníme revokační klíč k předchozí commitment transakci, a ten si uložíme. Co když se nyní Bob pokusí o podvod s publikováním transakce #2? Může to udělat? Samozřejmě, že může, transakce je platná. Když se ale podíváme na její konstrukt, tak zjistíme, že jeho výstup je zatížen časovým zámkem na 1 den. Alice nonstop monitoruje blockchain, a pokud takovouto transakci objeví, ihned využije revokačního klíče, který jí Bob zaslal, a jeho výstup mu sebere. Na rozdíl od výstupu pro Boba totiž u revokačního klíče není žádné časové omezení, a tak ho vlastně „předběhne“. Co se týká druhého výstupu (`to_remote`), tak

tam reálně žádná podmínka není, bitcoiny připadnou tedy ihned Alici. Výsledkem je, že se Bob pokusil o podvod publikováním staré commitment transakce, ale protože Alice neustále sledovala blockchain a ukládala si revokační klíče k předchozím transakcím, tak podvod detekovala a vše mu sebrala. Zde je nutné upozornit, že Alice musí být neustále online (nebo tuto činnost přenechat třetí straně, viz tzv. WatchTowers vysvětlené později), jinak by ji v tomto případě po 1 dnu Bob okradl.

Sestavení revokačních klíčů

Nyní se podíváme na kryptografické sestavení revokačních klíčů. Toto schéma je dle mého názoru zajímavé, pokud by ale na vás bylo již příliš náročné, klidně tento infobox přeskočte.

Nejprve trocha základů – privátní klíč `p` není nic jiného než náhodné číslo o velikosti 32 bajtů. Pro výpočet odpovídajícího veřejného klíče `V` musíme vynásobit bod `G` soukromým klíčem `p` (zde je myšleno násobení nad eliptickou křivkou). Bod `G` je tzv. generátor grupy, se kterou pracujeme, v našem případě tedy dříve zmíněná eliptická křivka.

Platí tedy, že

$$V = p * G$$

Dále můžeme využít faktu, že když sečtu dva klíče, tak dostanu třetí klíč:

$$V_A + V_B = V_C$$

V rámci zprávy `open_channel` odešle Alice Bobovi veřejný klíč `rev_base_pub` (ten zůstává identický po celou dobu životnosti kanálu), ke kterému má samozřejmě odpovídající privátní klíč `rev_base_key`. Zároveň mu stejným způsobem odešle veřejný klíč `rev_commit_pub`, ke kterému vlastní odpovídající privátní klíč `rev_commit_key`. Tento klíč se ale mění s každým revokováním commitment transakce. Stejným způsobem odešle Bob své klíče Alici, jen ve druhé zprávě zvané `accept_channel`. Výstup `to_local` u commitment transakce má vždy možnost utracení se znalostí revokačních klíčů a výstupem (cílem) je veřejný klíč `rev_pub`. Abychom tento výstup mohli utratit, musíme znát odpovídající privátní klíč `rev_key`. A teď to nejdůležitější – použitá kryptografie umožňuje **oběma stranám** si spočítat `rev_pub`, ale **pouze tomu čestnému** odpovídající `rev_key` (za podmínky, že protistrana podváděla). A jak je to technicky provedeno? Nejdříve si definujme skaláry `a` a `b`. Reálně se jedná o výstupy hashovacích funkcí, ale tyto detaily pro zjednodušení aktuálně vynechme. Veřejný revokační klíč `rev_pub` si mohu spočítat takto:

```
rev_pub = rev_base_pub * a + rev_commit_pub * b
```

Nyní můžeme použít výše uvedenou definici a veřejný klíč si v této rovnici nahradit privátním:

```
rev_pub = rev_base_key * G * a + rev_commit_key * G * b
```

Díky komutativní vlastnosti si to můžeme přepsat takto:

```
rev_pub = (rev_base_key * a + rev_commit_key * b) * G
```

Tím jsme vlastně dostali definici výpočtu privátního klíče `rev_key`:

```
rev_key = rev_base_key * a + rev_commit_key * b
```

V našem případě například Alice vlastní jen `rev_base_key` (sama si jej vygenerovala). Při úpravě stavu kanálu ale obdrží od Boba i „druhou část" k právě zneplatněné transakci, konkrétně `rev_commit_key`, což jí umožňuje si spočítat odpovídající privátní klíč `rev_key`, a je tak chráněna před publikováním revokované commitment transakce. Jelikož jsou privátní klíče pro každou stranu odlišné, označuji je `rev_key_A`, potažmo `rev_key_B`.

Nutnost rozdílných commitment transakcí

Představme si případ, že by Alice i Bob měli stejnou commitment transakci, která by strukturou vypadala takto:

<table>
<tr><th colspan="4">Commitment transakce (společná)</th></tr>
<tr><td colspan="2">Vstupy</td><td colspan="2">Výstupy</td></tr>
<tr><td rowspan="2">1M</td><td rowspan="2">Multisig adresa 2 ze 2 z funding transakce</td><td>0,8M</td><td>po 1 dnu → Alice
nebo
ihned Bob, pokud mám rev_key</td></tr>
<tr><td>0,2M</td><td>po 1 dnu → Bob
nebo
ihned Alice, pokud mám rev_key</td></tr>
</table>

Pokud by se jednalo o již zastaralou commitment transakci a například Bob by se pokusil o podvod jejím publikováním, mohl by sebrat prostředky (0,8M satoshi) Alici, protože dokáže spočítat `rev_key`. Tím by vlastně nebyl potrestán a ještě by se dokázal obohatit o bitcoiny, které mu nenáleží. Z tohoto důvodu je nutné, aby každá strana měla **rozdílnou commitment transakci** a pouze výstup `to_local` měl možnost utracení s využitím revokačních klíčů.

Úprava stavu kanálu

Nyní již tedy víme, že stav kanálu je vzájemná džentlmenská dohoda mezi dvěma uzly, kolik ze společných prostředků patří komu, kdy není nutné tento stav publikovat veřejně do blockchainu. Důvodem je, že protistrana drží v ruce podepsanou commitment transakci, díky které může své bitcoiny kdykoliv získat. Při úpravě stavu kanálu (odesílání nebo přijímání platby) se obě strany vlastně jen dohodnou, že si vzájemně podepíší nový stav (novou commitment transakci) výměnou za druhou polovinu revokačních klíčů k předchozímu stavu.

Jak ale k samotné úpravě stavu dojde? Jak asi správně tušíte, oba uzly si opět vymění několik zpráv. Vzhledem k tomu, jakým způsobem funguje internet, to ale nelze udělat naprosto atomicky. Musíme tedy v návrhu protokolu počítat s tím, že kterákoliv ze stran se může kdykoliv v průběhu odpojit a jít navždy offline nebo to alespoň předstírat. Za celou dobu výměny zpráv tedy nesmí dojít ke stavu, kdy by pro jednu ze stran bylo výhodnější přestat komunikovat a nějakým způsobem se na tom obohatit.

V našem případě bude Alice v obchodě u Boba a bude si tady kupovat novou mikinu za 100k satoshi. Celkově se vymění 4 zprávy a je zde nutné, aby komunikaci **začínal ten, kdo bude odesílat prostředky**, v našem případě Alice. Ta mu nejprve oznámí,

kolik bitcoinů mu chce zaplatit (100k satoshi), a odešle zprávu, ve které je mimo jiné ID vzájemného kanálu a **podpis k nové commitment transakci**, v níž je Bobův zůstatek o 100k vyšší. Bob si podpis ověří a přilepí ke své transakci. I když aktuálně vlastní 2 commitment transakce, které by beztrestně mohl odeslat do sítě, tak tím nijak nemůže Alici ohrozit. Buď by totiž publikoval tu starou, a to by okradl sám sebe, případně tu novou, která ale reflektuje správný stav. Alice tedy odesláním tohoto podpisu nic neriskuje.

V druhé zprávě posílá Bob Alici **svoji část pro tvorbu revokačních klíčů** k předchozí transakci (její zneplatnění). Ani toto nepředstavuje pro Boba riziko – ten totiž již drží podpis k nové commitment transakci, tu starou, která mu navíc připisuje menší zůstatek, prakticky nepotřebuje.

Ve třetí zprávě odesílá Bob Alici svůj **podpis k nové commitment transakci**. Ta má nyní sice dvě transakce, ale žádný podvod provést nemůže. Publikováním té staré, ke které ještě Bob nemá revokační klíče, by sice získala zpátky svých 100k satoshi, Bob by jí ale nepředal mikinu, protože by platba neproběhla úspěšně.

Jediné, co teď zbývá, je revokování zastaralé commitment transakce, kterou drží Alice. Ta ji zneplatní **odesláním své části revokačního klíče** Bobovi, který má nyní jistotu, že Alice původní commitment transakci již nemůže publikovat. Nyní tedy platba proběhla a Bob může spokojeně předat Alici mikinu v hodnotě 100k satoshi.

Uzavření kanálu

V ideálním případě bychom se uzavření kanálu měli pokusit vyhnout, protože budeme zbytečně platit on-chain poplatky a nebudeme moci nadále tento kanál používat na přijímání ani odesílání plateb. V některých situacích je ale uzavření kanálu vhodné, například je-li protistrana dlouhodobě offline a nedaří se nám ji jakkoliv kontaktovat. V tomto případě jsou totiž naše uzamčené bitcoiny v tomto kanálu nevyužité a vyplatí se je použít jinde. Dalším důvodem může být kanál, který kvůli špatné likviditě protistrany lze jen stěží využít na jakoukoliv platbu. Případně chceme část získaných prostředků přesunout na hardwarovou peněženku a pro konkrétní kanál již nemáme další využití.

Ať je váš důvod jakýkoliv, rozlišujeme 3 způsoby:

- Uzavření kanálu společnou dohodou.
- Vynucené uzavření.
- Pokus o podvod.

Uzavření kanálu společnou dohodou

Jedná se o **preferovanou formu** uzavření platebního kanálu. Iniciovat proces může kterákoliv strana, jež ve zprávě protějšímu uzlu odešle adresu (přesněji Bitcoin Script), na kterou si přeje přijmout svůj podíl z kanálu spolu s tím, jak vysoké navrhuje on-chain poplatky. Protistrana s poplatky buď souhlasí, anebo navrhne jinou výši. Zároveň odešle svoji adresu pro příjem zůstatku. Jakmile se domluví na výši poplatků a cílových adresách, oba si mohou sestavit **uzavírající transakci**.

Uzavírající transakce (oba drží stejnou)			
Vstupy		Výstupy	
1M	Multisig adresa 2 ze 2 z funding transakce	0,8M	ihned → **`Alice`**
		0,2M	ihned → **`Bob`**

Tato transakce je velmi podobná commitment transakci s tím rozdílem, že již neobsahuje žádné časové zámky ani revokační klíče – nejsou již potřeba. Obě strany v tomto případě dostanou svoje prostředky ihned po zahrnutí této transakce do bloku. Poplatky za uzavření platí ten, kdo kanál otevíral.

Vynucené uzavření

Tento typ uzavření využijeme v případě, že je protistrana offline či odmítá komunikovat. Vždy je vhodné se pokusit s provozovatelem uzlu nějakým způsobem spojit (na různých lightningových vyhledávačích, např. Amboss.Space lze na sebe zanechat kontakt), ale pokud se nám to nepodaří, nemáme jinou možnost. Technické provedení vynuceného uzavření je opravdu velmi triviální – stačí totiž publikovat do bitcoinové sítě poslední commitment transakci.

Commitment transakce #3 (držená Alicí)			
Vstupy		Výstupy	
1M	Multisig adresa 2 ze 2 z funding transakce	0,8M	po 1 dnu → **`to_local`** (Alice) *nebo* ihned **`to_remote`** (Bob), pokud má **`rev_key_B`**
		0,2M	**`to_remote`** (Bob)

V našem případě tuto transakci publikovala Alice, protože Bob nebyl dostupný. Bob získal svůj zůstatek ihned, jelikož výstup **`to_remote`** není nijak omezen, Alice si

musela na svých 800k satoshi 1 den počkat. Časový zámek je zde z toho důvodu, aby měl Bob případně šanci zareagovat, pokud by se nejednalo o aktuální commitment transakci. I když tento případ může svádět k pokusu o podvod publikováním neaktuální commitment transakce (Bob je přece offline, tak se nemůže bránit), rozhodně bych to nedoporučoval. Nikdy nemáte jistotu, zda Bob nemonitoruje blockchain (případně zda to nepřenechal třetí straně, jak si vysvětlíme později), a možnost potenciální ztráty všech prostředků za to nestojí. I v tomto případě platí on-chain poplatky ten, kdo otevřel kanál. Oproti preferovanému uzavření dohodou je to zpravidla mnohem dražší. Důvod je ten, že poplatky (rozdíl mezi výstupy a vstupy) musíme do commitment transakce zahrnout před podpisem, tedy v době tvorby této transakce při úpravě stavu kanálu. Nikdy ale nevíme, kdy budeme tuto transakci potřebovat použít a jak bude zaplněn mempool. Poplatky jsou tedy v commitment transakci **často velmi vysoké**. Zároveň platíme další poplatky za SweepOut transakci (případně více těchto transakcí, pokud jsme zrovna routovali platbu, jak si vysvětlíme později). Jaké z toho plyne ponaučení? Snažte se uzavírat kanál společnou dohodou a důkladně si promyslete, s kým si kanál otevřete.

Uzavření při pokusu o podvod

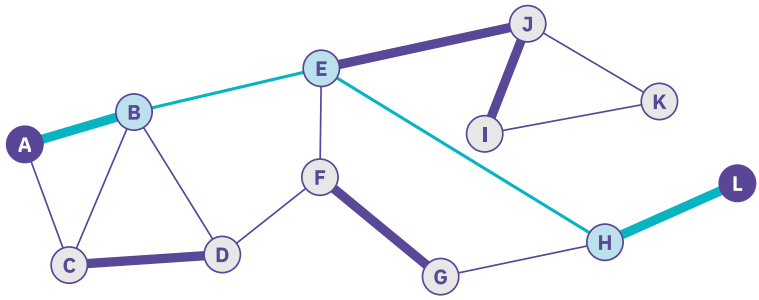
Vychytralý Bob odeslal do bitcoinové sítě commitment transakci, která není aktuální, a připisuje mu větší zůstatek než ta poslední. Protože Alice s touto situací počítá, ihned to detekovala a má 1 den (případně tolik, na jak dlouho byl nastaven časový zámek) na reakci. Po jednom dnu totiž prostředky připadnou Bobovi. Jaká bude její reakce? Vytvoří si **penalty transakci**, která bude mít jako vstup výstup z Bobem publikované commitment transakce, ale využije revokačních klíčů, kdy jednu polovinu vlastní a druhou získala od Boba při tvorbě nové commitment transakce. Tento výstup není zatížen časovým zámkem, a tak tímto způsobem prakticky předběhne Boba, který musí čekat. Důležité je to stihnout před vypršením časového okna, což občas může znamenat vyšší poplatky, ale zisk všech prostředků z celého kanálu za to v drtivé většině případů stojí. Bob se pokusil o podvod a jako trest mu Alice sebrala veškeré bitcoiny.

<table>
<tr><th colspan="4">Neaktuální commitment transakce #2 (publikovaná Bobem)</th></tr>
<tr><td colspan="2">Vstupy</td><td colspan="2">Výstupy</td></tr>
<tr><td rowspan="2">1M</td><td rowspan="2">Multisig adresa 2 ze 2 z funding transakce</td><td>0,6M</td><td><code>to_remote</code> (Alice)</td></tr>
<tr><td>0,4M</td><td>po 1 dnu → <code>to_local</code> (Bob)
nebo
ihned <code>to_remote</code> (Alice), pokud mám <code>rev_key_A</code></td></tr>
</table>

Penalty transakce (vytvořena Alicí)			
Vstupy		Výstupy	
0,4M	Výstup z neaktuální commitment transakce s využitím revokačních klíčů	0,4M	ihned → **`Alice`**

Routování plateb

Do principů routování (chcete-li přesměrování) plateb jsme částečně nahlédli v základní části knihy, úvod tedy bude vcelku stručný. Součástí Lightningu je tzv. gossip protokol, kterým se postupně šíří informace o nově vzniklých uzlech a navázaných kanálech. Právě díky těmto informacím si každý uzel postupně sestavuje graf všech existujících uzlů a kanálů mezi nimi. Co se týká kanálů, mezi veřejně známé informace patří primárně **celková kapacita** (na grafu níže zobrazeno tloušťkou jednotlivých kanálů) a **poplatky za přesměrování** (účtují si mezilehlé uzly po cestě). Pokud chce uzel **A** zaplatit uzlu **L** částku 100 000 satoshi a nemá s ním napřímo otevřený kanál, musí si najít optimální trasu v grafu. Prakticky se bude snažit sestavit cestu, která bude co nejkratší a po které budou nejmenší poplatky – v našem případě si uzel A zvolil modrou trasu.



Aktuální rozložení likvidity

Pokud hledáme optimální trasu, bylo by vhodné mimo jiné znát i aktuální rozložení likvidity. Vezměme v potaz kanál mezi uzly B a E o kapacitě 1M satoshi. Pokud bude B vlastnit pouze 50 000 satoshi a zbytek prostředků bude na straně E, platba o velikosti 100 000 satoshi touto cestou neprojde, jelikož uzel B nemá dostatek likvidity na své straně, kterou by mohl přeposlat. Prakticky to funguje tak, že tato trasa skončí chybou a uzel A se pokusí nalézt alternativní cestu.

> Existují dva důvody, proč se gossip protokolem nešíří informace o aktuálním rozložení. Tím prvním je soukromí – kolik ze sdílených prostředků patří vám a kolik protistraně je vaše soukromá věc. Druhým a pravděpodobně důležitějším aspektem je škálovatelnost. Bitcoin na první vrstvě má tu výhodu i nevýhodu, že o každé transakci ví všechny uzly v síti. To spolu s principem fungování blockchainu zajišťuje bezpečnost sítě, na druhou stranu je kvůli tomu omezeno množství transakcí, které tato vrstva zvládne vypořádat za jednotku času. Pokud bychom chtěli znát aktuální rozložení likvidity u všech kanálů v Lightningu, tak by jakákoliv platba (i pár desítek satoshi) musela být ihned propagována do celé sítě ke všem uzlům. Tím bychom propustnost ponížili na úroveň první vrstvy. Místo toho se pouze 1x za několik hodin odešle informace o nově vzniklých uzlech, kanálech a jejich parametrech. Pokud jedna trasa neprojde kvůli nedostatečné likviditě, zdrojový uzel si tuto informaci zapamatuje a příště zkusí alternativní trasu.

Navrhnout protokol, který zajistí přeposílání plateb lightningovou sítí, není triviální, máme na něj totiž hned několik požadavků:

- Za prvé musí být **bezpečný**. Kterýkoliv uzel se může kdykoliv pokusit o podvod a musí být zajištěno, že se tímto způsobem nelze obohatit o cizí prostředky.

- Dále musí být zajištěna **atomicita** transakce. Jinými slovy bitcoiny buď doputují do cíle, anebo zůstanou u odesílatele. I kdyby jakýkoliv uzel v libovolnou dobu přešel do offline stavu a již se nikdy nepřipojil (případně jen přestal komunikovat), platba se nesmí nikde trvale zaseknout.

- Důležitým aspektem je i **anonymita** – uzel uprostřed trasy by neměl vědět, kdo komu posílá bitcoiny, ani důvod platby. Jedinou informaci, kterou potřebuje znát, je částka, kterou má přeposlat.

- Pro uzly uprostřed trasy musí existovat nějaká **motivace**, aby částku přeposlaly dále a upravily si stav svých kanálů.

- O úspěšném přeposlání platby do cíle musí existovat nezpochybnitelný **důkaz**, aby cílový příjemce nemohl lhát, že mu žádné prostředky nedorazily.

- Toto vše musíme implementovat v rámci možností, které nám nabízí bitcoinová síť a její **pravidla**.

Úvod k routování plateb

Než se pustíme do technických detailů, představíme si princip routování plateb z nadhledu. V našem příkladě bude chtít Alice odeslat bitcoiny pomocí Lightningu Daniele, se kterou nemá napřímo otevřený platební kanál.

V prvním kroku Alice informuje Danielu, že jí chce odeslat 100k satoshi, které jí dluží. Nato si Daniela **vygeneruje náhodné číslo** o velikosti 32 bajtů zvané `payment_preimage`. Na tomto čísle není nic speciálního, pouze je tak velké, že jakékoliv uhodnutí třetí osobou je nereálné, a prozatím ho Daniela nesmí nikomu prozradit. Následně Daniela spočítá hash tohoto čísla, který nazýváme `payment_hash`. Jako hashovací funkce se v tomto případě používá SHA-256, platí tedy:

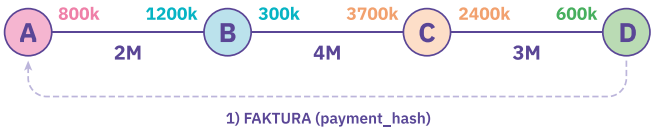
```
payment_hash = SHA256 ( payment_preimage )
```

Výše uvedený konstrukt si **dobře zapamatujte**, budeme se na něj odkazovat po zbytek knihy. Poté Daniela vytvoří **fakturu** – jedná se o speciální řetězec, ve kterém je zakódován veřejný klíč příjemce (Daniely), požadovaný obnos (100k satoshi), čas expirace faktury, `payment_hash` a další atributy, jež do detailu probereme v samostatné kapitole. Daniela tuto fakturu předá Alici, nejčastěji pomocí QR kódu. Faktura k Alici neputuje lightningovou sítí, ale libovolným komunikačním kanálem – zobrazením na webu, platebním terminálu nebo případně na displeji mobilního telefonu. Celý tento proces může být automatizovaný, faktura se často generuje bez zásahu člověka, například platební bránou na e-shopu, pokud si uživatel zvolí placení pomocí Lightningu.

> **Platba bez faktury**
>
> V případě uvedeného výše je nutné, aby někdo (uživatel nebo platební brána) vygeneroval fakturu. Až poté může proběhnout samotná platba. I když se jedná zatím o nejčastější způsob placení přes Lightning, v některých případech by bylo vhodné provést platbu tak, že iniciátorem bude odesílatel (např. donate na webové stránce). K tomu slouží techniky `keysend` nebo `LNURL-Pay`, které si probereme později. Pro tento příklad budeme počítat s tvorbou faktury.

Uvažujme čtveřici uzlů – Alice, Bob, Cyril a Daniela. Jak znázorňuje obrázek níže, Alice má vytvořený kanál s Bobem, Bob s Cyrilem a konečně Cyril s Danielou. O ostatních kanálech nebudeme zatím uvažovat. Každý kanál má jistou kapacitu a níže jsou vidět i aktuální zůstatky na jednotlivých stranách. Alice obdržela ve

faktuře veřejný klíč (ID) uzlu Daniely a využila znalostí grafu sítě k tomu, aby si spočítala, že nejkratší a nejlevnější cestou k Daniele je ta přes Boba a Cyrila. (*Mimochodem maximální částka, kterou lze touto trasou přeposlat, je 300k satoshi, což je maximální limit ve směru od Boba k Cyrilovi.*)



Alice má zároveň informaci o poplatcích, které si vezmou Bob a Cyril. Mezilehlé uzly si je účtují za přeposlání prostředků ze své strany kanálu k partnerovi (nikoliv naopak, z „reverzního" přesunu nic nemají). Poplatky mohou mít libovolnou hodnotu a skládají se z pevné části zvané `base_fee`, která není závislá na routovaném množství bitcoinu, a dále z variabilní části, která se označuje termínem `fee_rate`. Tato druhá část je závislá na přeposílaném obnosu a nejčastěji se vyjadřuje v `ppm` (parts per milion – kolik satoshi si ponechám za přeposlání 1 milionu satoshi). Jednoduchou matematikou si spočítáme, že například 100 ppm = 0,01 %. Bob i Cyril si budou shodně účtovat 1 satoshi za přeposlání platby (`base_fee`) a 100 ppm jako `fee_rate`. Při platbě 100k satoshi tedy budou poplatky u Boba 1 + 100 000 * 0,000 1 = 11 satoshi. Ty samé poplatky si účtuje Cyril. Alice si tedy musí připravit **celkovou částku 100 022 satoshi**.

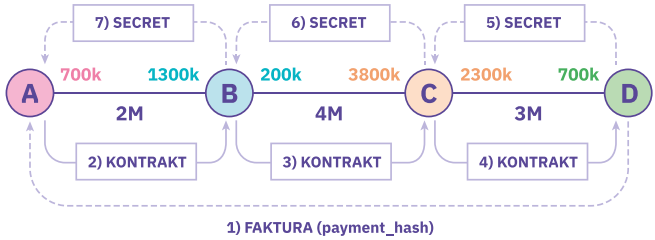
> **Jednotka msat**
>
> Na první vrstvě je nejmenší jednotkou 1 satoshi, který je definován jako jedna stomiliontina bitcoinu. Lightning však nativně pracuje v milisatoshi, který je definován jako 1/1 000 satoshi. Jakmile dojde k uzavření kanálu a vypořádání na první vrstvě, proběhne automaticky zaokrouhlení na nejbližší celý satoshi.

Nyní pojďme k samotné platbě. Alice uzavře s Bobem kontrakt, ve kterém se zaváže, že mu přenechá ze svého zůstatku 100 022 satoshi, pokud získá **důkaz**, že zaplatil Cyrilovi 100 011 satů. Rozdíl v částkách je poplatek (motivace pro Boba) a daným důkazem je `payment_preimage` vygenerované Danielou (náhodné číslo, které se po zahashování rovná `payment_hash`). Bob tedy obdržel od Alice `payment_hash` a informaci, za kým má jít dál. Velmi podobný kontrakt uzavře Bob s Cyrilem – tentokrát mu Bob slíbí, že mu

přepošle 100 011 satoshi, pokud získá `payment_preimage`, a že by se měl jít zeptat Daniely.

Konečně se naváže poslední kontrakt mezi Cyrilem a Danielou. Cyril jí slíbí, že jí přepošle 100 000 satoshi, pokud mu dodá takové `payment_preimage`, které se po zahashování bude rovnat hodnotě `payment_hash`. Daniela toto číslo jako jediná má, sama si ho totiž vygenerovala, a díky `payment_hash` ví, ke které faktuře patří. Nemá tedy nejmenší problém ho Cyrilovi sdělit, protože pokud to udělá, dostane od něj 100 000 satoshi. Po sdělení `payment_preimage` si ho Cyril ověří (zda `payment_hash = SHA256 ( payment_preimage )`), a pokud vše sedí, upraví se stav kanálu mezi Cyrilem a Danielou tak, že Daniela bude vlastnit o 100 000 více.

Nyní celý postup pokračuje obráceně – Cyril již zná `payment_preimage` a může u Boba požadovat 100 011 satoshi. Ten mu je po ověření vydá a sám se půjde dožadovat svých 100 022 satoshi u Alice. Finálním krokem je přeposlání této částky od Alice směrem k Bobovi, čímž je platba úspěšně dokončena. Alice nyní vlastní důkaz, `payment_preimage`, že platba proběhla v pořádku. Jediným způsobem, jak by se toto číslo k Alici mohlo dostat, je, že se postupně přeposlalo vytvořenou cestou a platbu tak lze považovat za dokončenou.



Vaše celkové prostředky jsou součtem „vašich stran“ přes všechny kanály, které máte otevřené. Na obrázku výše si můžete všimnout, že Alice vlastní nyní o 100k satoshi méně (v kanálu s Bobem). Naopak Daniela si těchto 100k satoshi připsala (poplatky nyní neuvažujme). Bob i Cyril mají stejně bitcoinů jako předtím, pouze se jim změnilo rozložení likvidity – v jednom kanálu 100k ztratili, naopak v druhém 100k + poplatky získali.

Tento návrh má ale prozatím několik problémů. Prvním z nich je, že například Bob si aktuálně nemůže být jistý, že mu Alice předá 100 022 satoshi, pokud sežene `payment_preimage`. Mohla by ho naprosto ignorovat, vydírat anebo se případně může stát, že se s ní nebude možné spojit. V tuto chvíli by na tom tratil Bob, který již odeslal prostředky Cyrilovi. Toto se řeší tak, že Alice „uzamkne“ část svých bitcoinů

do speciálního typu smart kontraktu. Ty jsou poté automaticky přesunuty Bobovi, pokud `payment_preimage` získá. Alice toto nemůže následně odvolat. I kdyby nebyla dostupná, Bob svoje prostředky dostane, pokud splní podmínky (sežene `payment_preimage`). Drobnou nevýhodou je pouze to, že Alice tyto prostředky nemůže po dobu uzamčení používat, ale bez této podmínky by to nefungovalo. Detailní popis tohoto smart kontraktu si probereme v následující podkapitole.

Máme nyní vše vyřešeno? Zdaleka ne. Co kdyby se Daniela rozhodla `payment_preimage` nevydat? Ekonomicky to sice nedává smysl, protože by nezískala svých 100k satoshi, ale stát se to může. Případně dojde k havárii jejího uzlu, nebude mít zálohu a uzamčené bitcoiny zůstanou ve smart kontraktu napořád – nezíská je ani Bob ani Alice, platba se zasekne. Jak se toto řeší? Absolutním časovým zámkem.

> **Časové zámky**
>
> V bitcoinovém ekosystému máme 2 druhy časových zámků (timelock). Tím prvním je **relativní časový zámek**, s kterým jsme se setkali již při úpravě stavu kanálů, kdy výstup transakce `to_local` je z bezpečnostních důvodů pozdržen o určitý počet bloků, například 144 (1 den). Tento časový zámek začíná běžet v době vytěžení transakce do bloku.
>
> Druhou možností je **absolutní časový zámek**, který je dán opět počtem bloků. Tentokrát ale jeho čas nezačíná vytěžením transakce, nýbrž se spouští ihned. V transakci je tedy omezení na **konkrétní datum a čas** (zapsáno v aktuální výšce bloku).

Každý smart kontrakt je opatřen absolutním časovým zámkem. Ten zjednodušeně řečeno říká, že Bob musí získat `payment_preimage` do dnešního dne do 17:00. Pokud jej sežene, dostane v něm uzamčené prostředky. Pokud se mu to z jakéhokoliv důvodu nepodaří, bitcoiny se navrátí Alici. Takže i kdyby se Daniela rozhodla platbu bojkotovat, prostředky se po určitém čase Alici vrátí a nezůstanou uzamčeny navždy. Důležitou vlastností ale je, že v dalším smart kontraktu mezi Bobem a Cyrilem musí být podmínka časově přísnější. Cyril bude mít na sehnání `payment_preimage` čas pouze do 16:40 a Daniela dokonce do 16:20. (*Časový rozdíl je konfigurovatelný, 20 minut je zde uvedeno pouze jako příklad.*) Proč je to nutné postupně snižovat? Představte si situaci, kdy by všechny uzly měly stejný limit do 17:00. Cyril by od Daniely získal `payment_preimage`, ale protože nebude hrát zrovna úplně fér, odešle toto číslo Bobovi v 16:59:59,9. Splní tím podmínku, bitcoiny mu automaticky připadnou, ale v Bobových silách nebude v tomto šibeničním termínu kontaktovat Alici a domluvit

se s ní. Bob by jí to tedy odeslal třeba o 2 vteřiny později. V Bitcoinu a Lightningu vládne matematika, na žádné dohady zde není prostor – Bob to nestihl, tudíž by se prostředky vrátily Alici. A přesně z tohoto důvodu je nutné časový zámek po cestě postupně snižovat.

Do třetice tu máme poslední problém – pokud by smart kontrakt mezi Alicí a Bobem byl pouze o znalosti `payment_preimage` ve prospěch Boba a časovým zámkem ve prospěch Alice, mohla by nastat nemilá situace. Daniela, která zpočátku jako jediná `payment_preimage` zná, by ho mohla použít nejen pro zisk svých 100 000 satoshi od Cyrila, ale dále by se obohatila i o prostředky, které by měly připadnout Cyrilovi a Bobovi. Udělala by to tak, že by tento smart kontrakt „vybrala" dříve, než jim tajné číslo sdělí. Reálně se toto řeší následobně: pro sebrání bitcoinů (po publikování commitment transakce) ze smart kontraktu nestačí pouze `payment_preimage`, ale je potřeba i podpis té strany, pro kterou byl kontrakt určen. Daniela nevlastní privátní klíč Boba ani Cyrila, proto nemůže zkonstruovat digitální **podpis** a tímto způsobem se neférově obohatit.

Několik doplňujících informací na závěr této podkapitoly:

- Uzel uprostřed cesty (např. Bob) ví pouze, kdo mu peníze posílá (Alice), komu je má přeposlat dále (Cyril) a o jak velkou částku se jedná. Bob ale už netuší, jestli prostředky odesílá Alice nebo někdo před ní. Stejně tak nemá šanci zjistit, zda je Cyril cílovým příjemcem či až někdo za ním. Díky tomuto konceptu (který si podrobněji popíšeme později) je zaručena **anonymita transakcí**.

- Všechny uzly po cestě **musí být online** a musí spolu kooperovat. Pokud by tomu tak nebylo, tak tato cesta selže a odesílatel musí najít alternativní. To, že určitá trasa selže, je vcelku normální a časté, nejčastějším důvodem je nedostatek likvidity pro přeposlání na mezilehlých uzlech.

- `payment_hash` lze považovat za jednoznačný **identifikátor** platby.

- Naopak `payment_preimage` jako nepopiratelný **důkaz** o úspěšném provedení platby.

HTLC

Poté, co Alice obdržela fakturu a vypočítala si cestu k Daniele, si upraví svoji commitment transakci s Bobem, a to tak, že si do ní přidá nový výstup.

Commitment transakce #652 (držená Alicí)			
Vstupy		Výstupy	
2M	Multisig adresa 2 ze 2 z funding transakce	0,8M	po 1 dnu → **`to_local`** (Alice) *nebo* ihned **`to_remote`** (Bob), pokud mám **`rev_key_B`**
		1,2M	**`to_remote`** (Bob)

Commitment transakce #653 (držená Alicí)			
Vstupy		Výstupy	
2M	Multisig adresa 2 ze 2 z funding transakce	0,7M	po 1 dnu → **`to_local`** (Alice) *nebo* ihned **`to_remote`** (Bob), pokud mám **`rev_key_B`**
		1,2M	**`to_remote`** (Bob)
		0,1M	ihned **`to_remote`** (Bob), pokud mám **`rev_key_B`** *nebo* pokud mám **`payment_preimage`** a podpis Boba → ihned **`to_remote`** (Bob) *nebo* pokud je po 17:00 → ihned zpět **`to_local`** (Alice)

Tento nový (žlutý) výstup se nazývá **Hash Time-Locked Contract**, zkráceně pouze HTLC, a je to speciální typ bitcoinového smart kontraktu, který nám umožňuje přesměrování plateb. Pojďme si ho nyní rozebrat více do detailu.

Jak je vidět v commitment transakci výše, Alice vzala 100k satoshi ze své strany kanálu a uzamkla je do HTLC, vytvořila nový výstup transakce. (*Reálně by tato částka byla ještě o 22 satoshi větší kvůli poplatkům, ale ty nyní pro přehlednost vynechme.*) V tuto dobu tedy disponuje pouze 700k satoshi, které může použít na další platby. Podobných HTLC může být v transakci několik, my ale budeme uvažovat pouze jedno. HTLC má 3 možnosti utracení:

- Prvním způsobem je, že tyto prostředky připadnou Bobovi, pokud bude mít revokační klíče. Jedná se o nám již známou ochranu Boba, pokud by Alice publikovala neaktuální transakci. V tomto případě by měl získat všechny prostředky, je proto nutné tuto možnost útraty přidat i do HTLC části transakce.

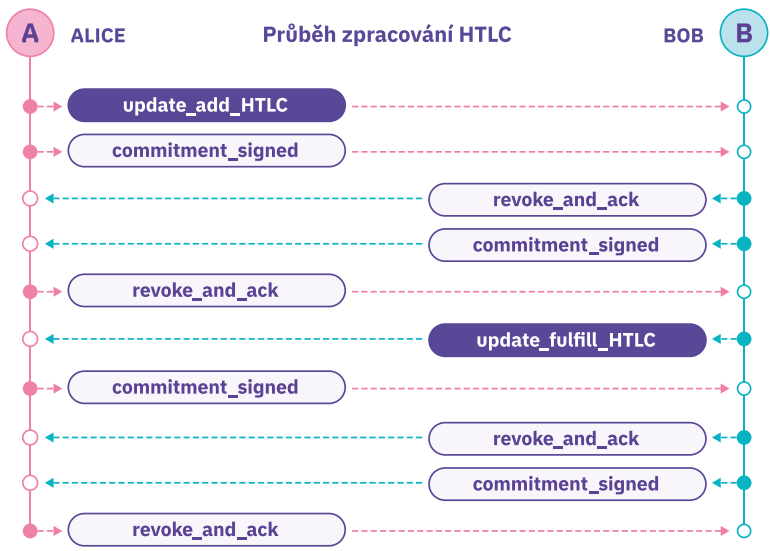
- Druhým způsobem je úspěšná platba. Pokud Bob dodá `payment_preimage` a zároveň svůj podpis do časového limitu 17:00, získá prostředky on.

- Poslední možností je vypršení absolutního časového zámku – přesně v 17:00 připadnou uzamčené bitcoiny zpět Alici.

Obdobné HTLC je i v commitment transakci u Boba – s tím rozdílem, že jsou prohozeny výstupy (při `payment_preimage` je výstupem `to_local`, u vypršení časového zámku naopak `to_remote`). Takovéto HTLC se zkonstruuje dále mezi Bobem a Cyrilem, v poslední řadě také mezi Cyrilem a Danielou. Všechny commitment transakce jsou strukturou stejné, jedinými rozdíly jsou dřívější limit na vypršení časového zámku a malinko nižší částka (ponížená o poplatek).

Průběh zpracování HTLC

Během úspěšného zpracování HTLC dojde mezi Alicí a Bobem k výměně 10 zpráv. Tato komunikace započne v době, kdy Daniela vygenerovala fakturu (obsahující `payment_hash`), předala ji Alici a ta si spočítala ideální trasu. Identická komunikace poté probíhá mezi Bobem a Cyrilem, následně mezi Cyrilem a Danielou.



V první zprávě `update_add_HTLC` odesílá Alice Bobovi informaci o tom, že si v rámci jejich kanálu chce vytvořit nové HTLC. Obsahem je ID kanálu (pokud by jich mezi sebou měli více), `payment_hash`, `cltv_expiry` (jedná se o hodnotu absolutního časového zámku – tedy limit, jaký má Bob na sehnání `payment_preimage` a vypořádání platby, detailně vysvětleno později) a informace, že dále má komunikovat s Cyrilem.

Bobovi toto stačí k tomu, aby si vytvořil novou commitment transakci, která vezme 100 022 satoshi z výstupu pro Alici a vloží je do nově zkonstruovaného HTLC.

Commitment transakce #653 (držená Bobem)			
Vstupy		Výstupy	
2M	Multisig adresa 2 ze 2 z funding transakce	0,7M	po 1 dnu → **`to_local`** (Bob) *nebo* ihned **`to_remote`** (Alice), pokud mám **`rev_key_A`**
		1,2M	**`to_remote`** (Alice)
		0,1M	ihned **`to_remote`** (Alice), pokud mám **`rev_key_A`** *nebo* pokud mám **`payment_preimage`** a podpis Boba → po 1 dnu **`to_local`** (Bob) *nebo* pokud je po 17:00 → ihned zpět **`to_remote`** (Alice)

Počet nezpracovaných HTLC

V každé commitment transakci může být v jednu chvíli více zatím nezpracovaných HTLC. Maximální limit je 483 – jedná se o limit pro velikost transakce v bitcoinové síti, aby byla stále platná. Právě kvůli tomu existují útoky na uzly, které tohoto využívají, a konkrétnímu uzlu vyčerpají všechny volné „sloty“ pro HTLC. V tomto případě je do vypršení časových zámků takovýto uzel nepoužitelný. Více o tomto útoku a možnostech obrany si povíme v pozdějších kapitolách.

Další 4 zprávy už nám jsou známé z kapitoly o úpravě stavu kanálů. Ve zprávě `commitment_signed` odesílá Alice Bobovi podpis k jím právě vytvořené commitment transakci, aby byla platná a mohl ji případně použít. Bob na to odpovídá `revoke_and_ack`, kdy Alici odesílá informace pro vytvoření **revokačních klíčů** k předešlé commitment transakci. Následně si Alice vytvoří svoji commitment transakci zahrnující HTLC výstup a Bob jí ve zprávě `commitment_signed` odesílá svůj podpis. Posledním bodem této části je zpráva `revoke_and_ack`, kdy Alice odešle svoji část **revokačních klíčů** k minulé transakci.

Nyní se přesuňme na konec našeho řetězu mezi Cyrila a Danielu. Konstrukce jejich vzájemného HTLC byla stejná jako v případě výše mezi Alicí a Bobem – pouze se dvěma rozdíly. Tím prvním je, že hodnota HTLC již není 100 022 satoshi, ale pouze 100 000 satoshi (poplatky si přivlastnili Bob s Cyrilem po cestě). Druhým rozdílem je, že Daniela `payment_preimage` k zaslanému `payment_hash` zná a nemusí se obracet na nikoho jiného. V jejím ekonomickém zájmu je toto tajné číslo Cyrilovi prozradit, protože díky tomu ihned získá 100k satoshi. Pojďme se ale nejprve podívat na to, co vše by se mohlo stát, kdyby se Daniela, případně kdokoliv jiný, rozhodli podvádět a případně přestali reagovat.

0,1M	ihned **`to_remote`** (Cyril), pokud mám **`rev_key_C`** *nebo* pokud mám **`payment_preimage`** a podpis Daniely → po 1 dnu **`to_local`** (Daniela) *nebo* pokud je po 16:40 → ihned zpět **`to_remote`** (Cyril)

Pokud se podíváme na HTLC výstup výše, který má Daniela ve své commitment transakci, jediným možným způsobem, jak může získat 100k satoshi je, pokud zveřejní `payment_preimage` a transakci podepíše svým podpisem. Pokud by nechtěla hrát fér, mohla by tuto transakci publikovat do bitcoinové sítě a vzít si svoje prostředky tvorbou transakce nové, která bude mít jako vstup prostřední cestu výše. Zpočátku by to mohlo vypadat, že na tom bude tratit Cyril, jelikož Daniela svoje prostředky získá a `payment_preimage` mu neprozradí. Opak je ale pravdou – unlocking script bitcoinové transakce musí obsahovat `payment_preimage`, Bob by se ho tedy stejně dozvěděl, jen by si ho musel přečíst z blockchainu. Zároveň by tímto došlo k uzavření kanálu mezi Cyrilem a Danielou. Časový zámek `CSV_delay` (zde 1 den) je zde ze stejného důvodu jako u všech `to_local` výstupů – ochrana pro protistranu, pokud by došlo k publikaci neaktuálního stavu.

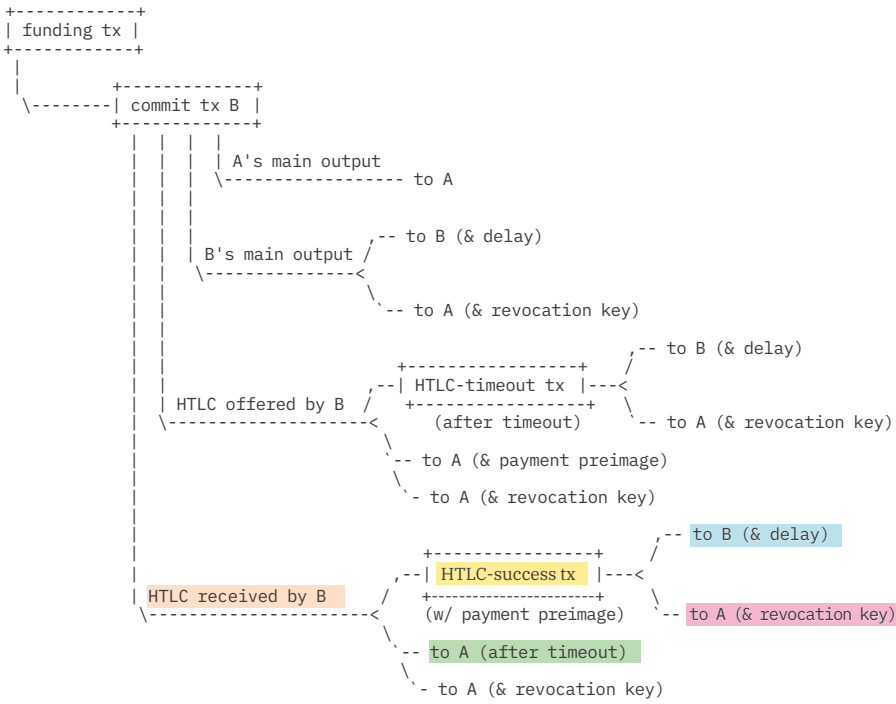
Dále se může stát, že kdokoliv po cestě přestane komunikovat v průběhu toho, kdy budou mezi uzly vytvořené aktivní HTLC. K tomu slouží absolutní časový zámek – bitcoiny se poté navrátí odesílateli. Ani zde není potřeba všechny commitment transakce na cestě odesílat do bitcoinové sítě – vzít si své prostředky zpět takto je až poslední možností. Mnohem efektivnější je se společně s ostatními (kteří jsou online) domluvit, že například Daniela přestala reagovat, a všichni si teď smažeme dané HTLC z commitment transakcí. Využívá se k tomu zpráva `update_fail_htlc`, která obsahuje mimo jiné i důvod selhání. Reálně dojde tedy k nucenému uzavření (publikováním commitment transakce) pouze toho kanálu, ve kterém je zapojena nereagující protistrana.

Jak je vidět na příkladech výše – odeslání commitment transakce do bitcoinové sítě je až tou poslední možností, kterou uzly využijí. Vždy při tom totiž dojde k nucenému uzavření kanálu. Veškeré tyto výstupy v HTLC tedy slouží spíše jako poslední možnost, pokud by někdo přestal reagovat. Mnohem efektivnější je se s uzly domluvit a HTLC buď smazat, nebo úspěšně vypořádat. Vraťme se ale k našemu příkladu. Daniela se rozhodne sdělit `payment_preimage` Cyrilovi. Udělá to tím způsobem, že mu odešle zprávu `update_fulfill_htlc`, která toto tajné číslo obsahuje spolu s dalšími detaily jako ID kanálu apod. Zde je nutné, aby si toto Cyril ověřil, a pokud hash odpovídá, můžeme přistoupit k již známému kolečku výměny 4 zpráv (`commitment_signed`, `revoke_and_ack`, `commitment_signed`, `revoke_and_ack`), kdy se vytvoří nové commitment transakce, tentokrát již bez jakéhokoliv HTLC výstupu. Hodnota připadající Daniele se navýší o počet bitcoinů, které byly uzamčeny v HTLC. Tímto stejným způsobem dojde k vypořádání mezi Cyrilem a Bobem, poté konečně mezi Bobem a Alicí. Alice získá důkaz o platbě – `payment_preimage`, je o 100k satoshi chudší a platbu lze považovat za úspěšnou.

<table>
<tr><th colspan="4">Commitment transakce #654 (držená Alicí)</th></tr>
<tr><td colspan="2">Vstupy</td><td colspan="2">Výstupy</td></tr>
<tr><td rowspan="2">2M</td><td rowspan="2">Multisig adresa 2 ze 2 z funding transakce</td><td>0,7M</td><td>po 1 dnu → to_local (Alice)
nebo
ihned to_remote (Bob), pokud mám rev_key_B</td></tr>
<tr><td>1,3M</td><td>to_remote (Bob)</td></tr>
</table>

Druhá úroveň transakcí

Informace, které jsme si řekli v předchozí kapitole, si musíme ještě rozšířit, protože reálně jsou HTLC o něco komplikovanější. Existuje tam totiž tzv. „druhá úroveň“ v rámci commitment transakcí. Na diagramu níže jsou v obdélníkových rámečcích znázorněny transakce, bez rámečku se jedná o nám již známé klasické výstupy. Kromě funding a commitment transakce zde existují ještě další transakce, na druhé úrovni, které se vážou na HTLC.



Zdroj: https://platbybudoucnosti.cz/schema-transakci

Představme si schéma výše jako Bobovu (B) stranu kanálu s Alicí (A). Vidíme zde výstup pro Alici, který není zatížen žádným časovým zámkem a zároveň výstup pro Boba, na který si musí počkat, než uplyne timelock `CSV_delay` (šance pro Alici zareagovat při podvodu). Případně tento výstup vezme Alice, pokud Bob podváděl a ona má revokační klíče. Zároveň zde máme dvě HTLC – jedno (HTLC offered by B), kdy je odesílatelem Bob, a druhé (HTLC received by B), kdy je Bob příjemcem. Pro náš příklad se budeme zabývat pouze tím HTLC, které nám zaslala Alice. Pro přehlednost je označeno oranžovou barvou.

Představme si, že my jsme Bob a Alice přes nás routuje platbu. Přidali jsme si tedy do commitment transakce HTLC a po chvíli sehnali od příjemce `payment_preimage`. Nyní se s Alicí chceme spojit a vyměnit tajné číslo za bitcoiny uzamčené v HTLC, které by nám měly náležet. V tuto chvíli ale Alice **přestane komunikovat**. Bob musí nyní odeslat commitment transakci do bitcoinové sítě (tím si uzavře s Alicí kanál) a vzít si prostředky, které mu náleží on-chain. Pokud by tak neudělal a došlo by k vypršení absolutního časového zámku, Alice by se shodou náhod mohla připojit zpět online a prostředky mu vzít.

Bob se po odeslání commitment transakce do sítě pokusí jít modrou cestou. Jelikož se ale jedná o výstup typu `to_local`, musí být opatřen relativním časovým zámkem, aby měla Alice šanci zareagovat, pokud by Bob odeslal neaktuální transakci, a mohla si tak vzít prostředky růžovou cestou s využitím revokačních klíčů. Zde obecně chceme, aby tento timelock byl **dlouhý klidně v řádu dnů**, pokud se jedná o kanál, ve kterém je uzavřeno velké množství bitcoinů. Nechcete totiž při havárii mít u takto velkého obnosu pouze málo jednotek hodin na znovuzprovoznění uzlu, než o vše můžete teoreticky přijít.

Problém ale je, že chceme, aby absolutní časový zámek připisující prostředky zpět Alici (zelená cesta) **byl co nejkratší**. Tyto bitcoiny jsou totiž po dobu uzamčení nepoužitelné pro jakoukoliv jinou platbu, jejich uzamčení na několik dní při běžném routování platby je tak nežádoucí.

Pokud by vše bylo uloženo v rámci jedné commitment transakce (jak jsme si zjednodušeně představovali v předchozích kapitolách) a modrá cesta (Bobovi, protože sehnal `payment_preimage`) by měla **delší časový zámek** než zelená cesta, kterou využije Alice, pokud Bob tajné číslo nesežene, mohlo by dojít k problému. Bob by neustále čekal, než si své prostředky bude moct vzít modrou cestou, ale Alice by se zničehonic připojila zpět a předběhla ho zelenou cestou. Pomocí tohoto způsobu by se tedy vyplatilo předstírat offline stav a okrádat protistranu.

Řešením jsou právě druhé úrovně transakcí, které umožňují rozdílné časové zámky pro výstupy. Tím se zaručí, že Bob projde modrou cestou dříve než Alice zelenou. Ihned po odeslání takovéto transakce do bitcoinové sítě se totiž vezmou prostředky z HTLC a uloží do nové transakce `HTLC-success tx`. Bob zde sice musí čekat, než si je bude moci přivlastnit, ale Alice ho již nemůže neférově předběhnout zelenou cestou. Jediná poslední cesta, která Alici zbývá, pokud by Bob podváděl s neaktuální transakcí a ona vlastnila revokační klíče, je vydat se růžovým výstupem.

Druhá úroveň transakcí může být na první pochopení trochu složitější, pokud si to ale člověk v klidu rozkreslí, začne mu to dávat smysl.

Doplnění – lokální platba kanálem

Nyní, když již známe princip fungování HTLC, tak si můžeme říct, že naprosto stejným způsobem probíhá platba, pokud odesílám prostředky někomu, s kým mám napřímo otevřený kanál.

Celý postup je tedy následující:

- Nejprve si u kamaráda v e-shopu něco objednám.
- Zobrazí se mi faktura obsahující `payment_hash`.
- Ve faktuře vidím, že příjemcem je někdo, s kým mám napřímo otevřený kanál, a tak mu nabídnu HTLC o hodnotě kupovaného produktu.
- Vyměníme si spolu podpisy a revokační klíče, abychom HTLC do commitment transakce zanesli.
- Kamarádův uzel mi odešle `payment_preimage` a díky tomu si opět vyměníme podpisy a revokační klíče pro novou commitment transakci. Nyní má ale moje strana kanálu o hodnotu produktu menší zůstatek.

Tento proces má dvě výhody. Tou první je, že **stejný kód** se využívá pro lokální platby i pro přeposílané platby. Druhou výhodou je soukromí – tím, že i lokální platba probíhá pomocí HTLC, kamarádův e-shop netuší, zda tato platba přichází přímo ode mne, anebo ji někdo úplně cizí přes můj uzel pouze přeposlal. (*Samozřejmě, pokud se budu na e-shopu muset registrovat a vyplnit tam své údaje, tak to kamarádovi bude asi jasné – tato informace ale není nijak sdílena po lightningové síti. V případě anonymní služby by to tak platilo.*)

Komunikace v síti

Onion routing

V rámci komunikace mezi uzly se používá onion routing, které je strukturou velmi podobné tomu, co se používá v anonymizační síti Tor. Cílem je vytvořit zprávu (v našem případě informace k přesměrování platby), která se odešle přes několik prostředníků, ale každý z nich si bude moci přečíst pouze tu část, jež je určena pro něj. Jinými slovy, jakýkoliv uzel po cestě ví pouze, od koho tuto zprávu dostal, komu ji má přeposlat a o jakou částku se jedná. Zbytek zprávy, který je určen pro následující uzel, si přečíst nemůže.

Využívá se k tomu tzv. cibulové šifrování ve vrstvách. Uvažujme o naší známé čtveřici sestávající z Alice, Boba, Cyrila a Daniely. Postupuje se odzadu – nejprve se vytvoří zpráva pro cílového příjemce, Danielu, která se zašifruje jejím veřejným klíčem (takže pouze Daniela si ji může přečíst). K této již zašifrované zprávě se přidá informace pro Cyrila a opět se to celé zašifruje, tentokrát ale s využitím veřejného klíče Cyrila. K tomuto již zčásti dvakrát zašifrovanému balíku se přidá zpráva pro Boba, a to vše se zašifruje veřejným klíčem Boba. Takovouto zprávu poté Alice odešle Bobovi.

Postup dešifrování je přesně opačný. Bob obdrží zprávu od Alice, pomocí svého soukromého klíče si ji dešifruje a zjistí její obsah, který je mu určen. Dále tam vidí i zašifrovaná data určené pro Cyrila. Ta si ale přečíst nemůže, protože nemá jeho privátní klíč. Odešle tedy tuto zprávu Cyrilovi, ten si ji opět dešifruje a zbytek odešle Daniele. Ta je již v tomto řetězci poslední, takže po dešifrování vidí cílovou zprávu, jež pro ni byla vytvořena Alicí. Nikdo po cestě se tedy nemohl dozvědět žádné informace, které mu nepřísluší.

A přesně proto se tomu říká někdy také cibulové šifrování, protože každý uzel po cestě si „jakoby sloupne" jednu vrstvu cibule a zbytek odešle dále. Alternativně by se to dalo připodobnit ke truhlám, kdy vezmu poklad pro Danielu a uložím ho do truhly. Na ni dám zámek, ke kterému má klíč pouze Daniela, a přilepím na ni cedulku, že je určena pro Danielu. Následně vezmu tuto truhlu a celou ji vložím do větší truhly, ke které má klíč pouze Cyril. Mohu mu tam přihodit nějaké drobné jako motivaci pro odeslání Daniele. Přiložím k ní papír, že je určena Cyrilovi, a celé to vložím do jedné velké truhly určené pro Boba, ke které má klíč pouze on.

V Lightningu se takto předávají doplňující informace k HTLC při přesměrování plateb. Představme si příklad, kdy Alice bude chtít odeslat 100 000 satoshi Daniele. Po přijetí faktury tedy Alice vytvoří zprávu pro cílového příjemce, Danielu. Ta obsahuje primárně **odesílanou částku** a dále hodnotu `min_final_cltv_expiry` – ta vyjadřuje počet bloků, které si příjemce vyhrazuje pro zpracování platby. Jinými slovy se jedná o určení absolutního časového zámku v HTLC mezi Cyrilem a Danielou. Pokud by aktuální blok měl pořadové číslo 720 000 a na faktuře bylo uvedeno jako `min_final_cltv_expiry` 9, tak by se Daniele odeslala tato zpráva:

Částka v satoshi	cltv_expiry	ID odchozího kanálu
100 000	720 009	(nevyplněno, jedná se o poslední uzel)

Pro Cyrila bude zpráva podobná. S tím rozdílem, že částka bude o něco vyšší (kvůli poplatkům), hodnota `cltv_expiry` se zvýší v tomto případě o 20 bloků a ID odchozího kanálu bude ten s Danielou.

Částka v satoshi	cltv_expiry	ID odchozího kanálu
100 010	720 029	ID (Cyril → Daniela)

Podobným způsobem se vytvoří i finální zpráva pro Boba.

Částka v satoshi	cltv_expiry	ID odchozího kanálu
100 020	720 049	ID (Bob → Cyril)

Tento onion paket se posílá součástí zprávy `update_add_htlc`, kterou jsme si probrali v kapitole o průběhu zpracování HTLC. Díky ní se každý uzel po cestě dozví informace, komu má dále platbu přeposlat (jak se dostat k cíli), ale zároveň zásluhou cibulového šifrování nezná celou trasu.

CLTV

V příkladu výše jsme hodnotu `cltv_expiry` vždy zvyšovali o 20, bez uvedení důvodu. Toto číslo se nazývá `cltv_expiry_delta` a každý uzel si je může libovolně nastavit per kanál. Vyjadřuje počet bloků, o které se zvětší absolutní časový zámek pro jednotlivá HTLC po cestě. V našem případě by se mohlo stát, že by Cyril získal `payment_preimage` od Daniely, odeslal by jí prostředky uzamčené v HTLC a šel by za Bobem, aby mu jej na oplátku odeslal on. Bob by ale začal hrát „mrtvého brouka" a nereagoval. Cyril má nyní čas do vypršení `cltv_expiry` na to, aby publikoval commitment transakci a vzal si prostředky on-chain (a spolu s tím uzavřel kanál). Hodnota `cltv_expiry_delta` je tedy rozdíl mezi expirací příchozího a odchozího HTLC a chrání přeposílatele před případnými podvody a ztrátou prostředků. Pokud je nastavena příliš nízko, tak se může stát, že to prostřední uzel nestihne. Naopak příliš vysoká hodnota může odrazovat od vybrání tohoto uzlu jako prostředníka pro přeposlání platby, jelikož přeposílané prostředky budou uzamčeny nepřiměřeně dlouhou dobu. Shrnutí:

- `min_final_cltv_expiry` – počet bloků, které si vyhrazuje na zpracování HTLC příjemce
- `cltv_expiry_delta` – počet bloků, které si vyhrazuje na zpracování HTLC přeposílající uzel
- `cltv_expiry` – aktuální výška bloku + hodnota `min_final_cltv_expiry` z faktury + sečtené všechny `cltv_expiry_delta` po cestě od aktuálního uzlu k příjemci

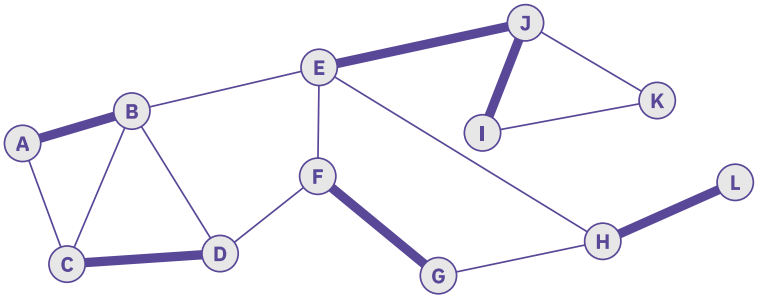
V Lightningu se klade velký důraz na soukromí a anonymitu. Když se vrátíme k příkladu s truhlami, tak vám zajisté mohlo dojít, že jakýkoliv uzel uprostřed by díky velikosti dané truhly mohl uhodnout, jestli je spíše na začátku, nebo na konci cesty. Jinak řečeno – pokud bude po dešifrování onion paketu ta část, kterou odesílám, dále velmi velká, tak se dá předpokládat, že po cestě bude ještě několik uzlů. Naopak pokud by byla velmi malá, můj přímý následník je pravděpodobně příjemcem.

Aby se takovémuto odkrývání informací zamezilo, tak se v Lightningu používá chytrý konstrukt, který zajistí, že **všechny odesílané onion pakety jsou stejně velké**. Technicky se to řeší tak, že každá zpráva vypadá, jako by byla odeslána dalším 19 uzlům (z toho plyne, že maximální počet uzlů po cestě je 20). Jinak řečeno, každý onion paket má velikost 1 300 bajtů (bez hlavičky).

Reálně konstrukce probíhá tak, že si vytvořím 1 300 bajtů náhodných dat, k nim zleva přidám zprávu pro Danielu, čímž velikost paketu zvýším, ale následně jej ihned zprava oříznu zpět na 1 300 bajtů. Tento proces opakuji pro každého příjemce. Na konci mi tedy vznikne zpráva, která například v první pětině obsahuje užitečná data, zbytek jsou náhodné bajty – nikdo po cestě ale neví, jestli se jedná pouze o „výplň“, nebo zda tam jsou zašifrované informace pro další uzly. Obdobně se zpráva doplňuje na výsledných 1 300 bajtů i při dešifrování. Tím, že si například Bob „odloupne“ svoji část, by totiž efektivně zprávu zkrátil. Před odesláním ji tedy doplní zpět na cílových 1 300 bajtů a odešle. Díky tomu žádný uzel po cestě nemůže odhadnout, kolikátý je v pořadí, ani kdo je odesílatel a kdo příjemce.

Gossip protokol

Jak jsme si již několikrát řekli, zdrojový uzel sestavuje cestu k uzlu příjemce. Aby toho mohl dosáhnout, musí mít informace o všech ostatních uzlech a jejich vzájemných kanálech. Dále potřebuje i podrobnější informace k jednotlivým kanálům jako je jejich kapacita a účtované poplatky, aby mohl vybrat ideální trasu. Jinak řečeno, potřebuje znát graf lightningové sítě.



Tato problematika se řeší tzv. gossip protokolem (do češtiny by se to dalo přeložit jako „šíření drbů“). Jedná se o šifrovanou P2P komunikaci mezi uzly, kde si navzájem sdílí informace, které se k nim dostaly. Pokud nějaký uzel například naváže nový kanál, předá tuto informaci sousedům, ti ji zvalidují a poté odešlou dále do sítě

k ostatním. Tímto způsobem se postupně šíří informace celou sítí a každý uzel si sestavuje svůj graf. Aby se zabránilo zbytečnému přetížení, tak se tyto informace předávají jednou za čas v balících obsahujících více zpráv najednou – než se tedy dostane informace o nově vzniklém kanálu ke všem uzlům, může to klidně trvat až jednotky hodin. To ale není nikterak na škodu, v Lightningu se s tím počítá.

> **Uzel bez kanálů**
>
> Velmi často se stává, že si člověk založí nový lightningový uzel a ihned po jeho startu se jde podívat na některý z lightningových vyhledávačů (probereme později v praktické části knihy), zda se tam již jeho uzel objevil. Realita je však taková, že tam tento uzel nenajde. Jak jsem psal výše, může trvat i hodiny, než by tato informace dotekla až k uzlu, z kterého přijímá data webový vyhledávač. V tomto případě by ale jakkoliv dlouhé čekání nevedlo k úspěchu. Uzel, který zatím nemá žádné kanály, se nešíří gossip protokolem. Důvod je jednoduchý – takovýto uzel nelze využít na příjem, odeslání ani routování plateb, tudíž by jen zbytečně zabíral ostatním uzlům místo v databázi. Jakmile navážete svůj první kanál, po určité době zde svůj uzel již najdete.

Gossip protokol se skládá ze tří druhů zpráv:

- **node_announcement**
- **channel_announcement**
- **channel_update**

Ve zprávě `node_announcement` se oznamuje vznik nového uzlu. Jejím obsahem je jeho ID (tedy veřejný klíč), síťová adresa (IPv4, IPv6 nebo Tor) pro připojení, alias (libovolný řetězec pro lepší identifikaci uzlu), podporované funkce (vývojáři takto mohou přidávat nové funkce za běhu), aktuální časová známka, digitální podpis a další.

Zpráva `channel_announcement`, jak již název napovídá, oznamuje vznik nového kanálu. Obsahuje jeho celkovou kapacitu, zkrácené ID, důkaz o vytvoření, podporované funkce, ID obou uzlů, mezi kterými byl vytvořen, jejich podpisy a další. Každý uzel si poté vytvoření tohoto kanálu ověřuje díky uvedeným podpisům a funding transakci v blockchainu, aby nedocházelo k zaspamování sítě vymyšlenými kanály.

ID kanálů
Jak jsme si již řekli v kapitole o konstrukci kanálu, jeho ID se získá takto:
ID_kanálu = TXID_funding_transakce `XOR` index_výstupu

Existuje ale i zkrácené ID (`short_channel_id`), na které můžete občas narazit, a využívá se například ve zprávách `channel_announcement`. Zkrácené ID je vlastně odkaz do blockchainu na funding transakci a její výstup. Jedná se o 8 bajtové číslo, kdy první 3 bajty jsou výška bloku, ve kterém se funding transakce nachází. Další 3 bajty jsou pořadí transakce v bloku a poslední 2 bajty jsou výstupní index dané transakce. Výstup se zapisuje buď jako celé číslo o velikosti 8 bajtů, například `774909407114231809`, případně pro jednodušší čtení takto – `704776x2087x1`.

Posledním typem zprávy je `channel_update`. Primárním důvodem existence této zprávy je oznamování poplatků (`base_fee` i `fee_rate`) a parametru `cltv_expiry_delta`. Dále tato zpráva obsahuje identifikaci kanálu (zkrácené ID), minimální/maximální velikost HTLC, časovou známku a podpis. Tato zpráva je odesílána spolu s `channel_announcement` oběma stranami, jelikož každý uzel může mít nastavené odlišné parametry, např. poplatky. Pokud kdykoliv během provozování uzlu jakýkoliv parametr změním, opět se odešle zpráva `channel_update` a ostatní uzly si díky tomu upraví svůj graf.

Ve všech těchto zprávách se využívá autentizace, aby si je nemohl kdokoliv jen tak vymýšlet. Pokud o nějakém kanálu nepřišla dlouhou dobu žádná informace, bude časem vymazán. To samé platí pro uzly – jakmile nebude existovat žádný kanál s tímto uzlem, ostatní si ho ze své databáze smažou. Některé implementace si do svého grafu přidávají ještě další informace jako vlastní skóre apod.

DNS Bootstrapping

Pro odeslání platby si potřebuji sestavit cestu ze svého uzlu k příjemci. Abych to mohl provést, musím mít graf lightningové sítě, který se mi neustále upravuje díky informacím od ostatních uzlů v rámci gossip protokolu. Pokud si vytvoříte úplně nový uzel, tak se musíte dozvědět alespoň o jednom dalším uzlu, abyste se k němu připojili a mohli gossip protokolem získávat informace o lightningové síti – jak se to technicky provede?

Stejný problém se řeší i na Bitcoinu a jedno z nejjednodušších řešení by bylo do kódu implementace natvrdo napsat seznam uzlů, které „vždy budou fungovat“ a budou sloužit jako rozcestník. Právě ta část věty v uvozovkách je velmi problematická, a pokud by se někomu podařilo vyřadit všechny uzly v tomto seznamu z provozu, nikdo nový by se do sítě nemohl připojit (pokud by si někde nesehnal alternativní

seznam). Existuje mnoho lepších řešení tohoto problému, se kterými se můžete setkat pod pojmem bootstrapping. My se podíváme na jeden, který se v Lightningu primárně využívá, a tím je DNS bootstrapping.

Na rozdíl od Bitcoinu potřebuji v Lightningu kromě adresy a síťového portu také veřejný klíč, jelikož z něj se odvozují další šifrovací klíče, které jsou použity pro veškerou komunikaci. Každá implementace má v kódu několik DNS serverů, které se pro redundanci využívají navzájem. Po čisté instalaci tedy provede váš uzel DNS dotaz na SRV záznam a obdrží veřejný klíč a číslo portu náhodného uzlu. S pomocí této informace provede druhý dotaz, tentokrát na záznam typu A, čímž získá IPv4 adresu (případně AAA pro IPv6). Těmito DNS dotazy tedy získáte informace o několika prvních uzlech, ke kterým se můžete připojit, a začít od nich postupně získávat informace o celé lightningové síti.

> **Spojení vs. kanál**
>
> Váš uzel může být síťově spojený s několika dalšími, není ale nutnou podmínkou mít s těmito uzly otevřený platební kanál. Typicky po instalaci čistého uzlu zjistíte, že jste spojeni s dalšími uzly, aniž byste zatím s kýmkoliv otevřeli kanál. Informace o těchto uzlech máte díky DNS boot-strappingu a využíváte je k získávání aktuálních informací o síti. Reálně tedy budete spojeni se všemi uzly, se kterými máte otevřený platební kanál (pro zpracování plateb i gossip zároveň), a navíc s několika dalšími náhodnými uzly, které slouží pouze pro výměnu zpráv v rámci gossip protokolu.

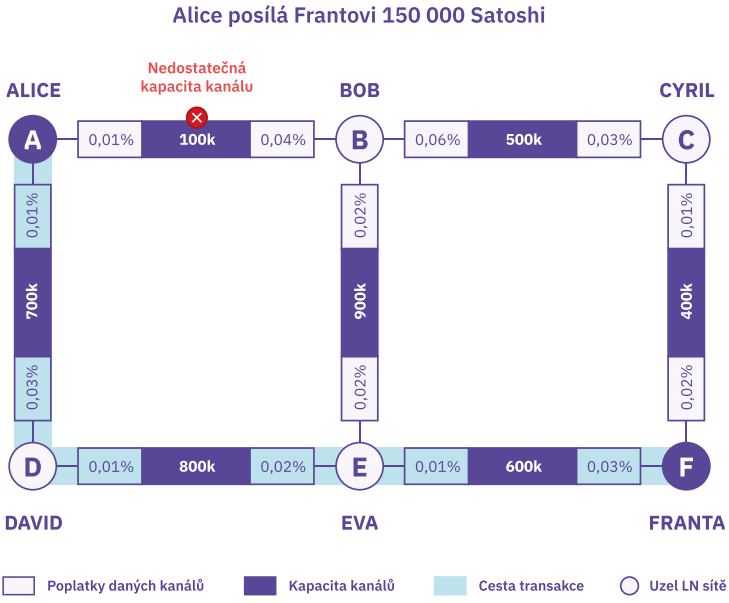
Pathfinding

Podívejme se nyní na pathfinding neboli **hledání ideální cesty** od odesílatele k příjemci. Tento výpočet probíhá na straně odesílatele a výsledná cesta je uložena do onion paketu (aby každý uzel věděl, komu má zprávu poslat dále). Pro tuto problematiku neexistuje žádná specifikace – naopak každá implementace Lightningu si to řeší po svém a je to do určité míry i konkurenční boj o to, komu se podaří vyvinout nejlepší algoritmus.

V dnešní době už většinou nebývá problém najít cestu z uzlu A do B. Síť je již dostatečně velká a dobře propojená. Mnohem těžší je ale najít ideální trasu. To znamená, aby byla co nejkratší, jednotlivé uzly si účtovaly co možná nejmenší poplatky a měly nastaveny krátké časové zámky (`cltv_expiry_delta`). Zároveň ale nalezená cesta musí mít dostatečnou likviditu v požadovaném směru, aby platba prošla. A to je ten největší problém – jak jsme si již vysvětlili dříve, v rámci gossip protokolu se nešíří z mnoha důvodů informace o aktuálním stavu kanálů. Víme

tedy, že kanál mezi Bobem a Cyrilem má kapacitu například 5 milionů satoshi, nevíme však, jakou část vlastní Bob a jakou Cyril. Pokud by veškeré prostředky byly na straně Cyrila, Bob by tímto kanálem nemohl již nic odeslat. Tuto informaci ale Alice jako odesílatel netuší, a tak s ní bohužel nemůže ve svém výpočtu kalkulovat.

Představme si zjednodušený graf lightningové sítě, který se bude sestávat pouze z 6 uzlů. Vidíme zde celkovou kapacitu jednotlivých kanálů (vyznačena v satoshi bílým písmem uprostřed) a zároveň poplatky, které si každý uzel účtuje za přeposlání platby v procentech. Tyto poplatky mohou být pro každý směr odlišné, protože si je určuje uzel pouze pro svoji odchozí likviditu. Reálně bychom v každém směru měli ještě navíc `cltv_expiry_delta`, minimální/maximální velikost HTLC (přeposílané částky) apod., ale pro přehlednost tam ponechme pouze poplatky.

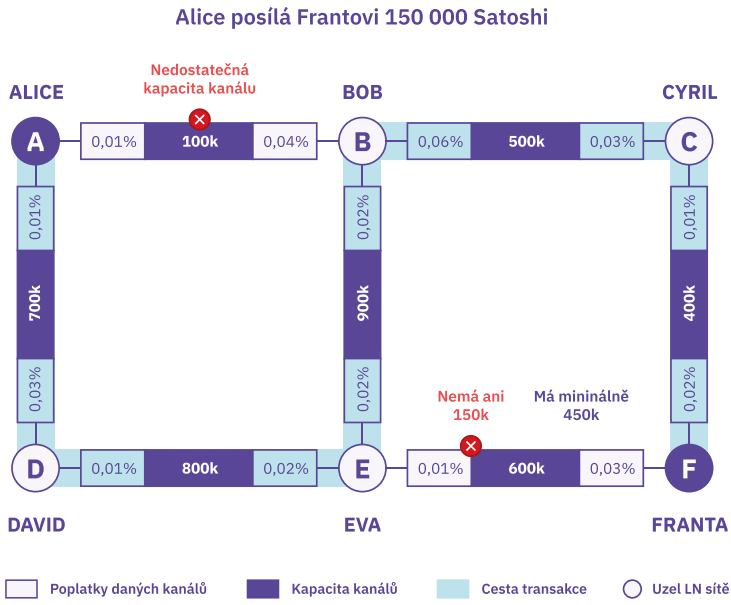


Vžijme se nyní do role uzlu A, který chce odeslat částku 150 000 satoshi uzlu F. Už nyní můžeme vyřadit kanál mezi uzly A a B, jelikož jeho celková kapacita je nižší než přeposílaná částka. Nás uzel si tedy spočítá 2 možné cesty:

- A – D – E – F
- A – D – E – B – C – F

Tyto cesty si následně seřadí podle určitých kritérií (poplatky, délka, časové zámky, interní skóre uzlů apod.) a bude zkoušet jednu po druhé od té nejlepší. V našem zjednodušeném případě budeme trasy hodnotit pouze podle poplatků.

Levnější trasou je ta první, kde zaplatíme celkem 0,01 % za přeposlání z uzlu D do E a pak tu samou částku z uzlu E do cílového F. (*Pokud odesíláme platbu, tak sami sobě poplatek neúčtujeme, proto se nás 0,01 % v kanálu mezi uzly A a D netýká.*) Zkusme tedy tuto trasu. Od uzlu E se nám ale vrátí zpráva o neúspěchu (`update_fail_htlc` typu `TEMPORARY_CHANNEL_FAILURE`), která nám říká, že aktuálně uzel E nemá dostatečnou likviditu pro přeposlání částky 150 000 satoshi uzlu F. Na jednu stranu je to špatná zpráva (platba neprošla), na druhou stranu si díky této informaci můžeme upravit náš graf. Jednak víme, že kanál mezi E a F nebudeme nadále využívat, a zároveň víme, jaký je přibližný stav tohoto kanálu. Díky každému neúspěchu se tedy dozvíme důležité informace. Zkusme nyní dražší a delší trasu A, D, E, B, C, F.



Tato trasa již projde a úspěšně zaplatíme 150 000 satoshi uzlu F. Stejně jako v případě výše jsme se ale dozvěděli důležité informace – každý kanál má v námi odeslaném směru na vzdálenější straně likviditu alespoň 150 tisíc satoshi. Díky úspěšným i neúspěšným platbám si tedy **můžeme náš graf sítě „vylepšovat“** o informace, které se gossip protokolem neodesílají, a zpřesnit tak naše algoritmy vyhledávající cestu. Je ale nutné upozornit, že tyto informace jsou platné pouze dočasně. Například to, že uzel E neměl dostatek likvidity na odeslání platby uzlu F, platilo jen tehdy, když jsme platbu odesílali. Klidně minutu nato mohla být routována platba v opačném směru, která likviditu otočila. Při navrhování ideálních pathfinding algoritmů se s tímto počítá a tyto informace jsou uloženy pouze po určitou dobu – poté jejich platnost vyprší.

Dále je nutno upozornit, že neexistuje jeden oficiální graf lightningové sítě, na kterém by panovala shoda napříč všemi uzly (jako například v případě blockchainu). Naopak každý uzel může mít graf mírně odlišný, poněvadž záleží na tom, které všechny zprávy ohledně vzniku a zániku kanálů se k němu stačily gossip protokolem v době výpočtu trasy dostat.

Asi nejhorší variantou, která může nastat, je „zaseknutá platba". Konkrétně tím myslím stav, kdy jeden z uzlů po cestě z nějakého důvodu přestane reagovat, a vám se tak nevrátí ani informace o úspěchu ani o chybě. Jak jsme si již vysvětlili dříve, díky absolutním časovým zámkům se platba nezasekne navždy a dříve nebo později se vám prostředky vrátí (anebo se odešlou do cíle) – toto čekání ale může v extrémních případech trvat hodiny až dny, záleží totiž na hodnotě `cltv_expiry`. Ta, kdyby byla nastavena na velmi krátký interval, tak by uzly po cestě riskovaly, že nestihnou v případě potřeby do stanoveného limitu odeslat commitment transakci do blockchainu. I když se podobný problém nestává často a většinou se díky němu síť **zbaví problémového uzlu**, který nereagoval (pokud nebude reagovat do stanoveného limitu, bude s ním vynuceně uzavřen kanál), tak to zajisté není příjemný stav. Nemůžeme totiž tuto platbu nijak stornovat (je zanesena do vzájemných HTLC mezi uzly po cestě), a pokud bychom ji poslali znova, tak se může stát, že do cíle doputují obě. Aktuálně na toto téma probíhají diskuse a navrhované úpravy lightningového protokolu by zmíněný problém do budoucna mohly minimalizovat.

Typy kanálů

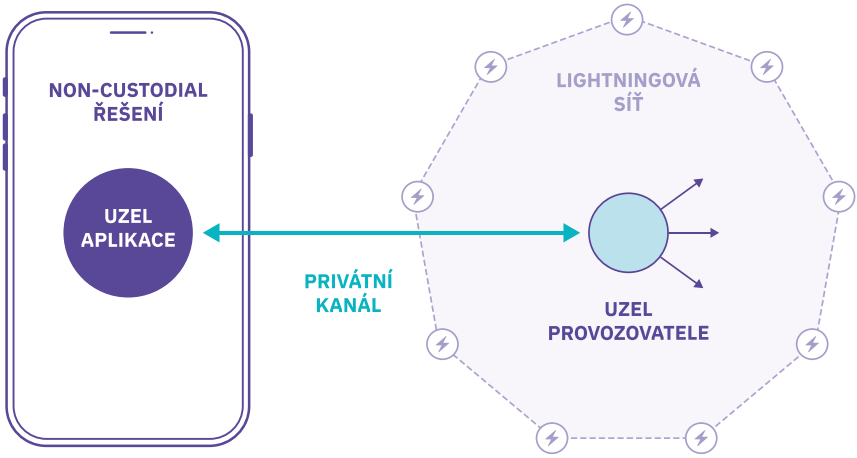
V této kapitole se podíváme na několik vlastností platebních kanálů, dle kterých je lze klasifikovat.

Privátní kanály

Pokaždé, když navazujete nový platební kanál s protistranou, máte na výběr, zda ho chcete vytvořit jako veřejný či privátní. Doposud jsme se v této knize bavili pouze o veřejných kanálech. Privátní kanál je kanál, jehož existence se **nešíří gossip protokolem**, a tudíž o něm ví pouze ty uzly, mezi kterými je navázán. Kvůli této vlastnosti například nelze přesně říci, kolik existuje kanálů v lightningové síti, ani jaká je jejich celková kapacita. Tuto informaci máme pouze o veřejných kanálech. Zároveň pokud uzel nemá navázán žádný veřejný kanál, ale například pouze jeden privátní, tak se informace o tomto uzlu také nešíří gossip protokolem, jelikož pro zbytek sítě se tváří jako uzel bez jakéhokoliv kanálu, tudíž jako nepoužitelný pro routování plateb. Z tohoto důvodu neznáme ani celkový počet uzlů v lightningové síti. Nejčastějším důvodem pro vytváření privátních platebních kanálů je fakt, že nechceme naši likviditu v něm uzamčenou nabízet ostatním pro routování

plateb. Protože o tomto kanálu nikdo jiný neví, lze ho používat pouze pro odesílání a přijímání plateb námi. Dává tedy smysl takovýto uzel navázat s kamarádem, případně například s oblíbeným obchodem. Kromě vyšší anonymity nám to zaručí, že se nám nepřesune veškerá likvidita na druhou stranu kvůli přesměrování velkých plateb, a tudíž se nestane tento kanál pro další platby nepoužitelným.

Dalším častým využitím privátních kanálů jsou non-custodial mobilní peněženky.



Běžnou praxí je, že provozovatel lightningové peněženky naváže privátní platební kanál ze svého uzlu (zde vyznačený zelenou barvou) na váš light-weight uzel v mobilní aplikaci (fialová barva). Veškeré platby poté putují do lightningové sítě skrze tohoto prostředníka, který musí zajistit dobré propojení na ostatní uzly. Není to však pravidlem, které by platilo ve 100 % případů – custodial řešení se chovají odlišně a některé peněženky vám umožňují navázat ze své aplikace kanálů více.

Routing hints

Když se podíváme na schéma výše, pokud budu chtít odeslat ze své mobilní peněženky platbu, nebude s tím žádný problém. Naleznu několik optimálních cest k příjemci a budu zkoušet jednu po druhé, kde všechny půjdou přes uzel provozovatele aplikace. Co když ale někdo bude chtít zaslat platbu mně? Jelikož můj jediný navázaný kanál je privátní, tak odesílatel netuší, jak vytvořit cestu k mému uzlu, protože o existenci tohoto kanálu nemá ponětí. Využívají se k tomu nápovědy zvané routing hints. Ty se nachází na faktuře, kterou vygeneruje moje mobilní peněženka, a zjednodušeně řečeno, říkají odesílateli, že mezi tyrkysovým uzlem a mnou existuje privátní kanál, a dále jeho parametry jako poplatky apod. Díky tomu může odesílatel vytvořit trasu až k mému uzlu. Drobnou nevýhodou je, že publikováním této faktury veřejně se všichni, kdo se k ní dostanou, dozví o existenci tohoto privátního kanálu.

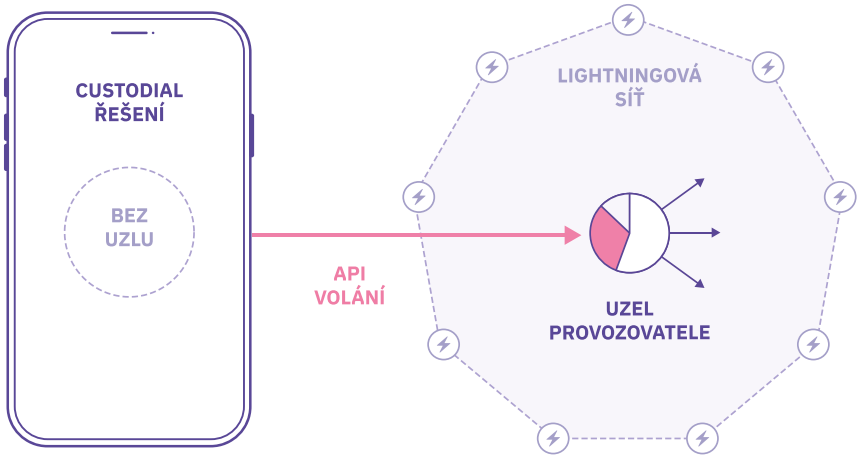
Turbo kanály

Tento typ kanálu nalezne primární využití také u mobilních peněženek. Otevření kanálu není nic jiného, než odeslání funding transakce do bitcoinové sítě a vyčkávání na vytěžení do bloku. To znamená, že každé otevření kanálu trvá několik desítek minut v závislosti na aktuálním zaplnění mempoolu, nastavených on-chain poplatcích a požadovaném počtu zanoření. Uživatelská přívětivost Lightningu by v tomto případě pro naprosté začátečníky tedy nebyla nikterak oslňující, poněvadž by nemohli mobilní peněženku používat ihned po stažení a nabytí prostředků, ale museli by čekat na vytěžení funding transakce. Z tohoto důvodu byly vymyšleny turbo kanály. Zjednodušeně řečeno se jedná o platební kanál, který lze začít **používat ihned**, ještě před vytěžením transakce do bloku. Toto řešení zásadním způsobem zpříjemňuje používání Lightning Network. Je to stejné, jako když v restauraci začnou připravovat a donesou vám váš oblíbený steak ještě před zaplacením.

Jedinou nevýhodou je potenciální risk double-spendingu. Protože transakce není na počátku zapsaná do bloku, mohl by ten, kdo kanál otevíral, použít po určitou dobu identické UTXO k vytvoření odlišné transakce. Každá platforma si to řeší po svém, ať již tak, že kanál iniciuje provozovatel peněženky ze svého uzlu, tak například otevíráním platebního kanálu z multisig adresy 2 ze 2, kde jedny privátní klíče drží „pro všechny důvěryhodná osoba či třetí strana“, která double-spending neprovede.

Hostované kanály

Základní princip fungování majority lightningových non-custodial peněženek jsme si vysvětlili v kapitole o privátních kanálech. Než se podíváme na hostované kanály, vysvětlíme si, jak fungují custodial peněženky, kde privátní klíče fyzicky nevlastníte, tudíž v konečném důsledku ani bitcoiny.



Zde jsou ve většině případů veškeré prostředky všech uživatelů uloženy na uzlu provozovatele. Vy pouze ze své mobilní aplikace dáváte příkazy tomuto uzlu, kolik satoshi chcete odeslat. Tyto prostředky se poté odečtou z vašeho podílu. Pokud naopak přijímáte nějaké bitcoiny, doputují fyzicky na uzel provozovatele, kde se vám o tuto částku zvedne váš stávající zůstatek. I když je toto řešení nejméně bezpečné a anonymní, z hlediska uživatelské přívětivosti je na tom nejlépe, jelikož odpadají poplatky za vytváření jakéhokoliv kanálu a čekání na vytěžení transakce.

Hostované kanály jsou někde mezi těmito dvěma světy. Jedná se o platební kanál, který **není zabezpečen blockchainem**, jelikož funding transakce vůbec **nebyla odeslána do bitcoinové sítě** (ani nikdy nebude). Veškeré vaše prostředky jsou stále drženy provozovatelem uzlu, ale mezi vaší mobilní peněženkou a tímto uzlem je navázán hostovaný kanál, který se chová velmi podobně jako ten běžný.

Představte si, že přijímáte platbu. Pokud budete používat kompletně custodial peněženku, tak uzel provozovatele vytvoří fakturu a přijme za vás veškeré prostředky. Čistě teoreticky se může stát, že bude tvrdit, že žádné prostředky neobdržel. Vy to ale nemáte jak dokázat, protože `payment_preimage`, náhodné číslo, které si generuje příjemce a slouží jako důkaz o platbě, si vytvořil provozovatel, nikoliv vy. Není tedy ve vašich silách rozhodnout, zda lže odesílatel platby nebo provozovatel vaší aplikace.

Naopak u hostovaných kanálů je mezi vaším light-weight mobilním uzlem a uzlem provozovatele otevřen jakýsi pseudokanál. Nevyužívají se zde klasické commitment transakce (blockchain se nevyužívá vůbec), ale uzly si mění jakýsi aktuální stav, který je commitment transakcím velmi podobný. Rozdíl je v tom, že fakturu i `payment_preimage` si generujete vy. Prostředky tedy sice při příjmu nikdy nedoputují až k vám, ale musíte provozovateli uzlu dodat vámi vygenerované `payment_preimage`, aby si mohl odemknout HTLC s dalším uzlem po cestě. Jinými slovy, tak jako tak vás může provozovatel okrást, tentokrát o tom **alespoň budete mít důkaz**. Nebezpečí toho, že provozovatel se všemi prostředky uteče nebo je zabaví „vyšší moc“, zde ale bohužel pořád zůstává.

Značnou **výhodou je tady ale soukromí** – v případě hostovaných kanálů si celou trasu vytváříte vy, provozovatel peněženky poté figuruje pouze jako jakýkoliv jiný uzel po cestě, který má pouze informace o částce a následujícím uzlu. V případě plně custodial řešení ví provozovatel naprosto vše od celé trasy po cílového příjemce, jelikož tuto cestu sestavoval. Další výhodou je, že takovéto hostované kanály lze míchat i s klasickými. Pokud budu odesílat platbu přes 7 uzlů a jeden kanál po cestě bude hostovaný, může mi to být naprosto jedno. Nutná důvěra (a případné riziko ztráty) je to vždy pouze pro ty dvě strany, které spolu tento kanál mají otevřený.

I když nejsou hostované kanály ideální a doporučoval bych spíše ty klasické, připadá mi to jako lepší řešení pro onboarding naprostých začátečníků, nežli je kompletně custodial řešení. Odpadají poplatky za funding transakci, nemusí se čekat na vytěžení této transakce, ale zároveň je zde alespoň malé zlepšení z hlediska soukromí. Po určité době je možné takovéto uživatele přesunout na plnohodnotné non-custodial řešení.

Wumbo kanály

Zpočátku vývoje Lightning Network byla omezena maximální celková kapacita jednoho kanálu na 0,167 7 BTC. To bylo zavedeno z důvodu, aby uživatelé v případě nějakých problémů nepřišli o velkou částku, protože samotná síť i jednotlivé implementace byly ještě v experimentálním stádiu vývoje. S postupem času a získanou důvěrou ale téměř všechny hlavní implementace zavedly tzv. Wumbo kanály. S nimi **tento limit odpadá** a dnes lze nalézt kanály o kapacitě i několik bitcoinů. Wumbo kanály tedy naleznou využití primárně u velkých provozovatelů, jako jsou uzly peněženek, obchodů, směnáren, služeb apod.

Podpora Wumbo kanálů se šíří gossip protokolem ve zprávě `node_announcement`. Díky tomu každý uzel v síti ví, kdo tyto velké kanály podporuje a kdo nikoliv.

Dual-funded kanály

Pokud otevíráme platební kanál standardní cestou, jsou po jeho vytvoření veškeré prostředky na naší straně. To znemožňuje zpočátku tímto kanálem přijmout jakoukoliv platbu a je tedy nutné počkat, než se nějaké satoshi přesunou na druhou stranu, abychom získali příchozí likviditu.

S dual-funded kanály toto již neplatí. Díky nim je možné, aby **obě strany** odeslaly své prostředky v rámci funding transakce. Tím pádem má kanál ihned po vytvoření jak příchozí, tak odchozí likviditu. Pokud se tato funkcionalita do budoucna uchytí, mohlo by to zásadním způsobem přispět k jednodušší správě likvidity.

Multi-funded kanály

Jedná se o několik kanálů, které byly otevřeny **v rámci jedné funding transakce**. Technicky je to proveditelné díky tomu, že commitment transakce se odkazuje na konkrétní výstup v rámci funding transakce. Díky této funkcionalitě lze znatelně ušetřit na on-chain poplatcích, pokud otevíráme velké množství kanálů naráz.

Anchor kanály

V rámci úpravy stavu kanálu si jednotlivé uzly mezi sebou vyměňují nové commitment transakce a zneplatňují ty původní. Tyto dopředu podepsané transakce se při běžném provozu neodesílají do bitcoinové sítě, pouze si je jednotlivé strany drží u sebe. K jejich využití dojde v případě, že druhá strana přestane odpovídat, a já jsem tedy nucen tuto commitment transakci publikovat, abych si vzal zpět svoji část prostředků.

Problém ale nastává, že k podpisu nové commitment transakce dochází **v době úpravy stavu kanálu**. V tuto chvíli do této transakce musíme zanést i případné on-chain poplatky. Pokud se podíváme do mempoolu, tak bychom mohli odhadnout potřebné poplatky k aktuálnímu stavu – rozhodně ale nevíme, jak veliké poplatky budou potřeba v době případného odeslání commitment transakce. K tomu totiž může dojít klidně i o několik týdnů později, a vzhledem k tomu, že nejčastějším důvodem je nemožnost komunikace s protistranou, tak ani jednostranná úprava on-chain poplatků nepřipadá v úvahu (došlo by k změně transakce a neseděl by podpis).

Dříve se to řešilo tím způsobem, že se poplatky nastavovaly **velmi vysoké**, aby bylo v budoucnu zajištěno, že se takováto transakce vytěží včas. Často tedy tak docházelo k tomu, že byly commitment transakce zbytečně přeplacené a v ojedinělých případech mohlo dojít ke stavu, že ani takovéto vysoké poplatky nemusely stačit.

Řešením jsou právě tzv. **anchor výstupy**. Jedná se o funkcionalitu, která lehce upravuje formát commitment transakce (přidává nové výstupy), aby bylo možné využít techniky **CPFP** (Child Pays For Parent). Tato technika umožňuje urychlit zapsání již odeslané transakce do bloku tím způsobem, že se do mempoolu odešle **další transakce**, která utrácí výstup té původní a má vysoké poplatky. Těžařům se poté vyplatí zahrnout do nejbližšího bloku obě, jelikož díky té druhé obdrží štědrou odměnu, která jim vykompenzuje zařazení té první levné commitment transakce. Tímto způsobem lze tedy efektivně zajistit, že moje commitment transakce bude zapsána do bloku včas, a odpadá mi nutnost odhadovat, jaké v budoucnu budou potřeba on-chain poplatky.

Aby toto fungovalo, je nutné mít připravené v rezervě (nejčastěji v on-chain peněžence na samotném uzlu) nějaké satoshi navíc, které se právě využijí pro toto „protlačení“ commitment transakce.

Zombie kanály

Jedná se o kanál, jehož protistrana dlouhodobě nereaguje a nelze ji jakkoliv kontaktovat. V tomto případě jsou bitcoiny zde uzamčené **nevyužitelné** a dává smysl takovýto kanál nuceně uzavřít a prostředky použít jinde.

Možnosti plateb

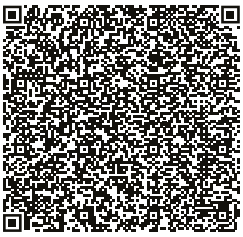
Faktury

Nejčastějším způsobem příjmu plateb v rámci Lightning Network jsou aktuálně faktury. Generuje je příjemce a primárně pomocí nich sděluje odesílateli `payment_hash` a ID cílového uzlu (veřejný klíč). Faktura jako taková není nic jiného než textový řetězec, který není nikterak šifrovaný (jeho obsah si může přečíst kdokoliv), pouze je zakódovaný do specifického formátu. Pro zvýšení uživatelské přívětivosti se nejčastěji můžete setkat s reprezentací faktury v podobě QR kódu.

> **Připomenutí**
>
> `payment_hash` = SHA256 (`payment_preimage`)

Oproti bitcoinovým adresám nejsou faktury statické – mají určitou časovou platnost a nelze je zaplatit vícekrát. Důvodem je, že k jednomu `payment_hash` uvedenému na faktuře odpovídá jedno `payment_preimage` (náhodné číslo, které si vygeneruje příjemce a díky němuž si jednotliví prostředníci připisují prostředky ze vzájemných HTLC). Pokud by někdo použil jednu a tu stejnou fakturu vícekrát a platba by směřovala přes uzel, který ji již jednou zpracoval, tak by tento uzel dané `payment_preimage` znal, aniž by musel kontaktovat cílového příjemce, čímž by se mohl nespravedlivě obohatit.

Příklad stejné faktury v QR kódu i textové reprezentaci:



lnbc25u1p3p2dq6pp530533lf8fuxthurlup3xtn84tjzhssmtu24h0kd6yapgqqd3tdcqdqu23jhxar0weskx6fqwpkxzarzvyhqcqpjsp5uqg0z6kpetv7uhedqnkpwq6cq6e7w383fc74chuctgmc6vjpp2mq9qtzqqqqqqysgqxqyjw5qrzjqwryaup9lh50kkranzgcdnn2fgvx390wgj5jd07rwr3vxeje0glcllc9md0duw8unqqqqqlgqqqqqeqqjqvlfagke8lfm28ptheehf52gz4q9e0mfwqf85txm4l0r97lvnxzvztsr4gnfdfgkqp26e0zqtw2c2q06y0aphttnsfjz20klz26t2kygq lefzau

Textová podoba faktury se skládá ze dvou částí:

- Lidsky čitelná část (zvýrazněno tučně)
- Datová část (zbytek)

Veškeré lightningové faktury začínají písmeny `ln`, za nimiž následuje specifikace bitcoinové sítě (`bc` pro mainnet, `tb` pro testnet a `bcrt` pro regtest). Nejčastěji se tedy setkáte s prefixem `lnbc`. Následuje číslo a za ním jeho jednotka. Ta je vyjádřena v bitcoinech s předponou **mili**, **mikro**, **nano** nebo **piko**. Detailněji to znázorňuje následující tabulka:

Symbol	Jednotka	Bitcoinu	Převod na satoshi
m	mili	0,001	*100 000
u	mikro	0,000 001	*100
n	nano	0,000 000 001	*0,1
p	piko	0,000 000 000 001	*0,000 1

Za mě osobně je pro běžné používání nejpodstatnější poslední sloupec. Pro většinu plateb v rámci Lightningu si stačí pamatovat, že hodnotu před písmenem **u** násobím stovkou, naopak hodnotu před písmenem **n** dělím deseti. Ve faktuře výše máme uvedeno **25u**, což znamená, že faktura byla vystavena na 2 500 satoshi. Podobně například **200n** odpovídá 20 satoshi a případně **2m** znamená 200 000 satoshi. Díky tomu můžete na první pohled z textové podoby zjistit, na jakou částku byla faktura vytvořena, aniž byste ji museli načítat vaší peněženkou.

Po této „lidsky čitelné části“ následuje datová část, která se skládá z časové známky, několika záznamů typu klíč/hodnota a podpisu. Díky aktuální časové známce (vyjádřené v Unix-timestamp, tedy počet vteřin od roku 1970) můžeme fakturu časově omezit – standardní platnost je 1 hodina. Podpis nám zaručuje, že fakturu opravdu vydala požadovaná protistrana. A konečně záznamy typu klíč/hodnota nám umožňují do faktury vložit libovolná data, ta nejčastěji používanájsou:

- `payment_hash` – hash `payment_preimage`, identifikátor platby
- **ID uzlu cíle** – jeho veřejný klíč
- **popis platby** – libovolné
- **expirace faktury**
- `min_final_cltv_expiry` – udáváno v počtu bloků, viz kapitola o Onion routing
- **fallback on-chain adresa** – lze specifikovat klasickou bitcoinovou adresu, kam budou prostředky zaslány, pokud lightningová platba selže
- **routing hints** – potřebné informace pro využití privátních kanálů
- velmi podobné zprávám `channel_update`
- obsahuje veškeré potřebné informace pro sestavení cesty jako ID uzlu, s nímž je privátní kanál navázaný, ID privátního kanálu, poplatky, `cltv_expiry_delta` apod.

Dekódovaná faktura výše:


```
{
    „destination“:
„02dfcda4fc5476a4a2e9ff609ac43ec1b0b8d6b95105fc2021ac05db5ede38fc75“,
    „payment_hash“:
„8be918fd274f0cbbf07fe06265ccf44c8578436be2ab77d9ba27428001b15b70“,
    „num_satoshis“: „2500“,
    „timestamp“: „1645556762“,
    „expiry“: „604800“,
    „description“: „Testovaci platba.“,
    „description_hash“: „“,
    „fallback_addr“: „“,
    „cltv_expiry“: „18“,
    „route_hints“: [
        {
            „hop_hints“: [
                {
 „node_id“:
„03864ef025fde8fb587d989186ce6a4a186895ee44a926bfc370e2c366597a3f8f“,
                    „chan_id“: „1837633505505686 2360“,
                    „fee_base_msat“: 1000,
                    „fee_proportional_millionths“: 100,
                    „cltv_expiry_delta“: 144
                }
            ]
        }
    ],
    „payment_addr“:
„e010f16ac1cad9ee5f2d04ec17035806b3e744f14e3d5c5f985a378d32410ab6“,
    „num_msat“: „2500000“,
    „features“: {
        „8“: {
            „name“: „tlv-onion“,
            „is_required“: true,
            „is_known“: true
        },
        „14“: {
            „name“: „payment-addr“,
            „is_required“: true,
            „is_known“: true
        },
        „17“: {
            „name“: „multi-path-payments“,
            „is_required“: false,
            „is_known“: true
        },
        „51“: {
            „name“: „unknown“,
            „is_required“: false,
            „is_known“: false
        }
    }
}
```

HODL faktury

Speciálním typem faktur jsou tzv. HODL faktury. Při běžné platbě si odesílající uzel přečte z faktury ID cílového uzlu a spočítá optimální cestu. Jednotlivé uzly si mezi sebou sestaví HTLC a příjemce **ihned**, jakmile to je možné, prozradí patřičné protistraně `payment_preimage` výměnou za prostředky uložené v HTLC. Tímto způsobem se uzamčené satoshi přesunou od odesílatele k příjemci.

U HODL faktur je zpracování téměř identické, jen s tím rozdílem, že příjemce neprozradí `payment_preimage` ihned, ale **čeká na splnění určitých podmínek**. Jako příklad uveďme, že si chcete koupit nějaký díl na auto, který je nutné vyrobit na zakázku. Prodejce vám v tomto případě vystaví HODL fakturu, kterou vy zaplatíte. Vám se sice prostředky z peněženky odečtou, ale platba se finálně nezpracuje, jelikož prodejce neuvolní `payment_preimage`. Místo toho čeká, až se mu váš díl na zakázku naskladní, a teprve poté toto náhodné číslo uvolní, čímž dojde k finalizaci platby. Případně by mohl platbu „stornovat“ (odesláním `update_fail_htlc`), čímž by se vám prostředky vrátily.

Zní to možná zajímavě, provází to ale několik potenciálních problémů:

- Uzel, který přeposílá platbu, **nemůže rozeznat**, zda se jedná o klasickou či HODL fakturu. V druhém případě vezme své prostředky, uzamkne je do HTLC (tudíž je nemůže dočasně používat), ale k vypořádání nedojde do vteřiny, jak je obvyklé, nýbrž to může trvat hodiny až dny. Jeho satoshi jsou tedy po tuto dobu nepoužitelné.

- S tím se váže i maximální limit 483 aktivních HTLC v rámci jednoho kanálu (limit pro velikost transakce v bitcoinové síti, aby byla stále platná). Pokud by se HODL faktury rozmohly, některé kanály by tímto mohly být dočasně kompletně vyčerpány, a tudíž by se nedaly používat.

- Pokud nestihne prodejce rozhodnout o výsledku (uvolnění `payment_preimage` či stornování) do limitu daného `min_final_cltv_expiry`, dojde k nucenému uzavření kanálu ze strany posledního uzlu na cestě. Aktuálně v tomto případě platí on-chain poplatky ten, kdo kanál otevřel, tedy je možné, že předposlední uzel jednak přijde o kanál, ale ještě za to bude muset zaplatit, aniž by udělal cokoliv špatně. On-chain poplatky se navíc budou platit hned dvakrát – za commitment transakci a poté SweepOut HTLC (druhá úroveň transakcí).

- Bohužel zde nemáte nikdy jistotu, že prodejce nebude podvádět. Pokud by mu díl nedorazil a on uvolnil `payment_preimage`, tak máte smůlu. Je zde tedy opět důležitá důvěra. Každopádně možná je toto řešení lepší, než zaplatit dopředu, a pak se dožadovat vrácení peněz zpět.

Keysend

Zatím jsme si představovali koncepty plateb, kdy bylo vždy nutné nejprve vytvořit fakturu, která obsahovala `payment_hash`. Tento krok se nám ale nemusí vždy hodit – například streamer bude chtít přijímat donate během živého přenosu. Pokud by si na to vystavil fakturu, tak to přináší dva problémy:

- Faktura má omezenou platnost.
- Lze ji zaplatit pouze 1x, takže první fanoušek by mu ji pro ostatní zneplatnil.

Co kdybychom ale chtěli někomu odeslat nějaké satoshi bez faktury? Jedním z řešení je tzv. **keysend**. V tomto případě pouze odesílatel a příjemce vědí, že je „něco jinak".

Aktuálně se pro kódování onion paketu používá TLV (Type-Length-Value). Tento elegantní formát nám mimo jiné umožňuje dovnitř šifrovaného paketu přidat libovolnou další informaci ve formátu klíč/hodnota.

Hlavním rozdílem u keysend je, že `payment_preimage` negeneruje příjemce, ale vytváří si ho sám odesílatel. Ten si tedy vygeneruje náhodné `payment_preimage`, k omu spočítá odpovídající `payment_hash` a celé to odešle cílovému uzlu. Hodnota `payment_preimage` je do onion paketu přidána díky TLV formátu a je zašifrována tak, že si ji může přečíst pouze finální příjemce. Ten po rozbalení paketu zjistí, že obdržel keysend platbu. Zároveň tímto způsobem od odesílatele obdržel i `payment_preimage`, které použije standardně pro vypořádání všech HTLC po cestě. Vzhledem k tomu, že bylo `payment_preimage` v původní zprávě zašifrované, tak si ho nikdo nemohl přečíst a vlastně ani žádný uzel uprostřed nevěděl, že odesílá keysend platbu.

Není zde tedy žádná faktura a cílem platby je ID uzlu – veřejný klíč, od toho název keysend. Tyto platby je většinou nutné povolit, ale ne veškeré implementace uzlů či peněženek je plně podporují. Určitou nevýhodou je, že odesílatel **nemá žádný důkaz** o proběhlé platbě. U standardního placení přes fakturu může odesílatel příjemci ukázat `payment_preimage`. Jediným způsobem, jak se k odesílateli toto náhodné číslo mohlo dostat, bylo úspěšným vypořádáním všech HTLC po cestě. V případě keysendu si ale toto číslo vygeneroval sám odesílatel, tento důkaz tedy nemůže použít.

Zároveň je vhodné do zprávy přidat nějaký popis (opět v rámci TLV), aby si příjemce mohl platbu k něčemu přiřadit, pokud nemá být anonymní. Pro obchodníky to tedy není moc výhodné, ale pro někoho, kdo chce přijímat anonymně platby (například sbírka), to může být zajímavé. Poslední nevýhodou, kterou je nutné zmínit, je to, že u keysendu nelze používat privátní kanály na straně příjemce, jelikož zde nemáme žádné routing hints jako na faktuře.

LNURL

LNURL je sada protokolů, které rozšiřují možnosti používání Lightning Network s cílem zlepšit uživatelskou přívětivost. Ve své podstatě se nejedná o nic jiného, než **URL odkaz na webovou službu** pomocí protokolu HTTPS (alternativně také na Tor Hidden Service), který je zakódovaný pomocí bech32 algoritmu.

LNURL1DP68GURN8GHJ7UM9WFMXJCM99E3K7MF0V9CXJ0M385EKVCEN
XC6R2C35XVUKXEFCV5MKVV34X5EKZD3EV56NYD3HXQURZEPEXEJXXEP
NXSCRVWFNV9NXZCN9XQ6XYEFHVGCXXCMYXYMNSERXFQ5

https://service.com/api?q=3fc3645b439ce8e7f2553a69e5267081d96dcd340
693afabe04be7b0ccd178df

V rámci LNURL se vždy jedná o komunikaci mezi lightningovou **peněženkou** (nejčastěji na mobilním telefonu) a **službou**, která běží na webovém serveru. Základní princip je takový, že uživatel nejprve pomocí peněženky naskenuje QR kód. Přívětivou vlastností je, že na první pohled není QR kód pro běžnou fakturu od toho pro LNURL **rozeznatelný**, uživatel tedy nemusí vůbec vědět, že na pozadí se děje „něco jiného". Po dekódování URL peněženka tuto stránku navštíví a odpovědí je JSON objekt. Existuje několik standardů LNURL, na ty nejpoužívanější se nyní podíváme. Ne každý standard je podporován všemi peněženkami, ty lepší jej ale podporují téměř všechny.

LNURL-payRequest

Idea je zde taková, že uživatel naskenuje QR kód LNURL a jeho peněženka na pozadí navštíví danou webovou adresu. Z té získá informace o platbě, primárně popis a částku. V závislosti na nastavení služby může být částka variabilní, např. v rozmezí 100 až 10 000 satoshi. Uživatel zadá, kolik chce zaplatit, a peněženka s těmito informacemi opět kontaktuje službu. Ta na tomto základě **vygeneruje fakturu** pro určenou částku, tu mu odešle a peněženka ji zaplatí.

Jedná se tedy o to, že v rámci QR kódu není uložena žádná faktura. Ta se vygeneruje a odešle uživateli až na základě nějakých informací, které on zadá. Velkou výhodou je, že tento LNURL QR kód **může být statický** (lze ho využít vícekrát), je to tedy vhodné a často používané řešení pro příjem nějakých dobrovolných a anonymních příspěvků například uveřejněním na webu.

Detailnější postup zpracování:

- **Uživatel** naskenuje QR kód. Alternativně klikne na odkaz začínající řetězcem `lightning:LNURL`...
- **Peněženka** provede HTTPS GET požadavek na dekódované URL.
- **Služba** vrátí JSON objekt, který obsahuje:
    - `callback` – URL služby, kam se mají odeslat detaily platby
    - `maxSendable` – maximální částka
    - `minSendable` – minimální částka (pokud je stejná jako `maxSendable`,
      uživatel nemá možnost výběru)
    - `metadata` – informace o platbě v textové či grafické podobě
    - `tag` – v tomto případě „payRequest“
- **Peněženka** uživateli zobrazí dialog, kde uvidí doménu služby, popis platby (případně i obrázek), a pokud má tu možnost, tak mu dovolí si nastavit částku. **Uživatel** vyplní požadované informace.
- **Peněženka** provede HTTPS GET požadavek na callback URL s informací, kolik chce uživatel zaplatit.
- **Služba** mu vrátí JSON objekt, který obsahuje příslušnou **fakturu** na danou částku.
- **Peněženka** fakturu zaplatí.

LNURL-payToAddress

Funkce placení na „Lightning adresu“ velmi zlepšuje uživatelskou přívětivost. Pokud ji totiž vaše peněženka podporuje, lze odeslat nějaké satoshi kamarádovi bez generování jakékoliv faktury a skenování QR kódu. Bohatě stačí znát jeho adresu ve formátu `uživatel@doména`, ke které má přiřazenou peněženku, a na ni prostředky odeslat. Například si můžete zkusit odeslat finanční příspěvek na `michal@platbybudoucnosti.cz`.

I když to tak na první pohled nevypadá, technické provedení je po znalosti LNURL-payRequest velmi triviální. Peněženka načte URL ve tvaru:
`https://<doména>/.well-known/lnurlp/<uživatel>`

Takže například:
`https://platbybudoucnosti.cz/.well-known/lnurlp/michal`

Odpovědí je již klasický JSON objekt představený v LNURL-payRequest. Ve skutečnosti tedy cílovou LNURL adresu nenačítáte z QR kódu, ale sestavíte si ji z lightningové adresy.

LNURL-withdrawRequest

Představme si případ, že na webové stránce, kde je možnost hrát ruletu, za satoshi vyhrajeme poměrně zajímavou částku a chceme si ji odeslat na naši mobilní peněženku. Bez použití LNURL bychom byli nuceni vygenerovat fakturu na mobilu a tu nějak nahrát na web, aby nám byla proplacena. Obzvlášť v případě, kdy ruletu hrajeme na počítači, to může být komplikovanější. Co kdyby nám stačilo pouze naskenovat mobilním telefonem QR kód z webu a peníze takto přijmout? Přesně na to existuje LNURL-withdrawRequest.

Detailnější postup zpracování:

- **Uživatel** naskenuje QR kód. Alternativně klikne na odkaz začínající řetězcem `lightning:LNURL`...
- **Peněženka** provede HTTPS GET požadavek na dekódované URL.
- **Služba** vrátí JSON objekt, který obsahuje:
    - `callback` – URL služby, kam se odešle faktura
    - `maxWithdrawable` – maximální částka
    - `minWithdrawable` – minimální částka (pokud je stejná jako `maxWithdrawable`, uživatel nemá možnost výběru)
    - `metadata` – informace o stránce v textové či grafické podobě
    - `tag` – v tomto případě „withdrawRequest“
    - `k1` – identifikace uživatele

- **Peněženka** uživateli zobrazí dialog, kde uvidí doménu služby, popis (případně i obrázek), a pokud má tu možnost, tak mu dovolí si nastavit částku, kterou si chce vybrat.
- **Uživatel** vyplní požadované informace.
- **Peněženka** provede HTTPS GET požadavek na callback URL, kam odešle právě vygenerovanou fakturu a identifikaci uživatele (`k1`).
- **Služba** vše zkontroluje a fakturu zaplatí.
- **Uživatel** obdrží požadované satoshi do své peněženky.

Ostatní LNURL specifikace

Zajímavým konceptem je určitě i **LNURL-auth**, které vám umožňuje se registrovat a přihlašovat ke službám pouze vaší lightningovou peněženkou bez jakéhokoliv vyplňování jména a hesla. Vše funguje pouze tak, že na webu uvidíte QR kód, ten naskenujete a jste přihlášen pod identitou vaší peněženky. Na pozadí vám služba totiž vygenerovala náhodná data, která jste podepsal s využitím vašeho privátního klíče. Kdokoliv se tedy zmocní vaší peněženky, má i vaši identitu (na webech, kde jste ji použil).

Posledním konceptem, který si představíme, je **LNURL-channelRequest**. Ten slouží k tomu, pokud nějaká služba nabízí možnost, že s námi otevře kanál (zdarma či za poplatek). Uživatel naskenuje QR kód a peněženka klasicky odešle HTTPS GET požadavek. Vrátí se mu informace o uzlu (ID, adresa, port), jeho identifikace a callback URL. Peněženka se síťově spojí s daným uzlem a provede další HTTPS GET požadavek na callback URL, kam odešle ID svého uzlu a informaci, zda má být kanál privátní. Následně cílová služba otevře kanál směrem k naší peněžence, případně k našemu uzlu.

Existují i další specifikace, které otevírají nové možnosti práce s Lightningem nebo případně rozšiřují ty výše uvedené. Jejich detailní specifikaci lze najít na GitHub stránce k LNURL.

Rozšíření a budoucnost Lightning Network

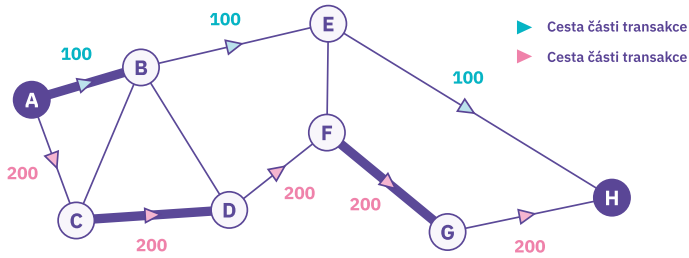
V této již poněkud pokročilejší kapitole v rámci teorie se podíváme na různá vylepšení a rozšíření Lightning Network. Některé z nich jsou stále ve fázi vývoje,

jiné už mohou být částečně či plně nasazené. Každá implementace Lightningu má totiž jinak stanovené priority, a tak je možné, že některá z funkcionalit nebude zatím dostupná všude.

Multi-part Payments

Dále jen MPP. Jedná se o funkcionalitu, která byla zavedena v průběhu roku 2020 a umožňuje jednu platbu **rozdělit na více menších**, kdy každá část putuje k příjemci vlastní cestou. Díky tomu je možné odesílat platby o vysoké hodnotě (například v řádech milionů satoshi), u nichž by byl bez této funkcionality problém s doručením, protože je potřeba série kanálů o velké kapacitě a dostatečné likviditě v odesílaném směru.

Jedním z nejnáročnějších problémů, jehož optimální řešení se aktuálně stále zkoumá, je optimalizace ideálního rozdělení transakce. Pokud odesílanou částku rozložím na mnoho částí, tak je u každé z nich sice větší šance, že projdou díky menším požadavkům na kapacitu a likviditu kanálů, na druhou stranu použiji více cest (tedy více uzlů) a s tím mi stoupá riziko selhání alespoň jednoho z nich. Každá implementace si toto řeší po svém v rámci částečného konkurenčního boje. Ideální je najít nějakou střední cestu, kdy se velká částka rozdělí na „rozumný počet“ menších.



Technicky to není nikterak složité. Podmínkou je, aby tuto funkcionalitu podporoval odesílatel a příjemce. Uzlům po cestě může být jedno, zda přeposílaná částka je celá, nebo se jedná pouze o její fragment, chovají se k ní naprosto stejně. Odesílatel rozloží částku na ideální počet částí a pro každou z nich nalezne trochu jinou cestu (odesílat všechny přes stejné uzly by nedávalo smysl). Příjemce se k přijatým platbám chová jako k HODL fakturám – neuvolňuje `payment_preimage` ihned, ale čeká, než k němu doputují všechny části. Neví sice, kolik jich je, ale ví celkovou částku (z faktury) a kolik aktuálně přijal. Jakmile se k němu dostanou všechny HTLC (jejich součet odpovídá částce z faktury), tak `payment_preimage` uvolní naráz a dojde k jejich vypořádání. Naopak odesílatel monitoruje, zda se mu nějaká část

platby vrátí (zpráva `update_fail_htlc`), a pokud se tak stane, odešle konkrétní část znovu, tentokrát jinou cestou.

Toto řešení má ještě další dvě výhody – jednak lehce zvyšuje anonymitu, protože rozdělí platbu na více částí a žádný uzel po cestě (pokud vše neteče přes jeden jako například v případě některých mobilních peněženek) neví celkovou částku. Za druhé je také díky tomu možné odeslat částku, která se rovná součtu odchozí likvidity všech mých kanálů. Jinými slovy, pokud budu mít v jednom kanálu 100 satoshi a v druhém 200 satoshi (vždy jako odchozí likviditu na své straně), mohu díky MPP poslat celkově až 300 satoshi.

Atomic Multi-path Payments

Dále jen AMP. Jedná se o jakousi kombinaci vylepšeného MPP spolu s keysendem. U klasických MPP plateb popsaných v předchozí kapitole mám teoreticky dva problémy. Tím prvním je, že každá část putuje lightningovou sítí se stejným identifikátorem `payment_hash`, a tudíž pokud někdo kontroluje více uzlů, může tyto jednotlivé části spolu korelovat. Druhým problémem je, že příjemci nic nebrání v tom, aby uvolnil vygenerované `payment_preimage`, i když přijal pouze například polovinu z celkové částky. Reálně by to sice pro něj nejspíše nemělo žádný význam, protože by okradl sám sebe, ale dostali bychom se do stavu, kdy byla platba zaplacena částečně, což nechceme. AMP je ve svém návrhu podobné keysendu – platba se odesílá na ID uzlu, není nutné dopředu vytvářet fakturu. Navíc je zde ale několik vylepšení. Prvním rozdílem je to, jak se chováme k `payment_preimage`. Stejně jako u keysend plateb jej netradičně generuje odesílatel, nyní jsou k němu však pro každou část platby přidána náhodná data. Reálné `payment_preimage` nutné pro vypořádání HTLC příjemce získá tak, že provede operaci XOR nad všemi přijatými „upravenými“ `payment_preimage`. Tím je zaručeno, že zpráva bude **atomická** – projde buď celá, anebo neprojde vůbec. Příjemce totiž potřebuje všechny části, aby si spočítal originální `payment_preimage`. I samotná korelace zpráv ostatními uzly je v tomto případě nemožná, protože je nelze k sobě nikterak přiřadit.

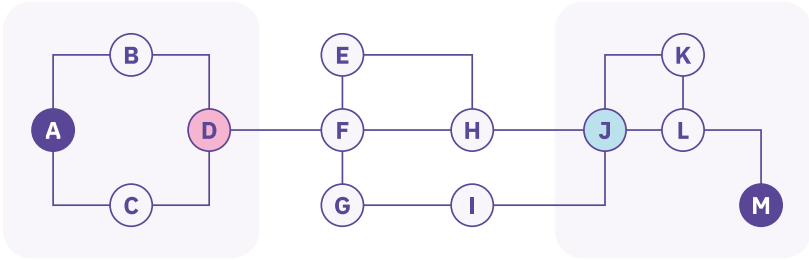
Druhou výhodou je, že AMP platby **podporují statické faktury** – jedná se o lehce pozměněný typ faktury, ve kterém je uloženo ID cílového uzlu s informací, že se má využít AMP. Tuto fakturu tedy lze vystavit na webu a přijímat platby od ostatních podobně jako s využitím LNURL-Pay. Navíc oproti keysend zde lze do faktury přidat routing hints, a tedy využívat i privátní kanály.

Jedná se zatím o celkem novou koncepci, kterou zdaleka nepodporují všechny implementace a mobilní peněženky. Dobrou zprávou ale je, že podpora musí být pouze na straně odesílatele a příjemce – pro uzly uprostřed zde není v routování žádný rozdíl.

Trampoline routing

Jedná se o funkcionalitu, která umožňuje primárně mobilním zařízením přenechat výpočet optimální trasy k příjemci nějakému jinému uzlu, kterému budeme říkat **trampolína** (na obrázku níže jsou tyto uzly označeny písmeny D a J). Problémem, který tato technologie řeší, je fakt, že udržovat neustále aktuální graf celé lightningové sítě může být pro mobilní zařízení paměťově náročné. Dále také samotná synchronizace zabere nějaký čas, pokud bylo zařízení delší dobu offline. Nakonec i výpočet optimální trasy může být s rostoucím počtem uzlů a kanálů pro mobilní zařízení čím dál náročnější.

Představme si příklad, kdy **A**lice (uzel A) chce zaplatit fakturu **M**artinovi (uzel M), a obě tyto instance jsou mobilní peněženky. V tomto případě jim stačí znát vždy pouze svoje nejbližší okolí (vyznačené fialovým čtvercem), nikoliv celý graf lightningové sítě.



Uzly v rámci gossip protokolu ve zprávě `node_announcement` mohou šířit informaci, zda jsou či nejsou trampolínou. **M**artin vygeneruje fakturu a specifikuje do ní trampolínu ze svého okolí, kterou zná, konkrétně **J**akuba. Tuto fakturu odešle **A**lici.

Alice po naskenování příslušného QR kódu zjistí, že musí kontaktovat trampolínu **J**akuba. Vybere tedy jinou trampolínu ze svého okolí, konkrétně Davida, a sestaví trasu **A**lice → **D**avid → **J**akub → **M**artin. Z této trasy sestaví speciální **„trampoline onion packet“** (jedná se o strukturu podobnou klasickému onion paketu). Nyní nalezne cestu k **D**avidovi pomocí kanálů ze svého okolí, konkrétně přes **A**lici a **B**arboru. Do klasického onion paketu pro **D**avida vloží trampoline onion paket, který mu říká, že má najít cestu k **J**akubovi. Jakou cestu zvolí, je už na něm. **D**avid (trampolína), který má znalost celé lightningové sítě, vybere jako ideální trasu k druhé trampolíně **J**akubovi přes **F**ilipa, **G**ábinu a **I**vana. Opět do onion paketu přibalí speciální trampoline onion paket, který **J**akubovi říká, aby našel cestu k **M**artinovi. Ten zvolí nejkratší cestu přes **L**ukáše.

Tímto způsobem tedy mohu využít dvou trampolín, kterým vlastně jen řeknu, koho mají kontaktovat, ale výpočet optimální trasy nechám na nich. Počet trampolín je libovolný a soukromí tímto specifickým způsobem routování není nijak narušeno.

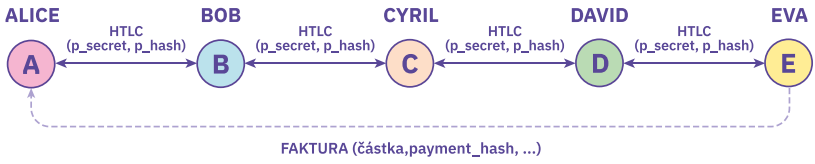
Trampolína **D**avid totiž ví pouze to, že odesílatelem je buď **B**arbora, anebo někdo před ní, a příjemcem je buď **J**akub, anebo někdo za ním. Trampoline routing lze kombinovat s MPP, kdy každá trampolína může platbu ještě dále rozdělit na několik menších podle svého uvážení.

Aktuálně tuto funkcionalitu zdaleka nepodporují všechny mobilní peněženky ani lightningové implementace. Každopádně pokud bude velikost sítě narůstat stejnou rychlostí jako doposud, masivní rozšíření tohoto řešení je velmi reálné.

Point Time Locked Contracts

Princip HTLC jsme si v této knize již vysvětlili a samotný termín byl použit nesčetněkrát. Pro osvěžení – jedná se o speciální výstup transakce s podmínkami, který zajišťuje, že lze odesílat platby i uzlům, s nimiž nemám napřímo otevřený kanál. V případě problémů lze transakci s tímto výstupem odeslat do bitcoinové sítě, čímž zamezíme podvodům, případně trvalému zaseknutí prostředků.

Jednou z hlavních nevýhod HTLC je fakt, že `payment_hash` je stejný pro všechny uzly po cestě. Tudíž pokud někdo ovládá více uzlů, přes které konkrétní platba putuje, mohl by čistě teoreticky odhalit odesílatele a příjemce. Použijme následující příklad, kdy Alice odesílá satoshi Evě přes Boba, Cyrila a Davida. Nejprve Eva vygeneruje `payment_preimage`, spočítá z něj `payment_hash` a ten odešle ve faktuře Alici. Následně se naváže série HTLC mezi uzly, kdy k vypořádání platby dojde vždy po předání odpovídajícího `payment_preimage`.



Dále počítejme s tím, že dva uzly po cestě (v našem případě konkrétně Boba a Davida) ovládá společnost, která chce deanonymizovat lightningovou síť. Protože se používá stejný identifikátor, mohou přijít na to, že se jedná o jednu a tu samou platbu a Alice nejspíše odesílá prostředky Evě. Využitím onion routingu sice Bob netuší, zda je odesílatelem Alice nebo někdo před ní, existují ale různé metody, které to mohou s určitou pravděpodobností potvrdit. Příkladem může být:

- **Celková délka trasy.** Prakticky v Lightningu má málokterá platba více než 4–6 prostředníků, protože s každým dalším se zvyšuje pravděpodobnost selhání.

Pokud je cesta mezi špehujícími uzly již velmi dlouhá, tak lze předpokládat, že sousedící uzly jsou odesílatelé nebo příjemci.

- **Detekce mobilní peněženky**. Jedná se o uzel, který je čas od času offline, velmi často se mu mění IP adresa a nepřesměrovává žádné platby.

- **Počet kanálů**. Pokud má sousedící uzel pouze 1 kanál, je to většinou odesílatel nebo příjemce. Výjimkou může být, pokud by měl nějaké další, které by byly privátní.

Výše popsaný příklad neznamená, že by používání HTLC nebylo soukromé nebo dokonce nějak nebezpečné. Aby útočník dokázal odhalit odesílatele a příjemce, musí ovládat větší množství uzlů, tyto uzly musí být zapojené do platby (ideálně na začátku a konci), a i když toto vše splní, nemůže si být odhalením jistý. Na základě různých heuristik může prohlásit, že s pravděpodobností 85 % je odesílatelem Alice a příjemcem Eva.

Právě tento problém řeší nový koncept – **Point Time Locked Contracts**, dále jen PTLC. Pokud jste tedy měli radost, že jste konečně pochopili princip HTLC, tak pro vás nemám dobrou zprávu. Existuje totiž reálná šance, že se v nejbližší době nahradí HTLC za PTLC. Na druhou stranu se zase můžete naučit něco nového. Než se konkrétně pustíme do principů PTLC, podíváme se lehce na kryptografii nad eliptickými křivkami.

Kryptografie nad eliptickými křivkami

Nejprve si definujme skalár, tedy celé číslo (1, 2, 3 až n) jako malé písmeno, např. `k`.
Dále si definujme bod na eliptické křivce jako velké písmeno, např. `P`.
Pro následující výpočty budeme potřebovat operaci **sčítání bodů** nad eliptickou křivkou. Definujme si jej klasickým symbolem `+`, kdy platí, že pokud sečtu dva body na křivce (např. `P` a `Q`), tak obdržím `R`, které je opět bodem na křivce. Tedy `R = P + Q`.

Dále budeme potřebovat operaci **násobení bodů** nad eliptickou křivkou. Definujme si jej klasickým symbolem `*`, kdy se vlastně jedná o několikanásobné sečtení toho samého bodu. Takže `k * P = P + P + P + … + P` (celkově `k`-krát). Výsledkem je opět bod na křivce. Platí, že pokud známe bod `P` a výsledek `k * P`, tak nelze zjistit `k` jinak, než vyzkoušením všech hodnot – jedná se tedy o velmi náročnou a při dostatečně velkém `k` až neřešitelnou úlohu. Existují sice sofistikovanější algoritmy řešící problém diskrétního logaritmu, z našeho pohledu jsou však stále extrémně pomalé. Mimo jiné má násobení distributivní vlastnost, tedy `a * P + b * P = (a + b) * P`. Definujme si privátní klíč `p` jako dostatečně velké náhodné číslo. Odpovídající veřejný klíč `V` je poté spočítán jako `p * G`, kde `G` je generátor dané grupy. Opět zde platí, že získat privátní klíč `p` se znalostí `G` a veřejného klíče `V` je nereálné. Poslední věcí, která je snad všem jasná, ale je vhodné ji zde připomenout, je fakt, že pokud znám pouze celkový součet `a + b + c`, tak nemohu zjistit jednotlivá čísla `a`, `b` ani `c`. Dokonce i kdybych znal například `a`, tak nemohu vypočítat přesně `b` ani `c`.

Vraťme se opět k příkladu výše – tedy Alice chce odeslat platbu Evě a jako prostředníky využije Boba, Cyrila a Davida. Podívejme se nyní, jak by takováto platba probíhala s využitím PTLC:

- Eva chce přijmout od Alice platbu. Vygeneruje si náhodný privátní klíč `p` (obdoba `payment_preimage`) a spočítá si k tomu odpovídající veřejný klíč `V` jako `p * G`. Tento veřejný klíč předá Alici v rámci faktury, stejně jako jí dříve v rámci HTLC předávala `payment_hash`.

- Alice si vypočítá optimální cestu k Evě. Ta vede přes Boba, Cyrila a Davida.

- Alice si vygeneruje 4 náhodná čísla (podle počtu mezilehlých uzlů) `a`, `b`, `c` a `d`. Číslo `a` si uloží pro sebe, `b` odešle Bobovi, `c` Cyrilovi a konečně `d` Davidovi. Příjemci, tedy Evě, odešle v rámci onion paketu celkovou sumu všech těchto čísel, tedy (`a + b + c + d`).

- Alice se domluví s Bobem, že si do svojí commitment transakce přidají PTLC. V něm bude určeno, že prostředky obdrží Bob, pokud sežene privátní klíč k veřejnému klíči (`a + p`) `* G`, což je z definice (`a + p`). Bob dále z onion paketu zjistí, že Cyril zná privátní klíč k veřejnému klíči (`a + p`) `* G + b * G`, což odpovídá (`a + b + p`).

- Bob si uvědomí, že pokud by získal od Cyrila hodnotu (`a + b + p`), tak by mohl spočítat potřebné (`a + p`), a to tím způsobem, že `(a + p) = (a + b + p - b)`. Samotné číslo b totiž zná, protože mu jej zaslala Alice v rámci onion paketu. Tím pádem by splnil PTLC podmínku s Alicí a získal satoshi z tohoto kontraktu. Vytvoří si tedy PTLC s Cyrilem, kde žádá privátní klíč (`a + b + p`).

- Jakmile Cyril obdrží toto PTLC od Boba, tak se z onion paketu dozví, že David zná privátní klíč (`a + b + c + p`), který se váže k veřejnému klíči (`a + b + p`) `* G + c * G`. Kdyby ho získal, tak by díky znalosti c (od Alice) mohl prostým odečtením spočítat potřebné (`a + b + p`) a přivlastnit si hodnotu PTLC od Boba.

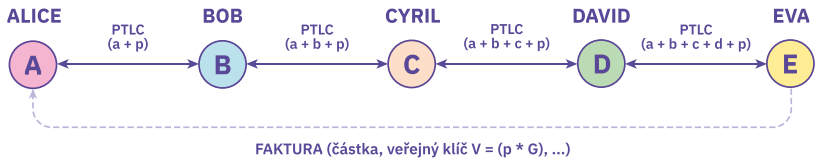
- To samé se opakuje u Davida. Ten si sestaví PTLC s finálním příjemcem Evou, kde žádá (`a + b + c + d + p`), aby si sám spočítal potřebné (`a + b + c + p`).

- Eva ale `p` zná, protože si jej sama vygenerovala. Dále zná i celkovou sumu čísel (`a + b + c + d`), jelikož jí ji odeslala Alice. Tím pádem může uvolnit (`a + b + c + d + p`) Davidovi, aby získala prostředky z jejich vzájemného HTLC.

- Nyní, když David zná (`a + b + c + d + p`), tak si může spočítat (`a + b + c + p`), protože `(a + b + c + p) = (a + b + c + d + p − d)`. Tím pádem si vezme prostředky od Cyrila.

- Stejným způsobem dojde k výpočtu a přijetí hodnoty uzamčené v HTLC u Cyrila a Boba.

- Konečně Alice obdrží od Boba (`a + p`). Jelikož Alice `a` zná (sama si ho vygenerovala), tak si může spočítat finální `p` jako `p = (a + p − a)`. Tím pádem získala důkaz o platbě a ta je nyní úspěšně vypořádána.



Díky tomuto konceptu je hodnota PTLC mezi jakýmikoliv dvěma uzly **rozdílná**, a tudíž i kdyby potenciální útočník ovládal více uzlů, nedokáže jednotlivé PTLC korelovat k jedné platbě.

PTLC neřeší jen soukromí – s určitou úpravou dokáže vyřešit i „**zaseknuté platby**“. Čas od času se totiž stane, že se transakce při platbě faktury zasekne. Může to být tím, že některý uzel po cestě přijal HTLC/PTLC a kvůli chybě přestal reagovat, případně se může jednat o někoho, kdo se rozhodne škodit. Jedná se poté o ne zrovna příjemnou situaci, jelikož platba „visí“ někde na cestě a nelze ji odeslat znova, protože by hrozilo, že se zaplatí obě. Odesílatel sice o svoje prostředky nepřijde, ale čekání na vypršení `cltv_expiry` může trvat hodiny až dny. Nejedná se o častý problém, ale pokud nastane, tak je uživatelská přívětivost Lightning Network značně ponížena.

Toto lze vyřešit s využitím PTLC tak, že odesílatel (Alice) **neodešle** příjemci (Evě) součet všech náhodných čísel v rámci onion paketu ihned. Naopak čeká, zda se podařilo sestavit sérii PTLC od Alice až k Evě. Jakmile se to podaří, Eva o tom informuje Alici v rámci speciální zprávy `ACK` (acknowledgement, potvrzení) a ta jí **až nyní** odešle sumu náhodných čísel. Odteď se pokračuje klasickým vypořádáním popsaným výše. Tímto způsobem lze tedy vyřešit zaseknutí PTLC při sestavování od odesílatele k příjemci. Limit na příjem `ACK` zprávy může být nastaven například na 1 minutu. Pokud jej do této doby neobdržím, **je bezpečné zkusit jinou trasu**. Nehrozí mi totiž, že by proběhly obě platby, jelikož Eva může začít s vypořádáním až po obdržení součtu

náhodných čísel. Pro všechny uzly uprostřed se nic nemění, nepotřebují tento nový typ zpráv podporovat. Jedinou výjimkou je routování `ACK` zprávy.

PTLC by sice šlo implementovat i nad dnešním ECDSA, ale elegantnější aplikace využívá Schnorr podpisů a signature adaptors. Tyto koncepty byly do Bitcoinu začleněny s aktivací Taprootu v roce 2021, a i když PTLC nejsou zatím při psaní této knihy reálně implementovány, jedná se díky výhodám, které přináší, pouze o otázku času, než první implementace spatří světlo světa.

BOLT 12

BOLT je zkratka pro **Basis of Lightning Technology**. Jedná se o sérii dokumentů, ve kterých je detailně technicky specifikováno, jak má Lightning Network fungovat, a jednotliví vývojáři by se tím měli řídit. Tato specifikace v době psaní této knihy obsahuje 11 dokumentů, které jsou postupně číslovány. Takže například BOLT#3 specifikuje formát on-chain transakcí, BOLT#4 popisuje onion routing, BOLT#7 obsahuje detaily P2P komunikace a gossip protokolu apod. Funguje to tak, že navrhnout nový (případně upravit stávající) BOLT dokument může kdokoliv. Pokud se tato navrhovaná změna uchytí ve více implementacích Lightningu, je poté prohlášena de facto za standard. My se nyní podíváme na BOLT dokument s číslem 12, který je aktuálně ve fázi návrhu a experimentální implementace.

Klasické faktury (popsané v BOLT#11) jsme si již rozebrali. Obsahují ID uzlu příjemce, celkovou částku, `payment_hash` a případně další prvky. Mají ale několik potenciálních drobných problémů:

- Fakturu lze zaplatit pouze 1x. Zaplacení té stejné faktury vícekrát se nedoporučuje z toho důvodu, že uzly po cestě již znají `payment_preimage`, a tak pro vypořádání HTLC nemusí vůbec kontaktovat příjemce.

- Faktura je nominovaná v satoshi. Při zatím celkem velké volatilitě bitcoinu je náročné cenit zboží přímo v satoshi. Proto jsem nucen fakturu generovat vždy těsně před samotným prodejem.

- Faktura aktuálně slouží pouze k příjmu, nikoliv k odeslání. Příklad využití odeslání „přes fakturu“ si popíšeme níže.

Jedním z řešení je právě **BOLT#12**, který faktury nahrazuje něčím, co se nazývá **Lightning offer**, což není opět nic jiného než řetězec, který je nejčastěji reprezentován ve formě QR kódu. V něm je primárně uloženo SKU kupovaného produktu (jedinečný kód typicky používaný obchody, který jasně popisuje a označuje daný produkt)

a dále ID uzlu. Po načtení tohoto QR kódu kontaktuje moje peněženka samotný cílový uzel a vyžádá si vygenerování faktury na konkrétní množství satoshi. Tuto fakturu poté peněženka zaplatí. BOLT#12 Offer lze tedy použít vícekrát.

Mimo jiné se zde počítá s **podporou předplatného**. Služba může například požadovat platbu 100 Kč každý měsíc a na tuto částku (v satoshi) bude generovat vždy novou fakturu. Další výhodou může být fakt, že BOLT#12 offers lze používat i k příjmu prostředků. Typickým případem může být automat, do kterého vložím bankovky, a budu chtít přijmout Lightning do své peněženky. Místo generování faktury na konkrétní částku na svém telefonu mohu pouze načíst QR kód z displeje automatu a přijmout tak zakoupené prostředky.

BOLT#12 Offer

Že vám to něco připomíná? Ano, BOLT#12 je velmi podobný LNURL. Výše popsaných funkcionalit lze samozřejmě dosáhnout s využitím nám již známých LNURL-payRequest a LNURL-withdrawRequest. Tím hlavním rozdílem je, že u LNURL se to řeší na aplikační vrstvě (peněženka kontaktuje webový server, který je pro LNURL nutný, jelikož se přistupuje na HTTPS URL), kdežto BOLT#12 to řeší přímo v rámci lightningového protokolu. K tomu se ale vážou i různé problémy.

Tím prvním je, že je potřeba upravit BOLT#4 – dokument, který popisuje sestavování onion paketů a routování těchto zpráv. Aktuálně totiž Lightning neumožňuje odesílat libovolné zprávy nějakému uzlu, proto se to řeší tak, že odešlu platbu například na 1 milisatoshi a do poznámky přidám požadovaný text. To není úplně čistý způsob, a tak se přišlo s efektivnějším způsobem – odesílání plateb bude odděleno od odesílání zpráv. Samotné poslání zprávy je mnohem jednodušší a rychlejší, protože neřeším žádné HTLC, `payment_preimage` apod. Aktuálně se počítá s routováním těchto zpráv zdarma. Zde ale nastává problém spamování nebo případného DoS/DDoS útoku. Z toho důvodu se bude muset pravděpodobně zavést nějaký limit a odesílání takovýchto zpráv nebude garantované, podobně jako u UDP spojení. To ale může vést k problémům s uživatelskou přívětivostí, pokud se vůbec nepodaří fakturu od uzlu získat. Vlastně se tím snažíme z Lightningu udělat jednak platební síť, ale zároveň i jakousi konkurenci k Toru – tedy k anonymnímu odesílání zpráv.

Dalším problémem je, že BOLT#12 obsahuje opravdu velké množství změn potřebných pro jeho implementaci (ne všechny jsem tomto textu rozebral), a tak je nutná kooperace mnoha implementací a peněženek. Vývoj takto rozsáhlých změn může mít nižší prioritu, pokud to lze zatím vcelku úspěšně nahradit pomocí LNURL, které je mnohem rozšířenější. Posledním zádrhelem, vázaným s výše

zmíněným předplatným, je skutečnost, že telefon je často v režimu spánku, a tak nemůže zaplatit fakturu. Je tedy nutné vymyslet způsob, jak v době odeslání faktury telefon probudit (push notifikace apod.).

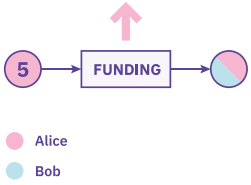
Až čas ukáže, jestli se BOLT#12 uchytí. Aktuálně se jedná o návrh, který je implementován od ledna roku 2021 jako experimentální funkce v rámci implementace Core Lightning.

Eltoo

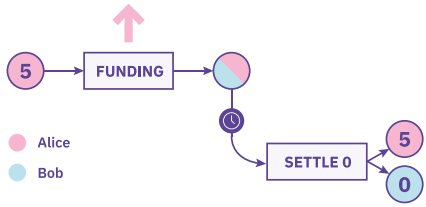
Eltoo (název pochází z anglické výslovnosti spojení L2, které označuje druhou vrstvu) je **nový princip platebních kanálů** pro Lightning Network. Zatím není nikde reálně implementován, protože vyžaduje upgrade v samotném Bitcoinu – konkrétně zavést flag `SIGHASH_ANYPREVOUT`, který se využívá při kontrole podpisu transakce.

U klasických platebních kanálů (někdy zvaných též LN-Penalty), které jsme do detailu probírali v teoretické části knihy, existuje nepěkná vlastnost spočívající v nutnosti si uchovávat veškeré potřebné informace, abychom mohli prokázat neplatnost **kterékoliv** z již revokovaných commitment transakcí. Pokud je kanál velmi využívaný a za svoji životnost měl statisíce aktualizací stavů, tak potřebné místo může být vcelku markantní. Zároveň je zde problém s tím, pokud byste omylem publikovali revokovanou commitment transakci. K tomu může dojít například při obnově ze zálohy, která nebude obsahovat naprosto poslední stav kanálu. Reálně se tak pokoušíte o podvod a riskujete veškeré své prostředky.

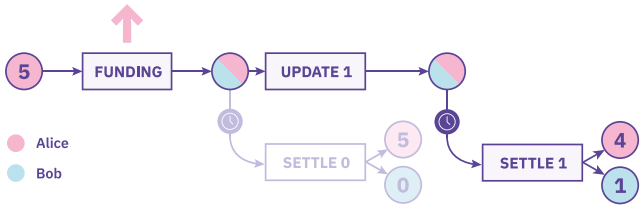
Pojďme se nyní podívat na příklad, jak fungují Eltoo kanály.



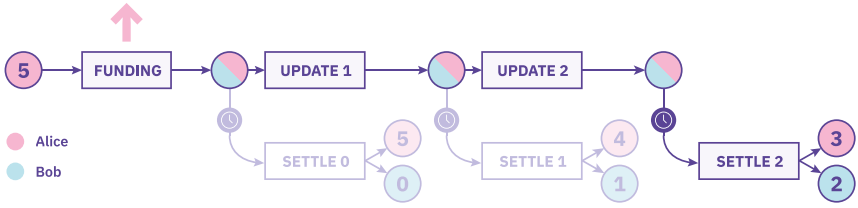
Na schématu výše jsou vstupy znázorněny kolečkem, transakce jsou v obdélníku. Představme si naše staré známé – Alici (růžová) a Boba (tyrkysová). Alice vytvořila funding transakci, jejímž vstupem je 5 bitcoinů, které vlastnila, a výstupem je multisig adresa 2 ze 2. Tato transakce byla publikována Alicí do bitcoinové sítě, což znázorňuje růžová šipka.



Na výstup z funding transakce napojíme takzvanou **settlement transakci**, která spravedlivě rozděluje prostředky mezi zúčastněné strany. Tedy 5 bitcoinů náleží Alici a Bobovi žádný. Tato transakce je z bezpečnostních důvodů opatřena relativním časovým zámkem (důvod si vysvětlíme později), takže k vypořádání nemůže dojít dříve, než tento zámek vyprší. Pro tento příklad si jeho hodnotu nastavme na 1 týden. Tuto settlement transakci neodesíláme do bitcoinové sítě, ponecháme si ji v paměti. Její obsah je pro obě strany kanálu identický.



Každá úprava stavu je reprezentována dvěma transakcemi – **settlement a update**. Alice odeslala tímto platebním kanálem jeden bitcoin Bobovi, tudíž vznikla nová dvojice těchto transakcí s pořadovým číslem 1. Update transakce utrácí multisig výstup z funding transakce, čímž efektivně zneplatňuje settlement 0. Není zde totiž žádný časový zámek, a tak díky ochraně proti doublespendingu je zaručeno, že se již nelze vydat cestou, kde Alice vlastnila 5 bitcoinů. Zároveň vzniká nová settlement transakce, která je rovněž opatřena relativním časovým zámkem a rozděluje prostředky mezi obě strany. Tentokrát má však Bob o jeden bitcoin více.

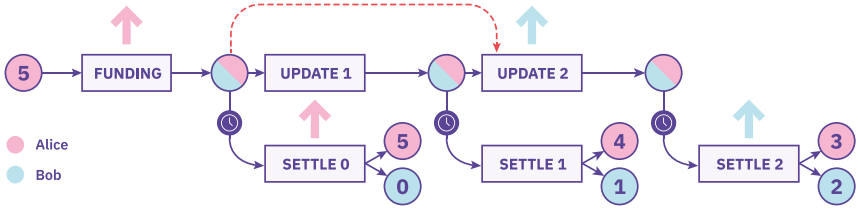


Alice opět odešle jeden bitcoin Bobovi a celý proces se opakuje. Vzniká tedy nová dvojice settlement a update transakcí s pořadovým číslem 2, které reprezentují aktuální stav. Takto by se dalo pokračovat do nekonečna.

Nyní to ale začne být zajímavé. Vzhledem k tomu, že do bitcoinové sítě byla zatím publikována pouze funding transakce, nic nebrání Alici v tom, aby odeslala settlement 0, která pro ni byla nejvýhodnější. Jak se proti tomu může Bob bránit? Velmi neefektivním řešením by bylo publikovat obě update transakce on-chain a poté si vzít prostředky settlement transakcí číslo 2. V tomto případě jsme ale vlastně řešení degradovali na klasické on-chain transakce, jelikož bychom vše, co se stalo v platebních kanálech, „zpětně přehráli" na blockchainu. To moc smysl nedává. My místo toho využijeme `SIGHASH_ANYPREVOUT`.

> **Sighash**
>
> Sighash flag určuje, která data v bitcoinové transakci musí být podepsána, čímž poskytuje určitou flexibilitu pro komplexní transakce s více účastníky. Pokud nějaká část transakce není podepsána, tak to znamená, že tuto nepodepsanou část lze upravit, a přitom zachovat transakci stále platnou. Například nejčastěji používaný flag `SIGHASH_ALL` znamená, že musí být podepsáno vše, a tudíž transakci nelze dodatečně nikterak změnit. Naopak námi využívaný `SIGHASH_ANYPREVOUT` říká, že vstup transakce nemusí být podepsán, takže mezi podepsáním transakce a jejím publikováním na blockchain **lze vstupy změnit**. Platnost transakce se poté ověřuje pouze na úrovni kompatibility skriptů (`scriptSig` a `scriptPubKey`). Jinak řečeno – lze vytvořit a podepsat transakci, ale přitom konkrétně nespecifikovat, jaké UTXO bude utrácet. Tento flag zatím nebyl do Bitcoinu implementován. Existuje na něj BIP118, a jestli bude někdy začleněn do samotného protokolu, ukáže až čas. Pokud totiž člověk či programátor nezná všechny jeho možné konsekvence, může být nebezpečný.

Bob by tedy detekoval pokus o podvod spočívající v publikaci transakce settlement 0 a reagoval by na to odesláním poslední update transakce, v našem případě update 2. Díky `SIGHASH_ANYPREVOUT` by ale mohl specifikovat, že jejím vstupem bude výstup funding transakce (znázorněno červeně), a tudíž tímto způsobem efektivně všechny mezilehlé update transakce přeskočit a doublespendovat settlement 0. Na zjednodušeném schématu níže bychom ušetřili pouze jednu on-chain transakci, pokud by ale platební kanál měl tisíce aktualizací, úspora by byla velmi znatelná. Následně Bobovi stačí pouze publikovat poslední settlement transakci a prostředky spravedlivě rozdělit. U Eltoo kanálů si tedy stačí pamatovat vždy **pouze poslední update a settlement transakci**.



Mezi výhody Eltoo kanálů patří jednodušší implementace, nižší požadavky na prostor pro ukládání aktuálního stavu a fakt, že transakce jsou u obou členů kanálu identické. Nehrozí zde problémy při obnově neaktuální zálohy a lze dokonce vytvářet platební kanály mezi více než dvěma účastníky. Určitou nevýhodou může být absence trestu pro podvádějící stranu (mimo zbytečně zaplacených on-chain poplatků) a skutečnost, že je potřeba upravit bitcoinový protokol. Pro podporu Eltoo kanálů není potřeba měnit celý Lightning Network stack. Pro implementaci stačí podporu tohoto nového typu platebních kanálů hlásit pomocí features bits ve zprávě `node_announcement` v rámci gossip protokolu. Oba typy platebních kanálů tedy mohou v Lightning Network existovat vedle sebe paralelně. Jejich zavedením by vlastně došlo k výměně pouze malé části protokolů z celého lightningového ekosystému – například gossip, onion routing apod. by zůstaly zachovány.

Jedná se o zajímavou koncepci platebních kanálů, kterou jsme si zde ale představili pouze velmi stručně. Pokud máte zájem o více detailů, doporučuji vám prostudovat si samostatný paper s názvem **eltoo: A Simple Layer2 Protocol for Bitcoin**. Autory jsou Christian Decker, Rusty Russel a Olaoluwa Osuntokun. Plný odkaz je uveden ve zdrojích.

Pickhardt Payments a #zeroBaseFee

Jedná se o iniciativu pro alternativní návrh pathfinding algoritmu. Autoři René Pickhardt a Stefan Richter ve své studii zjistili, že najít optimální cestu v grafu (pokud se využívá MPP, tedy rozdělení platby na více menších částí) je NP-úplný problém. Slovem optimální se myslí taková trasa, která bude mít nejvyšší spolehlivost a zároveň nejnižší poplatky. Tento nový algoritmus se v komunitě nazývá podle autora Pickhardt Payments. Pokud by se ale **upustilo od pevného poplatku** za přesměrování platby (`base_fee`) a zůstal by pouze proporcionální poplatek (`fee_rate`), tak by výpočet ideální trasy byl mnohem jednodušší a efektivnější. Jedná se ale o celkem razantní změnu Lightningu, takže zatím tato iniciativa zůstává v teoretické rovině.

Co se týká reálných čísel, tak momentálně v roce 2022 má necelých 40 % kanálů nastaven nulový `base_fee`, čímž buď výše uvedený návrh podporují, anebo pouze nechtějí tyto poplatky vybírat.

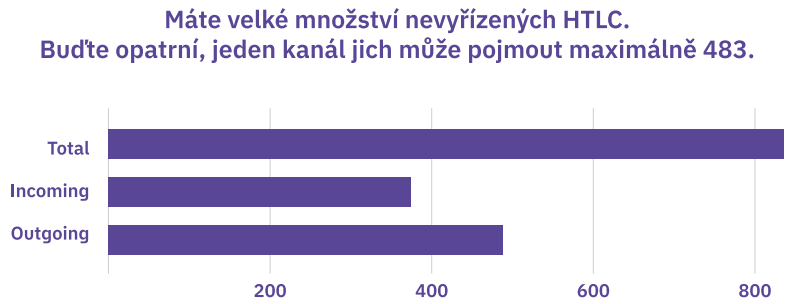
Zranitelnosti Lightning Network a útoky na ni

V poslední kapitole teoretické části se podíváme na příklady dvou zranitelností či problémů, s kterými se Lightning Network aktuálně potýká. V celkovém součtu jich existuje samozřejmě více, ale ty ostatní jsou již buď vyřešeny, anebo jejich praktická proveditelnost je extrémně náročná, takže zůstávají zatím spíše v teoretické rovině.

Griefing attack

Griefing attack není útok, při kterém by hrozila ztráta prostředků, ale jedná se spíše o DoS zranitelnost, která po určitou dobu zmrazí vaše prostředky a znemožní uzlu routovat jakékoliv transakce. Tudíž vaše kanály budou nepoužitelné a budete přicházet o potenciální zisk z routování.

V principu se jedná o zahlcení platebního kanálu (nebo i více kanálů) smyšlenými HTLC, kdy cílový uzel neodešle ihned `payment_preimage` a platba tak až do vypršení „zůstane viset“. Jak jsme si vysvětlili dříve, do HTLC se uzamknou ty prostředky, které mám v plánu přeposlat dále, a po dobu uzamčení s nimi nemohu nijak operovat. Existuje zde nám již známý limit 483 HTLC (v jednom směru) pro každý kanál, a ten pokud se vyčerpá, tak je kanál nepoužitelný v daném směru pro příjem, odesílání i přesměrování transakcí.



Motivací útočníka, který by takto vaše kanály zahltil, může být požadavek výkupného pro ukončení či nezahájení útoku, případně se může jednat o konkurenční boj mezi obchody a jejich lightningovými uzly.

V době útoku nemáte aktuálně téměř žádnou možnost obrany. Zbývá vám tedy pouze čekat na vypršení daných HTLC, případně nuceně tento kanál uzavřít. Kvůli velkému množství HTLC by se ale jednalo o velmi drahé uzavření platebního kanálu (viz kapitola o druhé úrovni transakcí) a vlastně by tím útočník vyhrál. Dříve byla

v implementaci LND chyba, vinou které došlo dokonce k automatickému uzavření takto napadeného kanálu – nyní je již opravena. Alternativou k tomuto typu útoku může být varianta, která takto vyčerpá veškerou kapacitu kanálu, i když stále budete mít „volné sloty" pro další HTLC.

Vývojář Joost Jager, který se tomuto problému dlouhodobě věnuje, vytvořil nástroj zvaný **Circuitbreaker**, který je jakýmsi firewallem pro Lightning a který by měl ztěžovat provedení podobného útoku. Jeho princip spočívá v tom, že si můžete nastavit maximální množství nepotvrzených HTLC pro každý kanál a limitovat případný spam. Na koncepčním řešení se aktuálně pracuje, ale vzhledem k tomu, že podobné útoky se reálně zatím naštěstí téměř nedějí, není tomu přiřazena od vývojářů vysoká priorita.

Na závěr bych chtěl upozornit na HODL faktury popsané dříve. Operátor uzlu totiž nemá žádnou možnost odlišit velké množství těchto typů faktur od probíhajícího útoku, takže je využívejte pouze tam, kde jsou opravdu nutné.

Channel probing

Channel probing je vlastnost Lightning Network, díky které lze částečně zjistit aktuální rozložení prostředků v platebním kanále, což by měla být soukromá informace.

Představme si rovnou příklad, kdy vlastníme uzel **A** a z nějakého důvodu chceme zjistit rozložení prostředků mezi uzly **C** a **D**. Jinak řečeno nás zajímá, kolik bitcoinů v platebním kanále o velikosti 4 miliony satoshi náleží oranžovému uzlu a kolik zelenému. Díky gossip protokolu známe celkovou kapacitu všech kanálů, ta je vyjádřena v milionech satoshi a znázorněna fialově. Zároveň víme aktuální rozložení v našem kanálu s uzlem **B**.

Pro provedení tohoto typu útoku budeme simulovat odeslání platby o **velikost 2M** satoshi uzlu **D**, aniž by nám vystavil jakoukoliv fakturu. Tato transakce bude tedy mít zcela vymyšlené `payment_hash`. Uzly **B** a **C** ale nemohou nijak zjistit, zda je `payment_hash` správný, protože nejsou finálními příjemci, a tak sestaví HTLC směrem k cíli, uzlu **D**. Během sestavování této platby mohou nastat následující tři případy:

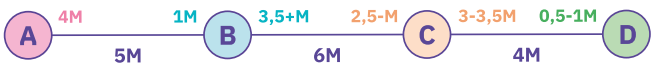
- Od uzlu **D** se nám vrátí chybová zpráva, že se jedná o neplatný `payment_hash`, ke kterému nemá odpovídající `payment_preimage`. Platba neprošla, ale v tuto chvíli

víme, že v kanálu mezi **C** a **D** je na straně **C** alespoň 2M satoshi. (*Zároveň to samé platí pro kanál B a C, ale to není náš primární cíl.*)

- Od uzlu **C** se nám vrátí zpráva `Temporary channel failure`, která značí nedostatečnou likviditu kanálu v daném směru. Nyní tedy víme, že v kanálu mezi **C** a **D** je na straně **C** méně než 2M satoshi, protože transakce neprošla až do cíle.

- Zpráva `Temporary channel failure` se nám může vrátit i od uzlu **B**, v tomto případě se o rozložení mezi uzly **C** a **D** nedozvíme nic a je vhodné najít alternativní trasu k uzlu **C**, kde nás nebude omezovat nedostatečná likvidita předchozích uzlů.

Dejme tomu, že nastal první případ, a vrátila se nám chybová zpráva o neplatném `payment_hash`. Uzel **C** tedy vlastní alespoň 2M satoshi. Nyní můžeme využít binárního půlení a odeslat transakci o velikost 3M satoshi. I tato platba skončí stejnou chybou. Víme tedy, že uzel **C** vlastní alespoň 3M satoshi v tomto platebním kanálu. Další pokus bude s hodnotou 3,5M satoshi a ten již skončí nedostatečnou likviditou uzlu **C**. Díky tomu jsme zjistili, že uzel **C** vlastní v kanálu s **D** něco mezi 3 až 3,5M satoshi, na uzel **D** tedy zbývá 0,5 až 1M satoshi. Pokud bychom chtěli přesnější údaj, lze s binárním půlením pokračovat. Výsledný stav:



Jak jsme si ukázali na tomto příkladu, pomocí generování vymyšlených transakcí lze částečně deanonymizovat rozložení prostředků v platebních kanálech. Aby měl útok nejvyšší pravděpodobnost úspěchu, je vhodné být k testovanému kanálu co nejblíže, abychom vyloučili nedostatečnou likviditu v předcházejících uzlech, ideálně s daným uzlem být napřímo spojen. Zároveň je ale nutné upozornit na to, že takovýto útok nějakou dobu trvá a že zjištěné informace jsou platné pouze v době testování. Pokud by tedy někoho napadlo tímto způsobem zjistit aktuální rozložení u většího množství kanálů, než by je všechny oskenoval, tak výsledky těch prvních by již nemusely být platné. Celé snažení útočníka mohou ztěžovat tzv. paralelní kanály, tedy stav, kdy mezi dvěma uzly je navázáno více kanálů. V tuto chvíli nemůže útočník žádným způsobem ovlivnit to, který kanál se použije, takže informace budou zkreslené.

Mnoho lidí channel probing nepovažuje za útok, ale za vlastnost lightningové sítě, díky které mohu zvyšovat pravděpodobnost úspěchu plateb tím, že budu průběžně skenovat svoje sousedy a této informace využiji v případě reálné platby. Jak se k uvedené vlastnosti postavíte vy, již nechám na vás, každopádně s aktuálním principem fungování lightningové sítě ji lze jen těžko odstranit.

PROVOZ VLASTNÍHO UZLU

V této části knihy se přesuneme od teorie k praxi a podíváme se na provozování vlastního lightningového uzlu. Lightning je nadstavba nad Bitcoinem, tudíž lightningový uzel potřebuje mít aktivní přístup do bitcoinové sítě pro vysílání transakcí otevírajících a zavírajících kanál, monitorování uzavření kanálu protistranou apod. **Uzel** se tedy v majoritě případů skládá z bitcoinového fullnodu, nad kterým běží implementace Lightningu. Je sice teoreticky možné provozovat uzel bez bitcoinového fullnodu (např. implementace LND se umí napojit na cizí instanci pomocí protokolu Neutrino), jedná se však o málo využívanou a obecně ne příliš doporučovanou variantu. V následujícím textu tedy budeme za pojem uzel považovat **kombinaci Bitcoinu a Lightningu v jedné instanci**. Předpokladem pro tuto kapitolu jsou alespoň základní znalosti administrace serverů (primárně Linux), síťových protokolů, skriptování a bezpečnosti.

Nebudu zde popisovat jedinou „univerzální a nejlepší" variantu pro provozování lightningového uzlu, protože nic takového neexistuje. Spíše se podíváme na různé možnosti, jejich výhody a nevýhody a poté bude na každém z vás, aby si vybral to řešení, které mu nejvíce vyhovuje. Zároveň nečekejte, že zde naleznete konkrétní příkazy, jak jednotlivé komponenty nainstalovat a provozovat. I když chápu, že by to bylo pro mnoho z vás přínosné, tak si musíte uvědomit, že ekosystém Lightningu se vyvíjí neuvěřitelně rychle, a tudíž jakýkoliv přesný postup instalace a konfigurace uvedený v této knize by byl po pár měsících zastaralý a částečně nepoužitelný. Kniha na toto není vhodná platforma. Místo toho si zde ukážeme obecné principy a možnosti provozování uzlu. Zároveň dostanete informace a odkazy na různé projekty, které vám s konkrétní implementací pomohou.

Proč provozovat vlastní uzel?

Proč bych vůbec měl lightningový uzel provozovat? Co tím získám? Na tuto otázku se dá odpovědět různými způsoby. Obecně toto téma lze rozdělit do dvou kategorií.

Buď jste obchodník nebo firma a rozhodnete se (alespoň částečně) integrovat platby pomocí Lightningu. To znamená, že vaším primárním důvodem je **podpora businessu**. V tomto případě vám vlastní lightningový uzel umožní přijímat platby tím nejpřirozenějším způsobem. Získáte kompletní nezávislost na jakékoliv třetí straně. Nikdo nebude mít přehled, kolik plateb váš business zpracoval, čímž zvýšíte svoje soukromí. Nikomu nebudete muset platit žádné poplatky za příjem plateb

či provoz a ani se nebudete muset spoléhat na třetí stranu, zda její platební řešení bude fungovat. Není zrovna příjemné si zavést do svého e-shopu lightningové platby skrze třetí stranu a poté číst nespokojené reakce zákazníků, že jim platba neprošla kvůli nedostatečné likviditě uzlu, kterou vy nemůžete nijak ovlivnit. Na druhou stranu musím upozornit, že instalace a údržba takovéhoto uzlu svépomocí nemusí být v začátku zrovna jednoduchá a levná záležitost. Záleží tedy na každém, jestli se vydá cestou vlastního uzlu či zda se spolehne na třetí stranu. V druhém případě bych však doporučil vybrat si kvalitního provozovatele s ověřenou historií. Drobnou výhodou této varianty je také fakt, že pokud nechcete příjmy z plateb uchovávat v bitcoinech, ale například je automaticky převádět na fiat měnu, tak tuto činnost můžete mít v rámci poskytovaného řešení.

Druhou a pravděpodobně častější variantou je provozování lightningového uzlu jako jednotlivec v rámci svého **hobby**. To znamená, že chcete získat větší soukromí, být naprosto nezávislý na jakékoliv instituci nebo si to prostě jen ze zvědavosti vyzkoušet. Co tím získáte? Mimo výše jmenované si rozhodně rozšíříte svoje znalosti Lightningu. Nejdřív s tím nejspíše budete bojovat a ne vše bude fungovat podle plánu hned od začátku. Pokud ale vytrváte, získáte hluboké povědomí o fungování jak Lightningu, tak i Bitcoinu. Nikdy nevíte, zda se Lightning (nebo nějaký jeho nástupce) nestane za několik let mainstreamem v oblasti plateb, a v tuto chvíli by vaše těžce nabyté zkušenosti měly cenu zlata. Nic není jednoduché a určitě vám zabere několik večerů a víkendů, než váš uzel bude fungovat naprosto přesně podle vašich představ. Proto to musí být v první řadě váš koníček. Věřím ale, že to minimálně za zkoušku stojí – prvotní investice pro takovéto domácí provozování uzlu není nikterak nákladná.

Zároveň vám v rámci uzlu poběží i Bitcoin fullnode, který validuje veškeré transakce a bloky, čímž činí síť bezpečnější. Nemusíte se tedy spoléhat na jakoukoliv třetí stranu, která vám může dávat nesprávné informace, a zároveň jí prozrazovat, o které adresy máte zájem (ty totiž pravděpodobně budou patřit vám).

Možnost výdělku

Jeden z nejčastějších dotazů začátečníků je, zda se dá provozováním uzlu vydělat. Když jsem s Lightningem začínal, tak mne samotného také odpověď na tuto otázku zajímala a na jakémkoliv fóru je to neustále omílané. Je to vcelku pochopitelné – těžba bitcoinu se přesunula postupem času do rukou profesionálů a specializovaných firem, takže takové to domácí těžení v dnešní době není pro každého. Jak je to ale u Lightningu? Lze si provozováním uzlu vydělat? Nebudu vás dlouho napínat a odpovím, že pokud budete uzel provozovat dobře a investujete do toho hodně času, znalostí a prostředků, tak skončíte přibližně

na nule, ale ne v mínusu. Představa rychlého zbohatnutí provozováním uzlu je pro naprostou majoritu lidí **mylná**.

> **Zisk**
>
> Zisk je definován jako **rozdíl mezi výnosy a náklady**. Pokud vynecháme osobní čas či studium a budeme brát v potaz pouze materiální náklady, tak tam lze započítat prostředky pro nákup nebo pronájem HW, elektřinu, internetové připojení a případně nějaké záložní zdroje elektřiny či sekundární internetové spojení, pokud to myslíme opravdu vážně. Dále budeme muset započítat on-chain poplatky za otevírání a zavírání kanálů, v neposlední řadě také něco málo utratíme za rebalancování kanálů (to si popíšeme později), případně nákup příchozí likvidity.
>
> Na straně výnosů máme pouze poplatky za přesměrování platby z naší strany kanálu na druhou, nic víc. I když si tyto poplatky můžeme nastavit libovolně, jedná se o vysoce konkurenční prostředí a vysoký poplatek by mohl znamenat, že si náš uzel nikdo nevybere a všechny platby budou putovat alternativní trasou, která bude levnější. Ideální je tedy nastavit poplatky tak, aby to byla maximální hodnota, kterou jsou lidé ještě ochotni za platbu zaplatit a nemají levnější alternativu.

Zpočátku budete rozhodně v mínusu, začínal tak téměř každý. Dobrou zprávou je, že pokud si odmyslíme jednorázové náklady na nákup zařízení, tak samotný provoz je vcelku levný. V závislosti na vaší aktivitě vás mohou stát on-chain poplatky spolu s rebalancingem stovky satoshi (případně až nižší jednotky tisíc) týdně, což je akceptovatelné. Horší zprávou je, že vaše výdělky budou ze začátku ještě o něco nižší.

Domácí provozování uzlu tedy musí být primárně vaše hobby. Postupem času, pokud se tomu budete aktivně věnovat, se ale můžete dostat do mírného plusu. Nečekejte však nějaké velké výdělky, a pokud je zisk vaší primární motivací, tak doporučuji porozhlédnout se jinde.

Existují sice uzly, které dokážou získat velmi slušné peníze, je jich však naprostá minorita. Tato informace není veřejná, ale co se tak proslýchá, tak opravdu zajímavé částky si dokáže vydělat ani ne procento těch nejlepších uzlů, které mají dlouhou historii, stovky platebních kanálů, součet jejich kapacity činí několik jednotek bitcoinů, jsou pravidelně spravovány, mají v komunitě nějaké jméno a jsou používány jako tzv. centrální huby, přes které se směruje obrovské množství transakcí.

My chceme čísla!

Dobře. Vím o uzlu, který je aktuálně v TOP 100 podle hodnocení Lightning Terminal Web. Jedná se o proprietární hodnocení uzlů, každopádně umístění mezi první stovkou značí, že se jedná o kvalitní uzel. Celková kapacita všech jeho 84 kanálů je necelých 12 BTC. Uzel je v provozu přibližně půl roku.

Za poslední měsíc přesměroval 3 486 plateb o celkové hodnotě 16 BTC, čímž na poplatcích získal celkem 912 tisíc satoshi. Za on-chain poplatky zaplatil 168 tisíc a za rebalancing 109 tisíc. Čistý měsíční zisk je tedy 647 tisíc satoshi.

Výše výdělku je však **velmi variabilní.** Se stejným množstvím kanálů i stejnou kapacitou můžete teoreticky vydělávat klidně až desetinásobek, v opačném případně i desetinu. Záleží totiž na tom, s kým si otevřete kanály, jak moc budou využívány, na kolik si nastavíte poplatky a v neposlední řadě na vašich schopnostech efektivně rebalancovat.

Hot-wallet risk

Ještě předtím, než se pustíte do provozování vlastního uzlu, tak si musíme uvědomit, že veškeré prostředky (bitcoiny) budou uloženy v tzv. „hot-wallet“. To znamená, že je budeme mít v peněžence, která bude neustále připojená k internetu, a hrozí zde riziko, že se je někomu podaří ukrást. Ukládat větší množství bitcoinů v hot-wallet se obecně nedoporučuje, od toho tu máme hardwarové peněženky. Při provozování uzlu se tomu ale nevyhneme (offline uzel by jaksi postrádal smysl), je proto důležité mít toto riziko neustále na paměti.

Naprostá většina implementací uzlů je open-source a zrovna ta část kódu, která zajišťuje bezpečnost při komunikaci po internetu s ostatními uzly, bude pravděpodobně ta nejvíce auditovaná. Aktuální hodnota všech bitcoinů uzamčených v platebních kanálech Lightningu dosahuje 3 miliard korun – útočníci mají tedy silnou motivaci nějakou chybu najít. Doposud se ale nic vážného, co by ohrozilo celou síť, nenalezlo. To je sice dobrá zpráva, nikdy však nemůžeme mít 100% jistotu, že v komunikačním protokolu nebo v samotné implementaci nějaká chyba, která by šla zneužít, není nebo se v budoucnu neobjeví.

Nemusí se vždy jednat jen o zranitelnost v implementaci či protokolu, mnohem častější je chyba v samotné konfiguraci. Pro příklad si vezměme nezkušeného administrátora, který vystaví do internetu nejen port pro P2P komunikaci s ostatními uzly, ale také port pro RPC volání (zjednodušeně řečeno se jedná o rozhraní pro správu uzlu). Pro samotný přístup k tomuto rozhraní je sice nutná autorizace ve formě **macaroon souborů** (něco jako webový cookie soubor, který ale umožňuje navíc definovat různá práva), ale protože si chce náš administrátor ulehčit práci,

otevřel si do internetu i webový server, kde má různé své statistiky ohledně uzlu. IT bezpečnost ale není zrovna jeho silný obor, a tak kvůli chybné konfiguraci webového serveru se lze pohybovat po filesystému serveru a přečíst si obsah administrátorského macaroon souboru, s kterým máte neomezený přístup k RPC rozhraní. Využití kombinace těchto dvou zranitelností je poté velice triviální. Podobných příkladů mohou být tisíce – toto je pouze jeden z nich, kterým jsem chtěl demonstrovat možná rizika spojená s hot-wallet peněženkou.

Technické možnosti

Pokud jste se rozhodli, že se vydáte cestou provozování vlastního bitcoinového a lightningového uzlu, je nutné si předem promyslet, jakým způsobem toho chcete docílit. Neexistuje totiž jedno univerzální řešení vhodné pro všechny, ale závisí to na tom, kolik peněz jste ochotni do provozu investovat, za jakým cílem chcete uzel provozovat, jak hluboké jsou vaše technické znalosti a jak moc si chcete chránit své soukromí. V následujících kapitolách se podíváme na tuto problematiku více podrobně.

Fyzické umístění uzlu a architektura

Dle mého názoru existují primárně dvě možnosti – buď budete uzel provozovat doma, anebo si pronajmete server u třetí strany, nejčastěji v cloudu.

Domácí provozování má tu výhodu, že je finančně velmi nenáročné. Většinou vám postačí minipočítač typu Raspberry Pi (případně různé alternativy) s procesorovou architekturou ARM, který lze s potřebným příslušenstvím pořídit za jednotky tisíc korun. Výhodou je také jeho velmi nízká spotřeba elektrické energie a možnost uchlazení bez použití aktivních větráků, takže o něm prakticky nebudete vědět. Na druhou stranu musíte počítat s tím, že v domácích podmínkách mohou čas od času nastat výpadky elektřiny či internetu. Ty lze v případě potřeby řešit záložními zdroji. Také se zde může stát, že budete omezeni výkonem daného zařízení, kde je prakticky nemožné upgradovat jednotlivé komponenty za výkonnější. Alternativou může být klasický server provozovaný doma, pokud vám to podmínky umožňují. Vzhledem k tomu, že v tomto případě máte kompletně vše pod svojí správou, je toto řešení vhodné pro **domácí použití jako hobby**.

Naopak **firemní zákazníci** častěji sáhnou po cloudovém řešení na procesorové platformě x86. Jedná se sice o mnohem dražší variantu, na druhou stranu však máte vyřešeno naprosto vše od chlazení, napájení, zálohování až po pevnou IP adresu a garantovanou dostupnost. Velkou výhodou je také možnost navýšení prostředků „za běhu“, pokud by ty současné nedostačovaly. Je zde však jeden menší problém – uzel potřebuje mít pro některé operace přístup k vašim privátním klíčům. Ty jsou poté uloženy v paměti samotného serveru, ke kterému mohou mít čistě teoreticky přístup i další osoby zajišťující chod cloudu. Při volbě kvalitního a ověřeného provozovatele je toto riziko velmi malé, přesto jsem na něj však chtěl upozornit. Určitým kompromisem může být housing vašeho vlastního serveru.

Určitě budou existovat i další možnosti a kombinace, tyto dvě jsou ale nejčastější. Ať se rozhodnete pro jakoukoliv z nich, tak berte na vědomí, že přechod z jedné na druhou nemusí být vždy jednoduchý. Například pokud se rozhodnete začít doma na minipočítači a postupem času bude váš uzel tak úspěšný, že výkon nebude dostačovat, tak migrace na úplně jinou procesorovou platformu nemusí být vždy bezproblémová.

Výběr implementace Lightningu

Aktuálně existuje několik konkurenčních implementací samotného Lightningu. Všechny by měly dodržovat standardy definované v BOLT specifikaci, a tudíž spolu vzájemně kooperovat. V této kapitole se podíváme na tři nejpopulárnější implementace více do hloubky.

LND (Lightning Network Daemon) od společnosti Lightning Labs je jednou z nejznámějších a nejpopulárnějších implementací Lightningu. LND je napsáno v programovacím jazyce Go pod MIT licencí a lze ho provozovat na Linuxu, Windows i MacOS. Ke stažení je k dispozici v rámci Docker image jako předpřipravené binární soubory nebo si jej lze zkompilovat ze zdrojového kódu. Jako backend nejčastěji slouží Bitcoin Core, ale umí komunikovat i s implementací btcd a nově také přes experimentální light-weight protokol Neutrino. Ten najde využití primárně pro mobilní implementace – LND nabízí SDK pro Android i iOS. Hlavní výhodou této implementace je velmi široké a dobře zdokumentované API rozhraní, které je k dispozici ve formě gRPC anebo HTTP REST. Díky tomu je nad LND vytvořeno **velké množství nadstaveb**, které vám umožňují spravovat uzel z webového rozhraní, automaticky rebalancovat kanály, nastavovat poplatky podle aktuálního rozložení likvidity apod. Tato implementace je také nejčastěji používána v rámci hotových lightningových platforem, které si probereme v následující kapitole. Společnost Lightning Labs dále stojí za projekty Pool a Loop, které umožňují provádět převody z on-chain na Lightning (a zpět)

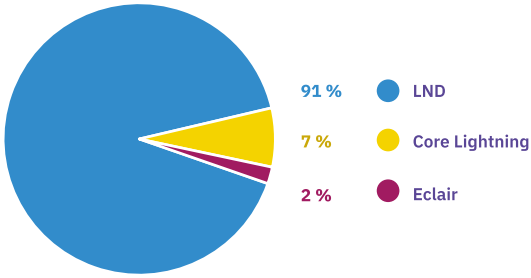
a prodávat či nakupovat likviditu. Do detailu si tyto koncepty probereme později. Jedná se tedy o jednoduchou a uživatelsky velmi přívětivou implementaci, která je vhodná pro začátečníky. Pokud jste v minulosti provozovali velmi vytížený uzel s několika stovkami kanálů, nemusela být tato implementace kvůli extrémně rychle rostoucí databázi úplně vhodná (databáze v některých případech dosahovala řádů desítek až stovek gigabajtů). Tento problém je však již od verze 0.15 vyřešen a aktuálně tímto neduhem LND již netrpí.

Druhou implementací, kterou si zde probereme, je **Core Lightning** od společnosti Blockstream. Donedávna, než došlo k přejmenování, byla známá pod názvem **c-lightning**. Tato implementace je napsaná v programovacím jazyce C a je k dispozici pod licencí BSD-MIT. Mezi podporovanými platformami najdeme pouze Linux a stejně jako LND je k dispozici ve formě Docker image, binárek nebo zdrojového kódu. Oproti jiným implementacím nabízí low-level přístup pro individuální úpravy. To znamená, že si můžete vyměnit určité subsystémy, jako je například databáze, samotná peněženka, systém pro komunikaci v rámci gossip protokolu apod. Lze si tedy například vyměnit subsystém pro držení privátních klíčů, aby podporoval HSM (*Hardware Security Module – fyzické zařízení pro správu klíčů a jednotlivé kryptografické operace*). Jako API podporuje JSON-RPC 2.0 přes Unix Domain soket. Hlavní výhodou této implementace je **modulární architektura s využitím pluginů**. Samotný démon se stará pouze o hlavní části Lightningu a jakékoliv další rozšíření je řešeno jako samostatný plugin, což není nic jiného než další proces, který s hlavním démonem komunikuje. Příkladem jsou pluginy pro watchtower (outsourcing hlídání publikování revokované transakce na třetí stranu), rebalance (automatické vyvážení likvidity v kanálech), autopilot (automatické otevírání kanálů) nebo třeba plugin pro zálohování. Ve výchozím stavu používá Core Lightning databázi SQLite3 a jeho výhodou oproti LND je pokročilejší zálohování, kdy při obnově není nutné uzavřít veškeré kanály. Core Lightning také často implementuje různé experimentální funkce, které ještě nejsou specifikované v rámci BOLT. Příkladem může být již dříve popsaný BOLT#12. Díky podpoře PostgreSQL databázového clusteru je vhodný pro firemní nasazení. Částečnou nevýhodou oproti LND může být menší množství komunitních nadstaveb a nástrojů.

Poslední implementací, kterou si představíme, je **Eclair** (francouzské slovo pro blesk) od společnosti ACINQ, která stojí mimo jiné za populární peněženkou Phoenix. Implementace je napsána v jazyce Scala pod licencí Apache 2.0. Eclair lze provozovat na Linuxu, Windows i MacOS. Pro vývoj dalších nástrojů publikuje JSON API. ACINQ se silně

zaměřuje na mobilní platformy, kde má nativní verzi lightningového uzlu pro Android i iOS – jedná se o lightweight implementaci, která nemá zdaleka všechny funkce, jako je např. přesměrování plateb, a počítá se u ní, že nebude neustále online. Díky tomu je ale dobře optimalizovaná pro mobilní zařízení. Eclair Mobile byla první non-custodial lightningová peněženka na mobilní platformy. Dnes ji nahradila již výše zmiňovaná peněženka Phoenix. Co se týká samotné implementace Eclair, tak ta se zaměřuje spíše na **firemní klientelu**, tedy na uzly o **velmi vysokém počtu transakcí**. Na této implementaci běží aktuálně největší a nejznámější uzel ACINQ, který má nyní přes 3 000 kanálů o celkové kapacitě 324 BTC. A to zde počítáme pouze veřejné kanály. Peněženka Phoenix navazuje s tímto uzlem kanály privátní, takže celkový počet může být řádově jinde. Pokud by vás zajímalo, jak je možné takto velký uzel provozovat, tak je to díky architektuře Eclair. Ta je silně paralelizovaná pomocí výpočtů ve více vláknech a procesech (například každý kanál běží odděleně ve svém vlastním sandboxu). Dále tato implementace umožňuje instalaci přes více serverů, kdy můžeme mít několik front-end serverů (před kterými je loadbalancer), které mezi sebe rozkládají CPU náročné výpočty a za nimi je back-end, jenž se stará o samotnou správu kanálů. Oproti předchozím implementacím je zde naprosté minimum jakýchkoliv rozšíření.

Zjistit procentuální zastoupení jednotlivých implementací není vůbec lehký úkol. Jelikož všechny dodržují BOLT specifikaci, je těžké přesně definovat, která implementace je kde použita. Částečně to lze vyčíst z výchozích hodnot v rámci gossip protokolu, jelikož většina uživatelů je nechává na výchozích hodnotách. Následující výsledky (z roku 2021) berte proto s velkou rezervou – jedná se spíše o kvalifikovaný odhad.



Existují samozřejmě i další implementace, ty však zatím nejsou tolik rozšířené. Záleží tedy na každém z vás, co bude nejvíce vyhovovat vašemu případu užití. V následujících kapitolách se pokusím popisovat vybrané nástroje co nejvíce nezávisle na jednotlivých implementacích. Každopádně vzhledem k velkému procentuálnímu zastoupení LND, vhodnosti pro začátečníky a faktu, že na něj existuje nejvíce

komunitních nadstaveb a nástrojů, budou určité kapitoly zaměřené spíše na tuto konkrétní implementaci. Věřím, že to takto bude přínosné pro majoritu čtenářů. Tím ale rozhodně netvrdím, že LND je nejlepší. Pokud to s Lightningem myslíte opravdu vážně, tak bych doporučoval implementací vyzkoušet více, a nakonec vybrat tu, která vám bude nejlépe vyhovovat.

Platformy

Samotné komponenty provozované na lightningovém uzlu se dají rozdělit do následujících tří kategorií:

- **Bitcoin fullnode**. Vzhledem k tomu, že Lightning potřebuje přístup do bitcoinové sítě pro vysílání transakcí, které otevírají nebo zavírají kanál, musíme na svém uzlu provozovat bitcoinový fullnode. Ten se zároveň využívá k monitorování, zda protistrana neodeslala již revokovaný stav kanálu. Nejčastěji to tedy znamená instalaci a konfiguraci známého Bitcoin Core.

- **Implementace Lightningu**. Podle vlastního výběru zvolíme LND, Core Lightning nebo Eclair. Tato komponenta se stará o šíření informací skrze gossip protokol, otevírání/zavírání kanálů, placení faktur, routování plateb apod. Zároveň komunikuje s bitcoinovým démonem kvůli přístupu do blockchainu.

- **Nadstavby a další nástroje**. Tyto komponenty nejsou povinné, ale zásadním způsobem zjednodušují správu samotného uzlu. Bez nich bychom totiž museli interagovat s bitcoinovým fullnodem i samotným Lightningem pouze pomocí příkazové řádky, což pro určité komplikovanější operace není ideální. Do této kategorie tedy spadají různé bitcoinové self-hosted blockchain explorery a také webové aplikace, které v grafické formě usnadňují správu uzlu. Dále sem patří komunitou vytvořené skripty, které se starají o rebalancing kanálů, automatické nastavování poplatků apod. Mnoho z těchto nástrojů si popíšeme v pozdějších kapitolách.

Jednou z pokročilejších variant je sestavení takovéhoto uzlu kompletně od základů svépomocí. Uživatel nejčastěji zvolí oblíbenou linuxovou distribuci, kde musí nejprve provést konfiguraci operačního systému a promyslet si hardening (zabezpečení). Následuje instalace a konfigurace komponent vypsaných výše, poté jejich vzájemné propojení. V rámci provozu je také potřeba udržovat veškeré komponenty aktuální. I když je tato varianta dle mého názoru nejlepší, protože máte naprostou kontrolu nad tím, jak je váš uzel sestaven a jak funguje, tak musím upozornit, že je vhodná pouze pro pokročilé uživatele. Jelikož na samotném uzlu budete mít určitý obnos

bitcoinů, jakákoliv chyba v konfiguraci může mít fatální důsledek. Nemusí se jednat pouze o ztrátu prostředků, ale také o možnost dysfunkce uzlu, pokud se například nepovede update některé komponenty a nebudete mít dobře nastavené zálohování.

Pokud si na stavbu vlastního uzlu zatím netroufáte, existují tzv. „předpřipravené platformy". Nejčastěji se jedná o speciálně upravené linuxové distribuce, které stačí pouze nahrát na minipočítač Raspberry Pi a zapnout ho. Obsahují již nakonfigurovaný Bitcoin fullnode, samotnou implementaci Lightningu a určité množství nadstaveb. Často je k dispozici také přívětivé grafické rozhraní, ze kterého lze celý uzel jednoduše ovládat. Jedná se o zajímavou variantu pro začínající uživatele, jelikož vše je již nainstalováno a nakonfigurováno. Většinou postačuje počkat několik hodin až dní pro synchronizaci blockchainu a poté lze s uzlem pracovat z velké části bez nutnosti práce s příkazovou řádkou. Pokud jste tedy začátečník anebo nemáte dostatečné znalosti administrace, může být jedna z těchto platforem řešením. Nyní si stručně projdeme ty nejběžnější:

- **Umbrel**. Jedná se o jednu z nejznámějších platforem pro domácí provoz lightningového uzlu. Instalace probíhá nakopírováním speciálního Umbrel OS na SD kartu, kterou vložíte do svého Raspberry Pi. K němu budete potřebovat ještě externí disk (ideálně SSD). Experimentálně lze Umbrel provozovat i na klasickém počítači. Na pozadí běží Bitcoin Core a LND v kontejnerech. Nově byla přidána podpora i pro Core Lightning. Umbrel je známý pro své graficky přívětivé webové rozhraní, z kterého lze uzel spravovat. Součástí něj je i vlastní AppStore, kde lze na několik kliknutí nainstalovat obrovské množství nadstaveb. Právě množství aplikací a jednoduchost provozu jsou hlavními přednostmi Umbrelu.

- **Citadel**. Jedná se o fork Umbrelu od vývojáře Aarona Dewese, který nebyl spokojený s tím, jaké bylo další plánované směřování vývoje Umbrelu, a tak si vytvořil svoje řešení. To je aktuálně s Umbrelem velmi podobné (dokonce existuje návod na migraci z Umbrelu na Citadel). Dle slov vývojářů se chtějí primárně zaměřit na komunitu a vyvíjet pod skutečnou free a open-source licencí, na rozdíl od Umbrelu. Dále také slibují mnohem častější aktualizace aplikací a možnost výběru Bitcoin fullnodu i implementace Lightningu. Tento projekt je však stále v počátku, takže o jeho úspěchu či neúspěchu rozhodne až čas.

- **RaspiBlitz**. Další velmi oblíbená platforma pro domácí provozování uzlu je RaspiBlitz. Jak již název napovídá, je také určena pro Raspberry Pi, kde stejně jako u Umbrelu pouze nakopírujete předpřipravený operační systém na SD kartu, vložíte do minipočítače a zapnete. Po spuštění a připojení pomocí SSH na vás čeká grafický průvodce, který vás provede prvotní konfigurací. Mimo jiné zde máte na výběr obnovu ze zálohy, případně zda chcete využívat LND nebo Core

Lightning. V druhém případě musíte ale počítat, že ne všechny nadstavby budou dostupné, jelikož mnoho z nich funguje pouze s LND. Po dokončení konfigurace se celý uzel ovládá přes grafické menu, ve kterém si můžete upravovat konfiguraci jednotlivých komponent, prohlížet logy, provádět zálohy a updatovat zařízení. Je zde také možnost z tohoto menu jednoduše instalovat další nadstavby, RaspiBlitz jich nabízí stejně jako Umbrel velké množství. Dále je zde navíc built-in podpora pro 3,5″ LCD displej, na kterém se vám zobrazují nejdůležitější informace o vašem uzlu. Oproti Umbrelu zde má člověk trochu více volnosti a možnosti úprav konfigurace.

- **myNode**. Svým základním konceptem je myNode velmi podobný Umbrelu. Opět se jedná o samostatný operační systém, který se nahraje na SD kartu a spustí v rámci Raspberry Pi (mimo jiné je zde i podpora pro Rock64, RockPro64 nebo klasický x86 virtuální stroj). Jako Lightning démon je zde na výběr pouze LND a správa uzlu se provádí primárně přes webové rozhraní. Tímto jsme si popsali myNode Community Edition, což je opensource verze, kterou si lze zdarma stáhnout. Dále je k dispozici placené myNode Premium, které po zadání licenčního klíče přidá do vašeho myNode další funkce jako například jednoduché upgrady z webového rozhraní, podporu výrobce a aplikace či funkce navíc. Zájemce si může objednat i myNode One, což není nic jiného než Raspberry Pi s 1TB SSD diskem, na kterém je již nainstalovaný myNode ve verzi Premium.

- **Embassy**. Projekt Embassy od společnosti Start9 je opět podobný výše uvedeným řešením. Případný zájemce má možnost si koupit samostatné Embassy, což je již sestavené a nainstalované Raspberry Pi, anebo si zakoupit samostatný operační systém EmbassyOS, pokud již například máte potřebný hardware. Alternativně je zde možnost si Embassy zkompilovat ze zdrojového kódu, v tom případě by to mělo být zdarma. Na výběr máte LND nebo Core Lightning a celé řešení je spravováno přes webové rozhraní. Součástí je i samostatný Marketplace s aplikacemi a nadstavbami, který je aktuálně zatím ale chudší než u konkurence.

- **Nodl.** Tento projekt prodává hotová hardwarová řešení, která jsou postavená na RockPi4. Pokud si zakoupíte Nodl One, dostanete samotné zařízení, které je plně šifrované a obsahuje několik základních aplikací pro provozování lightningového uzlu. Využívá se zde implementace LND. Druhou možností je zakoupení Nodl Dojo, které má o něco lepší hardware a nabízí prémiovou podporu pro služby Dojo a Whirpool do společnosti Samourai, které je partnerem. Můžete si tedy k tomuto uzlu připojit svoji vlastní Samourai peněženku a následně mixovat bitcoiny. Aktuálně je v nabídce i Nodl Cloud, kdy toto zařízení nemusíte fyzicky vlastnit, ale pouze si jej pronajmete.

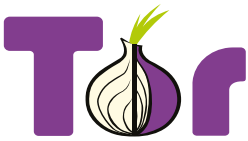
- **Raspibolt**. Nejlepší nakonec? Posledním (a mým oblíbeným) řešením je projekt Raspibolt, který neprodává žádná hotová řešení ani předpřipravené operační systémy, kde je již vše nakonfigurováno. Raspibolt je spíše takový kvalitní návod. Tento projekt si dává za cíl dobře nakonfigurovat samotné Raspberry Pi a operační systém Raspbian. Po nutném hardeningu si sami nainstalujete a nakonfigurujete Bitcoin Core, electrum server, LND a velké množství aplikací, rozšíření a nadstaveb. Jedná se tedy o návod, který vás krok po kroku vede zprovozněním vašeho vlastního uzlu kompletně od čistého operačního systému. Naučíte se, jak si dané komponenty stáhnout, ověřit podpisy a kontrolní součty, nainstalovat, nakonfigurovat a následně propojit mezi sebou. Začátečníci mohou pouze mechanicky kopírovat příkazy jeden po druhém, přičemž se zajisté hodně naučí. Pokročilí uživatelé mohou tento návod využít jako inspiraci a některé věci si udělat trochu jinak podle svého. Součástí není žádné webové rozhraní (mimo samotných aplikací), přes které by se celý uzel následně ovládal. Na druhou stranu budete mít velmi slušnou představu, jak vše funguje dohromady. U každé komponenty je i uvedeno, jak ji aktualizovat na novější verzi – toto je tedy poté již vaše zodpovědnost. Nějaké magické tlačítko „Update everything" ve webovém rozhraní zde nenajdete.

Další platformy zajisté budou vznikat, některé výše uvedené se budou měnit a případně zanikat. Berte tedy tuto kapitolu pouze jako přehled možností provozování lightningového uzlu. Dále je již na každém z vás, kterou variantu si vyberete. Osobně bych doporučil si vždy najít informace o daném řešení a podívat se třeba na aktuálnost jednotlivých aplikací. Zprovoznit si totiž uzel na platformě, kde poslední update vyšel před rokem, není zrovna ideální. Ze začátku bych možná doporučil se vyhnout placeným řešením, jelikož jsem osobně nenašel mnoho výhod oproti těm zdarma. Pokud se již rozhodnete si nějaké hotové zařízení pořídit, tak si ověřte, zda daná společnost posílá výrobky také do Evropy. Občas je kvůli dražšímu poštovnému v nabídce pouze USA, pak je to tedy nutné řešit jinak.

Na závěr jedno **upozornění k automatickým aktualizacím**. Je sice uživatelsky velmi přívětivé si ve webovém rozhraní kliknout na tlačítko *update*, chvíli počkat a mít aktuální verze všech komponent. Když se ale zamyslíte nad tím, co se děje na pozadí, tak už to tak skvěle vypadat nemusí. Software se nejčastěji připojí k aktualizačním serverům, stáhne si potřebné soubory a započne instalace a změny konfigurace. To vše na pozadí bez jakéhokoliv zásahu uživatele. Aby to bylo technicky možné, tak takovýto nástroj musí mít **nejvyšší oprávnění** na vašem uzlu, jelikož instaluje nové služby, upravuje konfiguraci, nastavuje pravidla do firewallu apod. Tím pádem mu nic nebrání v tom (čistě teoreticky) sebrat všechny vaše privátní klíče nebo například odeslat veškeré prostředky na cizí adresu. Samozřejmě, že by tím byla tato platforma odsouzena k zániku, vyloučit to ale nelze. Reálnější hrozbou je

třeba také útok na aktualizační server dané platformy, kdy útočník nahradí soubory aktualizace vlastním kódem. Tímto způsobem může poté ovládnout naráz všechna zařízení, která se automaticky aktualizují. Každá platforma si řeší aktualizace po svém a někde má uživatel alespoň minimální šanci se bránit. Berte ale prosím toto na vědomí, a pokud víte, že třetí osoba má skrze aktualizace plný přístup k vašemu uzlu, nedržte na něm velké množství prostředků. Raději si sestavte řešení vlastní, stahujte balíky z ověřených zdrojů, kontrolujte podpisy softwaru a případně monitorujte komunitní fóra.

Tor vs. clearnet



Další věcí, kterou si musíte rozmyslet, je to, zda a do jaké míry budete využívat Tor (pokud máte v rámci vaší implementace na výběr). Tento nástroj **zvyšuje anonymitu** provozovatele uzlu tím, že skrývá jeho IP adresu, a pro připojení ostatních uzlů se využívá Tor Hidden Service. Nevýhodou je naopak **pomalejší komunikace** a v ojedinělých situacích i případné problémy se stabilitou spojení. Je to dáno tím, že do komunikace s ostatními uzly vstupuje další vrstva (Tor), která pakety šifruje, a jedná se o případný Point-of-failure.

Rozhodnutí, zda využívat Tor, závisí na osobních preferencích. Pokud vám jde o maximální anonymitu, kdy nechcete do světa publikovat svoji IP adresu, rozhodně se vyplatí Tor používat. Naopak, jestli stavíte uzel pro nějaký obchod, firmu či pro svoji mobilní aplikaci, může mít pro vás větší prioritu rychlost a spolehlivost transakcí. Rozhodně to však neznamená, že by platba přes Tor byla nějak extrémně pomalá. Spíše se v ojedinělých případech (kdy se například routuje HTLC přes 5 uzlů, první čtyři cesty selžou a musí se využívat až ta pátá) můžete setkat s delším zpracováním platby.

Pokud se rozhodnete uzel provozovat přes clearnet (tedy bez Toru pomocí klasických IPv4/IPv6 adres), kromě sdílení své IP adresy tím nepřímo zvyšujete požadavky na bezpečnost daného řešení. Útočník si vaši IP adresu najde v kterémkoliv vyhledávači lightningových uzlů a dostane se mu informace, že zde běží Lightning démon a s největší pravděpodobností i Bitcoin fullnode. Dále může proskenovat otevřené porty a zjistit, že je zde do internetu publikované SSH a 2 webové služby pro správu uzlu. Administrátor se totiž rozhodl, že chce ke svému uzlu přistupovat vzdáleně, a na bezpečnost úplně nedbal. Bohatě pak stačí jakákoliv zranitelnost či chyba v konfiguraci libovolné služby publikované do internetu a útočník může získat plnohodnotný přístup k uzlu. K samotnému útoku bude motivovaný, jelikož si může **přibližně** zjistit, kolik bitcoinu je na daném uzlu:

Odhad množství bitcoinu na lightningovém uzlu

Součet kapacity všech kanálů je veřejně známá informace. Nikdy ale nevíme, kolik % kapacity je na jaké straně kanálu – můžeme to odhadnout na ½, protože pro routování plateb je výhodné mít stejnou příchozí i odchozí likviditu. Dále na samotném uzlu mohou být další bitcoiny v on-chain peněžence, které jsou připravené na budoucí otevírání nových kanálů. Pokud si tedy útočník takto odhadne množství a zjistí, že se jedná o desítky či stovky bitcoinů, bude velmi motivovaný najít případnou zranitelnost a uzel ovládnout.

Tímto chci jen říct, že provozování uzlu přes IP protokol zvyšuje požadavky na správnou konfiguraci z hlediska bezpečnosti. Při využívání Tor máte pro každou službu (pokud se tak rozhodnete) publikováno samostatné `.onion` hostname. I tak bych ale zvážil, které všechny služby chcete tímto způsobem publikovat. Tor totiž bezpečnost přímo nezvyšuje, spíše dané služby skrývá.

Pokud se podíváme na aktuální statistiku v době psaní této knihy (rok 2022), tak nyní 43 % uzlů svoji IP adresu zveřejňuje.

Jaké máme tedy možnosti ohledně Toru, clearnetu a anonymity obecně?

- **Tor-only**. Zde se počítá, že lightningový uzel umožňuje připojení (publikuje) pouze přes Tor Hidden Service adresu. Při této variantě se zachovává maximální soukromí. Na druhou stranu veškeré spojení jde skrze Tor, což přináší menší nevýhody popsané výše (rychlost, stabilita). Tento uzel může navázat spojení s ostatními skrze Tor i přes IP protokol, připojit se k němu mohou však pouze uzly, které komunikují v síti Tor. Toto řešení tedy mírně zmenšuje množinu uzlů, které s vámi mohou iniciovat otevření kanálu.

- **Hybrid mode**. Jedná se o mix, kdy uzel publikuje jak Tor hostname, tak IP adresu. Navázat s ním spojení lze přes oba protokoly. Komunikace s Tor uzly je routovaná přes Tor, naopak spojení na clearnet uzly jde vždy napřímo. Zde je pokryta maximální kompatibilita za cenu prozrazení IP adresy uzlu.

- **IP-only**. V této variantě kompletně bez využití Toru je uzel dostupný pouze přes IP protokol. Dle mého názoru se jedná o nejméně výhodnou variantu, jelikož nemůžeme navázat spojení s žádným uzlem, který je za Torem – těch je aktuálně většina.

- **VPN tunel**. Jedná se o možnost, jak využívat rychlejší komunikaci přes clearnet, ale zároveň nepublikovat do internetu adresu svého uzlu. Budete potřebovat VPN server (ať již vlastní či hostovaný jako službu), na kterém budete poslouchat na portu pro Lightning Network komunikaci (standardně TCP/9735). Veškerá

komunikace, která dorazí na tento port, bude poté přesměrována do VPN tunelu a odeslána na váš uzel. Do světa je tedy publikována pouze IP adresa VPN serveru, nikoliv uzlu jako takového.

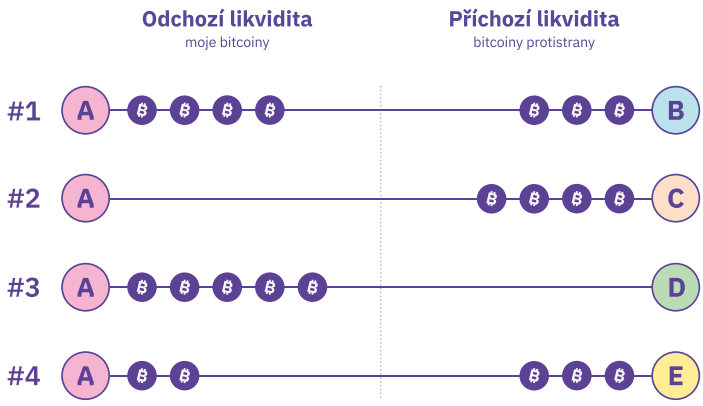
- **Tor tunel**. Toto řešení se velmi podobá předchozímu, ale využívá se zde Tor, nikoliv VPN. Pro konfiguraci budete potřebovat server (například anonymní VPS zaplacené pomocí bitcoinu), na kterém si nastavíte, aby poslouchal na portu TCP/9735 a veškeré přijaté pakety odesílal pomocí Tor kanálu na váš uzel. Ten poté poslouchá pouze přes Tor. Tímto způsobem bude opět uzel dostupný pro ostatní i přes clearnet a jediná IP adresa, která bude zveřejněna, je adresa VPS.

Vaše finální rozhodnutí tedy závisí na tom, do jaké míry preferujete rychlost a spolehlivost na úkor anonymity. Zároveň je nutné si promyslet, které služby budu s mým uzlem integrovat (mobilní peněženky, e-shop, platební terminály apod.), a zda je v nich podpora pro Tor.

Likvidita a poplatky

Než se pustíme do dalších kapitol, zopakujeme si problematiku likvidity a poplatků trochu více z praktické stránky. Základním pojmem je **kapacita kanálu**. Ta vyjadřuje množství uzamčených bitcoinů v konkrétním kanálu a je určena tou stranou, která kanál otevřela. Aktuálně není možné kapacitu v průběhu kanálu zvýšit nebo snížit – je tedy nutné si dopředu promyslet, jak velký kanál bude pro náš účel ideální.

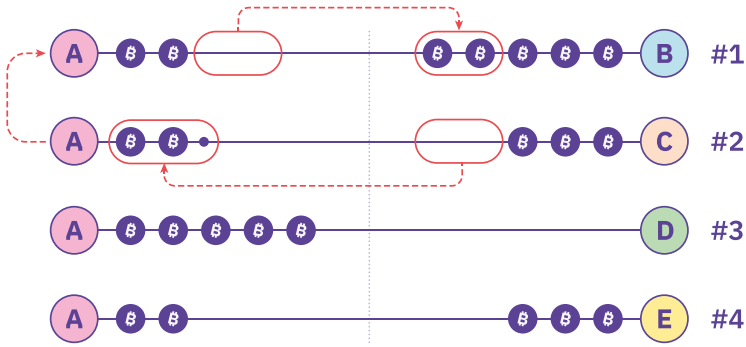
Kapacitu dále dělíme na **odchozí likviditu** (prostředky na naší straně kanálu, které můžeme odeslat) a **příchozí likviditu** (prostředky, které vlastní protistrana a můžeme je přijmout). Veškeré naše bitcoiny jsou poté součtem odchozí likvidity přes všechny kanály.



Pokud v nějakém kanálu nemáme žádné prostředky na naší straně (jako například kanál č. 2), nemůžeme ho použít pro odesílání. Je poté jedno, zda bychom tímto kanálem chtěli něco zaplatit, anebo by ho někdo jiný chtěl využít pro přesměrování platby. Transakce v tomto případě neprojde. To samé platí pro příjem a příchozí likviditu viz kanál č. 3.

Pokud přesměrováváme platbu, tak se nám pouze v jednom kanálu přesunou prostředky z naší strany na druhou a v druhém kanálu naopak. Ve výsledku tedy máme stejně prostředků, pouze jsou „jinak poskládané". Když máme nenulové poplatky za přesměrování platby, tak bychom měli mít na konci prostředků dokonce více. Ty se nám uloží do kanálu, který platbu přijal, jelikož požadujeme v rámci HTLC přijmout více, než následně odešleme. Poplatky si můžeme nastavit pro každý kanál zvlášť a nejčastěji se vyjadřují v ppm (parts per milion – kolik satoshi si ponecháme za přesměrování 1 milionu satoshi).

Přesměrování platby z kanálu #2 do kanálu #1

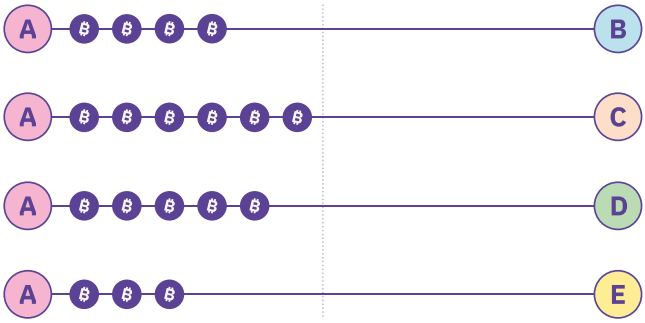


Dejme tomu, že bychom měli nastaveny poplatky 100 ppm pro kanál č. 1 a 180 ppm pro kanál č. 2. V tomto případě bychom tedy získali 0,01 % (to odpovídá 100 ppm) z celkové přeposílané částky. Poplatky se totiž počítají vždy v rámci toho kanálu, kterým odesíláme. Jinak řečeno – zbavujeme se odchozí (vlastní) likvidity, a tak si za to necháme zaplatit. Poplatky pro druhý kanál (180 ppm) jsou v tomto případě irelevantní.

Problematika příchozí likvidity

Jeden z nejčastějších problémů, který řeší téměř každý začínající operátor, je zisk příchozí likvidity. Pokud totiž máte nový uzel a otevřete například 4 kanály, veškeré prostředky budou na vaší straně, jelikož jste to vy, kdo je tam vložil. V tomto

případě takovým uzlem nemůžete žádné prostředky přijmout ani přesměrovat, pouze odeslat.



Jak to vyřešit? Existuje několik možností:

- **Kupte si něco** a zaplaťte pomocí Lightningu z vašeho uzlu. Tím pádem dojde k přesunutí prostředků v hodnotě ceny produktu na druhou stranu. Každopádně vybalancovat si takto všechny kanály je vcelku nákladné.

- **Požádejte kamaráda**, aby otevřel kanál směrem k vašemu uzlu. Pokud žádné takové kamarády nemáte, můžete zkusit veřejná diskusní fóra nebo sociální sítě. Bohužel je tam ale podobných požadavků mnoho, a tak šance na úspěch je relativně malá.

- Navázání kanálu směrem k vašemu uzlu si lze i **zakoupit**. Existuje několik služeb (do detailu si je popíšeme později), kde operátoři dobře propojených a úspěšných uzlů nabízí za určitý poplatek otevření kanálu směrem k vám.

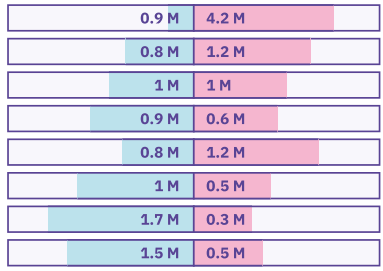
- Můžete se účastnit tzv. **rings of fire**. Ty spočívají v tom, že se několik uzlů domluví, a navzájem „do kruhu“ mezi sebou otevřou kanály. Každý z nich tedy získá jeden příchozí kanál a jeden odchozí. Jedno z řešení (Lightning Network Plus) si popíšeme později.

- Další možností je využít tzv. **submarine swaps**. Jedná se o služby (například LOOP, kterou si představíme v následujících kapitolách), které vám umožňují provádět převod z Lightningu na on-chain (a naopak). Tím pádem tedy můžete odeslat celou kapacitu kanálu na uzel služby (čímž získáte příchozí likviditu) a následně se vám na předem specifikovanou on-chain adresu tyto prostředky vrátí. Samozřejmě poníženy o poplatek, který si daná služba za tento swap účtuje.

- Pokud budete **provozovat úspěšný uzel** a dostanete se v různých žebříčcích na lepší místa, ostatní operátoři začnou k vašemu uzlu navazovat kanály sami od sebe, jelikož pro ně bude výhodné se spojit s dobře propojeným provozovatelem.

Rebalancing

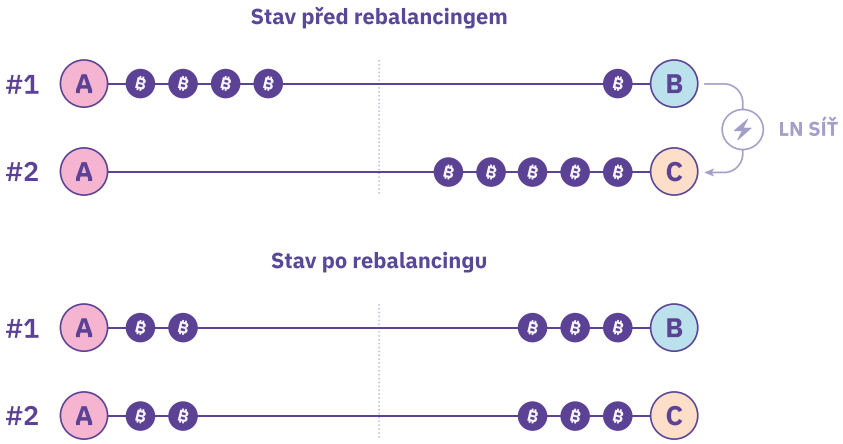
Zpočátku se může zdát, že nejlepší je mít každý kanál vybalancovaný přesně 50:50, abychom pomocí něj mohli odesílat platby oběma směry. Dosáhnout tohoto stavu je zajisté vhodné, každopádně ne vždy je tento poměr naprosto ideální. Postupem času totiž zjistíte, že některými kanály spíše odesíláte a jinými spíše přijímáte. Optimální podíl je tedy velmi individuální. Například můžete mít navázaný kanál s nějakou platební bránou, kterou lze označit termínem **liquidity sink** – jedná se o uzel, který velmi často přijímá platby, ale málokdy je odesílá. V tomto konkrétním případě tedy může být vhodné mít poměr 90:10 ve prospěch odchozí likvidity. Obecně je ale ideální mít možnost všemi kanály přijímat i odesílat.



Co ale dělat v případě, že máte téměř všechny prostředky v určitém kanálu na jedné, nebo druhé straně? A věřte, že se vám to po chvíli stane. První a nejméně pracnou možností je **nedělat nic** a doufat, že jednoho dne bude váš uzel přesměrovávat platbu v opačném směru a kanály se vybalancují automaticky. Šance, zda toto nastane, je hodně individuální – záleží na tom, jaké je postavení vašeho uzlu v síti, s kým máte navázány kanály a kudy teče nejvíce plateb. Někdy se také může stát, že byste takto čekali do nekonečna.

Druhou možností je **upravit poplatky**. Tím, že v jednom kanálu snížíte poplatky na naprosté minimum, zvýšíte šanci, že si váš kanál vybere někdo jiný pro přesměrování platby, a tudíž si v tomto konkrétním kanále změníte odchozí likviditu na příchozí. Naopak zvýšením poplatku šanci snižujete, a proto se nebude stávat tak často, že budou všechny prostředky u protistrany. Pokud tomu tak bude, dostane za to alespoň slušnou provizi. Po jakékoliv úpravě poplatků doporučuji počkat alespoň několik dní na vyhodnocení, zda to bylo úspěšné, jelikož určitou dobu trvá, než se gossip protokolem zpráva rozšíří ke všem a začnou váš kanál využívat.

Pokud by ani toto nepomohlo, můžete zkusit **rebalancovat kanály**. Rebalancing kanálu není nic jiného než platba sám sobě. Vytvoříte tedy transakci, jejímž odesílatelem budete vy (a konkrétně vyberete kanál, ve kterém máte většinu prostředků na své straně) a příjemcem budete opět vy, tentokrát ale budete chtít přijmout kanálem, kterému odchozí likvidita schází. Snad každá dnešní implementace lightningového uzlu vám v pokročilém nastavení umožňuje (na rozdíl od mobilních peněženek) specifikovat, jakým kanálem chcete platbu odeslat či přijmout. Tímto trikem tedy lze kanály vybalancovat.



Zní to skvěle, má to nějaké nevýhody? Ano, bohužel. Rebalancing není nic jiného než klasická platba, a tak za ni musíte zaplatit poplatky ostatním uzlům, které ji budou přesměrovávat. Pokud si spočítáte, že by vás rebalancing stál více, než kolik byste následně díky vybalancovanému kanálu mohli vydělat, tak se to nevyplatí. Některé kanály je zároveň tímto způsobem velmi těžké vybalancovat (například ty, které nazýváme liquidity sink), protože o to samé se snaží velké množství operátorů.

Často ještě vyvstává otázka, jak moc „agresivně" rebalancovat kanály. Dejme příklad, že jsme si spočítali, že náš výdělek bez jakéhokoliv rebalancování je 30 000 satoshi měsíčně. Pokud budeme rebalancovat, tak si vyděláme 150 000 satoshi. Bohužel ale 120 000 zaplatíme za poplatky při rebalancingu, takže to vyjde nastejno. Který z přístupů je v tomto případě lepší? Obecně se **zpočátku** přikláním k tomu druhému. I když je čistý zisk stejný, tak přes vás proteče mnohonásobně více plateb a **získáte důležité informace** o tom, kterým směrem jsou nejvyšší požadavky na tok prostředků. Na základně těchto informací poté můžete upravit svoji strategii a rebalancovat již méně agresivně.

Optimalizace uzlu

V závislosti na tom, jaký je primární důvod provozování vašeho uzlu, se bude lišit strategie týkající se navazování a správy kanálů. Rozdělme si to na tři kategorie – odesílání, příjem a routování plateb.

Pokud provozujete vlastní uzel primárně za účelem **placení**, tedy chcete být nezávislí na jakékoliv třetí straně a mít maximální míru soukromí a kontroly, tak to máte vcelku jednoduché. Bohatě vám postačí si otevřít několik kanálů k větším a dobře propojeným uzlům. Více uzlů bych doporučoval z důvodu redundance, kdyby jedna z protistran nebyla dostupná anebo by neměla dostatečnou likviditu v požadovaném směru. V tomto případě, pokud není vašim cílem přesměrování plateb, je vhodné si všechny kanály otevřít jako privátní. Následně doporučuji napojit si mobilní peněženku na vlastní uzel (vybrat takovou, která to podporuje) a získat tím maximální kontrolu nad svými prostředky.

Druhým důvodem může být **příjem plateb**. To využijí primárně obchodníci a majitelé e-shopů. Postup zde je obdobný jako v případě výše, ale s několika rozdíly. Prvním z nich je, že bych doporučoval si najít uzly jednotlivých mobilních peněženek (minimálně těch populárních), a navázat platební kanál přímo s nimi. Právě z těchto uzlů k vám nejčastěji bude putovat platba a přímý kanál bude jednak pro koncového uživatele mít nulové nebo minimálními poplatky, a zároveň se sníží pravděpodobnost, že platba neprojde. Dalším rozdílem je, že budete potřebovat mít naprostou většinu prostředků na druhé straně jako příchozí likviditu. Kupte si nějaké kanály nebo požádejte komunitu o otevření kanálu k vám. Jakmile přijmete větší množství plateb, budete muset opět řešit nedostatek příchozí likvidity – zde může být řešením například služba Loop Out (popsána později). Zároveň i zde zvažte, zda nechcete mít některé kanály privátní, aby vám likviditu nepřesunovalo routování plateb ostatním.

Posledním a asi nejčastějším důvodem provozování uzlu je **routování plateb** za účelem vydělávání si na poplatcích za přesměrování. Naprostá majorita uzlů je provozována za tímto účelem, tudíž zde máte velkou konkurenci čítající několik tisíc provozovatelů uzlů. Zde žádné jednoduché řešení neexistuje. Obecně se snažíte navázat kanály tak, abyste propojili napřímo ty části sítě, které jsou propojeny špatně – přes velké množství uzlů. Zároveň musíte nastavit adekvátní poplatky, aby vaše cesta byla využívána ostatními a vy jste tak maximalizovali svoje zisky. Jinými slovy se snažíte postavit most přes řeku, který propojí dvě velká města, a za přejetí si budete účtovat mýtné. Máte ale obrovskou konkurenci, která se podobný most snaží postavit také. Pokud již mostů existuje více, lidé budou preferovat ten s nejmenším mýtným. V tomto případě je nutné mít všechny kanály jako veřejné, aby byly propagovány gossip protokolem ke zbytku sítě.

Časté chyby při navazování kanálů za účelem routování plateb

Řada začátečníků se podívá na největší uzly v lightningové síti, které většinou patří provozovatelům známých peněženek, obchodů, směnáren nebo služeb, a naváže mezi nimi kanál, protože zde přece musí být velká poptávka po toku plateb. Poptávka zde velká zajisté je, každopádně není výjimkou, že za prvních několik dní nepřesměrujete jedinou platbu. Problémem je, že tito velcí hráči mají mezi sebou již kanály pravděpodobně vytvořeny, a tudíž ten váš nikdo používat nebude. Je to stejné, jako kdybyste se divili, proč nikdo nejezdí z Prahy do Brna po krásné nové štěrkové cestě, kterou jste vybudovali skrze vaší malebnou vesničku, když tu máme přímou dálnici. I kdyby tento přímý kanál byl vyčerpaný (veškerá likvidita bude na jedné straně), tak zde budou desítky dalších uzlů, se kterými budete soupeřit.

Naopak otevírat kanály mezi malými a naprosto bezvýznamnými uzly také není optimální cesta. Váš kanál bude sice jediným spojením, ale platba zde projde možná tak jednou za pár měsíců.

Samozřejmě, že vaší motivací pro běh lightningového uzlu může být kombinace výše uvedených důvodů, v tomto případě je nutné si svoji strategii upravit podle konkrétních potřeb.

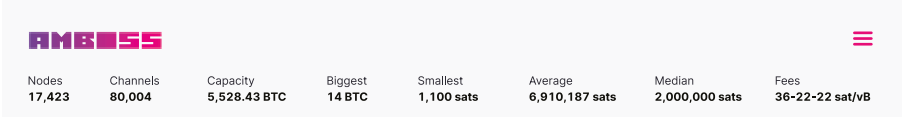
Lightningové vyhledávače

Lightningové vyhledávače jsou veřejně přístupné weby, které sbírají informace o všech uzlech a kanálech v rámci gossip protokolu a prezentují tyto výsledky v uživatelsky přívětivé podobě. Pokud si tedy chcete o nějakém uzlu zjistit všechny dostupné informace, jsou tyto služby ideálním místem. My se podíváme na dvě aktuálně nejpoužívanější databáze.

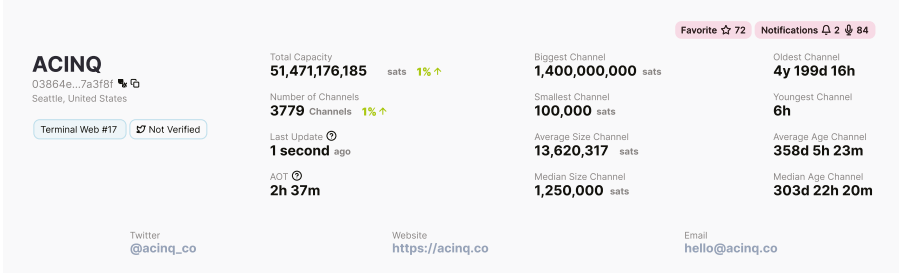
Amboss.space

Tou první je služba **Amboss.space**. Jedná se o aktuálně nejpropracovanější webovou stránku s informacemi o Lightning Network. V horní liště vidíte souhrnné informace o celé síti, jako je celkový počet uzlů, kanálů a další hodnoty.

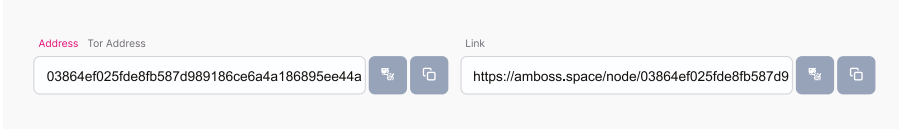
Amboss.space



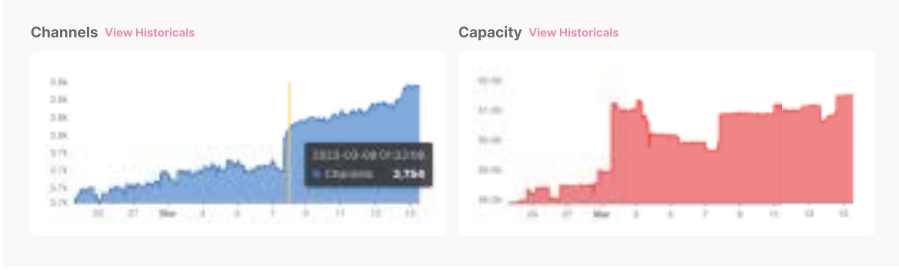
Dále zde máte možnost vyhledávat jednotlivé uzly, ať již podle jejich ID (veřejného klíče), anebo podle aliasu. Ke každému z nich zde naleznete data o celkové kapacitě, počtu kanálů včetně rozložení jejich kapacity a dalších informace. Velmi přívětivou funkcí je možnost přidat si sem osobní informace, jako jsou odkazy na sociální sítě či web, minimální akceptovanou velikost kanálu, kontakt a další. Primárně spojení je velmi užitečnou informací, protože díky němu lze poté kontaktovat provozovatele uzlu v případě problémů.



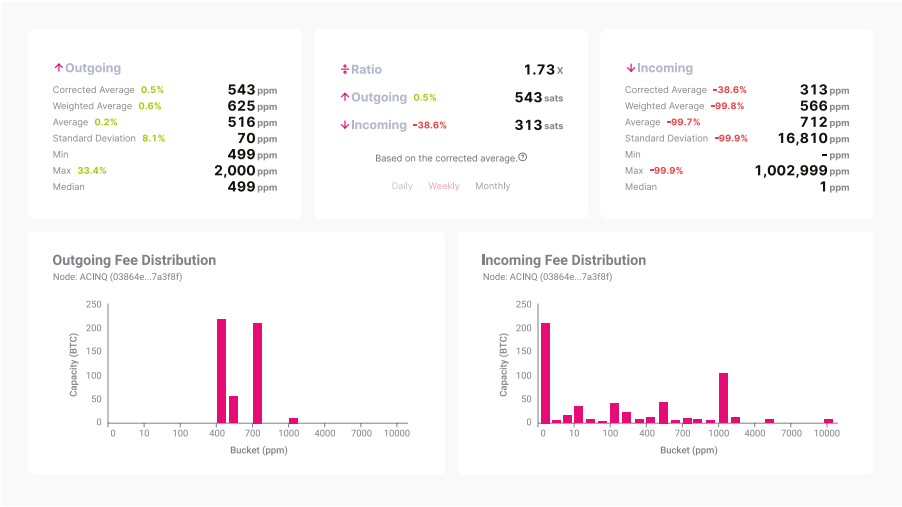
Samozřejmě zde nechybí adresy (IP, případně Tor, podle toho, co daný uzel publikuje), pokud byste s tímto uzlem chtěli navázat spojení.



Další zajímavou informací jsou grafy vývoje počtu kanálů a celkové kapacity. Zde můžete vidět, jak aktivní je tento uzel za poslední období.



Jednou z nejužitečnějších informací je report ohledně poplatků. Na levé straně vidíte informace k poplatkům, které si účtuje tento uzel za přesměrování platby ostatním (minimum, maximum, průměry), a dole máte grafické rozložení poplatků podle rozsahů. Na pravé straně je to identické, tentokrát ale pro poplatky, které si účtují ostatní za přesměrování platby k tomuto uzlu. Uprostřed poté vidíte poměr. Pokud by například odchozí poplatky byly téměř nulové a ty příchozí velmi vysoké, tak se s největší pravděpodobností bude jednat o liquidity sink (uzel, nejčastěji obchodník, který pouze přijímá platby, ale málokdy odesílá).



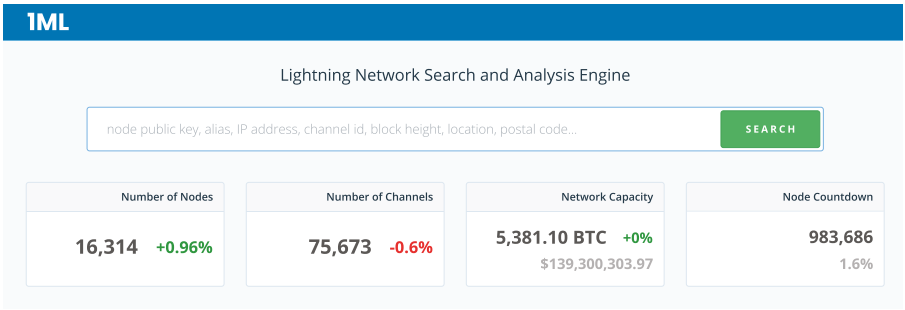
Dále na této stránce naleznete různé komunitní metriky jako LNnodeinsight, LnRouter a Terminal Web, které se snaží každému uzlu přiřadit určité skóre. Na jejím konci máte seznam všech navázaných kanálů. Součástí jsou i detailní informace, což je kapacita kanálu, stáří, poplatky (`base_fee` i `fee_rate`), hodnoty Time Lock Delta (`cltv_expiry_delta`) a další. Amboss.space mimo jiné pomocí červených šipek u každého kanálu signalizuje, zda náhodou není ve stavu `disabled`, a případně z jaké strany. Kanál se do tohoto stavu dostane automaticky v případě, pokud bude cílový uzel nedostupný anebo operátor tento kanál zakáže ručně. Lze tak například detekovat, že je uzel offline (všechny protistrany si tento kanál po nějaké době přepnou do stavu `disabled`).

Amboss.space umožňuje i přihlášení operátorů. To probíhá vygenerováním zprávy, kterou podepíšete na svém uzlu (ať již přes příkazovou řádku, případně přes nějakou grafickou nadstavbu), vložíte odpověď, a tím portál zjistí, jaký uzel vlastníte. Po přihlášení můžete například vyplnit informace o svém uzlu, jako je web, kontakty apod.

Na tomto webu (či obdobných alternativách) budete jako provozovatel uzlu zajisté trávit velké množství času. Díky chytrému zpracování přijatých dat dokážou tyto aplikace přehledně graficky prezentovat informace o jednotlivých uzlech, kanálech a celkové struktuře lightningové sítě.

1ML

1ml.com je alternativou k webu Amboss.space. Na úvodní stránce jsou opět vidět statistiky ohledně lightningové sítě a možnost vyhledávání.



Po vyhledání konkrétního uzlu se dozvíte informace jako celková kapacita, počet kanálů, adresy, proprietární hodnocení, detailní seznam kanálů apod. I zde je možnost vložit kontaktní informace.

Vzhledem k tomu, že informace, které zde lze nalézt, jsou velmi podobné těm u Amboss.space, nemá cenu je procházet znovu. Určitou nevýhodou 1ML oproti Amboss.space je způsob přihlášení. Pokud si totiž vlastní uzel chcete přivlastnit a vyplnit si o něm požadované informace (jako například kontakt), webová služba vám vygeneruje kód (většinou šestimístné číslo) a vy nyní musíte otevřít platební kanál o kapacitě rovné vygenerovanému kódu k uzlu provozovatele 1ML. Jinými slovy vás tato webová stránka nutí uzamknout stovky tisíc satoshi do kanálu s nimi jen kvůli přihlášení. Aktuálně již existují efektivnější způsoby a osobně mi to přijde zbytečné.

Záleží tedy na každém z vás, který vyhledávač vám bude vyhovovat. Je téměř jisté, že postupem času budou vznikat nové a prezentované informace se budou rozšiřovat, berte tedy tuto kapitolu jako takový aktuální obecný přehled.

Analýza uzlu pro navázání kanálu

V klasickém bitcoinovém protokolu si jsou všechny uzly rovnocenné. Vaše implementace se připojí k několika dalším fullnodům (a zároveň umožní ostatním připojit se k vám) za účelem přeposílání zatím nezpracovaných on-chain transakcí, jednotlivých bloků a podobně. V tomto případě vám je téměř jedno, s kým jste takto navázali spojení, pokud vám odesílá relevantní data.

V Lightningu tohle ale neplatí. I když se také jedná o decentralizovanou P2P síť, tak je asi všem jasné, že je velký rozdíl mezi připojením ke známému a dobře propojenému uzlu o velké kapacitě, a nějakému jinému provozovateli, který zrovna včera zprovoznil svůj Umbrel a operuje s jedním malým kanálem. Rozhodně zde neplatí, že čím větší uzel, tím lepší, každopádně každý provozovatel si musí pečlivě vybrat, s kým naváže platební kanál. Existuje několik faktorů, které je vhodné před jakýmkoliv navázáním kanálu zvážit. K jejich zhodnocení vám pomohou dříve představené lightningové vyhledávače a na základní faktory se nyní podíváme.

Stáří uzlu. Pokud uzel běží bez problémů již rok, je zde větší šance, že tomu tak bude i do budoucna. Často se totiž při navázání kanálu k nově vzniklému uzlu stane, že protistrana si Lightning pouze zkouší a po týdnu vše vypne. Nejenže poté přijdete o kanál, ale zároveň budete zbytečně platit on-chain poplatky za otevření a zavření.

Dostupnost. Rozhodně se nechcete propojit s někým, kdo je polovinu doby offline. Moje doporučení je požadovat dostupnost vyšší než 99 % – většina kvalitních uzlů to splňuje.

Kapacita a počet kanálů. Největší uzly v síti bývají již velmi dobře propojené a navázáním kanálu k nim budete jen další z tisíce. Naopak ty nejmenší mají zase malou konektivitu do zbytku sítě. Toto je tedy silně individuální a záleží to na vašich osobních preferencích a postavení vašeho uzlu v rámci sítě.

Trend počtu kanálů a kapacity. Ideálně se chcete spojit s uzlem, u kterého v průběhu času mírně roste počet kanálů a celková kapacita. Znamená to totiž, že provozovatel aktivně navazuje další spojení a ostatní jej navazují k němu. Naopak silně sestupný trend je varováním, že s uzlem není něco v pořádku (neodpovídá, je často offline, má přemrštěné poplatky apod.).

Poplatky. Podívejte se, jaká je struktura poplatků, které si uzel účtuje za přeposílání plateb ostatním. Pokud jsou extrémně vysoké, bude jakákoliv cesta přes tento kanál drahá a málokdy ji ostatní uzly zvolí. Zároveň zde budete mít problém i s rebalancingem odchozí likvidity. Pokud totiž nastaví protistrana vysoký poplatek i k vám, tak jej budete muset při rebalancingu vždy zaplatit, jelikož se jedná o poslední hop směrem k vašemu uzlu. Zároveň je vhodné se podívat i na to jaký je průměrný poplatek, který si účtují ostatní uzly k vašemu kandidátovi. Téměř nulové značí, že ostatní mají problém tímto kanálem odeslat nějaké prostředky, a tak jsou nuceni snižovat poplatky až téměř na nulu. Naopak velmi vysoké značí, že veškerá likvidita bude po většinu doby u protistrany.

Protokol. Pokud provozujete business a záleží vám primárně na rychlosti a spolehlivosti plateb, můžete preferovat spojení s uzly, které publikují IP adresu před těmi, které jsou dostupné pouze skrze Tor.

Komunitní hodnocení. Lightningové vyhledávače často publikují různé proprietární systémy hodnocení uzlu podle toho, jak dobře jsou propojené. Toto lze částečně také využít v rámci vaší individuální strategie.

Otevírání kanálů

Samotné otevření kanálu s protistranou je velmi jednoduché. Lze ho provádět z příkazové řádky vaší lightningové implementace, případně z nějaké grafické webové nadstavby. Nejprve je nutné se s cílem síťově spojit (některé nástroje to udělají na pozadí za vás) a potom jen specifikovat, jak velkou má mít kanál kapacitu, zda bude privátní, a nastavit si hodnotu on-chain poplatků za otevírající transakci. Pokud nespěcháte, doporučuji využít víkendů, kdy jsou poplatky obecně nižší. Osobně jsem zatím všechny své kanály otevíral s poplatkem 1 sat/vB. Další možností je při otevření přesunout určitou částku k protistraně – toto využívejte pouze v případě, že k tomu

máte důvod (například jste se domluvili s kamarádem), jelikož o tuto částku efektivně přijdete. Poté jen stačí čekat na vytěžení vaší transakce, její dostatečné zanoření do blockchainu a kanál máte otevřený. Veškerá kapacita je nyní na vaší straně.

> **Channel reserve**
>
> Při důkladném zkoumání narazíte na to, že každá strana kanálu si drží určitou rezervu ze svých prostředků, kterou nemůže utratit. Doporučováno je 1 %, tudíž při kapacitě kanálu 5 milionů satoshi bude mít každá ze stran 50 000 satoshi, které nemůže utratit. Je to zavedeno z toho důvodu, aby každá strana měla „co ztratit", pokud by odeslala již revokovanou transakci. Kdyby totiž jeden z účastníků neměl na své straně vůbec nic, čistě teoreticky by se mu vyplatilo zkusit podvádět. V tom nejhorším případě by totiž měl kvůli penalty transakci přijít o vše, což by však při nulovém zůstatku kanálu nebylo trestem. Výjimkou je prvotní otevření kanálu, kdy jsou všechny prostředky na vaší straně, v tomto případě protistrana žádnou rezervu nemá. Zároveň ale nedisponuje žádnou revokovanou commitment transakcí, prostřednictvím které by mohla podvádět.

V průběhu existence kanálu si můžete měnit různé parametry:

- Poplatky. Jak pevné (`base_fee`), tak ty proporcionální (`fee_rate`).
- `cltv_expiry_delta`. Někdy označováno jako Time Lock Delta. Počet bloků, které si vyhrazujeme na zpracování HTLC. U nižší hodnoty riskujeme ztrátu prostředků v případě nutnosti uzavřít kanál, zároveň se nám ale zvyšuje šance, že si nás někdo vybere pro přesměrování platby.
- `Min/Max_HTLC`. Minimální, nebo maximální hodnota, kterou jsme ochotni přeposílat či odesílat.

Ideální kapacita kanálu

Našli jsme si partnera, s kterým chceme otevřít kanál, a nyní jen řešíme, jaká je jeho ideální kapacita. Mnoho začátečníků dělá tu chybu, že otevírá příliš malé kanály. Pokud se omezíme pouze na **optimalizaci pro přesměrování plateb**, tak kanál s kapacitou menší než 1 milion satoshi obecně nemá moc smysl. Ideální hodnota neexistuje, ale obecně se jako optimální udává něco mezi 2 až 10 miliony satoshi. Proč nedoporučuji vytvářet veřejný kanál o malé kapacitě, například 200 000 satoshi? Důvodů je několik.

- Tento kanál bude omezen v tom, že maximální částka, kterou jím bude možné přesměrovat, bude 200 000 satoshi. A to jen v případě, že veškeré prostředky

budou na té správné straně. Kdyby byla kapacita v tomto kanálu ideálně rozložena, tedy 50:50, tak se vlastně omezíme na maximální částky 100 000 satoshi. Reálně se ale přes Lightning posílají i statisícové částky – ty by se tedy našemu kanálu vyhýbaly a my bychom nezískávali poplatky za přesměrování.

- U každého kanálu musíme zaplatit on-chain poplatek za otevření, a pokud k tomu dojde, tak i za uzavření. Ten ale není závislý na kapacitě kanálu. Jinak řečeno – za kanál o kapacitě 5 milionu satoshi zaplatíme stejně jako za kanál o velikosti 200 000 satoshi. V tom druhém nám ale bude trvat znatelně déle, než se nám tento vstupní náklad vrátí pomocí příjmu poplatků za přesměrování.

- U malého kanálu se bude často stávat, že bude veškerá likvidita na jedné, nebo druhé straně. To znamená velmi častou nutnost rebalancingu. A i po úspěšném vybalancování stačí jedna větší platba a můžeme začít znovu.

- V rámci pathfinding algoritmu jsou často upřednostňovány větší kanály, takže je dost možné, že vás pro přesměrování plateb bude využívat málokdo. Jaký je důvod? I když je to pro mnoho lidí možná překvapení, tak z praktického pozorování vyplynulo, že poměr odchozí a příchozí likvidity je statisticky rovnoměrné rozdělení. To znamená, že je stejná šance, že bude kanál v poměru 20:80 jako 70:30. Díky tomu se dá zjednodušeně říci, že pravděpodobnost úspěšného přesměrování platby je rovna přeposílané částce vydělené celkovou kapacitou kanálu. Z toho vyplývá, že pokud budu hledat ideální trasu a neznám konkrétní hodnoty odchozí/příchozí likvidity, tak se mi vyplatí využít větší kanály, jelikož je tam vyšší šance, že požadovaná likvidita bude v mém směru dostupná.

- Malé kanály obecně lightningové síti škodí. Vzhledem k tomu, že je často veškerá jejich kapacita na jedné straně anebo je nedostatečná, dochází kvůli tomu k neúspěšným platbám a uživatelům se snižuje uživatelská přívětivost Lightningu, protože musí déle čekat na nalezení úspěšné trasy.

Ano, chápu. Ne každý si může dovolit otevřít několik kanálů s kapacitou v jednotkách milionů satoshi. Pokud ale chcete úspěšné routovat velké množství plateb a přijímat díky tomu poplatky, je to bohužel nutnost. Podobné to je například u těžby bitcoinu, kdy musíte zpočátku investovat nemalé peníze do těžících zařízení.

Výše uvedený text je opravdu zaměřen na uzly, které plánují maximalizovat zisk z přesměrování plateb. Pokud chcete jen přijímat a odesílat platby pro osobní využití či si jen s kamarády vyzkoušet různé možnosti Lightningu, jsou malé kanály v pořádku. Jen bych se u nich vždy zamyslel, zda nedává v tomto případě smysl je vytvářet jako privátní.

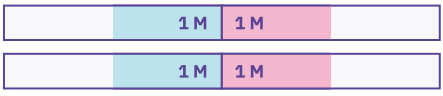
Naopak extrémně velké kanály přes 10 milionů satoshi musí mít svoje opodstatnění pro konkrétní uzel. Pokud ho nemají, bude s největší pravděpodobností lepší využít tyto prostředky pro navázání více kanálů.

Lightning Network Plus

Jedna z nejužitečnějších služeb, která usnadňuje otevírání kanálů, zajišťuje příchozí likviditu a pomáhá decentralizaci sítě, je Lightning Network Plus. Jedná se o webovou stránku, kde se provozovatelé uzlů mohou domluvit na tzv. rings of fire swapech. Tento swap má určitý počet účastníků (3 až 5) a podle toho jeho tvar připomíná buď trojúhelník, čtverec, nebo pentagon.

Představme si případ, že máme 2 miliony satoshi a chceme si otevřít další platební kanál. Zanalyzovali bychom si potenciální partnery a s vybraným uzlem navázali kanál. Zpočátku by ale byly všechny prostředky na naší straně jako odchozí likvidita.

My ale místo toho na webu Lightning Network Plus nalezneme swap o tvaru trojúhelníku. Pravidla tohoto swapu určují, že se bude jednat o kapacitu 2M satoshi a jsou v něm dva účastníci (jejich uzly), kteří čekají na třetího. Jakmile se tohoto swapu zúčastníme, vytvoří se kruh kanálů (někdy zvaný také Ring of Fire), kdy uzel A naváže platební kanál s B, B naváže s C a C zpět s A. Tímto způsobem získám 2 kanály (jeden iniciuji já, druhý vytvoří protistrana). Následně se přes komentáře na stránce s kolegy domluvíme, že si dočasně nastavíme nulové routovací poplatky a provedeme rebalancing. Ten spočívá v tom, že jeden z nás provede platbu sám sobě přes ostatní účastníky s polovinou kapacity kanálu, v našem případě by to byl 1M satoshi. Tímto způsobem se všechny kanály vybalancují do ideálního poměru 50:50.



Výhodou je, že mám při stejných nákladech (2M satoshi) otevřené dva kanály místo jednoho a zároveň jsou oba perfektně vyvážené – alespoň zpočátku. Pokud by se mi žádný z nabízených swapů nelíbil, lze si vytvořit vlastní a čekat, zda se k němu někdo přidá. Mám možnost si zvolit jeho kapacitu, požadovanou délku

trvání, a případně omezit uzly, které se smějí swapu zúčastnit – řekněme, že budu požadovat, aby účastníci měli alespoň 5 kanálů a jejich celková kapacita byla minimálně 10M satoshi.

Tato služba je mezi komunitou vcelku rozšířená, a to oprávněně. Pro běžné provozovatele uzlu se jedná o jeden z nejlepších způsobů otevírání kanálů. Má nějaké nevýhody? Ano, zaprvé si nemůžete vybrat, kdo se k vám do swapu přidá. Pokud budete mít smůlu, můžete „vyfasovat" někoho, kdo bude často offline, nastaví směrem k vám obrovské poplatky nebo nebude mít žádnou reálně použitelnou likviditu. Druhým negativem je, že doba, po kterou jste si navzájem nastavili nulové poplatky (kvůli počátečnímu balancování), musí být co nejkratší, jelikož zde hrozí, že tuto velmi levnou cestu využije někdo jiný a prostředky vám přesune jiným směrem, než požadujete. Vy z toho navíc nic nezískáte. Obě tato negativa jsou ale spíše drobnosti, jinak se jedná o velmi povedenou službu.

LN-Big a prodej likvidity

LN-Big je webová služba, kde si můžete za určitý poplatek koupit příchozí likviditu tím způsobem, že její správce otevře kanál směrem k vašemu uzlu. Jedná se o zajímavý nápad, jak získat další příchozí likviditu, protože to je něco, s čímž většina provozovatelů uzlů bojuje. LN-Big vám zaručí, že tento kanál bude otevřený minimálně po dobu 30 dní. Následně LN-Big vyhodnotí, zda se tímto kanálem přesunuly alespoň nějaké satoshi, a pokud ano, nechá ho s největší pravděpodobností nadále otevřený. V opačném případě může dojít k uzavření a použití těchto prostředků jinde.

Pokud bychom si například chtěli zakoupit kanál o celkové kapacitě 5M satoshi, aktuálně v době psaní této knihy bychom za to zaplatili poplatek 13 500 sat, což dělá 0,27 %. Není to však mnoho, když si vezmete, že za tuto částku zvýšíte příchozí likviditu svého kanálu o 5 milionů. Poplatek se skládá z on-chain poplatků za otevření uzlu, případné zavření a určitá část jde navíc provozovateli za „pronajmutí" těchto prostředků.

LN-Big provozuje aktuálně několik desítek uzlů a nelze si vybrat, z které instance k vám otevřou kanál. Všechny jsou si ale, co se týká konfigurace, velmi podobné.

Mimo LN-Big existuje několik jiných podobných provozovatelů, například Bitrefill, Thor, Yalls, Coincept a další. Někteří provozovatelé kvalitních uzlů nabízí obdobné služby přes sociální sítě. Princip je všude stejný, za určitý poplatek otevřou kanál k vašemu uzlu. Je poté na zvážení každého z vás, zda se vám to vyplatí. Určitě bych si ale udělal i důkladnou analýzu uzlu, ze kterého by k vám měl být kanál navázán.

Pool

Lightning Pool je non-custodial, peer-to-peer tržiště od společnosti Lightning Labs, které funguje pouze **s implementací LND**. Dochází zde k propojení uživatelů, kteří mají dostatek prostředků a prodávají svoji likviditu těm, kterým naopak příchozí likvidita schází a chtějí si koupit platební kanál.

Pro interakci s Poolem je zapotřebí mít nainstalovaný a nakonfigurovaný démon zvaný `poold`. Ten by měl neustále běžet na pozadí a služba se ovládá pomocí příkazové řádky s využitím nástroje `pool`. Alternativou je využití webové aplikace Lightning Terminal (také od společnosti Lightning Labs), která v sobě již Pool obsahuje. Lightning Terminal si představíme v následujících kapitolách. Terminal je také součástí mnoha hotových řešení, kde ho lze nainstalovat na pár kliknutí z rozhraní uzlu.

Abyste mohli s tímto nástrojem pracovat, musíte si podobně jako u směnárny otevřít účet. Pod pojmem účet se zde myslí speciální on-chain multisig adresa typu 2 ze 2. Nejprve je nutné na tuto adresu odeslat určité prostředky a nastavit si expiraci (například 3 měsíce). Privátní klíče k této adrese držíte vy a provozovatel služby Pool. Ve skriptu je uvedeno, že po vypršení expirace si prostředky můžete vzít i se znalostí pouze vašich privátních klíčů (ochrana proti tomu, abyste nepřišli o prostředky, pokud by provozovatel Poolu přestal existovat). Jinak se dají na tento účet klasicky nahrávat bitcoiny, vybírat, prodlužovat jeho platnost apod.

Jakmile máme vytvořený účet, můžeme se pustit do aukce. Existují zde 2 typy – **Bid** (potřebuji koupit kanál a tím získat příchozí likviditu) a **Ask** (prodávám svoji likviditu tím, že navážu kanál k ostatním). Představme si nyní příklad, že si chci koupit příchozí likviditu o kapacitě 5M satoshi. V tom případě pomocí příkazové řádky, případně z webu Lightning Terminalu, vytvořím Bid příkaz. V něm specifikuji celkovou sumu požadované příchozí likvidity, o kterou mám zájem (v našem případě to je 5M satoshi), na jak dlouho si chci tuto kapacitu koupit (aktuálně existují trhy s dobou trvání 2 týdny, 1 měsíc, 3 měsíce a 1 rok) a posledním povinným parametrem je cena, kterou jsem za tuto likviditu ochoten zaplatit vyjádřená v procentech celkové částky, v našem případě třeba 1 %. Nepovinně lze ještě specifikovat, jaká bude minimální velikost kanálu. Protože chceme získat jeden platební kanál, uvedeme 5M satoshi (jinak by se mohlo stát, že se k nám naváže více kanálů o celkové kapacitě 5M).

Obdobným způsobem ostatní uživatelé zadávají příkazy typu Ask. U nich se specifikuje opět trh (doba trvání), nabízená kapacita a kolik procent za toto propůjčení na konkrétní čas chci získat.

Tyto nabídky a poptávky jsou neveřejné. Přibližně jednou za 10 minut (s každým vytěženým blokem) dojde k vyhodnocení, a pokud se nalezne shoda mezi poptávkou a nabídkou, prodej se uskuteční. V našem případě by nám z našeho účtu u Poolu byly odečteny poplatky službě a za nákup likvidity, z účtu prodejce by naopak byl navázán požadovaný kanál směrem k nám.

Doba trvání kanálu je specifikována v uskutečněném obchodu. Aktuálně je v rámci LND zajištěno, že tento kanál nesmí být kooperativně zavřen dříve, než tato doba uběhne. Bohužel zatím ale není omezeno vynucené uzavření jednou stranu (force-close), takže pokud narazíte na nepoctivého prodejce, máte smůlu. Obdobně zatím není nijak ošetřen stav, kdyby protistrana byla poté často offline anebo měla jiné technické problémy. Případně může nastavit extrémně vysoké poplatky a tento kanál vám bude reálně k ničemu. Určitou nevýhodou také je, že dopředu nevíte, kdo k vám kanál otevře. Je zde sice proprietární rozdělení na 2 skupiny („kvalitní uzly“ a ostatní), žádnou záruku o poctivosti zde však nemáte. Zároveň cena, za kterou se dají kanály koupit, je oproti jiným službám, jež prodávají kanály, velmi vysoká. Cena je zde určena průnikem nabídky a poptávky, každopádně aktuálně jsou poplatky nižší než 1 % požadované kapacity spíše výjimkou. Na druhou stranu zde můžete zkusit štěstí jako prodejce.

Oproti jiným službám je zde určité množství nevýhod, což je možná ten hlavní důvod, proč není toto tržiště moc využívané. Aktuálně na nejobchodovanějším trhu s 3měsíčním trváním dojde přibližně k jednomu uskutečněnému prodeji denně. Každopádně až čas ukáže, zda bude o tuto službu do budoucna zájem a tvůrcům se podaří technicky vyřešit výše uvedené problémy, či nikoliv.

Liquidity Ads

Mezi jednotlivými implementacemi panuje konkurenční boj, a tak odpovědí na tržiště Pool od Lightning Labs jsou takzvané Liquidity Ads, s kterými přišla společnost Blockstream stojící za implementací Core Lightning.

U Liquidity Ads se využívá koncept dual-funded kanálů, tedy kanálů, na jejichž celkovou kapacitu se skládají v určitém poměru obě strany. Díky tomu je kanál částečně vyvážený už v počátku své existence. Bývá zde ale častý problém s tím, že je složité protistranu najít, a musí se využívat externí médium, jako je například Twitter.

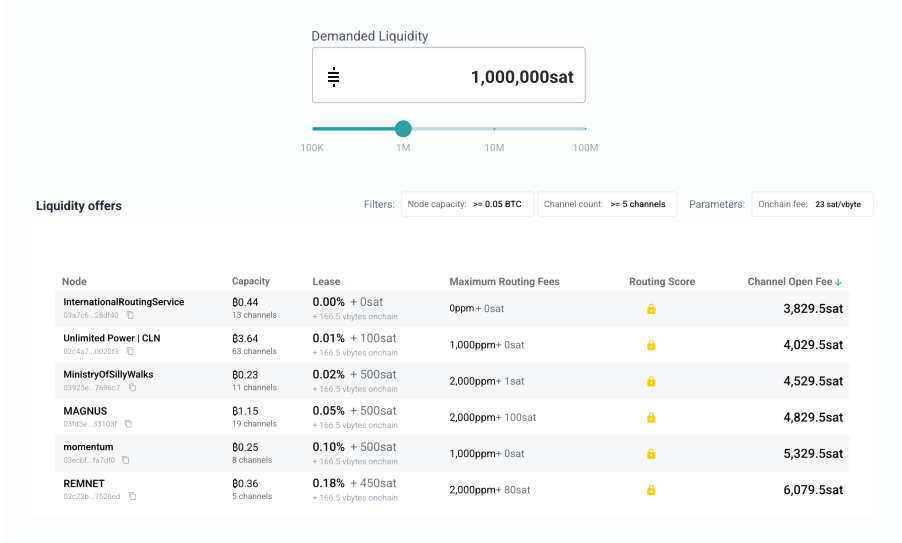
Liquidity Ads řeší tento problém tak, že moje ochota otevřít s někým kanál se šíří gossip protokolem, tudíž si informace může kdokoliv přečíst a zařídit se podle toho. Například si ode mne příchozí likviditu koupit. Není potřeba se domlouvat na sociálních sítích, ale využije se nativních protokolů Lightningu.

Pro zjištění, které všechny uzly jsou ochotny tímto způsobem prodat svoji likviditu, mohu využít příkaz `listnodes`, kde si zobrazím několik řádků před a po klíčovém slovu `option_will_fund`. Příklad:

```
$ lightning-cli listnodes | grep -B20 -A7 option_will_fund
...
„option_will_fund“: {
„lease_fee_base_msat“: „1000000msat“,          # 1000 satoshi si účtuji bez ohledu na velikost kanálu
„lease_fee_basis“: 10,                         # proporcionální poplatek v bazických bodech, tedy 0,1 %
„funding_weight“: 666,                         # velikost funding transakce
„channel_fee_max_base_msat“: „1000msat“,       #slib, kolik bude maximální base fee pro tento kanál
„channel_fee_max_proportional_thousandths“: 1, # slib, kolik bude maximální fee rate pro tento kanál
„compact_lease“: “029a000a0064000003e84c4b40“  # jiná reprezentace hodnot výše, využito při nákupu
}
```

Pokud bych chtěl této nabídky využít a koupit si kanál o velikosti 5M satoshi, tak mne to bude stát 1 000 satoshi jako pevný poplatek za nákup, dále 0,1 % z celkové kapacity, tedy 5 000 satoshi, a konečně 666 * zvolený on-chain poplatek (počítejme optimisticky s 1 sat/vB). Celková cena by tedy byla krásných 6 666 satoshi. Dále mi zde protistrana slibuje, že nenastaví vyšší poplatky než výše uvedené.

Alternativou jsou webové stránky, které tyto nabídky zobrazují online, například Liquidity Ad Marketplace na webu LnRouter.app:



Pro publikování těchto nabídek se využívá modul `funder`, ve kterém si lze definovat podmínky, za nichž jsem ochoten prodávat likviditu. Musím však mít dostatek on-chain prostředků, jinak otevření selže.

Následný nákup této likvidity je už vcelku jednoduchý. Řeknu pouze, od kterého uzlu likviditu kupuji, v jakém množství, mohu přidat nějaké své prostředky a vložím reprezentaci podmínek.

```
$ lightning-cli fundchannel -k id=02xxxxxxxxxxxxxxx45 amount=0.05btc
request_amt=0.05btc compact_lease=029a000a0002000003e80186a0
```

Oproti Poolu je zde výhoda, že vím, od koho si likviditu kupuji, a mohu si ho předem prověřit. Dále není potřeba nějakého centralizovaného tržiště, veškerá komunikace probíhá v rámci nativního gossip protokolu. Zároveň je zde možnost vložit do kanálu i své prostředky, aby byl zpočátku vybalancovaný. Jediná implementace, která Liquidity Ads zatím implementovala, je Core Lightning. Pokud tedy používáte LND, nemůžete jich využít.

Amboss Magma

Webové tržiště pro nákup a prodej likvidity s názvem Magma je součástí dříve popsaného lightningového vyhledávače a databáze Amboss.space. Stejně jako u řešení výše si zde mohu koupit příchozí likviditu tím, že někdo otevře platební kanál ke mně. Případně zde mohu naopak nabídnout své bitcoiny pro vytvoření kanálu ke kupujícímu, za což získám mnou určený poplatek. Toto řešení není závislé na žádné konkrétní implementaci Lightningu.

Před samotným nákupem platebního kanálu je nutné se na web Amboss.space přihlásit. To se provádí tím způsobem, že na svém uzlu podepíši zprávu, kterou mi web vygeneruje. Dále je vhodné si propojit Amboss s aplikací Telegram kvůli notifikacím ohledně prodeje či nákupu kanálů.

Na webu je následně k nalezení seznam prodejců likvidity. U každé z nabídek najdu:

- Informace a statistiky o uzlu prodejce včetně historie prodejů a **skóre** (důležité, viz níže).
- Množství zbývající likvidity k nákupu.
- Minimální a maximální nabízená kapacita kanálu.

- Cena za nákup (jak fixní, tak v závislosti na kapacitě).
- Slib maximálních hodnot poplatků pro tento kanál.
- Minimální délku trvání platebního kanálu.
- APR – procentuální možnost zhodnocení svých prostředků při prodeji kanálů.

Po výběru ideální nabídky si zvolím, jak veliký kanál chci zakoupit (pokud je na výběr), a potvrdím. Nyní má prodejce několik hodin na to, aby obchod schválil. Právě zde je vhodné mít připojený Telegram, jelikož o všech těchto krocích budete dostávat notifikace. Po schválení je mi zobrazena klasická faktura, kterou musím zaplatit. Prostředky zatím nejdou prodejci kanálu, ale jdou do úschovy ke službě Amboss. Nyní se čeká, až protistrana naváže platební kanál, vytěží se funding transakce a zároveň se informace o tomto kanálu dostane gossip protokolem až k uzlu služby Amboss. Až v tuto chvíli obdrží prodejce z úschovy prostředky za prodej, jež jsou lehce poníženy o poplatek, který si bere samotná služba za zprostředkování. Pokud si na stránce zaplatíte prémiový účet (zvaný Prime), poplatky službě jsou nižší. Jedná se o elegantní řešení, kdy je zaručeno, že žádná ze stran nemůže podvádět. Drobnou nevýhodou zde je ale nutnost důvěry ve službu Amboss Magma, která po určitou dobu drží poplatky za otevření kanálu a na celý proces dohlíží. Prodej kanálu je identický, pouze si vyměníte strany.

Jakékoliv pokusy o podvod – například dřívější uzavření kanálu nebo nastavení vyšších poplatků, než bylo domluveno při koupi, jsou viditelné v rámci gossip protokolu a Amboss je penalizuje snížením skóre, které je přiřazené ke každému uzlu. Pokud tedy nějaký bude mít skóre nízké, nedoporučuji s ním obchodovat.

V kapitolách výše jsme si popsali několik způsobů, jak otevírat kanály – od klasického otevření, které bude rozhodně nejčastější variantou, přes komunitní Lightning Network Plus až po několik způsobů nákupu příchozí likvidity. Nyní je již na každém z vás, které varianty budete využívat, proto berte tuto kapitolu jako jakýsi přehled jednotlivých možností.

Nástroje pro správu

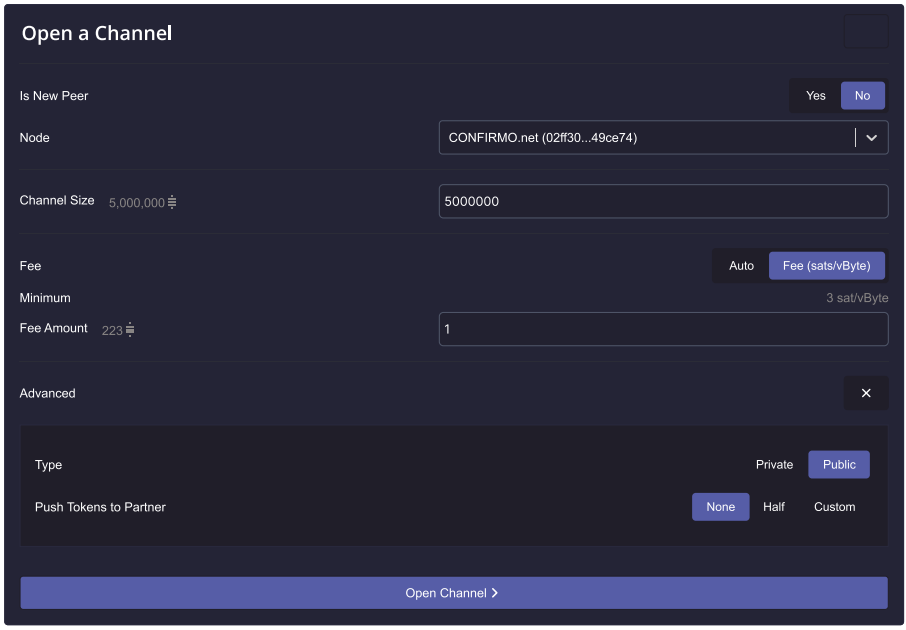
Ať si vyberete jako svoji primární implementaci LND, Core Lightning nebo Eclair – všechny se v základu ovládají přes příkazovou řádku. Na tom není nic špatného a má to v určitých případech své využití. Každopádně mnoho úkonů se lépe provádí v rámci nějaké webové nadstavby, kde máte nejrůznější informace o vašem uzlu uvedeny v tabulkách a graficky vykresleny. Zároveň správa kanálů je minimálně pro začátečníky s využitím těchto nadstaveb jednodušší a pohodlnější.

V této kapitole se podíváme na nejznámější nadstavby, které vám umožní váš uzel ovládat. Jedná se vždy o webové aplikace, které běží na vašem uzlu. Z jedné strany pomocí API komunikují s vaší lightningovou implementací a na druhé straně vám reprezentují požadované informace v grafické podobě. Způsob instalace těchto nástrojů se liší v závislosti na tom, jak máte svůj uzel postaven. Pokud jste si ho sestavili vlastnoručně, musíte si i tyto aplikace stáhnout, nainstalovat a správně nakonfigurovat. Jestliže ale využíváte různých hotových řešení, tak je jejich zprovoznění většinou otázkou několika kliknutí.

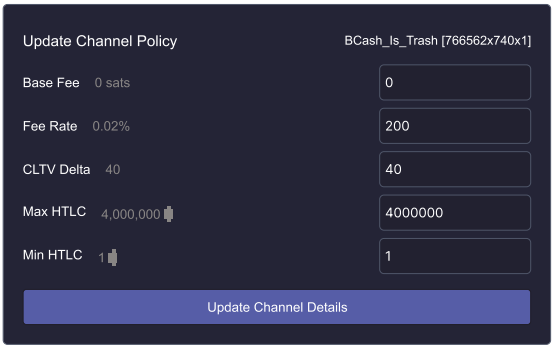
ThunderHub

ThunderHub je graficky velmi povedený open-source nástroj pro správu a monitoring uzlu postaveného nad LND. Na hlavní stránce máte přehled o veškerých vašich prostředcích – vidíte zde, kolik vlastníte bitcoinů v on-chain peněžence, která je v rámci LND, a zároveň i součet odchozí likvidity přes všechny vaše kanály, tedy kolik bitcoinů vlastníte v rámci Lightningu. Samozřejmostí je možnost odesílání a příjmu (jak on-chain, tak Lightning). Oproti klasickým mobilním peněženkám zde máte podporu mnoha funkcí od LNURL přes keysend až po definování speciálních parametrů faktur.

Přes toto webové rozhraní můžete také jednoduše otevírat kanály

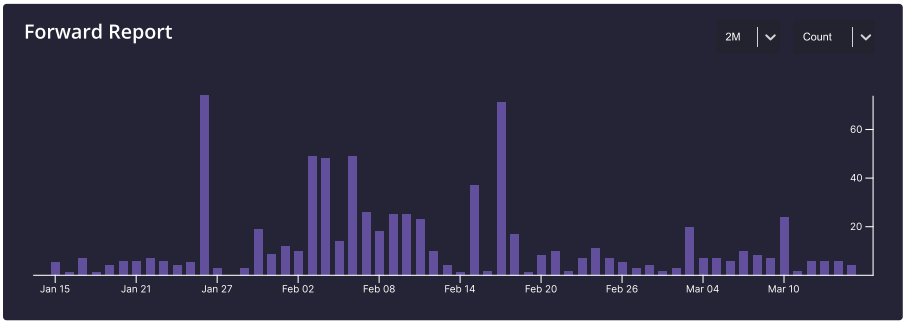


Samozřejmostí je také úprava vlastností kanálů, případně jejich uzavření.



Existuje opravdu velmi malé množství běžných činností, které by přes ThunderHub nešly realizovat. Máte zde přehled všech uzlů, s nimiž jste spojeni, včetně podrobných detailů. V přehledné tabulce vidíte opravdu velké množství informací o všech vašich kanálech. Samozřejmostí je i výpis provedených plateb, podrobnosti o přesměrovaných transakcích, součty příchozí i odchozí likvidity, množství nevyřízených HTLC s jejich detaily, výpis veškerých UTXO apod.

Kromě všech těchto statistických dat, která jsou většinou podpořena pěknými grafy, zde máte i možnost rebalancingu, generování reportů, provádění záloh, tvorby vlastních macaroon souborů (obdoba webových cookies, které umožňují navíc definovat různá práva, a využívají se v rámci autorizace na LND uzlu), provádění submarine swapů (výměna lightningových prostředků za on-chain) apod.

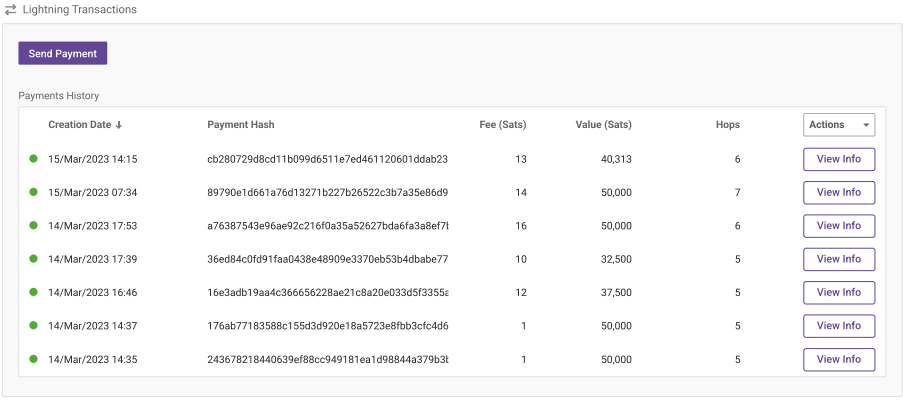


Velkou výhodou této grafické nadstavby je fakt, že vám umožňuje si z jednotlivých grafů a tabulek sestavit vlastní dashboard, na který si jednoduše poskládáte ty informace, které vás nejvíce zajímají. Při běžných kontrolách se poté nemusíte složitě proklikávat různými prvky menu, ale stačí si pouze otevřít předpřipravený dashboard, kde máte všechny informace pohromadě.

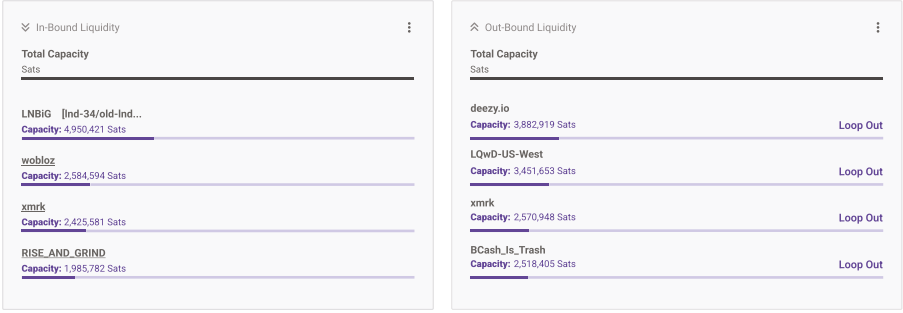
Ride The Lightning

Ride The Lightning (zkráceně RTL) je alternativou k výše popsanému ThunderHubu. Jeho výhodou je, že je kompatibilní s LND, Core Lightning i Eclair implementací, což není u podobných nástrojů moc časté.

Možnosti RTL jsou téměř totožné jako u ThunderHubu. Opět zde máte nepřeberné množství grafů, tabulek a informací o on-chain prostředcích, kanálech, transakcích, vygenerovaných fakturách apod.



Záložka on-chain vám umožňuje generovat nové adresy, sledovat veškeré přijaté i čekající transakce, odesílat prostředky a podobně. Na kartě Lightning naopak naleznete velké množství záložek pro navazování a úpravu platebních kanálů, přehled přesměrovaných plateb, generování reportů, provádění záloh, podepisování a ověřování zpráv atd. Můžete se zde i rychle podívat na vaši konfiguraci lightningové implementace, provést rebalancing, případně si propojit RTL s dalšími službami jako například Loop (probereme později).



Na některé úkony může být vhodnější a přehlednější ThunderHub, na jiné RTL. Pokud vám to vaše lightningová implementace dovolí, zkuste si obě rozhraní a uvidíte, které z nich vám bude více vyhovovat.

Nástroje, skripty a služby

V této kapitole si představíme různé nástroje, skripty a služby, které pomáhají operátorům se správou jejich lightningového uzlu. Na rozdíl od nástrojů typu Ride The Lightning nebo Thunderhub, které se starají o komplexní správu, se tyto

služby většinou zaměřují na určitou konkrétní činnost, jakou může být rebalancing kanálů, automatická úprava poplatků, otevírání více kanálů pomocí jedné transakce, reportování ziskovosti, správa likvidity apod. Některé se instalují přímo na váš uzel, jiné lze využívat jako externí službu.

Loop

Jedná se o službu od společnosti Lightning Labs, která funguje pouze s implementací LND a umožňuje provádět konverzi lightningových prostředků na on-chain (**Loop Out**), anebo naopak z bitcoinové adresy na Lightning (**Loop In**).

Loop Out se hodí primárně v případě, že jste někdo, kdo velmi často přijímá platby (např. obchodník podporující Lightning), a tudíž se vám stává, že po určitém čase je veškerá likvidita na vaší straně kanálu. Pomocí Loop Out si můžete všechny prostředky nebo jejich část přesunout například do coldstorage na hardwarovou peněženku a zároveň si znovu vytvořit likviditu pro příjem (reálně totiž dojde k odeslání lightningové transakce na uzel služby Loop, tudíž se likvidita přesune na druhou stranu). Dalším využitím může být rebalancing kanálů. Pokud otevřete kanál za účelem routování či přijímání plateb, zpočátku jsou veškeré prostředky na vaší straně a nemůžete tímto kanálem přijmout žádné bitcoiny. Pomocí Loop Out lze například polovinu takto směnit na on-chain bitcoin, tím si kanál vyvážit a směněné prostředky využít pro otevření kanálu dalšího.

Loop In je pravý opak. Odesláním bitcoinů na on-chain adresu dojde k jejich konverzi na Lightning a odeslání do vašeho uzlu. Získáte tím tedy likviditu pro odesílání satoshi a lze si takto kanál „dobít“, pokud jste již veškeré prostředky vyčerpali – odeslali na druhou stranu.

Jak lze očekávat, takováto služba není zdarma ani bez omezení:

Typ	Minimální částka*	Maximální částka*	Poplatky provozovateli
Loop Out	250 000 sat	120 000 000 sat	0,05 – 0,5 %
Loop In	250 000 sat	120 000 000 sat	0,1 – 1 %

*Minimální i maximální částka se může v průběhu času měnit, jejich aktuální hodnotu lze získat pomocí příkazu `loop terms`.

Co se týká používání, Loop si lze nainstalovat na svůj vlastní uzel pomocí démonu `loopd`, který se spojí s lokální instancí LND přes API. K samotnému používání služby přes příkazovou řádku poté slouží komponenta `loop`, pomocí které lze provádět jednotlivé činnosti (Loop In, Loop Out, dotaz na aktuální poplatek, monitoring probíhajících směn apod.).

```
$ loop out --channel 79xxx13 --amt 1100000 --conf_target 144 --max_swap_
routing_fee 2200 --verbose
Send off-chain:                          1100000 sat
Receive on-chain:                        1098638 sat

Estimated on-chain fee:                      146 sat
Loop service fee:                           1216 sat
Estimated total fee:                        1362 sat

No show penalty (prepay):                  30000 sat
Conf target:                               144 block
CLTV expiry delta:                           0 block

Publication deadline:                2022-02-14 16:08:14 +0100 CET

Max on-chain fee:                          36500 sat
Max off-chain swap routing fee:             2200 sat
Max off-chain prepay routing fee:             61 sat
```

V příkladu výše jsme započali Loop Out z kanálu s ID 79xxx13 o velikosti 1,1M satoshi. Na cílové přijetí si počkáme klidně až 144 bloků (tím se dá ušetřit na on-chain poplatcích) a za routování této platby chceme zaplatit maximálně 2 200 satoshi.

Alternativou je instalace Lightning Terminalu, kterou si probereme ihned v další podkapitole. Pokud vám webové rozhraní vyhovuje více než příkazová řádka, další možností je integrace `loopd` do nástroje Ride The Lightning, ze kterého lze jednotlivé činnosti provádět jednodušeji.

Celé řešení je trustless – pokud například provádíte Loop Out, vznikne on-chain transakce iniciovaná provozovatelem Loop služby, jejímž cílem je bitcoinový smart kontrakt s požadovanou částkou. Abyste si bitcoiny v něm uzamčené mohli vzít, potřebujete znát `payment_preimage`, které získáte, jakmile úspěšně proběhne lightningová platba. Pokud by cokoliv selhalo, bitcoiny budou po určitém čase navráceny zpět provozovateli služby Loop.

Kdyby vás zajímala reálná distribuce poplatků, pro Loop Out 1,1 milionu satoshi (za což jsme zaplatili celkem 3 513 satů) to může vypadat následovně:

- 1 216 satů (0,11 %) jako poplatek provozovateli služby Loop (viz tabulka výše).
- 2 162 satů (0,20 %) jako poplatek ostatním uzlům za routování platby na uzel Loop.
- 135 satů (0,01 %) jako on-chain poplatek za automatické vybrání bitcoinů ze smart kontraktu na svoji peněženku (SweepOut).

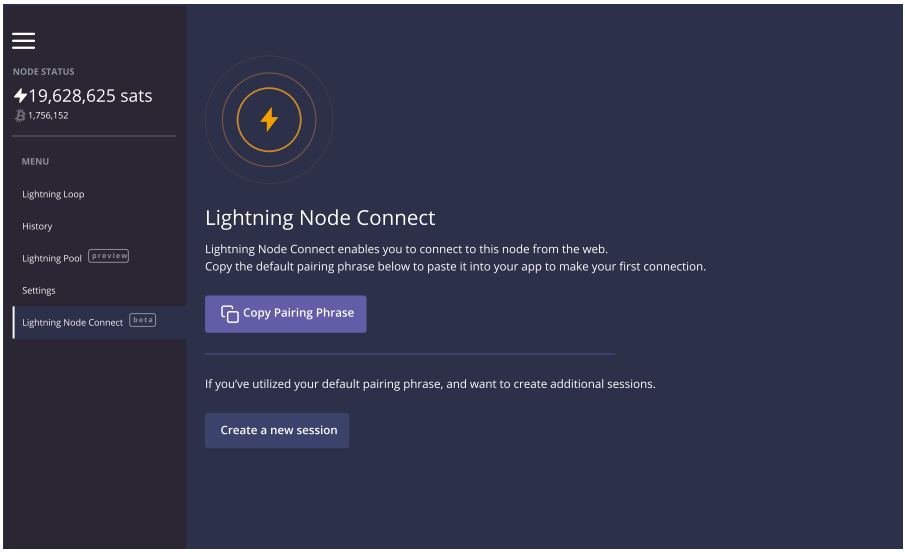
Celková směna z Lightningu na on-chain nás tedy stála v tomto případě 0,32 %. Jestli je to hodně nebo málo, si musí určit každý z vás. Tímto příkladem jsem chtěl demonstrovat, že se nesmí zapomenout na on-chain poplatky a poplatky za routování platby – tyto hodnoty většinou z marketingových důvodů na webu neuvidíte, tam se uvádí pouze poplatek provozovateli.

Na druhou stranu je fér říct, že tento Loop Out probíhal v době velmi nízkého vytížení bitcoinové sítě, a tudíž pokud ve vašem období nebude prázdný mempool, případně budete více spěchat, může se vám stát, že se dostanete na o něco horší čísla. Z toho, co jsem vypozoroval, tak poplatky se průměrně pohybují v rozmezí okolo 0,3 % až 0,7 %. Velmi drahý bývá právě poplatek za routování platby na Loop uzel, jelikož se jedná o známý liquidity sink, a tak si uzly účtují za přeposlání velmi vysoké poplatky.

Loop mi po technické stránce připadá jako dobrá služba, která funguje bezproblémově. Její nevýhodou jsou však vysoké poplatky, které ji činí téměř nepoužitelnou pro pravidelný rebalancing kanálů. Naopak pro obchodníka či firmu přijímající lightningové platby může být poplatek o výši okolo 0,5 % za uložení části přijatých bitcoinů do HW peněženky přijatelný.

Lightning Terminal

Lightning Terminal je aplikace, která se instaluje přímo na uzel a obsahuje v sobě již dříve představené tržiště Pool a službu Loop. Pokud tedy nejste fanoušci příkazové řádky a máte raději GUI, můžete tyto služby využívat z webového rozhraní Terminalu. Pro běh je potřebné mít na pozadí neustále spuštěný a správně nakonfigurovaný démon `litd`, pokud používáte Umbrel nebo nějaké jiné hotové řešení, častokrát je dostupný doslova na pár kliknutí.



První možností, jak se k Lightning Terminalu připojit, je přes webový prohlížeč přímo na vašem uzlu. Zde jsou možnosti omezené pouze na nástroje Pool, Loop, základní přehled kanálů a drobná nastavení. Mimo jiné si zde však můžete nakonfigurovat **Lightning Node Connect**. Jedná se o open-source nástroj pro end-to-end šifrované spojení mezi prohlížečem a mým uzlem, který může být ukrytý za Torem či NATem. Jinými slovy se jedná o autentizované spojení k mému uzlu odkudkoliv, kde mám přístup k internetu. Právě v tomto lokálním webovém rozhraní si mohu vygenerovat párovací frázi (10 slov), díky které lze toto spojení iniciovat.

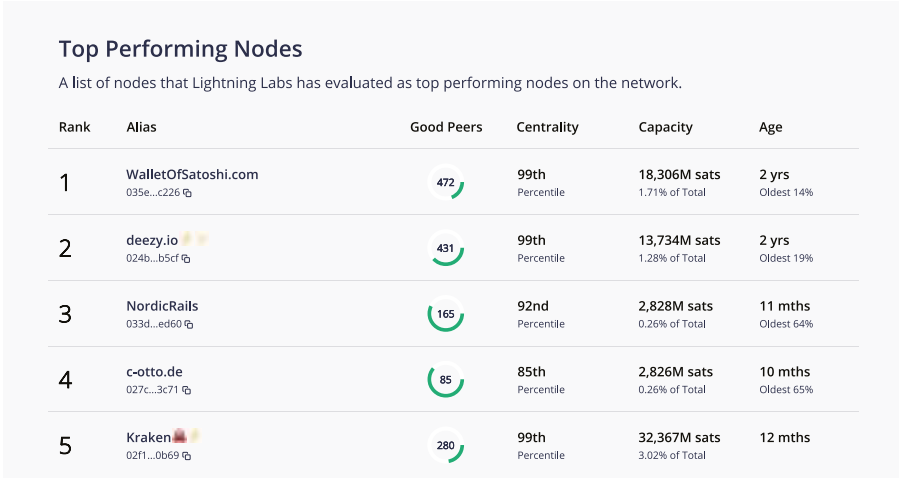
Pro vzdálený přístup je nutné navštívit stránku https://terminal.lightning.engineering a autentizovat se. Můj uzel pomocí volání API naváže šifrované spojení s proxy serverem, ke kterému se připojím webovým prohlížečem. V tomto druhém rozhraní je možností již podstatně více. Kromě klasických činností, jako je otevírání nových kanálů a jejich úpravy, zde mám detailní přehled o všech provedených platbách a přesměrovaných transakcích. Pokud jste tedy někde na cestách a chcete se podívat, jak se vašemu uzlu daří, je Lightning Terminal přes Lightning Node Connect zajímavou volbou.

Samotný Lightning Terminal provádí i jakési proprietární hodnocení jednotlivých uzlů. Ty jsou hodnoceny v těchto šesti kategoriích:

- Procento času, kdy je uzel online.
- Počet kanálů.
- Kapacita a likvidita v kanálech.
- Použitelnost kanálů.

- Stabilita kanálů.
- Počet kanálů s dobře propojenými uzly.

Záměrně nikde nejsou k dispozici přesné hodnoty, které musí daný uzel v těchto kritériích splňovat, aby síť nebyla absolutně unifikovaná. Pokud jich váš uzel splňuje alespoň 5 ze 6, je hodnocen jako „Stable“, pokud všechny, je hodnocen jako „Good“. Z těchto kritérií, do kterých se dále počítá i centralita uzlu, je poté sestaven žebříček.



Top Performing Nodes

A list of nodes that Lightning Labs has evaluated as top performing nodes on the network.

Rank	Alias	Good Peers	Centrality	Capacity	Age
1	WalletOfSatoshi.com 035e...c226	472	99th Percentile	18,306M sats 1.71% of Total	2 yrs Oldest 14%
2	deezy.io 024b...b5cf	431	99th Percentile	13,734M sats 1.28% of Total	2 yrs Oldest 19%
3	NordicRails 033d...ed60	165	92nd Percentile	2,828M sats 0.26% of Total	11 mths Oldest 64%
4	c-otto.de 027c...3c71	85	85th Percentile	2,826M sats 0.26% of Total	10 mths Oldest 65%
5	Kraken 02f1...0b69	280	99th Percentile	32,367M sats 3.02% of Total	12 mths

Spojení s těmito uzly by tedy mělo být jakousi zárukou, že se jedná o kvalitní uzel – jistí si tím však nikdy být nemůžete, a proto bych doporučoval si vždy uzel prověřit ještě jinde, než s ním navážete kanál. Zároveň platí, že pokud se váš uzel do tohoto seznamu dostane, tak se čas od času bude stávat, že ostatní budou navazovat kanály směrem k vám. Objevíte se totiž v Lightning Terminalu mezi doporučenými uzly.

Rebalance-LND

Rebalance-LND je pokročilý nástroj pro rebalancování, tedy odesílání plateb sám sobě tak, abych si v jednotlivých kanálech zajistil dostatek příchozí i odchozí likvidity. Ovládá se z příkazové řádky, a jak již název napovídá, je určen pouze pro implementaci LND.

Pomocí přepínače `-l` si mohu vypsat všechny kanály, které mají nedostatek odchozí likvidity a které by bylo vhodné vybalancovat:

```
$ ./rebalance.py -l
Channel ID:          784xxxxxxxxxxx625
Alias:               c-otto.de
Pubkey:              027xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxc71
Channel Point:       0d3xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx2ac:1
Local ratio:         0.123
Fee rates:           120ppm (own), 134ppm (peer)
Capacity:            2,000,000
Remote available:    1,730,325
Local available:     226,523
Rebalance amount:    751,901
```

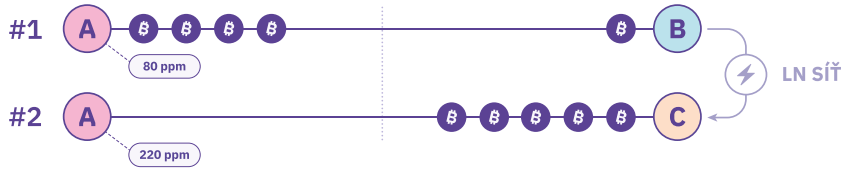
Zde vidíme, že jeden z našich kanálů má nedostatek odchozí likvidity (konkrétně pouze 226 523 satoshi, tedy 12 % kapacity), a tudíž je to vhodný kandidát pro rebalancing. Pomocí přepínače `-o` si naopak mohu vypsat ty kanály, které mají nedostatek příchozí likvidity. Nejjednodušší použití pro rebalancing je následující:

```
$ rebalance.py --amount 100000 --from 11111111 --to 22222222
```

Zde říkáme, že chceme odeslat 100 000 satoshi z našeho uzlu z kanálu `11111111` (tady si tedy zvýšíme příchozí likviditu) zpět sami sobě a přijmout transakci kanálem `22222222`, ve kterém nám tím pádem naroste odchozí likvidita. Je tady také možnost specifikovat pouze zdrojový/cílový kanál, v tom případě budou do páru vyzkoušeny všechny možnosti.

Pokud nebudu specifikovat ani částku, skript ji spočítá automaticky, a to tím způsobem, aby v každém kanálu bylo alespoň 1M satoshi jako odchozí i příchozí likvidita. Pokud by vám tato výchozí hodnota nevyhovovala, lze ji specifikovat pomocí `--min-local`, případně `--min-remote`. Další možností je využití přepínače `-A`, který nám zajistí, že se bude respektovat námi zvolená minimální příchozí/odchozí likvidita, i když pevně zvolíme danou částku k rebalancingu. Jinými slovy, pokud bychom skript zavolali s přepínači `--amount 50000 -A`, tak dojde k pokusu o rebalancing pouze v tom případě, že nebude dosaženo našich limitů definovaných parametry `--min-local`, resp. `--min-remote`. Tímto způsobem lze například postupně rebalancovat kanály po 50 000 satoshi ve smyčce, dokud nedosáhneme námi požadovaného rozložení likvidity.

Rebalance-LND vám nedovolí provádět balancing v případě, že by se to ekonomicky nevyplatilo. Co to znamená? Představme si následující dva kanály:



V tom prvním máme poplatky 80 ppm, ve druhém 220 ppm. Pro jednoduchost počítejme, že jedna kulička má hodnotu 1 milionu satoshi. Kdybychom v tomto případě chtěli rebalancovat 2M satoshi z kanálu #1 do kanálu #2, tak si tím v druhém kanálu zvýšíme odchozí likviditu a **čistě teoreticky** si můžeme v budoucnu vydělat až 440 satoshi, pokud by tímto kanálem prošla platba. Na druhou stranu ale tyto stejné 2M satoshi ztrácíme z kanálu #1 a zde **potenciálně přicházíme o zisk** v hodnotě 160 satoshi. Rebalance-LND určí, že je balancing ekonomicky výhodný pouze v případě, že součet potenciální ztráty (z kanálu, kterým odesíláme) a poplatků za rebalancing je menší než případný zisk. Jinak řečeno – nevyplatí se snažit se o perfektní balancing kanálů 50:50, když přesuneme prostředky z toho, který nám může vydělat velké množství satoshi do druhého, kde jsou naopak poplatky nízké. Právě tato funkcionalita je velmi užitečná, protože hlavně začátečníci často provádějí rebalancing, který se ale ekonomicky nevyplatí, a tak se dostávají celkově do ztráty. Důležité je také neměnit často poplatky pro jednotlivé kanály. K výpočtu, zda je daný rebalancing ekonomicky výhodný, totiž dochází v době spuštění příkazu. Pokud chvíli po rebalancingu kompletně změníme strukturu poplatků, nebude předchozí výpočet již platit.

Příklad úspěšného rebalancingu:

```
Trying route #1 (fee 2,500 mSAT, 49ppm)
793xxxxxxxxxxx377 to Alice                    (free, we usually charge 100ppm)
793xxxxxxxxxxx656 to Bob                      (fee    1,500 mSAT,    29ppm)
793xxxxxxxxxxx664 to muj-uzel                 (fee    1,000 mSAT,    20ppm)

Increased outbound liquidity on Bob (245ppm) by 50,000 sat
Increased inbound liquidity on Alice (100ppm configured for outbound)
Fee: 2 sats (2,500 mSAT, 49ppm)
Successful route:
793xxxxxxxxxxx377 to Alice                    (free, we usually charge 100ppm)
793xxxxxxxxxxx656 to Bob                      (fee    1,500 mSAT,    29ppm)
793xxxxxxxxxxx664 to muj-uzel                 (fee    1,000 mSAT,    20ppm)
     12,250 mSAT: expected future fee income for inbound channel (with Bob)
  -   2,500 mSAT: rebalance transaction fees
  -   5,000 mSAT: missing out on future fees for outbound channel (with Alice)
  =   4,750 mSAT: potential profit!
```

Jestliže bychom chtěli rebalancovat za každou cenu, můžeme použít `--reckless`, což způsobí, že na ekonomickou výhodnost nebude brán zřetel. V tomto případě můžeme poplatky za rebalancing limitovat pomocí `--fee-limit` nebo `--fee-ppm-limit`. Toto může být vhodné v případě, kdy v určitém kanálu chceme mít dostatek likvidity, nehledě na to, kolik nás to bude stát. Například nám tímto kanálem chodí naprostá majorita plateb do našeho e-shopu a poplatky za rebalancing nám vykompenzuje provize na prodaných produktech.

Příklad asi nejčastější chyby (Temporary channel failure – nedostatek likvidity v našem směru). V tomto případě je tento kanál po určitou dobu ignorován:

```
Trying route #2 (fee 3,150 mSAT, 62ppm)
795xxxxxxxxxxx913 to Martin                  (free, we usually charge 0ppm)
799xxxxxxxxxxx601 to Katerina                (fee    1,050 mSAT,    20ppm)
799xxxxxxxxxxx696 to David                   (fee       50 mSAT,     0ppm)
775xxxxxxxxxxx696 to Lucka                   (fee    1,050 mSAT,    20ppm)
793xxxxxxxxxxx664 to muj-uzel                (fee    1,000 mSAT,    20ppm)
Temporary channel failure
Ignoring 799xxxxxxxxxxx601 (Martin to Katerina)
```

Příklad stavu, kdy je zvolená cesta ekonomicky nevýhodná, a tak se neprovede:

```
Skipping route due to high fees (fee 4,550 mSAT, 90ppm)
793xxxxxxxxxxx377 to Alice                   (free, we usually charge 100ppm)
793xxxxxxxxxxx465 to Bob                     (fee    1,500 mSAT,    29ppm)
799xxxxxxxxxxx088 to Cyril                   (fee      950 mSAT,    18ppm)
700xxxxxxxxxxx145 to Daniela                 (fee    1,050 mSAT,    20ppm)
795xxxxxxxxxxx881 to muj-uzel                (fee    1,050 mSAT,    21ppm)
      5,000 mSAT: expected future fee income for inbound channel (with Daniela)
  -    4,550 mSAT: rebalance transaction fees
  -    5,001 mSAT: missing out on future fees for outbound channel (with Alice)
  =   -4,550 mSAT
Ignoring 793xxxxxxxxxxx377 (muj-uzel to Alice)
```

Podobných nástrojů pro rebalancing je více. Některé se ovládají z příkazové řádky, jiné jsou zakomponované do webových rozhraní (Ride The Lightning, Thunderhub). Rebalance-LND nabízí aktuálně asi největší možnosti (pro kompletní výčet funkcí navštivte GitHub repozitář tohoto projektu), proto je zde uveden jako příklad.

Charge-LND

Charge-LND je nástroj pro úpravu poplatků u jednotlivých kanálů na základě předem definovaných kritérií. Stejně jako rebalance-LND je určen pouze pro implementaci LND a ovládá se z příkazové řádky.

Základem tohoto skriptu je konfigurační soubor, ve kterém definujeme libovolný počet politik. Každá politika se skládá z následujících sekcí:

- **Název politiky**. Libovolný text ohraničený v hranatých závorkách.

- **Podmínky**. Skupina kritérií, vůči kterým je každý kanál porovnán. Charge-LND v tomto ohledu nabízí velmi široké množství podmínek – minimální odchozí likvidita, poměr příchozí a odchozí likvidity, stáří kanálu, ID kanálu, počet kanálů protistrany, samotná aktivita tohoto kanálu, jako je počet přeposlaných HTLC, hodnota přesměrovaných transakcí a mnoho dalšího. Tyto podmínky lze samozřejmě kombinovat.

- **Strategie a hodnoty**. Zde nastavujeme, jakou strategii pro naše poplatky chceme uplatnit, a jejich konkrétní hodnotu (jak `base_fee`, tak `fee_rate`, ale i další parametry).

Po vytvoření tohoto konfiguračního souboru a spuštění skriptu se vezme první kanál v pořadí a porovnává se s jednotlivými politikami odshora dolů (takže u nich záleží na pořadí), dokud u některé nevyhovují nastavené podmínky. Jakmile se taková politika nalezne, aplikují se na ni definované poplatky. Následně se identicky postupuje pro další kanál v pořadí. Je možné definovat speciální politiku s názvem `[default]`, která se bude aplikovat, pokud by žádné podmínky nevyhovovaly.

Před samotným produkčním nasazením tohoto skriptu je vhodné jej spustit s přepínačem `--check`, který zkontroluje syntaxi konfiguračního souboru. Velmi užitečným přepínačem je také `--dry-run`, který na standardní výstup vypíše veškeré změny, jež by skript provedl, ale reálně je nenastaví. Toto doporučuji používat zpočátku a do ostrého režimu se přepnout až v případě, že si budete konfigurací jistí.

Na GitHubu tohoto projektu lze nalézt kompletní dokumentaci a řadu ukázek. Příklady níže berte spíše jako obecný přehled syntaxe a možností, rozhodně je nenasazujte do produkce, protože z ekonomického hlediska nedávají většinou smysl.

```
# vice jak 90 % kapacity kanalu je u protistrany
# nastavime tedy vyssi poplatky
[vetsina_u_protistrany]
chan.max_ratio = 0.1
strategy = static
base_fee_msat = 1000
fee_ppm = 300

# na me strane kanalu je alespon 1M satoshi
# opet zvolime staticke poplatky
[milion_local_plus]
chan.min_local_balance = 1_000_000
strategy = static
base_fee_msat = 1000
fee_ppm = 150

# kanaly, kterymi jsem odeslal alespoň 10M satoshi za poslednich 7 dni
[preroutovano_alespon_10M]
chan.activity_period = 7d
chan.min_sats_out = 10_000_000
strategy = static
base_fee_msat = 1000
fee_ppm = 450

# lze si definovat kanaly, ktere mam otevrene s kamarady a pro ne aplikovat
jinou strategii
# jejich seznam, oddeleny carkami, muze byt vyjmenovany zde (snizime pro ne navíc
cltv_expiry_delta)
[kamaradi]
node.id = file:///home/chargelnd/friends_list.txt
strategy = static
base_fee_msat = 0
fee_ppm = 10
time_lock_delta = 20

# poplatky lze nastavit i proporcionalne podle aktualni vyvazenosti kanalu
# dale zde spcecifikujeme, ze minimalni hodnota upravy fee_rate je 10
[podle_likvidity]
strategy = proportional
base_fee_ppm = 1000
min_fee_ppm = 100
max_fee_ppm = 300
min_fee_ppm_delta = 10

# urcite kanaly lze naprosto ignorovat (napriklad si u nich chceme poplatky
nastavovat rucne)
[ignored_channels]
node.id = 02da8d5a759ee864438da617cfdb61c87f723fb76c4b6371b877d0347abe953a4f
strategy = ignore

# vychozi politika, pokud by se zadna nechytila
# zde nastavujeme i minimalni velikost preposilaneho HTLC na 200 000 satoshi
[default]
strategy = static
base_fee_msat = 1000
fee_ppm = 177
min_htlc_msat = 200000000
```

Existují i další nástroje pro automatické úpravy poplatků na základě předem definovaných podmínek. Je na každém z vás zjistit, jaký vám bude vyhovovat.

Podobné skripty je vhodné automatizovaně spouštět například jedenkrát denně či týdně. Osobně bych se ale vyhnul častým a agresivním změnám poplatků. Jednak to může učinit ekonomické výpočty rebalance-LND (pokud jej používáte) neefektivními, a zároveň je vhodné určitou dobu počkat, než budu vyhodnocovat, zda úprava poplatků přinesla kýžený efekt. Určitou dobu totiž trvá, než se tato informace rozšíří do celé sítě gossip protokolem a ostatní uzly vás začnou využívat. Pokud jde o mne, jsem tedy zastáncem spíše pomalejší a postupnější úpravy poplatků, kde jako hlavní faktor beru dlouhodobé statistiky přeposílání plateb tímto kanálem – hledání maximálního možného poplatku, který jsou ostatní ještě ochotní zaplatit.

Balance of Satoshis

Nástroj Balance of Satoshis (dále jen zkráceně bos) by se dal popsat jako švýcarský nůž pro práci s LND. Oproti předchozím nástrojům, které se zaměřovaly na rebalancing kanálů nebo úpravu poplatků, tedy na konkrétní činnost, umí bos tak nějak všechno. Jedná se tedy o nástroj, který rozšiřuje možnosti samotného LND a některé činnosti zásadně usnadňuje. Ovládán je opět pouze z příkazové řádky, díky čemuž lze jeho výstupy používat v různých dalších skriptech.

Po zadání `bos help` se nám vypíše nápověda a seznam všech použitelných komponent. Následně se s jednotlivými komponentami pracuje příkazem bos `<nazev_komponenty> <pripadne_argumenty>`. Pokud byste potřebovali více informací, lze zavolat příkaz `bos help <nazev_komponenty>`. Aktuálně Balance of Satoshis obsahuje 49 komponent a seznam se postupně rozrůstá o další. My se v následující tabulce podíváme na několik nejzajímavějších, abyste získali základní představu o tom, co tento nástroj nabízí. Pro kompletní dokumentaci vás opět odkážu na GitHub tohoto projektu.

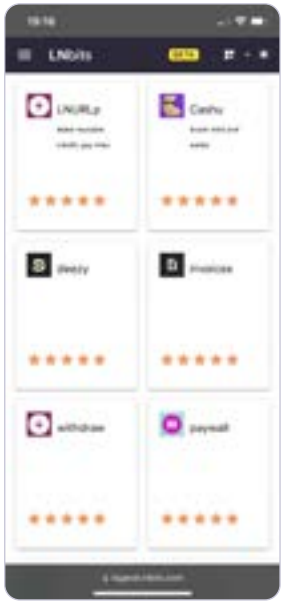
Komponenta	Popis
`accounting`	Umí vygenerovat a zobrazit různé reporty a statistiky. Například všechny zaplacené on-chain poplatky, přijaté či odeslané platby, faktury, detaily o přesměrování plateb apod. Statistiky lze omezit na určité časové období.
`chain-deposit`	Vygeneruje novou on-chain adresu pro příjem bitcoinů včetně QR kódu.
`chart-chain-fees`	Zobrazí statistiku a graf, kolik jste zaplatili na on-chain poplatcích za dané období. Spadá sem tedy primárně otevírání a zavírání kanálů, případně SweepOut HTLC.
`chart-fees-earned`	Zobrazí statistiku a graf, kolik jste získali za přesměrování plateb za určité období.

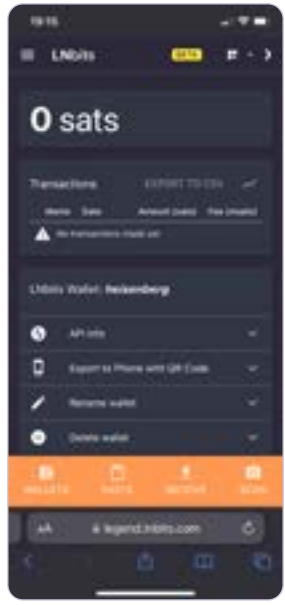
chart-fees-paid	Zobrazí statistiku a graf, kolik jste zaplatili na off-chain poplatcích za dané období, například za rebalancing. Tyto 3 komponenty s předponou chart lze tedy využívat, abyste si spočítali, zda jste celkově v plusu nebo mínusu.
clean-failed-payments	Smaže záznamy o všech neúspěšných transakcích.
fees	Umožňuje zobrazovat a upravovat poplatky pro jednotlivé kanály.
graph	Umožňuje zobrazit k zadanému uzlu (podle aliasu nebo ID) veškeré jeho kanály a jejich detaily. Jedná se tedy o velmi zjednodušenou podobu lightningových vyhledávačů typu Amboss.Space nebo 1ML, kterou lze volat přímo z vašeho uzlu.
inbound-channel-rules	Umožňuje definovat pravidla, která specifikují, kdo k vašemu uzlu může navázat platební kanál. Kromě nejčastější podmínky na minimální kapacitu kanálu zde lze definovat i například celkovou kapacitu protistrany přes všechny její veřejné kanály apod. Dále je možné nastavit zprávu, která se odešle v případě odmítnutí.
limit-forwarding	Lze limitovat, za jakých okolností budete přeposílat transakce, anebo přesměrování plateb úplně zakázat.
open	Umožňuje otevírat nové kanály, dokonce několik najednou v rámci jedné on-chain transakce (tudíž lze ušetřit na poplatcích), a využít externí peněženku jako vstup do těchto transakcí. Funguje to tak, že zadáte veškeré detaily nově otevíraných kanálů (tedy vždy ID uzlu a kapacitu) a začne vám běžet desetiminutové okno. V něm si musíte otevřít externí peněženku a najít funkcionalitu PayToMany (pokud je to podporováno). Sem vložíte výstup příkazu **bos open**, což je vlastně seznam on-chain adres a množství bitcoinů. Dále si nastavíte požadované on-chain poplatky a zprávu podepíšete. V žádném případě ji ale nesmíte z externí peněženky odeslat do sítě (publikovat), to za vás udělá bos. Pouze takto podepsanou raw transakci překopírujete zpět do terminálu s bos, a pokud to vše stihnete do 10 minut, začnou se kanály otevírat v rámci jedné transakce.
open-balanced-channels	Umožňuje otevírat dual-funded kanály, které jsou zpočátku vybalancované a na jejichž celkovou kapacitu se skládají obě strany. Je potřeba být na této činnosti domluven s protistranou (existuje určité pořadí příkazů, které obě strany musí dodržet) a podporovat funkcionalitu keysend, skrze kterou jsou vyměněny detaily kanálu.
probe	Umožňuje prozkoumat likviditu ostatních kanálů tím, že odešlu smyšlené HTLC a zjistím, jestli daná částka těmito kanály projde. Jedná se vlastně o implementaci Channel Probingu popsaného již v teoretické části knihy.
rebalance	Široké možnosti rebalancování kanálů.
send	Umožňuje odesílat keysend platby s mnoha různými nastaveními.
utxos	Zobrazí seznam všech vašich UTXO v LND on-chain peněžence.

LNBits

LNBits je webový nástroj pro správu lightningových peněženek s mnoha rozšířeními. Lze jej nainstalovat na svůj uzel jako samostatnou aplikaci, v tomto případě se pro backend může využít LND, Core Lightning a další. Druhou možností je používat LNBits, hostované u třetí strany pouze jako uživatel. Poté se ale jedná o plně custodial řešení se všemi riziky.

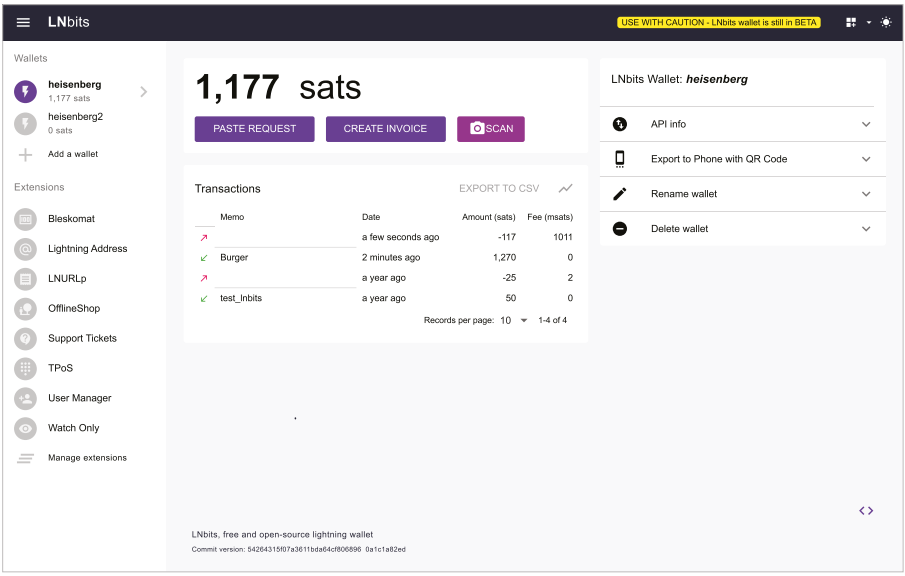
Při prvním načtení webové stránky zadám libovolné jméno mého účtu, a tím se mi na této konkrétní instanci vytvoří nová peněženka. Aktuálně je důležité si zapamatovat URL z prohlížeče, jelikož LNBits zatím nemá implementováno klasické přihlašování jménem a heslem. K návratu ke svým prostředkům slouží právě URL, kterou z pochopitelných důvodů není vhodné s nikým sdílet. Uložte si ji tedy například do záložek.





Na této nově vzniklé peněžence nebudu mít z počátku žádné satoshi. Oproti novému uzlu je ale mohu ihned začít přijímat, klasicky pomocí vygenerování faktury. Jako likvidita se totiž využívají kanály backendu, na který je LNBits napojeno. Uživatel peněženky nemá možnost kanály vytvářet a spravovat jejich likviditu, v tomto je plně odkázán na uzel, na kterém LNBits běží. Jakmile nějaké satoshi přijmete, můžete je identicky odeslat, opět se využívají kanály backendu. **Reálně totiž bitcoiny přijímá lightningový uzel**, LNBits si pouze pamatuje, kolik prostředků

vlastní ten který účet. Zároveň se uživatelé dostanou pouze k těm prostředkům, které sami přijali – nikoliv k prostředkům ostatních (leda by znali jejich URL), ani ke všem bitcoinům, které jsou uloženy na lightningovém uzlu. Z toho plyne hlavní riziko, pokud používám LNBits instanci hostovanou u třetí strany – správce uzlu může s mými prostředky dělat naprosto cokoliv, jelikož se jedná o nižší vrstvu a má k nim přístup. Pokud tedy chcete LNBits reálně používat, vlastní instance je velmi doporučována.



Díky tomu, že vše běží v prohlížeči, je extrémně jednoduché onboardovat nové uživatele. Stačí jim vytvořit peněženku, nahrát na ni nějaké satoshi a odeslat jim odkaz. Ten je možné otevřít klidně i z mobilního telefonu, jelikož design je plně responsivní. Jedná se poté o jakousi webovou mobilní peněženku. Každý uživatel má možnost si vytvořit více peněženek a prostředky si tak logicky oddělovat. Pokud plánujete LNBits využívat produkčně, bylo by vhodné, aby toto rozhraní bylo dostupné z internetu – některé funkcionality jako například LNURL to vyžadují.

LNBits má v sobě velké množství takzvaných rozšíření, které si uživatelé mohou do svých instancí přidávat a využívat je. Tato rozšíření přidávají do LNBits další funkcionalitu, a kdokoliv má zájem, může vytvářet vlastní. Na několik nejzajímavějších se nyní podíváme více do detailu.

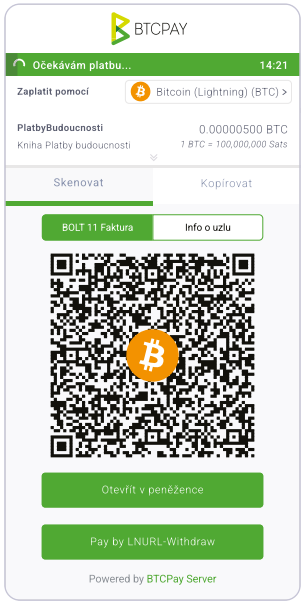
Komponenta	Popis
User Manager	Umožňuje spravovat uživatele – přidávat je, vytvářet jim peněženky, odesílat jim na ně odkazy apod. Ideální, pokud chcete spravovat peněženky své rodině či známým.
TPoS	Umožňuje vytvořit webovou stránku s plnohodnotným platebním terminálem, který umí přijímat lightningové platby. Ty poté putují do mé LNBits peněženky. Je možné vygenerovat odkaz, díky kterému lze tento terminál otevřít například na tabletu a používat TPoS jako způsob příjmu satoshi v obchodě.
Watch Only	Po zadání xpub (hlavní veřejný klíč) lze generovat adresy pro příjem on-chain bitcoinů do mé externí peněženky.
Support Ticket	Toto rozšíření umožňuje vytvořit webový support portál, kde se za jednotlivé dotazy platí pomocí Lightningu. Jakmile uživatel napíše dotaz, je vyzván k zaplacení drobného poplatku (buď pevný nebo podle délky dotazu). Po úspěšné platbě mi přijde notifikace (například na Discord pomocí webhook) s dotazem. Tyto dotazy si projdu a odpovědi poté přijdou zpět uživateli.
SatsPayServer	Jedná se o velmi jednoduchou verzi BTCPay Serveru (popsaný v následující kapitole). Toto rozšíření tedy vygeneruje stránku, kde uživatel musí do určité doby zaplatit fakturu za zakoupené zboží. Lze integrovat s e-shopem.
Paywall	S pomocí tohoto rozšíření můžete schovat určitou stránku (přesněji odkaz na ni) za paywall. Uživatelé tedy musí zaplatit lightningovou fakturu, aby byli na vaši stránku přesměrováni. Rozšíření si pamatuje, pokud již někdo zaplatil a přistupuje podruhé.
LndHub	Umožňuje propojit LNBits a mobilní BlueWallet peněženku.
LNURLp	Implementace LNURL-payRequest (popsané v teoretické části). Platba na statický QR kód.
LNURLw	Implementace LNURL-withdrawRequest (popsané v teoretické části). Příjem satoshi vyfocením QR kódu.
SpotifyJukebox	Umožňuje vytvořit webovou aplikaci s playlistem ze Spotify, kde se za každou spuštěnou písničku bude platit pomocí Lightningu. Vhodné například jako alternativa do baru.
OfflineShop	Velmi zajímavé rozšíření umožňující prodávat fyzické produkty pomocí Lightningu a být přitom offline. V nastavení rozšíření si nejprve definuji produkty, které prodávám, a jejich cenu. Následně si vytvořím seznam několika slov (například 20), které si zapamatuji nebo vytisknu. Ke každému produktu si vytisknu QR kód. Jakmile jej zákazník načte a zaplatí pomocí Lightningu, jeho peněženka mu zobrazí první slovo ze seznamu. Když je správné, vím, že zaplatil. Dalšímu zákazníkovi se zobrazí další slovo a takto celé opakovaně. Díky správným slovům lze tedy ověřovat, zda zákazníci opravdu zaplatili. Technicky se využívá LNURL-payRequest, kde jednotlivá slova (důkazy) jsou odesílány v rámci Success Message. Lze implementovat i s TOTP kódy (Google Authenticator a jiné), které mohu mít zobrazeny například na tabletu.

V rámci LNBits je samozřejmě jednotlivých rozšíření mnohem více a neustále přibývají nová. Ke své vlastní peněžence i všem výše popsaným rozšířením existuje dobře zdokumentované API – celé LNBits lze tedy ovládat z jiného skriptu či programu a využít tak funkcionality LNBits včetně jednotlivých rozšíření ve svých vlastních aplikacích. Pokud byste si LNBits chtěli vyzkoušet, lze využít Legend Demo server běžící na oficiálních stránkách https://lnbits.com.

BTCPay Server

Pokud ve svém obchodě plánujete přijímat platby bitcoinem a nechcete se spoléhat na třetí stranu, můžete k tomu využít nástroj BTCPay Server. Jedná se o open-source procesor pro příjem plateb kryptoměnami. Nejčastěji se používá jako platební brána v rámci integrace s e-shopem, případně jako terminál v kamenném obchodě. Uživateli je při placení zobrazeno okno, ve kterém vidí cílovou částku a případné další detaily o nákupu. Pokud to konfigurace umožňuje, tak zde má možnost výběru mezi klasickým on-chain bitcoinem nebo Lightning Network. V obou případech stačí vyfotit zobrazený QR kód (případně rovnou otevřít peněženku, pokud je provozována na stejném zařízení) a po zaplacení je platba je automaticky zprocesována.

Možnosti BTCPay Serveru jsou velmi široké, a jestli uvažujete o jeho využití, odkázal bych vás na rozsáhlou dokumentaci. My se v této kapitole stručně podíváme na technickou stránku související s Lightning Network.

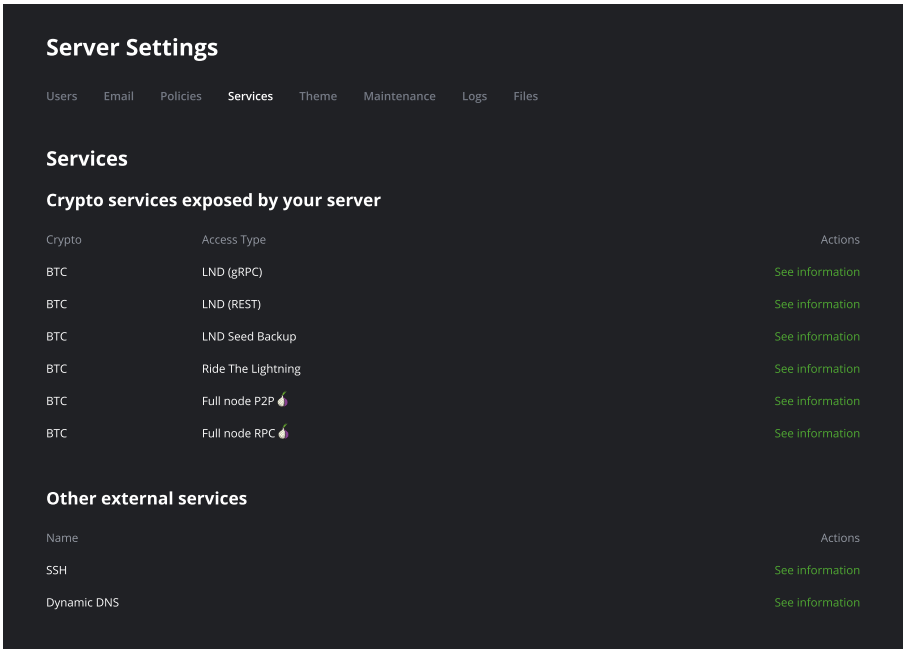


Existuje nespočet možností, jak tento software provozovat. Lze ho vybudovat doslova na pár kliknutí v cloudu (Luna-Node, Azure, Google Cloud apod.), můžete si ho ručně nainstalovat na svůj vlastní fyzický server, Raspberry Pi či na pronajatý virtuální server. Často bývá součástí hotových řešení pro provoz lightningového uzlu (Umbrel, myNode, Nodl atd.) a v neposlední řadě jej lze využívat tak, že infrastruktura bude nainstalována u třetí strany a vy k němu budete mít pouze uživatelský přístup do webového rozhraní.

BTCPay Server je balík nástrojů pro příjem plateb kryptoměnami, který se skládá z vašeho vlastního bitcoinového fullnodu, block exploreru (NBXplorer), databáze a samotného softwaru BTCPay. Pro příjem plateb v rámci Lightningu je na výběr aktuálně LND, Core

Lightning i Eclair. Dále zde můžete doinstalovat různé další nadstavby (tzv. additional fragments), např. Thunderhub, Ride The Lightning, Lightning Terminal, BlueWallet LNDHub apod. Jedná se tedy o podobné řešení, jakými jsou již předpřipravené implementace lightningového uzlu (Umbrel, Citadel, myNode, RaspiBlitz…), ale zde konkrétně se zaměřením na příjem plateb. Instalace je většinou vcelku jednoduchá, veškeré nástroje jsou již předkonfigurované a vy si pouze vyberete ty, o které máte zájem.

Co se týká samotného Lightningu, tak po instalaci BTCPay Serveru si musíte nejprve otevřít nějaké platební kanály a zajistit likviditu, abyste mohli přijímat platby. Zároveň je poté nutné průběžně udržovat dostatek příchozí likvidity, aby transakce neselhávaly. Pro otevírání kanálů a jejich případný balancing lze využít nadstaveb, které jsou přístupné v sekci Services.



Mimo využití lightningové implementace, která je součástí BTCPay serveru, se můžete připojit i k externímu uzlu – to může být vhodné v případě, že již provozujete dobře napojený uzel. Pokud byste chtěli vaši implementaci Lightningu, která je součástí BTCPay, vyměnit během jeho používání za jinou, tak je to možné, pouze je nutné pozavírat všechny kanály a přesunout si prostředky dočasně jinam. Následně budete s budováním kanálů a likvidity začínat od nuly. Poslední věcí, na kterou bych chtěl upozornit, je fakt, že pokud máte nainstalovaný BTCPay Server u třetí strany a využíváte jej pouze jako uživatel, tak veškeré lightningové platby jdou do

peněženky provozovatele. Ten je potom musí spravedlivě rozdělit mezi jednotlivé uživatele. Pro využívání Lightningu je tedy vlastní instance BTCPay téměř nutností.

BTCPay Server je oblíbený nástroj pro příjem plateb kryptoměnami. Nabízí mnoho pluginů pro integraci s jednotlivými e-shopy, a i když to zabere nějaký čas, je to mnohonásobně snadnější, než si platební bránu vytvářet kompletně od základů. Jen bych zde chtěl upozornit, že pokud se rozhodnete podporovat i Lightning, neobejdete se bez pravidelné správy vašeho uzlu, otevírání kanálů a zajišťování likvidity.

Lightning Polar

Lightning Polar je nástroj pro simulaci kompletní lightningové sítě na vašem počítači. Umožňuje vám na několik kliknutí rozběhnout Bitcoin Core jako backend a na něj připojit libovolné množství lightningových uzlů. Pojďme se rovnou podívat na příklad.

Na obrázku výše je vidět vytvořená lightningová síť, která sestává z jednoho Bitcoin Core a 6 lightningových uzlů (2x LND, 2x Core Lightning a 2x Eclair). Mezi jednotlivými uzly si můžete libovolně otevírat platební kanály, u kterých je v grafické podobě vidět aktuální likvidita. Vzhledem k tomu, že všechny uzly běží v Dockeru a celá síť na Bitcoin regtestu, je sestavení podobného schématu otázkou pár minut.

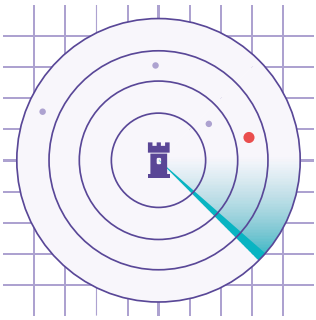
```
$ docker ps --format „{{.Image}}\t\t {{.Names}}“
polarlightning/eclair:0.6.1                 polar-n1-erin
polarlightning/eclair:0.6.1                 polar-n1-frank
polarlightning/clightning:0.10.0            polar-n1-dave
polarlightning/clightning:0.10.0            polar-n1-carol
polarlightning/lnd:0.14.1-beta              polar-n1-bob
polarlightning/lnd:0.14.1-beta              polar-n1-alice
polarlightning/bitcoind:22.0                polar-n1-backend1
```

Bitcoin regtest

Jedná se o mód Bitcoinu, ve kterém je vytvořen lokální privátní blockchain, který není s nikým sdílen, a můžete si upravovat jeho parametry, jak potřebujete. Například těžit jednotlivé bloky pouze na jedno kliknutí a se získanými bitcoiny libovolně nakládat. Využívá se primárně pro vývoj a testování nástrojů, u kterých není potřeba komunikace s ostatními uzly. Vytěžené bitcoiny samozřejmě nemají žádnou cenu.

Grafické prostředí nástroje Lightning Polar vám na několik kliknutí umožňuje těžit bitcoiny, přidávat další uzly, otevírat kanály, vytvářet a platit faktury apod. Můžete si zde otevřít terminálové okno a pracovat s danou implementací napřímo z příkazové řádky. Pokud vyvíjíte nějaký nástroj, rozhodně uvítáte možnost se na jednotlivé instance připojit pomocí různých síťových rozhraní. Uzly lze restartovat, vypínat, odebírat a podobných sítí můžete mít více. Jedná se o povedený nástroj, jenž využijí spíše pokročilí uživatelé, kteří si z určitého důvodu chtějí nasimulovat vlastní síť s využitím různých lightningových implementací.

Watchtowers



Jedná se o službu, která poskytuje podporu v případě, že by operátor uzlu, s kterým máme navázaný platební kanál, publikoval neplatnou (revokovanou) commitment transakci v období, kdy náš uzel z nějakého důvodu **nebude online**, a tudíž se nebudeme moci bránit. Watchtower neustále monitoruje blockchain, a pokud v něm detekuje revokovanou commitment transakci, zareaguje publikováním penalty transakce, která odešle celkovou kapacitu kanálu poškozené straně.

Jde o architekturu typu **server/klient**, kdy serverem je samotná Watchtower služba, která hlídá publikování neplatných commitment transakcí pro své klienty, lightningové uzly. Watchtower může být provozována v rámci lightningového uzlu anebo ji lze nainstalovat na dedikovaný server. Tato služba je anonymní, jelikož watchtower nemusí znát o svých klientech žádné informace. Technicky to funguje tak, že s každou úpravou kanálu by měl lightningový uzel odeslat potřebné informace své watchtower. Těmito informacemi jsou lokátor, jak detekovat neplatnou commitment transakci v blockchainu, a zašifrovaná penalty transakce. Právě až po publikování neplatné commitment transakce si může watchtower rozšifrovat penalty transakci a odeslat ji do bitcoinové sítě. Protože penalty transakce potřebuje správný podpis a ani nemůže být publikována dříve, watchtower si nemůže vaše prostředky nijak přivlastnit.

Pokud máte na svém uzlu větší obnos prostředků, je vhodné mít alespoň jednu aktivní watchtower, ideálně více. Nikdy totiž nevíte, co se s vaším uzlem, elektřinou či internetovým připojením stane v době, kdy zrovna budete na dovolené na druhé straně zeměkoule. Pro produkční využití je vhodné mít záložní zdroje napájení, případnou alternativní konektivitu a odzkoušený automatický start veškerých programů na uzlu po výpadku. Nikdy si ale nemůžete být jistí, že nějaká komponenta neselže. Krátkodobé výpadky jsou akceptovatelné, jelikož aby reálně došlo k okradení, museli byste být offline déle, než je hodnota `CSV_delay` (relativní časový zámek pro příjem prostředků `to_local` při publikování commitment transakce), což je obvykle nastaveno na nižší jednotky dnů a zároveň by se protistrana musela pokusit o podvod.

Každá implementace Lightningu si watchtowers aktuálně řeší po svém a my se nyní podíváme, jak je to prakticky řešeno u dvou nejpoužívanějších.

LND

Od verze 0.7.0 je součástí LND jak watchtower server (samotná služba), tak i klient, který se připojuje na jiné watchtowers. Není tedy potřeba instalace žádného dalšího separátního nástroje.

Pro využití **serverové části** je nutné mít LND zkompilované se subserverem `watchtowerrpc` a poté do konfigurace přidat `watchtower.active=1`. Následně se můžeme podívat na podrobnější informace příkazem:

```
$ lncli tower info
{
        „pubkey“:
„03281d603b2c5e19b8893a484eb938d7377179a9ef1a6bca4c0bcbbfc291657b63“,
        „listeners“: [
                „[::]:9911“
        ],
        „uris“: [

„03281d603b2c5e19b8893a484eb938d7377179a9ef1a6bca4c0bcbbfc291657b63@1.2.3.4:9911“
        ]
}
```

Zde vidíme, že watchtower poslouchá na všech rozhraních a výchozím portu TCP/9911. Řetězec v poli `uris` můžeme sdílet ostatním, jelikož jej budou potřebovat, aby naši watchtower mohli využívat – zde použité ID je odlišné od ID uzlu. Samotná databáze zneplatněných commitment transakcí (`watchtower.db`) může objemově velmi rychle narůstat a velikost v řádu GB není po delší době provozu výjimkou. Pokud umožníte využívat vaši watchtower celému internetu, vezměte tuto skutečnost v potaz. Co se týká **klientské části**, kterou využijete spíše než tu serverovou, tak je nutné nejprve přidat do konfigurace LND řádku `wtclient.active=true`. Následně si můžeme přidávat nové watchtowers následujícím příkazem:

```
$ lncli wtclient add
03281d603b2c5e19b8893a484eb938d7377179a9ef1a6bca4c0bcbbfc291657b63@1.2.3.4:9911
{

}
```

Zda je konkrétní watchtower aktivní a zda od nás přijímá informace o revokovaných commitment transakcích, zjistíme tak, že parametr `active_session_candidate` má hodnotu `true` při výpisu podrobností:

```
$ lncli wtclient tower
03281d603b2c5e19b8893a484eb938d7377179a9ef1a6bca4c0bcbbfc291657b63
{
        „pubkey“:
„03281d603b2c5e19b8893a484eb938d7377179a9ef1a6bca4c0bcbbfc291657b63“,
        „addresses“: [
                „1.2.3.4:9911“
        ],
        „active_session_candidate“: true,
        „num_sessions“: 1,
        „sessions“: []
}
```

Důležitým parametrem nacházejícím se v konfiguraci LND je `wtclient.sweep-fee-rate`, který definuje, jaký je maximální poplatek za penalty transakci. Ve výchozím stavu je nastaven na 10 sat/vB. Sami zvažte, zda nechcete tento parametr navýšit pro případ, že by k „záchraně“ od watchtower došlo ve chvíli, kdy bude mempool extrémně plný. Obecně je dobré mít watchtowers více, jelikož velké množství z nich je aktivních pouze krátkou dobu (pravděpodobně kvůli rychle rostoucí velikosti `watchtower.db`). Zároveň platí, že najít kvalitní watchtower, která je provozována dlouhodobě zdarma, není jednoduché.

Core Lightning

Jak už je u Core Lightningu zvykem, i watchtower je zde řešena skrze samostatný plugin.

V případě **serverové části** je využíván nástroj **The Eye of Satoshi (TEOS)**, což je implementace watchtower serveru napsaná v jazyce Python, jež je v souladu s navrhovaným standardem BOLT#13, který se právě nasazením watchtowers zabývá. Pro spuštění je nutný Bitcoin Core jako backend a po instalaci máme k dispozici samotného démona `teosd`, který by měl neustále běžet na pozadí, a k němu příslušný binární soubor `teos-cli`, pomocí něhož watchtower můžete ovládat. Například informace o vaší aktuálně běžící watchtower lze zjistit příkazem `teos-cli get_tower_info`. Oproti LND se tedy jedná o samostatný program, který není napřímo součástí Core Lightningu.

Pro **klientskou část** je nutné mít aktivovaný plugin s názvem `watchtower client`. Zaregistrování nové watchtower probíhá následovně:

```
$ lightning-cli registertower
0230053e39c53b8bcb43354a4ed886b8082af1d1e8fc14956e60ad0592bfdfab51@1.2.3.4:9814
{
   „available_slots“: 10000,
   „public_key“:
„02d5001fb4204fd9ab26f06c7869e3820906be565c5b449ecfb0251af4eff1c752“,
   „subscription_expiry“: 4753
}
```

Zde je vidět, že máme k dispozici 10 000 „slotů“ pro úpravu commitment transakce, a registrace k této watchtower nám vyprší za 4 753 bloků, tedy přibližně za měsíc. Poté je nutné watchtower obnovit, může se tedy jednat o placenou službu. Po aktivaci tohoto modulu se do všech registrovaných watchtowers odesílají informace o zneplatněných commitment transakcích automaticky. Pro vypsání všech registrovaných watchtowers můžeme využít následující příkaz:

```
$ lightning-cli listtowers
{
   „towers“: [
      {
         „id“:
„0230053e39c53b8bcb43354a4ed886b8082af1d1e8fc14956e60ad0592bfdfab51“,
         „netaddr“: „1.2.3.4:9814“,
         „status“: „unreachable“,
         „available_slots“: 9998,
         „pending_appointments“: [
            „fff5ceaff045b0e4ddf11936d66c02a0“,
            „0698bd6c0e1f74c11857b99ef6009156“
         ],
         „invalid_appointments“: []
      }
   ]
}
```

Ve výstupu vidíme, že zaregistrovaná watchtower je aktuálně nedostupná, zbývající počet slotů a dvě zneplatněné commitment transakce (zde zvané `appointments`), které čekají na odeslání, jakmile bude watchtower opět online.

Zálohování a obnova

Z klasického on-chain Bitcoinu jsme zvyklí, že vše podstatné je uloženo v rámci blockchainu – kolik vlastníme bitcoinů, veškeré naše transakce a podobně. Pokud tedy o svůj fullnode přijdeme, stačí mít poznamenaný způsob, jak se dostat ke svým privátním klíčům, a všechny tyto informace si znovu stáhneme od ostatních uzlů.

V Lightningu to ale tak jednoduché není. Jelikož zde již neplatí, že o všech transakcích vědí všichni, je zodpovědnost za správné zálohování na vás. I když je běžná realita taková, že rozhodně ne každý zálohuje pravidelně veškerá svá osobní data, tak u Lightningu to je **naprostou nutností**. V případě ztráty dat, ať již softwarovou či hardwarovou chybou, případně externím přičiněním, jako je krádež či požár, můžeme totiž nenávratně přijít o veškeré své prostředky na našem uzlu. Zálohování by mělo být ideálně automatické a běžet na pozadí, protože u manuálních záloh se dříve nebo později stane, že na ně člověk zapomene. Zároveň je více než vhodné tyto zálohy přesouvat na vzdálené zařízení, které je ideálně umístěno v jiné lokalitě. Přesun záloh do jiné složky na tom samém zařízení opravdu není zálohou. V následujících kapitolách si projdeme zálohování i obnovu podle různých kategorií a implementací. Budou zde popsány obecné postupy, pro konkrétní návod příkaz po příkazu se rozhodně podívejte vždy do dokumentace. Různé hotové lightningové platformy si mohou řešit zálohování odlišně, například níže popsané body mohou být seskupeny pod jedno tlačítko ve webovém rozhraní dané platformy.

Konfigurace

Začneme možná malinko netradičně, a to zálohou konfigurace. Tím se myslí záloha konfigurace veškerých nástrojů, které na uzlu používám – bitcoinový backend, implementace Lightningu, různé další webové nadstavby a nástroje. I když v případě ztráty těchto dat nepřijdu o žádné prostředky, rozhodně ztratím čas jejich novou konfigurací. Právě na zálohu konfigurace se často zapomíná, přitom bohatě stačí vybrané konfigurační soubory pravidelně přesouvat na vzdálené úložiště. Jen si dejte pozor na to, kdyby v nich byly uloženy nějaké citlivé údaje. S tím vám pomůže například komprimovaný a šifrovaný archiv, který bude vznikat jedenkrát denně. Nejenomže se ke konfiguračním souborům nedostane nikdo nepovolaný, ale zároveň budete mít možnost vrátit se ke konkrétnímu datu.

On-chain (LND)

Součástí každého lightningového uzlu je i on-chain peněženka, ve které máme bitcoiny pro otevírání kanálů. Při vytváření nového LND uzlu se nám zobrazí klasických 24 slov (**seed phrase**) a možnost k nim přidat cipher seed passphrase (zjednodušeně řečeno „25. slovo“). Pozor na to, že i když to na první pohled vypadá stejně jako téměř všem známá **BIP39 mnemonic seed phrase** (a dokonce se používají stejná slova), tak se jedná o jiný formát zvaný **aezeed cipher seed**. I když cílem je to stejné, tedy vygenerování hlavního klíče, tak interně je tento formát odlišný. Umožňuje verzování, uložení data vzniku peněženky, rozpozná špatnou passphrase, kterou si lze dokonce v průběhu používání změnit. Pro zálohu je tedy potřeba si těchto 24 slov zapsat (nikoliv digitálně, však to již znáte) a zapamatovat

passphrase. Při obnově formou založení nového uzlu se z výše uvedených informací budou generovat bitcoinové adresy a LND se bude dívat, zda se na nich nenachází nějaké prostředky. Ty bych měl po určité době vidět ve své peněžence.

On-chain (Core Lightning)

U Core Lightningu je to odlišné. Zde je nutné zálohovat soubor zvaný `hsm_secret`. Jedná se o binární soubor velikosti 32 bajtů, takže ideálně si jeho obsah zobrazte v šestnáctkové soustavě a zapište mimo uzel, podobně jako seed phrase. Ani zde nedochází v průběhu času ke změnám obsahu, takže postačí jedna záloha. Pro obnovu on-chain prostředků obsah tohoto souboru uložím pod názvem `hsm_secret` a nastavím správná oprávnění (linuxově 0400), bez kterých Core Lightning nenastartuje. Pokud máte odjinud BIP39 mnemonic seed phrase, můžete jej pomocí nástroje `hsmtool generatehsm` převést na `hsm_secret`.

Off-chain (LND)

Nyní se dostáváme k záloze a obnově prostředků uložených v platebních kanálech, tedy v adresách typu multisig 2 ze 2. Zde LND využívá takzvaných **Static Channel Backups** (dále jen **SCB**). V nich jsou uložené informace o všech vašich kanálech, jakými jsou funding transakce, celková kapacita, veřejný klíč (ID) protistrany, poslední známé síťové adresy uzlů a podobně. Rozhodně v nich **nenajdete aktuální stavy kanálů**, tedy jaká část kapacity patří té které straně. S každým otevřením kanálu je přidán záznam do SCB, s uzavřením je naopak vymazán. V průběhu životnosti platebního kanálu se v něm informace neupravují. SCB jsou šifrované pomocí klíče, který je odvozen ze seed phrase, tedy je jednak bezpečné tuto zálohu přesouvat na vzdálené uložiště (například cloud), pokud věříte onomu šifrování, a zároveň to zajišťuje, že nikdo nemůže bez znalosti seed phrase vaši zálohu použít a obnovit na svém uzlu. Doporučovanou metodou je tedy SCB (konkrétně soubor `channel.backup`) neustále monitorovat a přesouvat na vzdálené úložiště při každé změně jeho obsahu.

Naopak zálohovat aktuální stav kanálů, konkrétně soubor `channel.db`, vývojáři LND nedoporučují. Pokud máte aktivní uzel o mnoha kanálech a pravidelně rebalancujete na pozadí, tak není výjimkou, že se vám budou přidávat/odebírat HTLC (tudíž měnit commitment transakce zapsané v `channel.db`) i několikrát za vteřinu. A zde je riziko toho, že i kdybyste zálohovali třeba každých 5 vteřin, tak záloha nemusí být aktuální. Tím pádem by mohlo dojít při obnově k publikování neplatné (revokované) commitment transakce, takže k pokusu o podvod, jehož následkem je ztráta všech bitcoinů z kanálu. Pokud tedy nejste extrémně technicky

zdatní a nedokážete zajistit, že budete mít vždy naprosto poslední zálohu souboru `channel.db`, tak stavy kanálů takto nezálohujte a využijte doporučovaného SCB.

Obnova probíhá tak, že předám nově vzniklému uzlu SCB zálohu (`channel.backup`) a on se pokusí se všemi uzly spojit (proto si ukládal poslední známé síťové adresy). Těm oznámí, že došlo k obnově uzlu a má se aktivovat protokol **Data Loss Protection (DLP)**, který je součástí specifikace BOLT#2. Ten spočívá v tom, že protistrana ihned publikuje poslední commitment transakci, čímž dojde k nucenému uzavření kanálu a prostředky, které byly na naší straně, bychom po určité době měli mít zpět ve své on-chain peněžence. U obnovy pomocí SCB tedy **vždy dojde k uzavření kanálů**, a tudíž dochází pouze k obnově prostředků, nikoliv kanálů jako takových.

Možná vás teď napadne, že v tomto případě vlastně spoléháme na poctivost protistrany v tom smyslu, že bude publikovat poslední commitment transakci, a nikoliv nějakou, která by pro ni byla výhodnější. Ano, je tomu tak. Protistraně se ale většinou podvádět nevyplatí, protože si nemůže být jistá, zda nemáme z minula nastavenou watchtower nebo zda ztrátu poslední commitment transakce jen nepředstíráme. Případná ztráta veškerých prostředků za to většinou nestojí. Jinak řečeno – abychom kvůli konceptu SCB přišli o nějaké bitcoiny, tak bychom museli ztratit aktuální databázi kanálů, využívat obnovy pomocí SCB, mít co do činění s nepoctivou protistranou, a taková protistrana by musela být ochotna vsadit všechny své prostředky na to, že nelžeme a opravdu aktuální commitment transakci nemáme.

Z důvodů uvedených výše je tudíž vhodné se zbavovat zombie (neaktivních) kanálů, jelikož ty nelze pomocí SCB obnovit. Zároveň bych zde chtěl upozornit na fakt, že vinou uzavírání kanálů není SCB vhodný nástroj pro migraci na jiný uzel.

Off-chain (Core Lightning)

Jak již je zvykem, Core Lightning řeší zálohování jinak. Existuje zde několik způsobů a my se nyní teoreticky podíváme na ty aktuálně nejvíce doporučované.

První možností záloh jsou **dvě aktuální instance databáze** kanálů. Toto řešení funguje pouze v případě, že máte jako backend databázi SQLite3, což je výchozí varianta Core Lightningu. Pro konfiguraci pouze stačí pozměnit parametr `wallet` a za dvojtečku přidat cestu k druhé databázi.

```
wallet=sqlite3:///home/user/.lightning/bitcoin/lightningd.sqlite3:/my/
backup/lightningd.sqlite3
```

Doporučuje se druhou databázi pojmenovat identicky, tedy `lightningd.sqlite3`. Reálně tedy při každém uložení nového stavu kanálu dojde k současnému zápisu do obou databází. Základním předpokladem je, že se druhá databáze nachází na vzdáleném úložišti, které je k vašemu uzlu připojené skrze NFS či obdobný protokol. Záloha není nikterak šifrovaná, proto se nedoporučuje využívat například veřejný cloud. U této varianty je nutné zajistit spolehlivý zápis na cílové zařízení – pokud by totiž tato vrstva selhala, byla by záloha neaktuální a mohlo by dojít ke ztrátě všech prostředků. Pro obnovu pouze překopírujte databázi ze vzdáleného úložiště a přejmenujte na `lightningd.sqlite3`, pokud se jmenovala jinak. Při této obnově nedochází k zavírání kanálů.

Druhou možností je využít pluginu **backup**, který je napsaný v jazyce Python a technicky je velmi podobný řešení uvedenému výše. Opět je podporován pouze SQLite3 backend, zálohy jsou nešifrované a je vhodné mít ke svému uzlu připojený vzdálený filesystem. Plugin se poté stará o průběžné kopírování databáze na druhou lokaci pomocí `db_write` (hook, který je zavolán pokaždé, když dojde k zápisu do databáze). Pro prvotní konfiguraci je potřeba vypnout Core Lightning, provést jednorázovou tvorbu zálohy a pak po specifikaci tohoto modulu v konfiguračním souboru poběží záloha automaticky na pozadí. Pro obnovu se používá binární soubor `backup-cli` s přepínačem `restore`, který je součástí backup pluginu.

Core Lightning nabízí i další pokročilé možnosti, mezi které můžeme zařadit redundanci na úrovni filesystemu (hardwarový RAID, softwarové řešení na bázi BTRFS, ZFS apod.). Dále zkušení uživatelé, kteří využívají jako backend databázi PostgreSQL, mohou využít provozování této databáze v clusteru, kde dochází k automatické replikaci na ostatní nody, a je zde zaručen failover mód, pokud by jeden node v rámci clusteru selhal.

Připojení peněženky na vlastní uzel

Pokud provozujete vlastní uzel, možná vám bude dávat smysl jej používat i pro své osobní lightningové transakce. Zdaleka uživatelsky nejpřívětivějším způsobem je použití mobilní peněženky, která umožňuje se na váš uzel napojit a využívat ho. Tímto řešením totiž lze plně využívat možností vlastního uzlu i na cestách, případně v obchodech. Nejste poté odkázání na custodial peněženky, kde bitcoiny reálně nevlastníte, ani na non-custodial aplikace, u kterých naopak často bývá problém se správou kanálů a nedostatečnou likviditou. Díky připojení mobilní peněženky na váš uzel můžete pohodlně **využívat veškerých prostředků na uzlu**, a hlavně vašich platebních kanálů. Pokud jich máte více a mají dostatečnou likviditu, nebudete mít problémy ani s většími transakcemi. Zároveň tato varianta nabízí oproti jiným řešením maximální soukromí.

Na druhou stranu ale je férové přiznat, že toto řešení je použitelné jen pro minoritu uživatelů Lightningu. Využijí jej pouze velmi zkušení uživatelé, kteří provozují vlastní lightningový uzel a chtějí ho využívat i pro své osobní platby. Naprostá většina ostatních uživatelů se poté rozdělí mezi custodial a non-custodial řešení, kdy jim postačí si stáhnout vybranou mobilní peněženku a žádný uzel na separátním zařízení neprovozovat.

Existuje řada peněženek, které do určité míry umožňují napojit se na vlastní uzel. Příkladem může být vcelku známá peněženka BlueWallet, kterou lze mimo základního custodial módu provozovat i s připojením na vlastní uzel skrze LndHub wrapper. Dalšími příklady mohou být peněženky Zap, Zeus a jiné. A právě na Zeus se podíváme v následující podkapitole. Neznamená to, že by tato peněženka byla bezkonkurenční, ale aktuálně patří rozhodně k tomu lepšímu, co lze na trhu najít. Na praktickém příkladu si tedy ukážeme, jak takovouto peněženku připojit na váš LND uzel, a co vše tím získáte. Způsob samotného napojení a možnosti budou u konkurenčních aplikací velmi podobné, vyberte si tedy tu variantu, která bude nejvíce vyhovovat vám.

Zeus

Zeus je mobilní peněženka, která je dostupná jak v iOS, tak v Android verzi, a je uzpůsobena pro správu vlastního uzlu. Jinak řečeno, bez vlastního uzlu ji **nelze provozovat**. Umožňuje připojení jak k LND, Core Lightning, tak i Eclair implementaci, což je zajisté její velká výhoda. V tomto příkladu si ukážeme připojení k LND. To je prakticky řešeno přes API, je tedy nutné zajistit síťovou komunikaci mezi mobilním telefonem a vaším uzlem, konkrétně s REST API, které bývá velmi často z bezpečnostních důvodů nastaveno pouze na localhost rozhraní. Existuje několik možností, jak tuto komunikaci realizovat:

- Nejméně bezpečnou variantou je povolit přístup k tomuto rozhraní do celého internetu. Budete potřebovat veřejnou IP adresu a případně, pokud provozujete uzel za NATem, nakonfigurovat také port-forwarding.

- Druhou možností je vystavit REST API pouze do interní sítě. V tomto případě můžete peněženku používat jen na identické síti, nejčastěji z domova. Existuje zde ještě možnost řešení s využitím VPN spojení do stejné sítě, kde běží váš uzel. Takto lze váš uzel spravovat bezpečně a odkudkoliv, pouze je nutné se nejprve připojit na VPN.

- Poslední možností je využít Tor Hidden Service, pokud váš uzel komunikuje i přes tento protokol.

Zeus je částečně integrován do hotových platforem typu Nodl, myNode, RaspiBlitz či Umbrel, s nimiž bude připojení snadnější. I když je přístup k API autentizovaný ve formě macaroon souborů (jeho obsah v hexadecimálním formátu musíte vložit při konfiguraci spojení), dobře si rozmyslete, která varianta je pro vaše využití nejlepší. Osobně bych API rozhraní do celého internetu vystavovat nedoporučoval a využil například VPN nebo Tor. Dále je vhodné provádět validaci certifikátů, abyste odhalili případné MiTM útoky.

Po úspěšném připojení můžete provádět většinu úkonů jako z plnohodnotných rozhraní typu Ride The Lightning nebo ThunderHub. Lze samozřejmě odesílat a přijímat platby (Lightning i on-chain), je zde podpora pro pokročilé funkce, jakými jsou AMP, MPP, LNURL i keysend. Mimo jiné zde můžete vidět přehled všech vašich platebních kanálů, včetně detailů, upravovat routovací poplatky a případně otevírat kanály nové. Samozřejmostí je i přehled přesměrovaných plateb za určité období včetně kompletního detailu veškerých vašich činností a transakcí. Pokud tedy hledáte jednu aplikaci, která poslouží jako kvalitní mobilní peněženka a zároveň vám naplno umožní spravovat váš uzel, může být Zeus vhodnou volbou.

ZÁVĚR A PODĚKOVÁNÍ

Veškeré hlavní myšlenky o přínosu Lightning Network již byly zmíněny v závěru základní části knihy, takže bych se zde pouze opakoval. Pokud jste celou knihu dočetli až sem, měli byste mít slušné obecné povědomí o teoretickém fungování Lightningu i o jeho praktickém využití. Nebyl zde však prostor veškerá specifika probrat do naprostého detailu, jinak by tato kniha měla mnohonásobně více stránek. Další samostudium již tedy bude na vás. Lightning je technologie, která se vyvíjí velmi rychle, takže je stejně potřeba neustále sledovat novinky a případné změny. Jako výchozí bod můžete využít následující kapitolu, ve které jsou uvedeny veškeré zdroje, ze kterých bylo čerpáno.

Zároveň mi zde dovolte poděkovat několika lidem. V první řadě mé drahé manželce za to, že se se mnou po dobu psaní této knihy a neustálého studia a testování funkcionalit Lightningu, které na několik měsíců vyplnily veškerý můj volný čas, nerozvedla. Dále bych chtěl také poděkovat celému týmu Braiins publishing za veškerou pomoc při vydávání této knihy. A v neposlední řadě také děkuji následujícím lidem (seřazeno abecedně), kteří mi poskytli důležitou zpětnou vazbu a doplňující návrhy či opravy před samotným vydáním této knihy:

- Dominik Stroukal
- Filip Kodýtek
- František Šimek
- Jáchym Černý
- Jan Dvořák
- Jan Karas
- Jan Veselý
- Miloš Bém

ZDROJE

Zde uvádím v abecedním pořadí všechny zdroje, které jsem při studiu problematiky a psaní této knihy používal. Jelikož je fenomén Lightning Network velice mladou záležitostí, jedná se výlučně o on-line články, studie a weby samotných služeb spojených s Lightningem. Zájemce, kteří stojí o to si některé adresy rozkliknout a nastudovat, odkazuji na web `www.platbybudoucnosti.cz/kniha-zdroje`, kde naleznou zdroje v uživatelsky přívětivé podobě odkazů. Zdroje na webu budu navíc průběžně aktualizovat.

1. 1ML – Lightning Network Search and Analysis Engine
   https://1ml.com
2. A Byte's Journey; Current State of Lightning Network Privacy
   https://abytesjourney.com/lightning-privacy
3. Alza; Michal Mikle; myNode
   https://www.alza.cz/mynode-bitcoin-lightning-full-node-recenze-zkusenosti
4. Amboss Space – Lightning Network Explorer
   https://amboss.space
5. Bicoin Magazine; First-ever Dual-funded mainnet Lightning Channel opened
   https://bitcoinmagazine.com/technical/first-dual-funded-lightning-channel-opens
6. Bitcoin Magazine; Explaining Basis of Lightning Technology (BOL
   https://bitcoinmagazine.com/technical/explaining-bolt-12
7. Bitcoin Magazine; Good Griefing: A lingering vulnerability on Lightning Network that still needs fixing
   https://bitcoinmagazine.com/technical/good-griefing-a-lingering-vulnerability-on-lightning-network-that-still-needs-fixing
8. Bitcoin Optech; Channel jamming attacks
   https://bitcoinops.org/en/topics/channel-jamming-attacks
9. Bitcoin Optech; Point Time Locked Contracts (PTLC)
   https://bitcoinops.org/en/topics/ptlc
10. Bitcoin.org; Running a Full Node
    https://bitcoin.org/en/full-node
11. Blockstream; Core Lightning
    https://blockstream.com/lightning
12. Blockstream; eltoo: A Simple Layer2 Protocol for Bitcoin
    https://blockstream.com/eltoo.pdf
13. Blockstream; PeerSwap
    https://blockstream.com/assets/downloads/2021-11-16-PeerSwap_Announcement.pdf
14. BlueWallet – Bitcoin wallet and Lightning wallet for iOS and Android
    https://bluewallet.io
15. Bolt.fun; The HODL Invoice
    https://bolt.fun/guide/invoices/hodl-invoice
16. BOLT12; Offers: Lightning's Native Experience, Everywhere
    https://bolt12.org

17. Bottlepay; Lightning Node Performance: Testing TPS
https://bottlepay.com/blog/bitcoin-lightning-node-performance

18. BTCPay Server
https://btcpayserver.org

19. BTC-slovník; Bitcoinový slovník naučný
https://btc-slovnik.cz

20. Builder's Guide; Builder's Guide to the LND Galaxy!
https://docs.lightning.engineering

21. Citadel
https://runcitadel.space

22. Coindesk; Ready to Wumbo: LND Enables More, Larger Bitcoin Transactions on Lightning
https://www.coindesk.com/tech/2020/08/20/ready-to-wumbo-lnd-enables-more-larger-bitcoin-transactions-on-lightning

23. Cointelegraph; What is the Lightning Network in Bitcoin, and how does it work?
https://cointelegraph.com/bitcoin-for-beginners/what-is-the-lightning-network-in-bitcoin-and-how-does-it-work

24. Coldbit; What types of mnemonic seeds are used in Bitcoin?
https://coldbit.com/what-types-of-mnemonic-seeds-are-used-in-bitcoin

25. Derp Turkey; Revocable transactions with LN-Penalty
https://www.derpturkey.com/revocable-transactions-with-ln-penalty

26. Dr. Carsten Otto; Osobní stránky
https://c-otto.de

27. Fanis Michalakis; What are Anchor Outputs
https://fanismichalakis.fr/posts/anchor-outputs

28. Fanis Michalakis; What Are Hosted Channels
https://fanismichalakis.fr/posts/what-are-hosted-channels

29. Fanis Michalakis; What Are PTLCs
https://fanismichalakis.fr/posts/ptlcs

30. Fanis Michalakis; What Are Turbo Channels
https://fanismichalakis.fr/posts/turbo-channels

31. Gerg Walker; Learn Me a Bitcoin
https://learnmeabitcoin.com

32. Github; Accumulator; Charge-LND
https://github.com/accumulator/charge-lnd

33. Github; ACINQ; eclair
https://github.com/ACINQ/eclair

34. Github; Alex Bosworth; Balance of Satoshis
https://github.com/alexbosworth/balanceofsatoshis

35. Github; Alex Bosworth; Run LND
https://github.com/alexbosworth/run-lnd

36. Github; Andreas M. Antonopoulos; Mastering Bitcoin
https://github.com/bitcoinbook/bitcoinbook

37. Github; Andreas M. Antonopoulos; Mastering the Lightning Network
https://github.com/lnbook/lnbook

38. Github; Apotdevin; ThunderHub – Lightning Node Manager
https://github.com/apotdevin/thunderhub

39. Github; BIP173 – Bech32
https://github.com/bitcoin/bips/blob/master/bip-0173.mediawiki#Bech32

40. Github; Bitcoin Transcripts; 2022-03-01 Lightning Network Panel
https://github.com/bitcointranscripts/bitcointranscripts/blob/master/london-bitcoin-devs/2022-03-01-lightning-panel.md

41. Github; BlueWallet; LndHub
https://github.com/BlueWallet/LndHub

42. Github; Breez; Breez Mobile Client
https://github.com/breez/breezmobile

43. Github; C-Otto; Rebalance-LND
https://github.com/C-Otto/rebalance-lnd

44. Github; ElementsProject; Backing Up Your C-Lightning Node
https://github.com/ElementsProject/lightning/blob/master/doc/BACKUP.md

45. Github; ElementsProject; Core Lightning (CL)
https://github.com/ElementsProject/lightning

46. Github; Fiatjaf; LNURL Documents
https://github.com/fiatjaf/lnurl-rfc

47. Github; Jamaljsr; Polar
https://github.com/jamaljsr/polar

48. Github; Lightning Labs; Lightning Terminal
https://github.com/lightninglabs/lightning-terminal

49. Github; Lightning Network; Aezeed
https://github.com/lightningnetwork/lnd/tree/master/aezeed

50. Github; Lightning Network; Issue 5594 – BOLT12
https://github.com/lightningnetwork/lnd/issues/5594

51. Github; Lightning Network; LND
https://github.com/lightningnetwork/lnd

52. Github; Lightning Network; Private Altruist Watchtowers
https://github.com/lightningnetwork/lnd/blob/master/docs/watchtower.md

53. Github; Lightning Network; Recovering Funds From lnd
https://github.com/lightningnetwork/lnd/blob/master/docs/recovery.md

54. Github; Lightning transactions: From Zero to Hero
https://github.com/t-bast/lightning-docs/blob/master/lightning-txs.md

55. Github; Lightning; Basis of Lightning Technology (BOLT)
https://github.com/lightning/bolts

56. Github; Lightningd; A simple and reliable backup plugin
https://github.com/lightningd/plugins/tree/master/backup

57. Github; LNBits; LNBits Leged
https://github.com/lnbits/lnbits-legend

58. Github; Niteshbalusu11; Balance of Satoshis Comands
https://github.com/niteshbalusu11/BOS-Commands-Document

59. Github; Ride The Lightning
https://github.com/Ride-The-Lightning/RTL

60. Github; Rootzoll; RaspiBlitz
https://github.com/rootzoll/raspiblitz

61. Github; Sr-gi; BOLT13
https://github.com/sr-gi/bolt13/blob/master/13-watchtowers.md

62. Github; Talaia Labs; The Eye of Satoshi (TEOS)
https://github.com/talaia-labs/python-teos

63. Github; T-Bast; Lightning Documentation
https://github.com/t-bast/lightning-docs

64. Github; Zeus
https://github.com/ZeusLN/zeus

65. ION Lightning Network Wiki
https://wiki.ion.radar.tech/tech/bitcoin

66. Jameson Lopp Blog; Lightning Liquidity Management Guide
https://blog.lopp.net/lightning-network-liquidity-management-guide

67. Jamespon Lopp; Lightning Network Resources
https://www.lopp.net/lightning-information.html

68. Kevin Rooke; Lightning Commerce Is Coming
https://www.kevinrooke.com/post/lightning-commerce-is-coming

69. Lightning Hood; Lightning Resources
https://lightninghood.com/lightning-resources

70. Lightning Labs
https://lightning.engineering

71. Lightning Labs; Announcing lnd 0.11-beta: Let's Get Ready to Wumbo!
https://lightning.engineering/posts/2020-08-20-lnd-v0.11

72. Lightning Labs; Announcing lnd 0.12-beta: Improving the LND Developer Experience
https://lightning.engineering/posts/2021-01-28-lnd-v0.12

73. Lightning Labs; Loop
https://lightning.engineering/loop

74. Lightning Labs; Pool
https://lightning.engineering/pool

75. Lightning Network Developers; LND Overview and Developer Guide
https://dev.lightning.community/overview

76. Lightning Network Plus Blog; LND: Tor & Clearnet – How to setup hybrid-mode
https://lightningnetwork.plus/posts/137

77. Lightning Network Plus Blog; Start accepting Bitcoin Lightning Network payments as a business or as a freelancer
https://lightningnetwork.plus/posts/90

78. Lightning Network Plus
https://lightningnetwork.plus

79. Lightning Node Info; Set up a Watchtower and a Client on the Lightning Network
https://www.lightningnode.info/advanced-tools/watchtower

80. LightningDev Mailing List; An update on PTLCs
https://lists.linuxfoundation.org/pipermail/lightning-dev/2020-April/002659.html

81. LightningDev Mailing List; Anchor Outputs Spec & Implementation Progress
https://lists.linuxfoundation.org/pipermail/lightning-dev/2020-March/002607.html

82. LN.Guide; A beginners guide to the Lightning Network
https://bitcoiner.guide/lightning

83. LN-Big
https://lnbig.com

84. LNBits
https://lnbits.com

85. LnRouter; Lightning Payment Speed 2022
https://blog.lnrouter.app/lightning-payment-speed-2022

86. LnRouter; Liquidity Ad Marketplace
https://lnrouter.app/liquidity-ads

87. LnRouter; Terminal Score Debugger
https://lnrouter.app/scores/terminal

88. Medium; ACINQ; Adding “Data loss protection“ to eclair
https://medium.com/@ACINQ/adding-data-loss-protection-to-eclair-598c62494096

89. Medium; Blockstream; c-lightning Opens First Dual-Funded Mainnet Lightning Channel!
https://medium.com/blockstream/c-lightning-opens-first-dual-funded-mainnet-lightning-channel-ada6b32a527c

90. Medium; Blockstream; Setting Up Liquidity Ads in c-lightning
https://medium.com/blockstream/setting-up-liquidity-ads-in-c-lightning-54e4c59c091d

91. Medium; Fulgur Ventures; An Overview of Lightning Network Implementations
https://medium.com/@fulgur.ventures/an-overview-of-lightning-network-implementations-d670255a6cfa

92. Medium; Roy Sheinfeld; The Breez Release Candidate: Getting Lightning Ready for the Global Takeover
https://medium.com/breez-technology/the-breez-release-candidate-getting-lightning-ready-for-the-global-takeover-b5d1f9756229

93. Medium; Steven Ellis; How to Bootstrap a Profitable Lightning Node
https://medium.com/@steven_75507/how-to-bootstrap-a-profitable-lightning-node-8de72beac59c

94. Medium; Steven Ellis; How to Migrate LND From myNode or Umbrel to RaspiBolt
https://medium.com/@steven_75507/how-to-migrate-lnd-from-mynode-to-raspibolt-ubuntu-server-e6089a92eae7

95. Medium; StopAndDecrypt; Running Bitcoin & Lightning Nodes Over The Tor Network (2021 Editio
https://stopanddecrypt.medium.com/running-bitcoin-lightning-nodes-over-the-tor-network-2021-edition-489180297d5

96. MobyCrypt; Turn Your Self-hosted Lightning Node to Public in 10 minutes
https://www.mobycrypt.com/turn-your-self-hosted-lightning-network-node-to-public-in-10-minutes

97. Muun blog; Turbo Channels
https://blog.muun.com/turbo-channels

98. myNode
https://mynodebtc.com

99. Nodl
https://www.nodl.it

100. Phoenix FAQ
https://phoenix.acinq.co/faq

101. Polar – Regtest Lightning Networks
https://lightningpolar.com

102. Random Things; Guide To Networking for LND
https://random.engen.priv.no/archives/574

103. RaspiBolt
https://raspibolt.org

104. Rene Pickhardt; Osobní stránky
http://www.rene-pickhardt.de

105. River Financial; Point Time Locked Contract (PTLC)
https://river.com/learn/terms/p/point-timelocked-contract-ptlc

106. Sergei Tikhomirov; Lightning Probing 2
https://s-tikhomirov.github.io/lightning-probing-2

107. Sergei Tikhomirov; Lightning Probing
https://s-tikhomirov.github.io/2021-03-27-lightning-probing

108. Stack Exchange; Bitcoin – What are the downsides of Hodl Invoices?
https://bitcoin.stackexchange.com/questions/91168/what-are-the-downsides-of-hodl-invoices

109. Start9; Embassy
https://start9.com/latest

110. SuredBits; Payment Points – Part 2: "Stuckless" Payments
https://suredbits.com/payment-points-part-2-stuckless-payments

111. SuredBits; Payment Points Part 1: Replacing HTLCs
https://suredbits.com/payment-points-part-1

112. SuredBits; PTLC Proof of Concept
https://suredbits.com/ptlc-proof-of-concept

113. SuredBits; Schnorr Applications: Scriptless Scripts
https://suredbits.com/schnorr-applications-scriptless-scripts

114. The Block; Lightning Labs activates Wumbo channels, increasing bitcoin payment channel capacity
https://www.theblockcrypto.com/linked/75485/lightning-labs-wumbo-channels-bitcoin-payment-channels

115. Trezor Hardware Wallet
https://trezor.io

116. Umbrel – a personal server for everyone
https://getumbrel.com

117. Voltage blog; How Taproot improves the Lightning Network
https://blog.voltage.cloud/how-taproot-improves-the-lightning-network

118. Voltage Blog; Keysend Payments explained
https://voltage.cloud/blog/bitcoin-lightning-network/keysend-payments-explained-voltage-technical-series

119. Wallet Of Satoshi
https://www.walletofsatoshi.com

120. Wikipedia; Lightning Network
https://en.wikipedia.org/wiki/Lightning_Network

121. Wikipedia; SegWit
https://en.wikipedia.org/wiki/SegWit

122. YouTube; Aantop; Bitcoin Q&A: Eltoo, and the Early Days of Lightning
https://youtu.be/o6eFZ5aI9N0

123. YouTube; CuriousInventor; Bitcoin Lightning Transactions & Protocol Deep Dive
https://www.youtube.com/watch?v=to8XItlplac

124. YouTube; Chaincode Labs; Eltoo & the Far Future with Christian Decker
https://youtu.be/3ZjymCOmn_A

125. YouTube; Jonathan Levi; Bitcoin Lightning Network Routing Fees Explained w/ Carsten Otto
https://youtu.be/v1yfqFzJoW4

126. YouTube; Tadge Dryja, Neha Narula; MIT MAS.S62 Cryptocurrency Engineering and Design
https://www.youtube.com/playlist?list=PLUl4u3cNGP61KHzhg3JIJdK08JLSlcLId

BRAIINS Publishing

Michal Novák
Lightning Network: Platby budoucnosti

Jazyková redakce: Daniela Hozdová
Technická redakce: Miloš Bém, Jan Dvořák, Jan Karas, Filip Kodýtek, Dominik Stroukal, František Šimek, Jan Veselý
Grafická úprava a sazba: Sabina Heyová
Design obálky a grafika: Jiří Chlebus
Ilustrace: Tomáš Nadymáček
Design konzultace: Robert Blecha
Marketingová strategie: Kristian Csepcsar
Odpovědný redaktor: Jáchym Černý

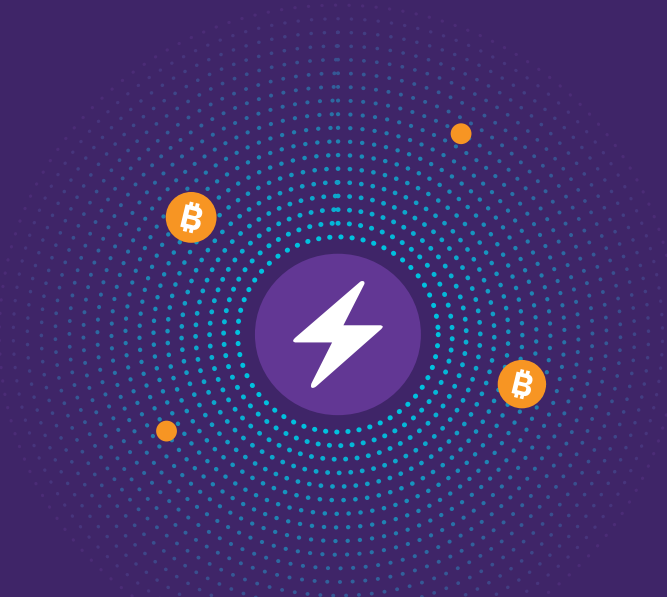
Vytiskla tiskárna Havlíčkův Brod
216 stran, vydání první

Vydalo nakladatelství Braiins Systems, 2023
braiins.com/publishing

Pochvaly či připomínky posílejte na: michal@platbybudoucnosti.cz
Lightning adresa: michal@platbybudoucnosti.cz
Web: https://platbybudoucnosti.cz
Twitter: @PBudoucnosti

LIGHTNING NETWORK

PLATBY BUDOUCNOSTI



Technologie Lightning Network se pomalu, ale jistě stává standardem pro rychlé a levné placení bitcoinem. V této unikátní publikaci, jejíž tištená podoba vznikla díky podpoře české a slovenské bitcoinové komunity, se dozvíte vše. Od úvodu pro naprosté začátečníky, přes detailní popis samotné technologie, až po návod, jak provozovat vlastní uzel.

„Tato kniha je výjimečná. Může se bez problémů postavit vedle knihy Andrease Antonopoulose Mastering LN a pro mnohé bude nejen čtivější, ale i srozumitelnější.“

ALEX PILAŘ

ISBN 978-80-908709-4-9

BRAINS Publishing