Princip fungování Ripple sítě

Petr Kraus

12. října 2018

Obsah

1 Úvod

XRP je **kryptoměna**, vytvořená a nadále vyvíjená společně s platební sítí **RippleNet** společností Ripple [**?**]. Tento systém byl vytvořen pro banky, za účelem zefektivnit transakce mezi dvěma různými bankami. Na rozdíl od většiny kryptoměn nevyžívá pro ukládání dat blockchain, nýbrž jeden soubor - **ledger**, který však využívá podobných principů k zajišťování správnosti a konzistence.

2 RippleNet

RippleNet je decentralizovaná peer-to-peer platební síť, ve které se XRP používá. Klienti v ní mohou upravovat data pouze transakcemi. Data jsou sdružována do jednoho souboru, který se nazývá **XRP Ledger**. Blockchain se skládá z jednotlivých bloků *(souborů)* s daty, kam se transakce zapisují. Každý blok odkazuje na blok předchozí a následující - **velmi zjednodušeně** je blockchain *linked list* [**?**]. Ripple zapisuje data do jednoho verzovaného po celé síti sdíleného souboru. Pro získání zůstatku v peněžence je v blockchainu nutné projít všechny bloky, Ripple veškeré zůstatky ukládá do každé verze ledgeru. Ačkoli využívá podobných principů nejedná se o blockchain.

XRP Ledger verze *(data o transakcích, zůstatcích apod.)* se mění každých pár sekund, a to když se síť shodne na obsahu konkrétní verze ledgeru, poté je daná verze uzavřena a data se začnou zapisovat do nové. Uzly, které validují a mezi sebou, ověřují transakce, za tuto službu získají možnost hlasovat o změnách v protokolu nikoliv finanční odměnu.

V RippleNet se lze zaregistrovat jako tzv. **gateway**, které umožňují odeslání či převod různých měn nebo aktiv, jako například USD, EUR, CZK, ropa, zlato a podobně. Namísto s XRP se pak pracuje s **IOU tokeny** [**?**]. Gateway může zmrazit IOU tokeny jakékoliv peněženky. Je to opatření proti převodům ilegálně nabytých finančních obnosů, které bylo přidáno, aby vyhovovalo bankám, respektive jejich nadřízeným vládám. Ripple doporučuje gateway uzlům zároveň provozovat validační server pro větší bezpečnost.

2.1 XRP Ledger

Uchovává veškerá data o síti a jeho verze se mění každých pár sekund. Každá verze musí projít schválením *(Konsensus proces)*, pokud se síť shodne na obsahu ledger verze, je uzavřena a poté její obsah nelze změnit. Všechny verze ledgeru jsou od verze 32570 uloženy na full history serveru provozovaném Ripplem [**?**]. V průběhu schvalování vzniká více kandidátů na novou verzi, z nichž je vybrán jeden. Každá verze obsahuje platná schválená data o celé síti, aktuální ve chvíli schválení.

Verze mají dva identifikátory. Jedním je **ledger index**, případně *sequence number*, který se s každou verzí zvyšuje o jedničku. Pokud by měla aktuální schválená verze index 100, předchozí verze má 99 a následují bude mít 101. Druhým je **ledger hash**, který je digitálním otiskem obsahu ledgeru. V průběhu

schvalování nové verze ledgeru vznikají různí kandidáti se stejným indexem, ale odlišným hashem. Z těchto kandidátů se vybere jeden a ostatní jsou zahozeni. Verze obsahuje množinu transakcí, které se týkaly pouze předchozí verze a metadata přesného efektu transakcí na tato data v ledgeru, což umožňuje ověřitelnost její historie.

Transakce uložené ve verzi XRP ledgeru jsou úspěšně vykonané transakce (**tes** class result code transakce (**??**)) a doposud nevykonané transakce z **tec** výsledkové třídy. Při odeslání nové transakce, musí být její výsledek kontrolován z již uzavřených verzí ledgeru, jedině tak jsou výsledky stoprocentně platné a neměnné.

Dále každá ledger verze obsahuje hash identifikující verzi předcházející, celkové množství XRP tokenů, čas potřebný k uzavření verze zaokrouhlený na sekundy, data účtů, data transakcí, čas uzavření verze digitálně a v UNIX formátu [**?**].

2.2 XRP Ledger protokol

Peer-to-peer XRP Ledger síť se skládá z mnoha nezávislých XRP Ledger serverů, které akceptují a zpracovávají transakce. Klientské aplikace podepisují a odesílají transakce na tyto servery, které rozešlou ucházející se transakce *(candidate transakcions)* po celé síti. Příklady takových aplikací zahrnují mobilní a webové peněženky, gateways finančních institutů, a elektronické obchodní platformy.

Servery, které přijímají, předávají a zpracovávají transakce mohou být buď **tracking servers** nebo **validující**. Hlavním úkolem tracking serverů je distribuce transakcí od klientů a reakce na příkazy s úpravou dat v ledgeru. Validující servery dělají to samé, navíc odesílají zprávy o validaci. Validujícím či tracking serverem se může stát kdokoli, stačí k tomu příslušný rippled software.

Při přijímání nových transakcí od klientských aplikací a kandidátů se používá poslední zvalidovaný ledger jako počáteční bod. Servery odešlou transakční kandidáty svým uzlům, čímž umožní propagaci transakcí po celé síti. V ideálním případě by se kandidáti dopropagovali ke každému serveru v síti, ale z časových důvodů nezpracovávají všechny servery všechny transakce, nýbrž každý server zpracovává určitou množinu. Pro zajištění zpracování a validace stejných transakcí po celé peer-to-peer XRP ledger síti se využívá proces zvaný **konsensus** *(Consensus)*.

2.2.1 Konsensus

Servery sdílí informace o ucházejících se transakcích *(kandidátech)*. Během tohoto procesu se validátoři shodnou na konkrétní podmnožině kandidátů, která se uloží do ledgeru. Každý server vyhodnocuje návrhy od specifické množiny serverů, zvané **server's trusted validator** - důvěryhodní validátoři daného serveru, nebo také *Unique Node List (UNL)*. Tyto důvěryhodné validátory reprezentují podmnožinu sítě, které je kolektivně důvěřováno, že se úmyslně nepokusí vysílat do sítě falešná nebo zmanipulovaná data.

Ucházející transakce, které nejsou odsouhlaseny, jsou při zpracování typicky vynechány, zůstávají mezi kandidáty a mohou být *(znovu)* zpracovány v následující verzi ledgeru (viz **??**). V některých případech mohou být transakce nezdařilé jen na neurčitou dobu. Například pokud se navýší poplatek za transakci na více než je v transakci zapsáno, může být transakce zvalidována ve chvíli, kdy se poplatek sníží. Aby se takovým případům předešlo, lze nastavit ledger index, kdy se transakce zruší.

2.2.2 Validace

Validace je druhá etapa konsensus procesu verifikující, zda servery dostaly stejný výsledek a deklaruje ledger verzi jako finální. Může být rozdělena na dvě části:

- Výpočet výsledné ledger verze z množiny sítí vybraných transakcí.
- Porovnání výsledků a deklarace, že je ledger verze validní, pokud se na ní shodlo dostatečné množství validátorů.

2.2.3 Výpočet a sdílení validací

Když je konsensus proces dokončen, každý server individuálně spočte nový ledger z vybrané množiny transakcí. Výsledek severy počítají podle následujících pravidel:

1. Začít s předcházejícím zvalidovaným ledgerem.
2. Kanonicky seřadit odsouhlasenou vybranou množinu transakcí, aby byly každým serverem zpracovávány stejně.
3. Zpracovat každou transakci podle instrukcí a odpovídajícím způsobem upravit data v ledgeru. Neúspěšně zpracované transakce označit tec class výsledkovým kódem. Opakovatelné selhané transakce umístit nakonec kanonického pořadí, aby byla v následující ledger verzi znovu zpracována.
4. Upravit hlavičku ledgeru odpovídajícími daty, jako je ledger index, identifikující hash předchozí verze - *"rodičovského ledgeru"*, přibližný čas uzavření a hash obsahu ledgeru.
5. Výpočet identifikujícího hashe této nové verze ledgeru.

Kanonické pořadí je určeno podle accoutKey, pořadí odeslání a transaction_id. AccountKey je výsledek operace XOR nad hashem množiny transakcí ke zpracování [**?**].

2.2.4 Porovnání výsledků

Každý validátor odešle svůj výsledek ve formě podepsané zprávy s hashem ledger verze, kterou vypočetl. Tyto zprávy umožňují každému serveru porovnat jejich výsledek s jeho uzly.

Pokud se na nové verzi ledgeru shodla "supermajorita"uzlů *(80%)*, je daná verze zvalidována s pořadovým číslem (indexem verze ledgeru) N+1.

V případě, že server dostane od více než 20% svých uzlů odlišně vypadající ledger, než spočetl on *(nelze dosáhnout supermajority)*, začne jej přepočítávat.

Pokud se nedokáže supermajorita shodnout na výsledné podobě ledgeru, je opakován konsensus proces. K této situaci může dojít při velkém zatížení sítě nebo její vysoké latenci. XRP ledger dynamicky přizpůsobí poplatek za transakci[**?**].

3 Peněženka

Peněženka slouží pro uložení obnosu, kontrolu zůstatku, případně odeslání zůstatku. V peněžence se vytvoří účty, kde každý obsahuje dva, respektive čtyři důležité řetězce: **Veřejný klíč**, **Privátní klíč**, **Adresa účtu** a **Family seed *(Secret)***. Kde jsou uživateli obvykle známi pouze dva, a to adresa účtu a Secret [**?**]. *Veřejný* a *privátní klíč* dohromady tvoří tzv. ***key pair***, oba klíče jsou hexadecimální řetězce, kryptograficky odvozené od Secret. RippleAPI umožňuje klíče získat, ale ve většině peněženek je uživateli zobrazena pouze adresa účtu a Secret. Adresa se odvozuje od veřejného klíče a umožňuje prohlížení historie transakcí, aktuálního zůstatku a slouží jako adresa pro přijetí transakce. Pro manipulaci se zůstatkem se používá key pair jako jeden objekt nebo secret.

Na rozdíl od jiných kryptoměn, musí být Ripple peněženka, respektive účet na peněžence aktivován, a to odesláním minimálně 20 XRP na zvolený účet. Prakticky si uživatel pořídí Ripple peněženku, vytvoří účet, koupí ve směnárně minimálně 20 XRP a odešle jej na vytvořený neaktivní účet, který se tím aktivuje. Odesláním méně než 20 XRP tokenů na neaktivní účet vede k jejich ztrátě. Aktivaci peněženek Ripple zavedl, jako opatření proti vytváření prázdných a nevyužívaných účtů. 20 XRP je zároveň nejnižší možný zůstatek na účtu.

4 Transakce

Transakce je jediný způsob, jak může klient změnit obsah ledgeru. Každá transakce, která se začne zpracovávat, se vyřídí za 4 sekundy. Síť je konstantně schopna zvládnout 1500 transakcí za sekundu, díky škálovatelnosti již zvládla i 50000 [**?**]. Při každé transakci je spáleno (zničeno) malé množství XRP, konkrétně poplatek za danou transakci *(transaction fee)*, který je momentálně 10 drops ($drop = 10^{-6} XRP$) za strandardní transakci[**?**]. Veškeré XRP tokeny budou při momentální rychlosti pálení zničeny za přibližně 400 tisíc let[**?**].

Aktuální minimální poplatek se odvíjí a automaticky přizpůsobuje od vytíženosti sítě. Pro jeho výpočet slouží vzorec (**??**). Base_fee_xrp vyjadřuje základní poplatek v XRP. Load_factor je násobitel základního poplatku za transakci, určovaný vytížeností serverů [**?**].

$$fee = base\_fee\_xrp * load\_factor \tag{1}$$

Ripple API umožňuje transakci nastavit různé parametry, například maximální nebo konkrétní množství XRP, které bude v rámci poplatku zničeno. Pokud je nastaveno maximální množství XRP, které je zájemce ochoten za transakční poplatek zaplatit, nemusí být veškeré XRP využity. V případě pevně nastaveného poplatku za transakci, je spáleno přesně nastavené množství XRP, a to i v případě, že je částka absurdně vysoká.

Dalším příkladem parametru transakce je počet verzí nebo konkrétní index verze ledgeru, do které musí být transakce provedena, poté bude zničena. V některých případech se může stát, že transakce není ihned vyřízena, aby v síti nezůstávala nevyřízená transakce, doporučuje se nastavit jeden z těchto parametrů.

RippleNet má několik typů transakcí, mimo převod XRP a IOU tokenů například změnu nastavení účtů, vytváření podmíněných transakcí, rušení transakcí. Příklady:

- **Payment** - Převod mezi dvěma účty.
- **Order** - Objednávka směny měn.
- **Escrow** - Zamčení XRP dokud není splněna zapsaná podmínka a následné odeslání na zvolený účet. Ekvivalent smart contracts v Ethereum síti.
- **Settings** - Změna nastavení účtu ve XRP ledger.

4.1 Průběh transakce

Transakce probíhá ve čtyřech následujících fázích:

1. **Prepare** - Naformátování nepodepsané transakce zvoleného typu. Vrací transakci ve vhodném formátu pro podepsání, společně s dalšími informacemi, včetně poplatku za transakci.
2. **Sign** - Kryptografické lokální podepsání transakce. Vrací podepsanou transakci jako hexadecimální string a její ID.
3. **Submit** - Odeslání transakce na připojený server. Vrací, zda byla transakce úspěšně odeslána.
4. **Verify** - Ověření, zda byla transakce skutečně zvalidována. Transakce může i přes úspěšné odeslání na síť selhat [**?**]. Ripple označuje tento krok jako nezbytný, ale skutečně slouží pouze pro ověření validace. Při validaci se kontroluje *result code*, který se dělí do tříd uvedených v tabulce (**??**).

RippleAPI může fungovat i bez internetového připojení, což zajišťuje vyšší bezpečnost. Pro připravení transakce offline, musí být specifikované **fee** *(poplatek za tranasakci)*, **sequence** *(N-tá transakce z daného účtu)*, **maxLedgerVersion** *(nejvyšší verze ledgeru, kde může být transakce uplatněna)*. Funkce, které lze volat v offline modu:

- preparePayment (a většina *prepare-* metod)

- sign
- generateAddress
- computeLedgerHash [**?**]

Kategorie	Prefix	Popis
Pouze požadované náklady	tec	Transakce nesplnila účel, ale poplatek byl zaplacen a zničen. Definitivní výsledek pouze ve zvalidovaném ledgeru.
Selhání	tef	Transakci nelze provést v momentální ani budoucí verzi ledgeru, nebo mohla být již provedena.
Lokální chyba	tel	Na rippled serveru došlo k chybě kvůli lokálním okolnostem, jako je vysoké zatížení .
Znetvořená transakce	tem	Transakce byla nevalidní, kvůli špatné syntaxi, konfliktním příkazům, špatnému podpisu, nebo něčemu jinému.
Opakovat	ter	Transakci se nepodařilo uplatnit, ale může být uplatněna později.
Úspěch	tes	Transakce uspěla *(není chyba)*. Definitivní výsledek pouze ve zvalidovaném ledgeru.

Tabulka 1: Tabulka result kódu transakcí [**?**]

4.2 Rozdělení odeslaných transakcí

Po odeslání transakce do sítě se na základě poplatku zhruba do tří kategorií:

- Transakce, s nastaveným nižším poplatkem než server vyžaduje, je odmítnuta.
- Transakce s dostatečně vysoko nastaveným poplatkem je přijata a zpracována.
- Transakce s dostatečně vysoko nastaveným poplatkem vůči místnímu minimálnímu poplatku odvozeného od zatížení konkrétního serveru, kam byla transakce odeslána, ale příliš nízkým poplatkem vůči požadavkům celé sítě. V tomto případě je při splnění určitých pravidel transakce přijata a uložena do fronty pro zpracování v další ledger verzi. [**?**]

4.2.1 Transakce ve frontě

Aby se transakce dostala do fronty, musí kromě dostatečného lokálního poplatku splňovat i následující pravidla:

- Transakce musí být správně formátová a autorizovaná validním podpisem.
- Transakce se zadaným *AccountTxnID* nemůže být ve frontě.

- Adresa odesilatele může mít maximálně 10 transakcí ve frontě. Musí mít dostatečné prostředky k zaplacení všech poplatků a odesílaných obnosů XRP.
- Pokud transakce ovlivňuje, jak odesilatel autorizuje transakce, nemůže ve frontě být za touto transakcí žádná další transakce tohoto odesilatele.
- Pokud má transakce zadaný *LastLedgerSequence*, musí být větší než aktuálni index ledgeru plus 2.

Ve frontě se transakce řadí podle výše poplatku za transakci, transakce s vyšším poplatkem je zařazena dříve, než transakce s poplatkem nižším. Řazení není podle absolutní ceny transakce, nýbrž ceny relativní k minimálnímu poplatku za transakci. Pokud je ve frontě více transakcí odeslaných z jedné adresy, řadí se podle času odeslání. Pořadí ve frontě určuje pořadí přidávání transakcí mezi množinu transakcí ke zpracování, ale neovlivňuje pořadí jejich vykonaní, pořadí vykonání transakcí je určeno podle kanonického pořadí [**?**].

4.3 Komentovaný příklad kódu transakce

```
// Pripojeni knihovny
const RippleAPI = require('ripple-lib').RippleAPI;

// Nastaveni serveru, se kterym bude aplikace komunikovat
const api = new RippleAPI({
  server: 'wss://s2.ripple.com'
});

// Ulozeni potrebnych retezcu
const sourceAddress = "rwSDFhSoqba3zCZGigrPYNeyWQj38r6E58";
const sourceSecret = "snAPpR3UxQVNw9fHFNgdKqaUWfTkW";
const destinationAddress = "rPWFMpUXWeSD2QnA7UKXsTewnYpvBgJ66T";

// Vytvoreni objektu transakce
const paymentTrans = {
    "source": {
        "address": sourceAddress,
        "maxAmount": {
            "value": "0.000001",
            "currency": "XRP",
        }
    },
    "destination": {
        "address": destinationAddress,
        "amount": {
            "value": "0.000001",
            "currency": "XRP",
        }
    },
}

// Pripojeni k serveru
api.connect().then(() => {
```

```
    console.log('Connected');

// Vytvoreni objektu platby, ve formatu citelnem pro ripple
  return api.preparePayment(sourceAddress, paymentTrans);
}).then(preparedPayment => {
    console.log('Payment prepared');

// Podpis transakce
  const signedData = api.sign(preparedPayment.txJSON,sourceSecret);
    console.log('Payment signed');

// Odeslani transakce na server
  return api.submit(signedData.signedTransaction).then(data => {
        console.log('Payment submited');
    console.log('Tentative result: ', data.resultCode);
        console.log('Tentative message: ', data.resultMessage);
    });
}).then(() => {

// Odpojeni od serveru
  return api.disconnect();
}).then(() => {
    console.log('done and disconnected.');

// Ukonceni programu
    process.exit();
}).catch(console.error);
```

Reference

[1] Official Ripple website https://ripple.com/

[2] Bitcoin Wiki https://en.bitcoin.it/wiki/Block

[3] Medium https://medium.com/@AlexCarrithers/xrp-vs-ious-on-ripple-what-are-they-and-which-are-banks-using-257023fc578e

[4] XRP official developer documentation https://developers.ripple.com/data-api.html#ledger-objects

[5] XRP Community blog https://xrpcommunity.blog/keys-are-key-secret-keys-signing-transactions/

[6] Steemit https://steemit.com/ripple/@ripplerm/ripple-understanding-passphrase-secret-key-and-address-account

[7] Ripple official website https://ripple.com/xrp/

[8] Ripple official developer documentation https://developers.ripple.com/transaction-cost.html

[9] Turf pool http://www.turfpool.com/ripple-burn-rate

[10] Ripple official developer documentation https://developers.ripple.com/rippleapi-reference.html

[11] XRPchat https://www.xrpchat.com/topic/3847-front-running-and-other-questions/?page=2&tab=comments#comment-38102

[12] Ripple official developer documentation https://developers.ripple.com/consensus.html

[13] Ripple official developer documentation https://developers.ripple.com/server_info.html

[14] Ripple official developer documentation https://developers.ripple.com/transaction-results.html

[15] Ripple official developer documentation https://developers.ripple.com/transaction-queue.html

[16] Wikipedia https://cs.wikipedia.org/wiki/Ripple

[17] Coindesk https://www.coindesk.com/ripple-medieval-banking-digital-twist/

[18] Finder https://www.finder.com/ripple

5 Dodatek A: Slovník

- XRP / XRP token - Token kryptoměny XRP
- XRP ledger / ledger / - Soubor s daty o Ripple síti
- Kandidát / Ucházející se transakce - Nová transakce odeslaná do sítě, jejíž kód se doposud nevykonal.
- Uzel - peer
- Konsensus / Consensus - Proces validace nové verze ledgeru (Význam v čj - shoda jistého společenství, na základě faktů a emocí).
- Pálení / zničení - Tokeny XRP, které jsou zničeny šmazány a nelze je nijak vrátit.