Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Diplomová práce

Bezpečnost IoT při integraci s technologií Blockchain

Bc. Sami Salama

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Bc. Sami Salama

Systémové inženýrství a informatika
Informatika

Název práce

Bezpečnost IoT při integraci s technologií Blockchain

Název anglicky

IoT security in regards to integration with Blockchain

Cíle práce

Hlavním cílem diplomové práce bude ověření vhodnosti integrace technologií Blockchain a IoT v kontextu bezpečnosti. Dílčími cíli bude:
- analýza stávajícího stavu IoT z hlediska bezpečnosti,
- experiment pro ověření vhodnosti integrace technologie Blockchain s IoT,
- vyhodnocení experimentu a formulace závěrů.

Metodika

Metodika řešené problematiky bude založena na studiu odborné literatury a analýze informačních zdrojů zaměřených na IoT a Blockchain. V teoretické části budou vymezeny pojmy IoT a Blockchain. Dále bude zkoumán současný stav výzkumu a vývoje spojující tyto dvě technologie.

Praktická část se bude zabývat provedením vhodného experimentu pro ověření řešení v kontextu bezpečnosti. Výsledky budou poté vyhodnoceny a ze získaných poznatků formulovány závěry.

Doporučený rozsah práce

65 – 75 stran

Klíčová slova

Blockchain, Internet of Things (IoT), decentralizace, bezpečnost

---

Doporučené zdroje informací

ALAM, Tanweer, 2019. Blockchain and its Role in the Internet of Things (IoT) [online]. [cit. 2019-05-23]. DOI: 10.32628/CSEIT195137. Dostupné z: http://arxiv.org/abs/1902.09779

FRANCO, Pedro, 2014. Understanding Bitcoin: Cryptography, Engineering and Economics. John Wiley & Sons, Incorporated. ISBN 9781119019152.

MOUGAYAR, William a Vitalik BUTERIN, 2016. The business blockchain: Promise, practice, and application of the next Internet technology. Hoboken, New Jersey: John Wiley & Sons, Incorporated. ISBN 978-111-9300-335.

PENTTINEN, Jyrki T. J., 2017. Wireless communications security: Solutions for the internet of things. The Atrium, Southern Gates, Chichester, West Sussex, United Kingdom: John Wiley & Sons, Incorporated. ISBN 9781119084419.

Předběžný termín obhajoby

2019/20 LS – PEF

Vedoucí práce

Ing. Jan Masner, Ph.D.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 26. 8. 2019

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 14. 10. 2019

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 25. 03. 2020

Čestné prohlášení

Prohlašuji, že jsem svou diplomovou práci "Bezpečnost IoT při integraci s technologií Blockchain" vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne Bc. Sami Salama

Poděkování

Rád bych touto cestou poděkoval Ing. Janu Masnerovi Ph.D. za cenné rady, věcné připomínky a vstřícnost při konzultacích a vypracování diplomové práce.

Bezpečnost IoT při integraci s technologií Blockchain

Abstrakt

Cílem diplomové práce je ověření vhodnosti integrace technologie blockchain a IoT (Internet of Things) v kontextu bezpečnosti. Práce je rozdělena na teoretickou a praktickou část. V teoretické části práce jsou vymezeny pojmy IoT a blockchain a je zkoumán současný stav výzkumu a vývoje spojujícího tyto dvě technologie.

V praktické části práci je navržen a vytvořen modelový případ bezpečnostně kritické aplikace IoT integrující blockchain technologii. Toto řešení blockchain sítě je implementováno pomocí platformy Hyperledger Fabric a souvisejícího frameworku Hyperledger Composer. Dále je provedena simulace IoT zařízení v nástroji Node-RED odesílajícího do této sítě data, která jsou následně ukládána na blockchain.

K ověření vhodnosti integrace je proveden experiment pomocí nástroje hping3, kdy je na komponentu vytvořené blockchain sítě proveden DoS (Denial of Service) útok. Pomocí měřícího nástroje Apache JMeter je sledován vliv útoku na funkčnost sítě a výkonnostní ukazatele. Následně jsou vyhodnoceny výsledky experimentu a ze získaných poznatků formulovány závěry.

Klíčová slova: Blockchain, Internet of Things (IoT), decentralizace, bezpečnost, Hyperledger Fabric, Hyperledger Composer, síť, DoS, experiment

IoT security in regard to integration with Blockchain

Abstract

The objective of this thesis is to verify the suitability of blockchain and IoT (Internet of Things) integration in the context of security. The thesis is divided into theoretical and practical part. The theoretical part defines the terms IoT and blockchain and examines the current state of research and development combining these two technologies and describes the blockchain platform Hyperledger Fabric.

There is designed and created a model case of security critical IoT application integrating blockchain technology in the practical part. This blockchain network solution is implemented using the Hyperledger Fabric platform and the associated Hyperledger Composer framework. Next, the IoT device is simulated in the Node-RED tool that sends data to this network, which are then stored on the blockchain.

To verify the suitability of the integration, an experiment is performed using the hping3 tool, where a DoS (Denial of Service) attack is performed on a component of the created blockchain network. Using the Apache JMeter, the impact of the attack on network functionality and performance indicators are being monitored. The results of the experiment are evaluated, and conclusions are formulated.

Keywords: Blockchain, Internet of Things (IoT), decentralization, security, Hyperledger Fabric, Hyperledger Composer, network, DoS, experiment

Obsah

1 Úvod

Blockchain technologie byla poprvé široce popularizována poté, co byla použita jako základ pro kryptoměnu Bitcoin. Později našla uplatnění i v jiných oblastech. Blockchain se začal implementovat v různých odvětvích kvůli svým vlastnostem transparentnosti, integrity a přesunutí důvěry od institucí směrem k algoritmům. Protože neexistuje univerzální řešení, které by bylo vhodné pro všechny případy užití blockchain technologie, vznikají různé blockchain platformy. Jednou z nich je open-source platforma Hyperledger Fabric cílená na podnikové využití s potřebou privátního prostředí.

Další technologie, která narůstá na popularitě je IoT (Internet of Things), neboli Internet věcí. Aktuální trend naznačuje stále narůstající počet připojených zařízeních k Internetu, což vede k otázkám, jak sledovat totožnosti, kontrolovat interakce a provádět transakce těchto zařízení bezpečným a efektivním způsobem. Protože se z IoT zařízení často získávají citlivá a bezpečnostně kritická data, je třeba brát v úvahu ochranu těchto dat po celou dobu jejich životního cyklu včetně jejich bezpečného uložení.

Ovšem technologie IoT se potýká se značným množstvím bezpečnostních hrozeb. Právě integrace IoT technologie s blockchain technologií může přinášet řešení k celkovému zlepšení důvěryhodnosti a odolnosti proti neoprávněné manipulaci s daty. Je to dáno vlastností blockchain technologie, která využívá pokročilé techniky kryptografie a algoritmy k ochraně uložených dat. Tato práce se zabývá ověřením vhodnosti spojení těchto dvou technologií v kontextu bezpečnosti. K ověření je proveden experiment, kdy je na komponentu blockchain sítě proveden útok, a je sledován vliv útoku na funkčnost sítě a výkonnostní ukazatele. Následně jsou vyhodnoceny výsledky experimentu a ze získaných poznatků formulovány závěry.

2 Cíl práce a metodika

2.1 Cíl práce

Hlavním cílem diplomové práce je ověření vhodnosti integrace technologií Blockchain a IoT v kontextu bezpečnosti. Dílčí cíle práce jsou:

- teoretické vymezení pojmů IoT a blockchain
- analýza stávajícího stavu integrace IoT a blockchain technologií
- návrh a vytvoření modelového řešení bezpečnostně kritické aplikace integrující IoT a blockchain technologie implementovaného pomocí platformy Hyperledger Fabric a souvisejícího frameworku Hyperledger Composer
- provedení experimentu nad vytvořeným řešením ověřující vhodnost integrace blockchain technologie s IoT technologií z hlediska bezpečnosti
- vyhodnocení experimentu a formulace závěrů

2.2 Metodika

Metodika řešené problematiky je založena na studiu odborné literatury a analýze odborných informačních zdrojů zaměřených na IoT a blockchain. V teoretické části jsou vymezeny pojmy IoT a blockchain. Dále je zkoumán současný stav výzkumu a vývoje spojující tyto dvě technologie.

Praktická část se zabývá navržením a vytvořením modelového případu bezpečnostně kritické aplikace integrující IoT a blockchain technologie implementované pomocí open-source blockchain platformy Hyperledger Fabric v1.2 a souvisejícího frameworku Hyperledger Composer 0.20.9. Tímto postupem je vytvořena blockchain síť dodavatelského řetězce, se kterou komunikuje simulované IoT zařízení. Toto simulované IoT zařízení odesílá citlivé údaje do blockchain sítě, které jsou následně zaznamenávány na blockchain. Simulované IoT zařízení je vytvořené pomocí open-source nástroje Node-RED.

Dále je proveden experiment ověřující řešení v kontextu bezpečnosti. V rámci experimentu je proveden DoS (Denial of Service) útok cílený na nasazenou blockchain síť a je zkoumán vliv tohoto útoku na funkčnost a výkonnost sítě. Útok je proveden pomocí nástroje hping3 a vliv útoku je zkoumaný měřícím nástrojem Apache JMeter. Výsledky experimentu jsou poté vyhodnoceny a ze získaných poznatků formulovány závěry.

3 Teoretická část

3.1 Blockchain

Technologie blockchain byla poprvé široce popularizovaná poté, co byla použita jako základ pro kryptoměnu Bitcoin. Bitcoin poprvé představil v roce 2009 Satoshi Nakamoto ve své práci nazvané „Bitcoin: A Peer-to-Peer Electronic Cash System“ (Nakamoto, 2008). Bitcoin se tak stal první používanou implementací důvěryhodné peer-to-peer elektronické měny. Vlastnosti blockchain technologie podporující Bitcoin jsou decentralizovaný přístup, kde žádný server nedrží kontrolu nad sítí, neměnitelnost, kde již uložená data nemohou být smazána a schopnost provádět platební transakce bez nutnosti zásahu třetí strany (Bagchi, 2017) (Zheng, 2017).

Od té doby vzniklo mnoho dalších forem elektronické měny vytvořených pomocí podobných struktur. Ovšem nevznikaly jen elektronické měny (kryptoměny), blockchain se začal implementovat v různých nových konceptech. Takovou zásadní a významnou aplikací je chytrá smlouva (anglicky „smart contract“) (Novo, 2018). Chytrá smlouva je počítačový protokol, který zajišťuje, ověřuje nebo vynucuje vyjednání či provedení kontraktu. Poskytuje schopnost přímo sledovat a provádět komplexní dohody mezi stranami bez nutnosti lidské interakce (Szabo, 1997). Chytrá smlouva je podrobněji popsána v kapitole 3.1.10.

Uvádí se, že mezi hlavní výhody, které blockchain technologie přináší, patří:

- bezpečnost
- distribuce a decentralizace
- soukromí
- transparentnost a integrita
- důvěrnost (Lisk, 2019)

Blockchain je přirozeně bezpečný, protože využívá výkonnou asymetrickou kryptografii, která využívá kombinace veřejného a soukromého klíče. Adresy uzlů nejsou přímo spojeny s identitou uživatele, takže nemůže dojít k úniku identit. Tímto způsobem blockchain nabízí vyšší úroveň zabezpečení pro jednotlivého uživatele, protože odstraňuje potřebu slabých a snadno odhalitelných hesel a online identit (Lisk, 2019) (Baset, 2018).

Decentralizovaná povaha blockchainu odstraňuje potřebu se spoléhat na centrální autoritu, což činí systém spravedlivější a bezpečnější. Místo toho využívá mechanismů konsensu k ověřování transakcí a zaznamenávání dat nenarušitelným způsobem. Data

v blockchainu se neukládají na jediné zařízení, ale místo toho jsou distribuovaná mezi ostatní zařízení v peer-to-peer (P2P) síti (Lisk, 2019).

Soukromí uživatele (uzlu) v blockchain síti je docíleno skrytím jeho identity pomocí kryptografie. Totožnost uživatele tak nelze odhalit. Navíc k využívání blockchainu nejsou vyžadovány žádné osobní údaje, místo toho je identita a vlastnictví aktiv spojeno s uživatelským zařízením. Nehrozí tedy odcizení osobních údajů, jak je tomu v případě centrálních serverů, které mohou být napadnuty (Lisk, 2019).

Transparentnost blockchainu vyplývá ze skutečnosti, že jsou transakce veřejných adres volně k nahlédnutí. Pomocí veřejné adresy uživatele lze prohlížet všechny transakce provedené konkrétním uzlem. Integrita dat je zaručena možností zaznamenávat transakce nenarušitelným způsobem. Své uplatnění nachází nejen ve finanční sféře, ale i například ve zdravotnictví, kde si uživatel může ověřit pravost léčiv. Dalšími oblastmi využití mohou být například státní volby a dodavatelské řetězce (Lisk, 2019) (Nofer, 2017).

Blockchain pomocí mechanismů konsensu a kryptografii umožňuje důvěryhodným způsobem zaznamenávat informace a odstraňuje potřebu zprostředkovatelů služby. Navíc při použití chytrých smluv není pochyb, zda strana splnila své závazky, protože se tak děje automaticky na základě jasně definovaných pravidel (Lisk, 2019) (Baset, 2018).

3.1.1 Typy blockchain

Veřejný blockchain

Veřejný blockchain je takový, který je k dispozici každému uzlu, který se chce zúčastnit. Každý uzel zde má právo číst, posílat transakce a zúčastnit se proces konsensu. Proces konsensu je proces, ve kterém se rozhoduje, které bloky se přidají do řetězce. Veřejný blockchain je zabezpečený kryptoekonomií, což je kombinace ekonomických pobídek a kryptografických ověřování pomocí mechanismů konsensu jako jsou PoW a PoS (Buterin, 2015) (Lastovetska, 2019). Mechanismy konsensu jsou popsány v kapitole 3.1.9.

Privátní blockchain

V privátním blockchainu mají práva k zápisu pouze uzly ze specifické organizace nebo autorizované uzly, které jsou pozvány k účasti. Uplatnění privátního blockchainu zahrnuje například správa databáze společnosti, kdy veřejná účast uzlu nemusí být nutná (Buterin, 2015) (Lastovetska, 2019).

Blockchain konsorcia

Posledním uvedeným typem je blockchain konsorcia, kde je proces konsensu řízen předem vybranou sadou uzlů, proto je považován za částečně decentralizovaný. Tato struktura blockchainu se může skládat z několika organizací (Buterin, 2015) (Lastovetska, 2019).

3.1.2 Architektura

Blockchain je distribuovaná decentralizovaná databáze obsahující časově uspořádaná fakta, která se nazývají transakce. Tyto transakce jsou seskupeny do takzvaných bloků. Bloky jsou propojeny pomocí ukazatelů, které představují kryptografické řetězce. Tento řetězec, neboli hašová hodnota, slouží k identifikaci předchozího bloku. Každý blok v blockchainu obsahuje hašovou hodnotu předchozího bloku, tím vzniká řetězec bloků, kde každý blok ověřuje integritu celého blockchainu před ním. Bloky mohou být připojeny na konec blockchainu a manipulace s předchozími bloky je detekovatelná. Blockchain je tedy rezistentní proti neoprávněné manipulaci (Luu, 2016) (Bano, 2017).

3.1.3 Síťová architektura

Tradičně World Wide Web využívá síťovou architekturu klient-server. V tomto případě jsou všechny informace uloženy v centralizované databázi na serveru, která je řízena správci s požadovanými oprávněními. V modelu s centralizovaným serverem dochází při rostoucím množství připojených klientů ke zpomalení celé sítě (Lastovetska, 2019).

Naopak v případě P2P sítě může znamenat větší množství uzlů zlepšení výkonu sítě. P2P architektura je používaná v distribuované blockchain síti. Dochází zde k tomu, že každý účastník v síti udržuje, schvaluje a aktualizuje nové záznamy. Platnost a bezpečnost dat je zajišťována každým z účastníků sítě (Lastovetska, 2019).

Peer lze chápat jako výpočetní jednotku v síti, často se peer označuje jako uzel. Peer poskytuje část svých výpočetních prostředků ostatním účastníkům v síti bez nutnosti zásahu centrální jednotky jako je například server. Výpočetním prostředkem může být například diskové uložiště, výpočetní výkon nebo šířka pásma sítě (Lisk, 2019) (Mougayar, 2016).

Ze základní povahy P2P vyplívá menší náchylnost k napadení, zneužití nebo odcizení dat. Žádná centrální autorita nemůže ovládat síť ani data v ní uložená. Nemůže se například stát, že by banka, jakožto centrální autorita a vlastník dat, zamrazila účet a peníze na něm uložené kdykoliv uzná za vhodné (Lisk, 2019) (Mougayar, 2016).

P2P architektura hraje hlavní roli v blockchain, jakožto nový systém komunikace, kde mohou jednotlivé uzly komunikovat přímo mezi sebou prostřednictvím zabezpečené, distribuované a decentralizované sítě. Přestože je účast všech uzlů v síti odkrytá, mohou být všechny informace a identity v blockchain skryté prostřednictvím kryptografie (Lisk, 2019) (Mougayar, 2016).

3.1.4 Uzly

Základem technologie jsou uzly, které zajišťují fungování blockchain sítě. Uzly mohou být jakákoliv elektronická zařízení, počítače, telefony i tiskárny. Podmínkou je, aby toto výpočetní zařízení bylo připojeno k internetu a vlastnilo IP adresu. Každý uzel v síti udržuje celkovou kopii blockchainu a v některých případech i zpracovávají transakce (Lisk, 2019).

V případě užití blockchain ve světě kryptoměn jako je například Bitcoin, udržuje každý uzel kopii tzv. účetní knihy. Když chce uzel (uživatel) zaslat své finanční prostředky jinému uzlu, tak pomocí svého privátního klíče podepíše zprávu, která uvádí, komu se mají tyto finanční prostředky připsat a jaké je jejich množství. Uzel poté tuto zprávu vyšle do sítě a každý zúčastněný uzel v síti získá kopii zprávy. Poté každý uzel může nezávisle ověřit platnost zprávy a podle toho aktualizovat jeho vnitřní databázi. Tím se systém ověřování kryptoměn liší od tradičního finančního systému, kde je účetní kniha (databáze) spravována finančními institucemi. Klienti musí vkládat důvěru v tyto instituce, že jejich databáze nebude narušena vnitřními ani vnějšími útočníky. Interní procesy a metody, které předcházejí těmto útokům, nejsou obecně odhaleny veřejnosti. Naopak, kryptoměny (například Bitcoin) databázi zveřejňují a vytvářejí volně dostupné softwarové protokoly k její ochraně. Tyto protokoly jsou navrženy tak, aby odolávaly útokům, takže uživatelé (uzly) nemusejí dávat svou důvěru v žádnou entitu (Franco, 2014).

Specifickým typem uzlu, který provádí verifikaci nového bloku před jeho přidáním do blockchain sítě, je tzv. těžař. Těžaři[1] jsou uzly, které přispívají svými výpočetními prostředky k ověřování transakcí. Za ověřování transakcí mají šanci získat transakční poplatky a získat tak odměnu v podobě kryptoměny. Tento proces je obecně známý jako těžba (anglicky „mining“) nebo kování (anglicky „forging“) (Evans, 2019). Detailněji je tento proces popsán v kapitole 3.1.10.

[1] Mining Node, česky „těžařský uzel“, dále jen jako „těžař“ (česky „horník“).

Uzly v blockchain se dělí v základu na dva druhy – úplné uzly (anglicky „full nodes“) a odlehčené uzly (anglicky „lightweight nodes“).

Úplné uzly

Úplné uzly obsahují kopii blockchainu, jeho historie a všech bloků v něm vytvořených. Jejich hlavní funkcí v blockchainu je udržování konsensu mezi ostatními uzly a ověřování transakcí. Tím, že uchovávají celou kopii blockchainu jsou považovány za bezpečnější a umožňují různé funkce jako okamžité odesílání a privátní transakce (Evans, 2019).

Specifickou podkategorií úplných uzlů jsou tzn. *ořezané úplné uzly*. Tyto uzly jsou omezeny předem nastavenou velikostí, kterou mohou uložit. Aby toho dosáhly začnou nejdříve stahovat bloky od začátku řetězce, a jakmile dosáhnou nastaveného limitu (např. 550 MB), odstraní nejstarší bloky a uchovají si pouze jejich hlavičky a pozici v řetězci. Přestože jsou ořezané, považují se za úplné uzly, a proto také mohou ověřovat transakce a účastnit se konsensu (Evans, 2019).

Další podkategorií úplných uzlů jsou tzv. *archivní úplné uzly*. Jedná se o specifický typ úplného uzlu, který uchovává všechny historické stavy každého bloku. S každým přidaným blokem jsou stavová data přidávaná a odstraňovaná z nejnovější verze stavu. Ovšem archivní uzel si všechna ořezaná data uchovává. Data poté lze použít k zobrazení historického stavu jakéhokoliv bloku v blockchainu. Archivní uzel je více než úplný uzel a není vyžadován k běhu plně ověřeného blockchainu. Archivní úplné uzly mohou být dále rozděleny na ty, které mohou přidávat nové bloky do blockchainu, a na ty, které nemohou. Mezi typy uzlů, které patří pod archivní úplné uzly, patří těžaři (Mining Nodes), stakers (Staking Nodes)[2], autoritní uzly (Authority Nodes) a masternodes (Martinez, 2018a).

Těžaři jsou uzly (úplné nebo odlehčené), které vycházejí z mechanismu konsensu PoW[3], takže jejich cíl je prokázat, že dokončily práci potřebnou k vytvoření bloku. Aby mohli tento cíl split, potřebují od ostatních uzlů v síti získat data o aktuálním stavu blockchain a požadovaných parametrech dalšího bloku v řadě. Jinou možností je, že je těžař archivním uzlem (Evans, 2019).

[2] Stakers – autor práce nenalezl vhodný překlad tohoto slova, proto bude v práci slovo dále užíváno v jeho anglické podobě. Význam slova „stake“ – peněžní nebo obchodní zájem, vklad, sázka, investice, podíl nebo účast na něčem bez jistoty zisku.

[3] Proof of Work, česky „důkaz o práci“, dále jen jako PoW.

První uzel, který za použití výpočetního výkonu hardwaru (CPU[4], GPU[5], anebo ASIC[6]) vyřeší náročný kryptografický problém, vyšle své výsledky do sítě, kde jsou ověřeny dalšími úplnými uzly. Tím se dosáhne konsensu a uzlu je uděleno právo k přidání bloku do blockchainu. Těžař je za svou práci odměněn předdefinovaným množstvím kryptoměny včetně transakčních poplatků. Tato odměna se nazývá coinbase (Evans, 2019).

Výhodami této metody jsou snadné pochopení konceptu prokázání účasti uzlu a příležitost spolupráce s ostatními uzly za účelem zvýšení míry přijímaných odměn. Nevýhodami jsou vysoká spotřeba energie, monopolní chování výrobců ASIC a vysoké počáteční náklady s nejistotou návratnosti investic (Evans, 2019).

Stakers jsou uzly, které vychází z mechanismu konsensu PoS[7]. Uzel udržuje mince a na oplátku získává úrok jako odměnu. Existují různé podoby PoS, ale základním principem je, že vydělávaní kryptoměny lze přirovnat k účasti v loterii. Uzel, který bude další, jenž přidá nový blok do blockchainu a získá tak odměnu, je vybírán na základě předdefinované sady pravidel a faktoru štěstí. Faktory štěstí jsou ovlivněny několika vlastnostmi uzlu – stáří mince, tedy doba, po kterou uzel drží své mince, dále množství mincí a jejich poměr ke zbytku sítě. Tento přístup není náročný na hardware, hlavní je udržovat svou peněženku nonstop online. Podmínkou je, aby byl uzel úplný archivní uzel. Z vlastností stakers uzlů vyplývají výhody jako nízká bariéra vstupu a nízká spotřeba energie. Nevýhodou je systém založený na štěstí (náhodě) (Evans, 2019) (Saleh, 2018).

Autoritní uzly narušují decentralizovanou povahu blockchain s cílem získání určitých výhod jako je zvýšená rychlost. V takovéto síti je definován pevný počet autoritních uzlů, které jsou zvoleny buď hlasováním komunity, nebo vývojovým týmem. Úkolem těchto uzlů je opět vytváření a ověřování bloků a zároveň distribuce informací uživatelům v síti. Uzly, které v síti nejsou autoritními uzly, jsou odlehčené uzly, které jsou závislé na vysílaných datech, aby mohli operovat v blockchainu. Mezi mechanismy konsensu, které využívají

[4] CPU (Central Processing Unit) – centrální procesorová jednotka.

[5] GPU (Graphics Processing Unit) – grafický procesor.

[6] ASIC (Application Specific Integrated Circuit) – integrované čipy se specifickým aplikačním určením.

[7] Proof of Stake, česky „důkaz o vkladu“, dále jen jako PoS.

principy autoritních uzlů, patří například Delegovaný PoS (DPoS), PoA[8] a Delegovaný BFT[9] (DBFT) (Evans, 2019).

Typem uzlu, který nemůže přidávat nové bloky do blockchainu je *masternode***.** Tento typ uzlu slouží k uchovávání záznamů o transakcích a jejich ověřování. Masternode ovšem také přináší odměny. Odměny jsou udělovány za zabezpečení sítě. Předpokládá se jeho nepřetržitý online provoz a uzamčení určité finanční sumy jako pojistky. Výhodami tohoto typu uzlu jsou zdroje pasivního příjmu, levná údržba a prospěšné vlastnosti pro síť, které přinášejí odměny. Nevýhodami jsou vysoké vstupní investice a složitější proces konfigurace (Evans, 2019) (Hlůšková, 2020).

Odlehčené uzly

Odlehčené uzly nebo také SPV (Simple Payment Verification) jsou peněženky, které neobsahují kopii celého blockchainu, ale jen hlavičky bloků z nejdelšího řetězce, čímž šetří úložné místo. Protože neobsahují kopii blockchainu, jsou závislí na úplných uzlech, kterých se dotazují na aktuální stav blockchainu a informací o transakcích. Bez úplných uzlů by odlehčené uzly nebyly schopné ověřovat transakce (Evans, 2019).

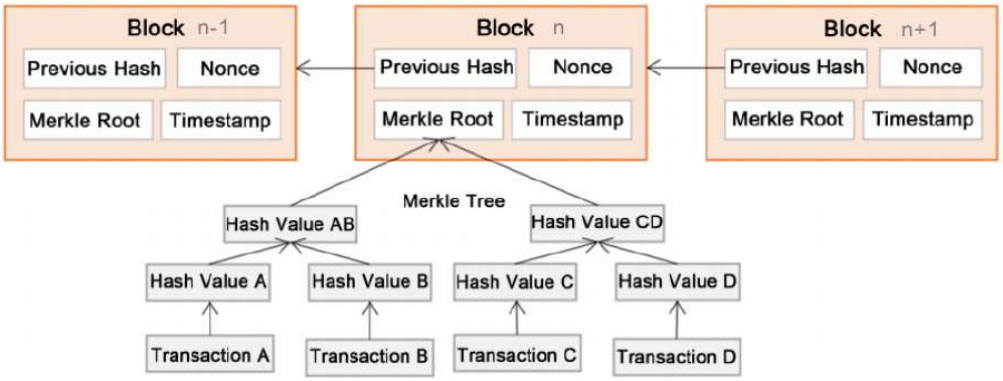
Ověřování je spolehlivé jen v případě, kdy je síť ovládána především čestnými uzly. Pokud síť ovládají škodlivé uzly, může útočník oklamat odlehčené uzly smyšlenými transakcemi. Obranou může být přijímání výstrah o neplatných blocích od úplných uzlů a vyzvat uživatele ke stažení celého blockchainu a odhalit tak nekonzistenci (Nakamoto, 2008).

[8] Proof of Autority, česky „důkaz autority“, dále jen jako PoA.

[9] Byzantine Fault Tolerance, volně přeloženo „Byzantská odolnost vůči chybám“, dále jen BFT. Vychází z teoretického problému dvou armád.

3.1.5 Bloky

Všechny transakce v síti jsou seskupovány do bloků, které jsou udržovány a vytvářeny všemi uzly v síti. Struktura bloku se skládá z hašové hodnoty, hašové hodnoty předchozího bloku, dále tzn. „*nonce*“, který zabraňuje škodlivým uzlům v zaplavení sítě, seznamu transakcí a časového razítka. Seznam transakcí je obvykle v bloku uložen v podobě Merkleova stromu (hašový strom). Merkleův strom je popsán později v kapitole o hašování. Bloky jsou přidávány do blockchainu těžaři procesem těžení (mining) (Hong, 2017) (Zheng, 2017). Příklad struktury blockchainu je ilustrován na obrázku 1.



Obrázek 1 - Struktura blockchainu a bloků v něm (Hong, 2017).

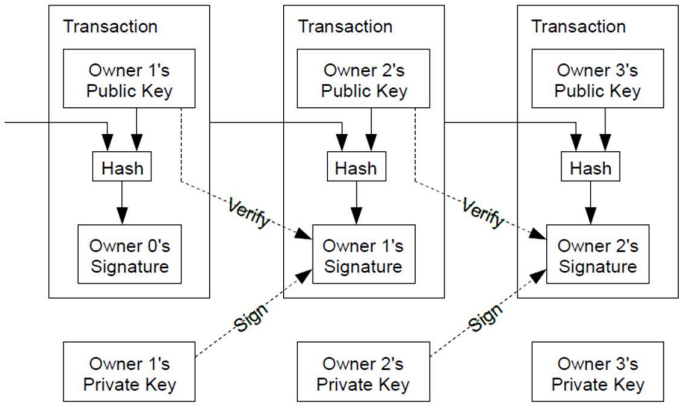
Nonce

Pojmem nonce se obecně v kryptografii označuje zpravidla náhodné číslo, které je použitelné pouze jednou. Nonce se do hašové hodnoty bloku přidává za účelem zvýšení obtížnosti podvržení daného bloku. Kdyby se útočníkovi povedlo replikovat celý blok i jeho transakce a vytvořit tak falešný blok, stále by musel najít takovou hodnotu nonce, aby se hašová hodnota podvrženého bloku rovnala s původní hodnotou (Hordějčuk, c2008-2019). Nonce se také využívá v procesu těžení v rámci PoW konsensu. Snahou těžaře je vyřešit náročný kryptografický problém. K nalezení výsledku software těžař postupně zvyšuje nonce a zkontroluje, zda zvolený nonce generuje správnou hašovou funkci bloku (Franco, 2014). Tento mechanismus je podrobněji popsán v kapitole o PoW konsensu (3.1.9).

3.1.6 Transakce

Transakce se skládá ze seznamu vstupních transakcí (TxIn) a seznamu výstupních transakcí (TxOut). Výstupní transakce se v případě Bitcoin skládá z adresy příjemce a částky. Majitel prostředků digitálně podepíše předchozí transakci svým veřejným klíčem,

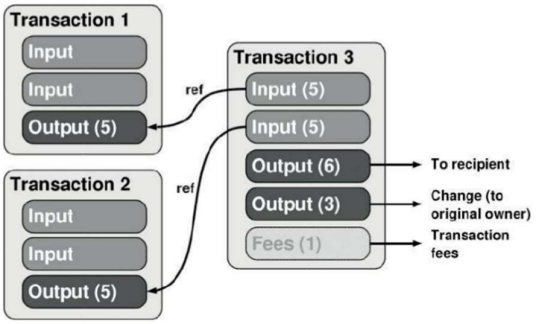
tím vznikne unikátní kryptografický haš. Majitel předchozí transakce poté podepíše haš svým soukromým klíčem (Nakamoto, 2008) (Franco, 2014). Zjednodušené znázornění procesu je popsáno na obrázku 2 níže. Počet vstupních i výstupních transakcí může být vyšší.



Obrázek 2 - Mechanismus podepisování transakcí popsaný dle (Nakamoto, 2008).

Transakční vstup (TxIn) obsahuje odkaz na předchozí transakční výstup spolu s podpisem, který dokazuje jeho pravost. Podpis musí být proveden se soukromým klíčem sdruženým s veřejným klíčem. Pokud se podpis neshoduje, je transakce považována za neplatnou a je sítí vynechána (Franco, 2014) (Nakamoto, 2008).

Dalším kritériem platnosti transakce je výsledek kontrolního součtu. Aby byla transakce platná, musí být suma částek vstupů větší nebo rovna sumě částek výstupu. Pokud existuje rozdíl mezi vstupy a výstupy, pak je tato částka považována za transakční poplatek, který dostanou těžaři jako odměnu za ověření a zahrnutí transakce do bloku (Franco, 2014).



Obrázek 3 - Příklad transakce (Franco, 2014).

Na obrázku 3 je znázorněný příklad transakce. V tomto příkladě chce odesílatel zaslat částku kryptoměny (o hodnotě 6) příjemci. Protože odesílatel nevlastní konkrétní sumu, ale pouze dva výstupy o hodnotě 5, musí vytvořit novou transakci, která spojuje tyto dva výstupy. Poté odešle výstupem požadovanou částku příjemci a zahrne do transakce i výstup s adresou, kam se zašle zbytek. Součástí transakce je i odměna pro těžaře, která slouží jako

motivace pro těžaře k ověření a zahrnutí transakce do dalšího bloku. Před odesláním transakce do sítě musí odesílatel tyto dva výstupy digitálně podepsat, aby se dokázalo, že vlastní adresy, na které tyto výstupy odkazují. První uzel, který v síti dostane transakci, ověří její pravost. Jeli transakce validní, tak jí odešle na další uzly v síti (Franco, 2014). Ověření pravosti probíhá v několika krocích:

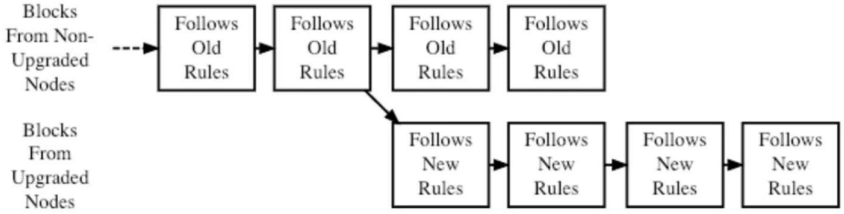
- zkontroluje, zda předchozí výstupy odkazovány transakcí existují
- dále zkontroluje, zda suma výstupů je větší nebo rovna sumě vstupů
- a zkontroluje, zda jsou všechny výstupy podepsány soukromým klíčem odpovídající veřejnému klíči sdruženým s adresou, na kterou odkazuje (Franco, 2014)

3.1.7 Forky

Hard fork

Hard fork je radikální změna v protokolu, která je permanentní odchylkou od předchozí verze. Každá takováto změna, která není kompatibilní s předchozí verzí protokolu, je považována za hard fork. Většinou se jedná o změny mechanismu konsensu v síti, jako jsou například přechod z PoW na PoS a implementace nových uzlů (Evans, 2019).

Hard fork (viz. obrázek 4) zahrnuje rozdělení blockchainu zneplatněním transakcí potvrzených uzly, které nebyly upgradovány na novou verzi softwaru protokolu. Uzly provozující předchozí verzi již nejsou nadále akceptovány novější verzí (Frankenfield, 2019a).

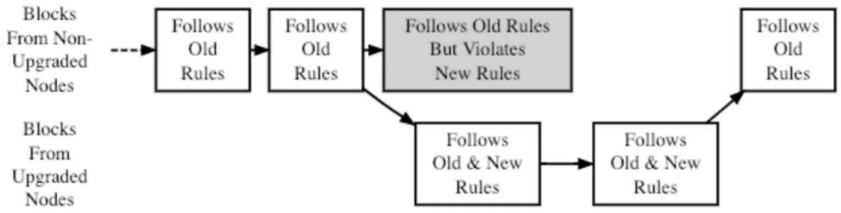


Obrázek 4 - Ukázka hard fork – neaktualizované uzly odmítají nová pravidla, což vede k rozdělení blockchainu (Frankenfield, 2019a).

Úplné uzly rozhodují o budoucnosti sítě hlasováním, pokud jich více jak 51 % nesouhlasí o návrhu, je návrh odmítnut nebo se vytvoří hard fork. V případě hard fork se blockchain rozdělí na další řetězec. Nejznámějším příkladem je Bitcoin Cash, který vznikl jako hard fork z Bitcoinu (Evans, 2019).

Soft fork

Na rozdíl od hard fork, soft work (viz. obrázek 5) nevyžaduje, aby se všechny uzly aktualizovaly a dohodly na nové verzi. Vyžaduje se pouze upgradování uzlů většiny těžařů, aby se prosadila nová pravidla. Jedná se pouze o takovou změnu protokolu, kde se zneplatní jen předchozí bloky či transakce. Soft fork je zpětně kompatibilní, protože staré uzly rozeznají nové bloky jako platné (Frankenfield, 2019b). 2^{256} klíč složen z 256 binárních jednotek.



Obrázek 5 - Ukázka soft fork – Bloky porušující nová pravidla se stanou zastaralými upgradovanou těžařskou většinou (Frankenfield, 2019b).

3.1.8 Hašování

Nedílnou součástí blockchain technologie je funkce hašování. Hašovací funkce je algoritmus, který má na vstup data libovolné délky a výstupem je bitový řetězec pevné délky nazývaný hašová hodnota. Vstupem může být například blok transakcí. Kryptografická hašová funkce musí mít následující klíčové vlastnosti:

- z hašové hodnoty musí být výpočetně nemožné získat vstupní data. Jedná se o klíčovou vlastnost v mechanismu konsensu PoW
- je výpočetně nemožné vytvořit stejnou hašovou hodnotu pro různé vstupy (Franco, 2014)
- hašová hodnota je vždy stejná pro stejný vstup
- rychle a efektivně vytvářet hašovou hodnotu ze vstupu
- malá změna na vstupu se projeví velkou změnou na výstupu (Lisk, 2019)

Hašování poskytuje odesílaným datům jistotu, že s nimi nebylo manipulováno dříve, než došla k příjemci. To se ověří tak, že se porovnají hašové hodnoty ze stejných dat odesílatele a příjemce, pokud se rovnají, lze s jistotou prohlásit, že s nimi po cestě nebylo manipulováno (Lisk, 2019) (Dasgupta, 2019).

V blockchainu je hašování používané k reprezentaci aktuálního stavu blockchainu a zajištění jeho neměnitelnosti. Každá transakce obsahuje určitou bitovou informaci, například

odeslanou částku, adresu příjemce a odesílatele, anebo časovou známku. Všechny tyto informace jsou zkombinované do hašové hodnoty, která se nazývá TXID (identifikační kód transakce). Tato hašová hodnota (TXID) se používá k identifikaci a potvrzení, že se transakce uskutečnila (Lisk, 2019).

První blok blockchainu se nazývá blok geneze, který po spojení a ověření svých transakcí vytvoří svou hašovou hodnotu. Když se vytvoří nový blok, tak se hašová hodnota předchozího bloku spojí se všemi transakcemi nového bloku a pomocí hašové funkce vznikne nová hašová hodnota. Tento proces se opakuje pro každý nový blok přidaný do blockchainu. Tím vznikne neměnitelný řetěz, kde hašová hodnota každého bloku odkazuje na hašovou hodnotu předchozího bloku (Lisk, 2019) (Lastovetska, 2019).

Součástí datové struktury blockchainu jsou ukazatele a propojené seznamy (viz. obrázek 6). Ukazatele jsou proměnné, které uchovávají informace o lokalitě jiné proměnné. Propojené seznamy jsou posloupnosti bloků, kde každý blok obsahuje specifická data a odkazuje na následující blok pomocí ukazatele. Na první blok (blok geneze) neodkazuje žádný ukazatel, protože se jedná o první blok v řetězci. Současně koncový blok má ukazatele bez hodnoty (Lastovetska, 2019).

Obrázek 6 - Datové struktury ukazatelů v blockchainu (Lastovetska, 2019).

Ukazatele neobsahují jen hašovou hodnotu adresy předchozího bloku, ale i hašovou hodnotu svého bloku. Pokud by chtěl útočník napadnout a změnit data nějakého bloku v řetězci, musel by změnit data i všech následujících bloků, protože by svým zásahem změnil hašovou hodnotu (Lisk, 2019).

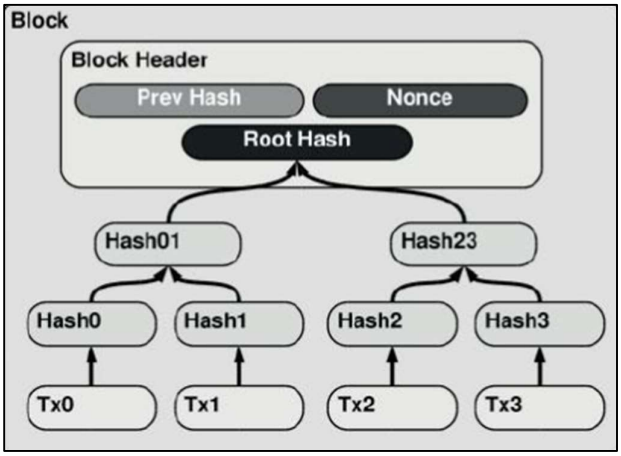
Merkleův strom

Merkleův strom, neboli hašový strom, je datová struktura hašových hodnot, která se v blockchainu používá k bezpečnému a efektivnímu ukládání dat (transakcí). Koncept Markleova stromu patentoval Ralph Merkle v roce 1979. Merkleův strom řeší problém, který by nastal, kdyby měla hašová hodnota v bloku obsahovat hodnotu bloku, hodnotu

předchozího bloku i všech transakcí v něm obsažených (Lisk, 2019). Tento přístup by přinesl několik nevýhod.

- Pokud by se změnila jedna z transakcí, bylo by potřeba aktualizovat bitový řetězec a znovu vypočítat hašovou hodnotu. Uzly by musely uchovávat bitový řetězec bloku v paměti. Pokud by byla nahrazena transakce uprostřed bitového řetězce za větší transakci, musela by být uprostřed řetězce alokována paměť, což je nákladná operace.
- K ověření, zda je transakce součástí bloku, by musel být k dispozici celý blok. Teprve potom, by bylo možné vypočítat hašovou hodnotu a ověřit ji.

Tyto problémy právě řeší Merkleův strom (Franco, 2014). Algoritmus prochází blok transakcí a vygeneruje hašovou hodnotu, která slouží k ověření platnosti těchto dat na základě původních transakcí. Blok transakcí není zahašován v jednom kroku, místo toho je každá transakce zahašována samostatně, přičemž jsou v dalších krocích transakce spojovány a hašovány společně až nakonec vznikne hašová hodnota pro celý blok transakcí (Lisk, 2019).



Obrázek 7 - Merkleův strom transakcí uvnitř bloku (Franco, 2014).

Na obrázku 7 lze vidět, že je nejdříve vytvořen binární strom složený z hašových hodnot individuálních transakcí (označené jako Tx0 až Tx3), které se nazývají listy. Binární strom je graf, ve kterém má každý rodič dva potomky. Hašová hodnota rodičovského uzlu je složena z hašových hodnot každého z potomků. Tento proces je znázorněn na obrázku jako Hash01, který je složený z Hash0 a Hash1. Nakonec je vytvořena hašová hodnota kořenového uzlu, tedy kořenový haš nebo Merkleův kořen (Franco, 2014).

3.1.9 Mechanismy konsensu

Protokol konsensu je tvořen sadou pravidel, které určují, jak funguje komunikace a přenos dat mezi uzly. Konsensu se dosáhne tak, že se většina uzlů shodne na tom, co je pravda, a co by mělo být zaznamenáno do blockchainu. Protože je blockchain decentralizovaný distribuovaný systém bez centrální autority, musejí se distribuované uzly shodnout na platnosti transakcí (Binance, 2019a). Mechanismus konsensu tedy představuje způsob, kterým uzly dosáhnou společné shody při validaci bloků a jejich transakcí, což umožňuje bezchybné fungování blockchain sítě, kde se uzly jednomyslně shodují pro jednu datovou hodnotu, a to i v případě, že jsou některé uzly nespolehlivé nebo jsou nedostupné.

Mechanismus konsensu v blockchainu plní dvě klíčové funkce.

- umožňuje aktualizaci blockchainu a zajišťuje pravost každého bloku v řetězci
- zabraňuje samostatné entitě k ovládnutí blockchainu a jeho rozdělení (Lisk, 2019)

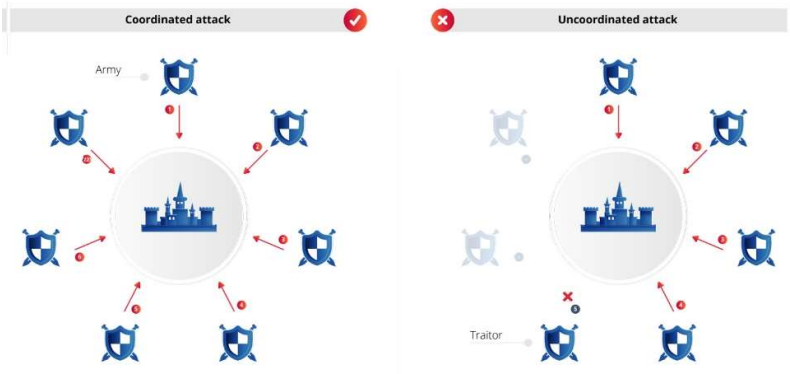
Existuje několik mechanismů konsensu. Mezi nejběžnější implementace patří PoW a PoS. Každý z nich má nějaké své výhody a nevýhody z hlediska zabezpečení, funkčnosti a škálovatelnosti (Binance, 2019a).

V případě využití blockchainu ve světě kryptoměn slouží mechanismy konsensu k ochraně blockchainu před útočníky. Jedním z útoku je například dvojitá útrata, což je útok, při kterém útočník utratí určitou sumu kryptoměny a poté se pokusí transakci zvrátit vysláním své vlastní verze blockchainu s tím, že tuto transakci nezahrne do řetězce. Mechanismy konsensu dále motivují účastníky ke správě blockchain sítě formou odměn a pobídek. Tím se zvyšuje konkurence při potvrzování dalších bloků v řetězci, což má pozitivní efekt na fungování sítě. Mechanismy jsou navrženy takovým způsobem, který je náročný jak z časového hlediska, tak z hlediska výpočetní náročnosti. Mechanismy se liší v závislosti na blockchainu, ve kterém se bloky validují. Všechny mechanismy však mají stejný cíl – zajistit absolutní jistotu, že všechny informace, které se ukládají do blockchainu, jsou čestné a přesné (Lisk, 2019).

BFT

Byzantine Fault Tolerance (zkráceně BFT) řeší problém, jak se mohou v distribuované decentralizované síti uzly shodnout na nějakém rozhodnutí, když je pravděpodobné, že mohou některé uzly fungovat nekorektně nebo jednat nečestně. Toto je otázka tzn. Problému

dvou armád (anglicky „Byzantine Generals' Problem"), ze kterého vychází BFT, znázorněného na obrázku 8.



Obrázek 8 - Vizualizace Problému dvou armád. Nalevo je úspěšný koordinovaný útok. (Lisk, 2019).

Problém dvou armád byl koncipován v roce 1982 jako logické dilema, ve kterém se řeší otázka, jak se mohou byzantští generálové shodnout na dalším tahu. Předpokladem je, že má každý generál svou armádu rozmístěnou okolo města určeného k dobytí. Cílem generálů je se jednotně shodnout buď na útoku, nebo ústupu. Pokud by alespoň jeden z generálů provedl jiný úkon než ostatní, znamenalo by to totální porážku. Problém ovšem nastává v komunikaci, protože jeden generál je schopný komunikovat s druhým jen prostřednictvím kurýra. Zprávy se tedy mohou ztratit, zpozdit, být zfalšovány nebo jinak znehodnoceny (Binance, 2019b) (Lisk, 2019).

V kontextu blockchainu je každý generál reprezentován jako uzel v síti, kde musejí všechny uzly dosáhnout společného konsensu ohledně aktuálního stavu systému. Pokud se tak nestane a uzly nedosáhnou společného konsensu, dojde k selhání. Je tedy zapotřebí, aby byly alespoň dvě třetiny uzlů spolehlivé a čestné. Pokud bude většina uzlů v síti škodlivých a nečestných, pak je systém náchylný k selhání (Binance, 2019b) (Lisk, 2019).

BFT je tedy vlastnost systému, která dokazuje, že je systém schopný pokračovat v činnosti i přesto, že některé uzly nemusí fungovat nebo se mohou chovat škodlivě. Existuje více možných řešení Problému dvou armád, a tím pádem i více způsobů návrhu systémů s BFT vlastností. To vede k mechanismům konsensu, které jsou dále popsány (Binance, 2019b) (Lisk, 2019).

PBFT

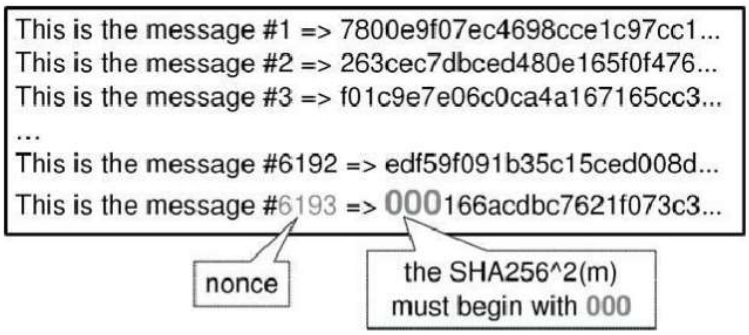
Practical Byzantine Fault Tolerance (PBFT) je mechanismus konsensu, který nezahrnuje žádný typ zdrojů, ale využívá mechanismus založený na přístupu BFT. Nejprve

je mezi uzly zvolen vůdčí uzel, který rozhoduje o validaci transakcí a publikuje blok mezi všechny zbylé uzly v blockchain síti. Transakce je do bloku zahrnuta jen v případě, že dvě třetiny těžících uzlů ověří její správnost. Protože se vůdčí uzel často mění, není tento přístup považován jako centralizovaný (Salman, 2019) (Zheng, 2017).

PoW

Proof of work (zkráceně PoW, česky „důkaz o práci“) je první mechanismus konsensu, který byl navržen a představen v Bitcoin a je široce používán dalšími kryptoměnami (Lisk, 2019). Je založen na myšlence, že jednou z možných obran před útoky je vyžadování, aby klient požadující službu prokázal, že byla provedena určitá práce. Tato práce může představovat nějaký výpočetně náročný problém. Tento problém musí být náročný na vyřešení ale výpočetně rychlý na verifikaci. Je-li tento systém správně navržen, nebude pro legitimní uživatele nepříjemností, ale pro útočníky bude odrazující (Franco, 2014) (Nakamoto, 2008).

Bitcoin používá k PoW částečnou hašovací inverzi (partial hash inversion). Částečná hašovací inverze vyžaduje, aby hašová hodnota bloku transakcí odpovídala určitému vzoru. Tento vzor, kterému odpovídá, určuje že hašová hodnota musí začínat určitým počtem nulových bitů. Název hašovací inverze znamená, že PoW musí invertovat (nalézt) určitý vzorec v části hašové hodnoty (Franco, 2014).



Obrázek 9 - Ukázka procesu částečné hašovací inverze v rámci PoW (Franco, 2014).

Na obrázku 9 je na řetězec „This is the message“ aplikovaná částečná hašovací inverze. Hašovací funkce použita v ukázce je SHA256[10]. Částečná hašovací inverze v tomto příkladu vyžaduje, aby prvních 12 bitů (3 hexadecimální znaky) bylo nulových. Náhodné číslo

[10] SHA (Secure Hash Algorithm, česky „bezpečný algoritmus hašování“) je jedna z kryptografických hašovacích funkcí. SHA-256 generuje unikátní hašovací hodnoty o pevné délce 256 bitů (32 bajtů). Je to jednosměrná funkce, nelze ji tedy zpětně dešifrovat (Asolo, 2018a).

(nonce) je připojeno k textu a slouží k vyřešení částečné hašovací inverze. Aby bylo nalezeno řešení, tedy vyřešena částečná hašovací inverze, musí se nalézt taková hodnota nonce, která společně s hašovou hodnotou zprávy odpovídá částečné hašové hodnotě, tj. hašová hodnota začíná alespoň třemi nulovými znaky. Na obrázku se zvyšuje hodnotu nonce, dokud není nalezeno řešení. Pro efektivní hašovou funkci je z hlediska výpočetního výkonu nepodstatné, zda se nonce hádá od jedničky, anebo se hádá náhodně (Franco, 2014).

Vyřešení částečné hašovací funkce je výpočetně náročné. Naopak ověření, zda byla provedena určitá práce k vyřešení, je výpočetně nenáročné. Obtížnost PoW je jednoduše nastavitelná úpravou počtu požadovaných nulových bitů, kterými musí vyřešená hašová hodnota začínat (Franco, 2014).

Obtížnost PoW je průběžně upravována, což zajišťuje, že rychlost vytváření nových bloků zůstává konstantní a úměrná množství výpočetní síly věnované síti. Je to dáno tím, že nově vytvořený blok je validní jen v případě, že jeho hašová hodnota je menší než cílová hašová hodnota stanovená protokolem (PoW) – tedy vzorem, který částečná hašovací inverze vyžaduje, což, jak již bylo zmíněno, odpovídá počtu počátečních nulových bitů hašové hodnoty. Toto souvisí s procesem ověřování a přidávání nových platných bloků do blockchainu, tj. těžby (mining) (Franco, 2014) (Zheng, 2017). Těžba je podrobněji popsána v kapitole 3.1.10.

PoS

Konsensuální mechanismus PoS byl představen v roce 2011 na fóru Bitcointalk s cílem vyřešit problémy spojené s PoW mechanismem. Proof of stake lze volně přeložit jako „důkaz o vkladu“ a slovo stake jako sázku či vklad. V PoS algoritmu je uzel, který validuje blok, a je vybírán pseudonáhodným volebním procesem ovlivněný kombinací faktorů. Mezi tyto faktory patří stáří vkladu (sázky), náhodnost a bohatství uzlu (Binance, 2019c).

Proces ověřování a přidávání bloků se v PoS nazývá „kování“, na rozdíl od těžení (PoW). Uzly, které se chtějí podílet na procesu kování, musí uzamknout určité množství kryptoměny jako svůj vklad do sítě. Čím větší je vklad uzlu, tím větší má uzel šanci, že se stane dalším validátorem, který uková nový blok do blockchainu. Aby nebyly zvýhodněné jen uzly s největším bohatstvím (uzamčeným podílem) jsou do výběrového procesu přidány další metody. Mezi tyto metody patří náhodnost a stáří vkladu (Saleh, 2018).

Metoda náhodnosti vychází z výběru validujícího uzlu dle kombinace nejnižší hašové hodnoty a nejvyššího vkladu. Protože je velikost vkladu veřejná informace, mohou ostatní uzly předpovědět, kdo bude dalším validátorem (Binance, 2019c) (Saleh, 2018).

Metoda stáří vkladu je založena na vynásobení počtu dní, po který byla měna uzamčena jako vklad s množstvím měny, které je vloženo. Když je uzel vybrán a vytvoří nový blok, jeho stáří vkladu je vynulováno a musí počkat určitou dobu, než bude moci vytvořit další blok. Tím se zabrání tomu, aby uzly s velkým vkladem ovládly blockchain (Li, 2017a).

Pokud je uzel vybrán jako validátor a vytvoří další blok, zkontroluje platnost transakcí uvnitř bloku, podepíše blok a přidá jej do blockchainu. Na oplátku uzel obdrží transakční poplatky, které jsou součástí bloku jako odměna. Když už uzel nechce být validátorem je po určité době jeho vklad společně s obdrženými odměnami uvolněn. Tato časová doba slouží k tomu, aby síť mohla ověřit, že uzel nepřidal do blockchainu podvodné bloky (Binance, 2019c) (Saleh, 2018).

Bezpečnost mechanismu je dosažena tím, že vklad uzlu funguje jako finanční motivace k nevytváření či validování podvodných transakcí, protože pokud síť detekuje podvodnou transakci, tak uzel ztratí část svého vkladu a právo se účastnit na dalším validování bloků. Dokud je tedy vklad vyšší než odměna, uzel by více ztratil, než by mohl získat podvodem (Binance, 2019c) (Lisk, 2019) (Li, 2017a).

I tak je ovšem možné, aby uzel mohl ovládat síť a potvrzovat podvodné transakce, a to v případě, že by uzel vlastnil většinu vkladu na síti. Tento druh útoku je znám jako Útok 51 %, který bude popsán v kapitole 3.1.12. V závislosti na kryptoměně by to bylo ovšem nepraktické, protože by musel podvodný uzel získat 51 % oběhu. Což by bylo nesmírně nákladné a většina účastníků sítě by pravděpodobně opustila síť, kdyby jedna strana vše zkupovala, zatímco jiní by zvyšovali cenu, aby odradili útočníka. Mezi hlavní výhody PoS patří energetická nenáročnost a bezpečnost. Metoda náhodnosti zvyšuje decentralizaci sítě, protože pro vytváření bloků již nejsou výhodné skupinové těžby (mining pools, viz kapitol 3.1.10) (Binance, 2019c) (Lisk, 2019) (Li, 2017a).

PoS je vhodný mechanismus konsensu k vyřešení BFT, protože jsou všechny validující uzly sledovány zbytkem sítě a jejich identity ve formě adres jejich peněženek známy. Vzhledem k tomu, že BFT vyžaduje, aby byly dvě třetiny ověřujících uzlů upřímných, sledování jejich individuálních identit pomáhá udržovat funkční stav sítě (Lisk, 2019).

DPoS

Delegated Proof of Stake (česky „delegovaný důkaz o vkladu“, zkráceně DPoS) vyvinul Daniel Larimer v roce 2014. Je to mechanismus konsensu, který využívá k dosažení konsensu hlasování v reálném čase kombinované se sociálním systémem reputace. Každý uzel, který drží určitý vklad, může mít vliv na dění v síti, proto může být považován za nejméně centralizovaný konsensuální protokol v porovnání s ostatními. Proto je považovaný za účinnější a demokratičtější verzi PoS mechanismu (Lisk, 2019) (Binance, 2019d) (Yang, 2019) (Zheng, 2017).

Úkolem delegáta je zabezpečit síť a validovat transakce. Na oplátku získává odměny, které bývají úměrně rozděleny mezi voliče. V jednom okamžiku může na síti existovat pouze určitý počet delegátů. Tento počet se liší dle implementace, například v případě Lisk[11] je to 101. Delegáti jsou voleni držiteli určitého vkladu. Hlasovací síla je úměrná velikosti tohoto vkladu, který je uzlem držen. Je zásadní, aby byli delegáti vybíráni s ohledem na zájmy sítě, protože udržují funkční a bezpečný chod sítě. V některých případech je zapotřebí, aby delegáti prokázali svou angažovanost tím, že vloží určitý vklad na časově uzamčený účet, který je zkonfiskován v případě nečestného chování (Lisk, 2019) (Yang, 2019).

Úkoly delegáta jsou následující:

- zajistit nepřetržitý provoz jejich uzlu
- shromažďovat transakce v síti do bloků
- ověřovat transakce a vysílat bloky do sítě
- spravedlivě a demokraticky řešit existujících problémy spojené s konsensem, což DPoS umožňuje

Delegáti nemají pravomoc měnit transakce, ale protože jsou validátory těchto transakcí a bloků, mohli by některé transakce vyloučit z bloku. To by však mělo jen velmi malý dopad, protože by tyto vyloučené transakce byly zahrnuty v dalším vytvořeném bloku. Transakce by se tak jen mírně zpozdila. Navíc by toto nečestné chování vedlo ke ztrátě důvěry v tohoto delegáta a jeho vyloučení zbytkem sítě v příštích volbách. Proto je mechanismus DPoS založený na hlasovacím systému přímo závislém na reputaci delegátů (Lisk, 2019) (Binance, 2019d) (Yang, 2019).

[11] Lisk je kryptoměna a blockchainová aplikační platforma využívající DPoS konsensus zajištěný 101 demokraticky zvolenými delegáty (Lisk, 2019).

DPoS je oproti mechanismu PoW schopný zpracovat větší množství transakcí. Výběr delegátů umožňuje, aby byly transakce validovány během několika sekund, namísto minut jako je tomu u PoW. PoW mechanismus je stále nejpoužívanější ve světě kryptoměn, protože je stále považován za nejbezpečnější mechanismus. Ovšem PoS a DPoS nachází uplatnění i v jiných případech užití, hlavně tam, kde je zapotřebí efektivně a rychle zpracovávat velké množství transakcí, například v oblasti internetu věcí (IoT) (Lisk, 2019) (Binance, 2019d).

3.1.10 Chytré smlouvy

Pojem chytré smlouvy byl poprvé konceptualizován na počátku 90. let Nickem Szabem, který koncept porovnával s digitálním prodejním automatem. Stejně jako v prodejním automatu je vybrána položka, vloží se požadované množství hotovosti, a pokud je platba zaregistrovaná, zboží je vydáno (Lisk, 2019).

Chytré smlouvy (anglicky „smart contracts“) formují a zabezpečují vztahy mezi uživateli v počítačové síti. Zásady chytrých smluv jsou odvozeny od právních principů, ekonomické teorie a bezpečnostních protokolů. Pomocí kryptografických a jiných bezpečnostních mechanismů lze zajistit algoritmicky specifikovatelné vztahy zabraňující porušování zásad, odposlechům nebo útokům třetích stran (Szabo, 1997).

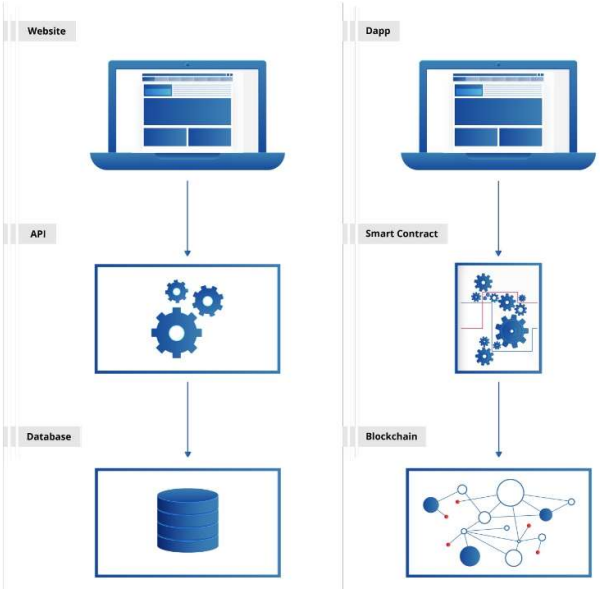
Chytrá smlouva je jedním z případů využití, které blockchain technologie přinesla. Je to smlouva napsaná v kódu a vložená do konkrétního blockchainu. Kód donucuje dodržování pravidel, podmínek, času a dalších potřebných zásad. Po splnění podmínek se automaticky spustí požadovaná funkcionalita. Protože jsou tyto podmínky psané v programovacím jazyce, tak jsou objektivnější a kvantifikovatelné, což je zásadní rozdíl od tradičních smluv psaných v lidském jazyce, které mohou být subjektivní a odlišně chápány (Lisk, 2019) (Mougayar, 2016).

Chytrá smlouva je uložena na decentralizovaném blockchainu, takže není zpracovávána a spravována centrální entitou, která by mohla mít zájem v manipulaci s daty. Protože blockchain využívá mechanismy šifrování, je chytrá smlouva a její historie odolná proti vnější manipulaci. Jakýkoliv pokus o její manipulaci je odhalen ostatními uzly v síti, které této změně mohou zamezit. Jakákoli manipulace by vyžadovala obrovské množství výpočetního výkonu a nebyla by finančně ani logisticky praktická. Chytrá smlouva tedy přidává na důvěryhodnosti sítě (Lisk, 2019) (Mougayar, 2016).

Automatizovaná, ověřitelná a důvěryhodná povaha chytrých smluv přináší spoustu případů využití, které aktuálně využívají centralizovaný systém, kde se strany musejí

spoléhat na prostředníky, jako jsou právní infrastruktury. Tento přístup přináší velké náklady na zpracování dokumentace a vymáhání smluv. Využitím chytrých smluv lze snížit riziko podvodu, nákladů spojených s dokumentací a vymáháním smluv. Dohody jsou prováděny a vymáhány automaticky. Tím se snižuje i plýtvání času, což je důležitý zdroj (Lisk, 2019) (Mougayar, 2016).

Dalším využití, které chytré smlouvy přinášejí, jsou blockchainové aplikace, které se nazývají „dapps“. Blockchainová aplikace (viz. obrázek 10) je uživatelsky orientované rozhraní, které spojuje koncového uživatele s blockchain technologií pomocí kombinace chytrých smluv. Dapps používá chytré smlouvy k připojení na konkrétní blockchain, na kterém jsou založeny. Poskytují koncovému uživateli bezpečnou, efektivní a flexibilní možnost, jak se připojit na blockchain přes front-end. Lze tak vyměňovat kapitál, majetek nebo cokoliv hodnotného ověřitelnou bezkonfliktní metodou bez nutnosti centrální entity představující prostředníka směny (Lisk, 2019) (Mougayar, 2016).



Obrázek 10 - Porovnání tradičního webu (vlevo) a blockchainové aplikace (dapp, vpravo) (Lisk, 2019).

3.1.11 Těžba (mining)

Těžba (anglicky „mining“) je v proces přidávání nových bloků do blockchainu, respektive verifikování transakcí. Proces těžení je rovněž zodpovědný za zvyšování objemu měny v oběhu. Je také jedním z klíčových prvků, které umožňují síti fungovat v architektuře P2P, tedy decentralizovaně, aniž by byla zapotřebí centrální autorita (Binance, 2019e). V

případě PoW konsensu implementovaného v Bitcoin se jedná o přispívání výpočetního výkonu horníky k vyřešení hašových hodnot, kteří jsou odměněny transakčními poplatky všech transakcí obsažených v bloku (Zheng, 2017).

Proces těžení

Kdykoliv je v síti provedena nová transakce, je tato transakce přijata těžaři, kteří transakce organizují do bloků. Těžaři tedy vytvářejí hašové hodnoty pro každou transakci v bloku. První transakce zaznamenaná v bloku se nazývá coinbase. Označuje se tak transakce, která odměňuje těžaře a zvyšuje objem měny v oběhu (Binance, 2019e).

Hašové hodnoty transakci jsou uspořádány do Merkleova stromu, který je popsán v kapitole 3.1.8. Zde jsou vytvářeny hašové hodnoty pro dvojice transakcí, tato hašová hodnota je poté uspořádaná do dvojice s další hašovou hodnotou, čímž vznikne jedna společná hašová hodnota. Tento proces se opakuje tak dlouho, dokud není dosaženo jedné kořenové hašové hodnoty, která představuje kombinaci všech předchozích hašových hodnot. Vytvořená kořenová hašová hodnota se poté společně s hašovou hodnotou předchozího bloku, číslem zvaným nonce a dalšími parametry umístí do hlavičky celého bloku. Opět se aplikuje hašovací funkce, jejíž výstupem je hašová hodnota identifikující nově vytvořený blok (Binance, 2019e) (Wang, 2019).

Aby byl blok považován za validní a byl úspěšně zařazen do blockchainu, musí být jeho hašová hodnota menší než hledaná hašová hodnota stanovena protokolem konsensu. Hledaná hašová hodnota se nazývá hašovací obtížnost (Binance, 2019e). Jak již bylo popsáno v kapitole o PoW protokolu (3.1.9), těžaři řeší problém částečné hašovací inverze. K nalezení řešení zvyšují číslo nonce, dokud nenaleznou takovou hašovou hodnotu bloku, která odpovídá nastavené obtížnosti. Obtížnost je stanovena určitým počtem nulových bitů, kterým musí výsledná hašová hodnota začínat. Tím více těžařů se zapojuje do procesu těžení, tím vyšší je obtížnost těžení (Franco, 2014). Obtížnost je protokolem pravidelně upravovaná tak, aby byla rychlost vytváření nových bloků konstantní a úměrná množství výpočetní síly v síti (Binance, 2019e) (Wang, 2019).

Poté co těžař úspěšně nalezne platnou hašovou hodnotu, vyšle blok do sítě. Všechny ostatní uzly zkontrolují, zda je hašová hodnota platná, a pokud ano, přidají vytvořený blok do své kopie blockchainu a pokračují v procesu těžení dalších bloků. Někdy se stane, že dva těžaři vyšlou platný blok do sítě ve stejnou chvíli, tím vzniknout konkurenční bloky, které jsou oba platné. Ostatní těžaři pokračují v těžení dalších bloků na základě bloku, který

přijmou jako první. Tato soutěž mezi těmito bloky pokračuje do doby, než je další blok těžen na základě pouze jednoho z bloků. Druhý blok, na základě, kterého se již dále netěží, se nazývá osiřelý blok nebo zastaralý blok. Těžaři tohoto bloku přepnout na řetězec vítězného bloku a pokračují v těžení (Binance, 2019e) (Wang, 2019).

Neexistují vstupní bariery k zahájení těžení, kdokoliv se může připojit a začít těžit. Není potřeba vyžadovat povolení či dodržovat soubor pravidel či postupů ke vstupu na síť. Rovněž nemohou již zavedené subjekty zakládat dohody, které by zabránily vstupu nových účastníků. Vstupem nových účastníků se snižují odměny pro všechny těžaře, kteří se v síti nacházejí. Tím se zvyšuje hodnota měny a těžaři tedy musí stále zrychlovat proces těžení, aby byli konkurence schopní. Hodnota měny se bude zvyšovat do té doby, dokud nebo mezní cena posledního těžaře, který vstoupí do procesu, rovna očekávané odměně. V tomto okamžiku síť dosáhne rovnováhy, která může být narušena pouze vnějšími faktory, jako je například další růst hodnoty těžené měny (Franco, 2014).

Mining pools

Mining pools, v překladu „těžební skupiny“ vnikly jako reakce na stále se zvyšujících obtížnost těžení a potřebu nákladných hardwarů k těžení. Těžební skupiny jsou shromáždění těžařů, kteří spojují svou výpočetní sílu do těžebních skupin a sdílejí mezi sebou získanou odměnu. Vytvářením těžebních skupin získávají těžaři předvídatelnější příjem než v případě, že by těžily jednotlivě. Sdílený příjem těžební skupiny je úměrný výpočetnímu výkonu jednotlivých těžařů v některých případech snížený o poplatek, který slouží jako příjem pro provozovatele těchto těžebních skupin. Další výhodou je účastnící těžebních skupin nemusejí uchovávat kopii celého blockchainu ani zpracovávat všechny příchozí transakce. Provozovatelé těžebních skupin těmto těžařům poskytují pouze kopie hlaviček bloků (Franco, 2014).

3.1.12 Útoky

Dvojitá útrata

Dvojitá útrata je pokus o útok, kdy se dvě různé transakce pokusí utratit stejné finance. Útočník se při tomto útoku pokusí utratit určitou sumu peněz a poté vyšle svou vlastní verzi blockchainu, která neobsahuje tuto transakci. Tento útok je o to častější v decentralizované síti, kde není žádná centrální autorita, která by tomuto zabránila. Obrana proti tomuto útoku je využívání mechanismů konsensu, například v případě Bitcoinu je to PoW protokol. Jak

již bylo popisováno, transakce jsou ukládány do bloků. Tím více je těchto bloků seřazeno do řetězce pomocí hašových funkcí, tím jsou transakce v těchto blocích více zabezpečeny. Pokud by chtěl útočník zmanipulovat nějakou transakci uloženou v určitém bloku v blockchainu, musel by následně znovu vytěžit všechny následující bloky od tohoto určitého bloku. Útočník by zároveň musel udržet krok se stále se rozšiřujícím řetězcem blockchainu, což by vyžadovalo obrovskou výpočetní sílu. Ovšem je stále teoreticky možné úspěšně provést dvojitou útratu. Dále jsou popsány typy útoků, které dovolují dvojitou útratu uskutečnit (Franco, 2014) (Asolo, 2018b).

Útok 51 %

Útok 51 % je útok, při kterém útočník ovládá alespoň polovinu výpočetního výkonu sítě. Útočník, který by ovládal takové množství výpočetního výkonu, by mohl provést svůj privátní fork blockchainu, který by dále soukromě budoval (těžil) sám. Mezi tím by mohl utrácet finanční prostředky na původním blockchainu budovaným poctivými těžaři. Tyto transakce by nezahrnoval do svého privátního forku blockchainu. Poté by svůj privátní blockchain, který by byl delší, vyslal zpátky do sítě. Protože by tento blockchain byl delší, těžaři by ho přijali jako validní a útočník by mohl znovu utratit finance v něm obsažené (Franco, 2014) (Asolo, 2018b) (Dasgupta, 2019).

Race Attack

Race Attack je druh útoku, kdy uzel příjme platbu za zboží či službu na základě nedostatečně ověřené transakce. Jedná se o transakci, která byla vyslána do sítě, ale doposud nebyla zahrnuta do žádného bloku. Útočník může zaslat transakci k uzlům nacházejících se v blízkosti obchodníka, a odlišnou transakci ostatním uzlům v síti. Transakce, kterou by obchodník přijal, by obsahovala platbu obchodníkovi za jeho zboží či službu. Druhá transakce by obsahovala platbu útočníkovi. Může se stát, že tato druhá platba bude ověřena, přijata ostatními uzly a zahrnuta do blockchainu jako validní transakce. Obchodník již poskytne útočníkovi zboží či službu v očekávání platby, ale nedostane zaplaceno. Obranou proti tomuto útoku je vyčkat, až bude transakce zahrnuta do alespoň jednoho bloku (Franco, 2014) (Asolo, 2018a) (Dasgupta, 2019).

Finney Attack

Finney Attack objevil Hal Finney v roce 2014. Útočník, který je také těžařem, vytěží blok, ale zatím ho nevyšle do sítě. Místo toho do bloku zahrne transakci, která zasílá platbu

z jedné své peněženky do druhé své peněženky. Než tento blok útočník odešle do sítě, zakoupí zboží či službu od obchodníka a provede platbu ze své peněženky na peněženku obchodníka. Obchodník na základě toho, že je tato transakce vyslána do sítě, poskytne útočníkovi zboží či službu. Útočník ovšem zároveň s transakcí vyšle do sítě podvodný blok, který obsahuje platbu mezi jeho peněženkami. Protože existuje časová prodleva mezi vytěžením bloku a jeho vysláním do sítě, může se stát, že těžaři nejdříve přijmou a validují blok obsahující podvodnou transakci a útok se tak podaří (Franco, 2014) (Asolo, 2018b) (Dasgupta, 2019).

Sobecké těžení

Při sobeckém těžení se útočník soustředí na rozšiřování své privátní větve blockchainu, kterou nezveřejňuje ve veřejné síti. V rozšiřování pokračuje, dokud je tato privátní větev delší než veřejná větev, na které pracují ostatní těžaři v síti. Když se délka veřejné větve přibližuje délce privátní větve, útočník tuto větev zveřejní. Všichni těžaři, kteří dodržují pravidla konsensu musí tuto delší větev přijmout, protože to dokazuje, že na ní bylo využito více výpočetního výkonu. Tím získá útočník výhodu, protože jeho výpočetní výkon nebyl promrhán jako v případě ostatních těžařů a je schopný sesbírat více odměn. Na rozdíl od výše popsaných útoků je tento útok cílený na těžaře namísto příjemců finančních prostředků (Franco, 2014) (Zheng, 2017).

Pro útočníka je výhodné zvát ostatní těžaře do své těžařské skupiny a pro těžaře je výhodné se k této skupině připojit. Toto může pokračovat až do doby, kdy těžařská skupina ovládá většinu sítě. Takto sobecká těžařská skupina poté neuvolňuje do sítě bloky, které vytěží, ale místo toho pravidelně uvolní celou větev. Takže ostatním těžařům se nevyplácí dále těžit a pravděpodobně opustí sít. Negativním důsledkem je soustředěná kontrola těžby, kdy skupina může transakce cenzurovat nebo vyřazovat z blockchainu. Vzniká tak rozpor s filozofii otevřenosti (Franco, 2014).

„Nic v sázce“

Útok zvaný „Nic v sázce“ je mířený na PoS protokol. Vzhledem k tomu, že v případě PoS protokolu není generování nových bloků výpočetně náročné, jako je tomu u PoW protokolu, je pro validujíc uzly výhodné vytvářet konfliktní bloky na všech možných větvích forku blockchainu bez následků, tedy jinak řečeno nic není v sázce. Další motivací je, že pokud by uzel vytvářel bloky (v případě PoS „koval“) jen na jedné větvi a jedna z ostatních

větví by se prodloužila, tak by uzel neprofitoval z doby strávené kováním na kratší větvi (Li, 2017a).

Útočník, který chce uskutečnit dvojitou útratu, vytvoří fork blockchainu. Na původní větvi uskuteční transakci a na druhé své větvi zašle peníze sám sobě. Útočník pak ková jen na své větvi, zatímco ostatní ve svém zájmu kovají na obou větvích, nakonec se může tato větev útočníka stát nejdelší větví. Ostatní ji tedy přijmou jako hlavní větev, protože je nejdelší. Obranou je přístup, který vývojáři kryptoměny Ethereum řeší ve svém protokolu PoS. Uzly, které validují, musejí vložit určitý vklad. Pokud validátor vytvoří podvodnou transakci, přijdou o část svého vkladu společně se schopností pokračovat v účasti na konsensu. Vytvoří se tak ekonomický podnět k čestnému chování (Ray, 2018) (Martinez, 2018b).

The "Long Range" Attack

Další typ útoku na PoS protokol se nazývá „Long Range". Útočník se zde pokouší pozměnit historii blockchainu vytvořením forku z již vytvořeného bloku. Lze toho docílit tak, že má útočník kontrolu nad účtem, který má velký vklad v minulém bloku. Pokud v současném bloku nemá žádný vklad, ale má velký vklad v bloku minulém, může tento podíl využít na vygenerování nového bloku. Útočník může kompromitovat soukromý klíč staršího účtu, který momentálně nevlastní žádný podíl, ale v minulosti vlastnil většinový podíl v síti. Účet, který nevlastní žádný podíl, tedy vykazuje nulový vklad, je méně chráněný nežli ostatní účty. Takový účet umožňuje útočníkovi vytvořit fork z minulého bloku, který může předstihnout současný řetězec svým podílem v minulosti. Protiopatřením proti útoku může být například zavedení kontrolních bodů k jeho omezení dosahu do minulosti. Kontrolní bod označuje blok, do kterého je blockchain neměnný. Tento kontrolní blok může být spravován centrálním serverem, který průběžně definuje správnou podobu blockchainu. Dalším možným protiopatřením je, aby uzly nepřijímaly změnu forku, která měnila bloky příliš daleko v minulosti (Li, 2017b) (Sharma, 2018).

3.1.13 Bezpečnost

Bezpečnost blockchain technologie je dosažena za použití mnoha mechanismů včetně pokročilých kryptografických technik a matematických algoritmů. Je základní technologií v kryptoměnových systémech, ale také v aplikacích, kde je zapotřebí dosažení neměnitelnosti a bezpečnosti dat (Binance, c2017-2019). Některé takovéto systémy/aplikace

jsou popsány v kapitole 3.2.2., kde jsou zmíněné oblasti jako chytrá města, chytré zemědělství, zdravotnictví, dodavatelské řetězce a mnoho dalších stále přibývá (Hassija, 2019). Mnoho konceptů hraje role v bezpečnosti blockchain technologie, ale mezi zásadní z nich patří neměnitelnost, konsensus a kryptografie. Neměnitelnost poukazuje na schopnost blockchainu zabránit změnám již potvrzených transakcí (transakce viz. kapitola 3.1.6). Konsensus poukazuje na schopnost uzlů v distribuované síti dosáhnout shody na stavu sítě a platnosti transakcí. Neměnitelnost a konsensus společně vytváří datovou bezpečnost v blockchain síti, protože konsensus zajišťuje dodržování pravidel a shodu zúčastněných stran na stavu sítě a neměnitelnost zajišťuje integritu dat transakcí již potvrzených bloků dat (bloky viz. kapitola 3.1.5) (Binance, c2017-2019) (Baliga, 2019). Mechanismy konsensu a jejich variace jsou podrobně popsané v kapitole 3.1.9.

Bezpečnostní mechanismy blockchainu se spoléhají na kryptografii, konkrétně na kryptografické hašovací funkce. Hašové hodnoty vytvořené pomocí hašovacích funkcí slouží jako unikátní identifikátory bloků dat, které se navzájem řetězí. Hašování se podílí na bezpečnosti blockchain technologie, zajištění neměnitelnosti, na mechanismech konsensu a také na asymetrické kryptografii k digitálnímu podepisování transakcí (Binance, c2017-2019). Podrobně jsou principy hašovacích funkcí popsány v kapitole 3.1.8 a s tím související těžení v kapitole 3.1.10.

3.2 Internet of Things

Pojem Internet of Things (dále jen IoT) v překladu Internet věcí odkazuje na rozsáhlou množinu všech druhů zařízení připojených k Internetu, jako jsou například bezpečnostní kamery, inteligentní chladničky a tiskárny, autonomní vozidla atd. IoT zahrnuje i taková zařízení, která doposud nebyla objevena a není představa o jejich budoucí podobě. Klíčovou myšlenkou IoT je nabídnout užitečné funkce prostřednictvím nepřetržité konektivity kdykoliv a kdekoliv a dále usnadnit životy lidí přechodem z informační společnosti na skutečně propojenou společnost (Penttinen, 2017) (Hassija, 2019).

IoT zařízení mohou mít různé podoby a účely. Mohou provádět měření fyzikálních vlastností a shromažďovat informace prostřednictvím senzorů. Jiná zařízení mohou na základě získaných dat ze senzorů analyzovat informace, které mohou být posílány dále ke zpracování. Některá zařízení mohou být schopná provádět více takových úkolů. Obecně by zařízení mělo být schopné komunikovat prostřednictvím připojení. Může komunikovat na malou vzdálenost, jako jsou například technologie NFC (Near Field Communication), RFID

(Radio Frequency Identification), Bluetooth, nebo na větší vzdálenost pomocí Wi-Fi či celulárních rádiových sítí (GSM[12]), anebo fixní sítě jako například ADSL (Asymmetric Digital Subscriber Line). Zařízení IoT je tak schopné doručit nebo přijmout zprávy od protistrany, což může být jiné zařízení (Penttinen, 2017).

3.2.1 Trendy

Aktuální trend naznačuje narůstající počet připojených zařízeních k Internetu. Dle reportu IoT Analytics (Lueth, 2019) bylo v roce 2018 celkem ve světě používáno 17,8 miliard připojených zařízeních, z toho 7 miliard IoT zařízení, kam se dle uvedeného reportu nezapočítávají smartphony, tablety, laptopy nebo pevné telefonní linky. V roce 2025 by počet IoT zařízení mohl vzrůst až na 22 miliard, resp. 34,2 miliard připojených zařízení celkem. Tento odhadovaný počet IoT zařízení nebere v úvahu zařízení, která byla pořízena v minulosti, ale již nejsou používána. Předpověď dle International Data Corporation (2020) uvádí 41,6 miliard připojených IoT zařízení v roce 2025.

Dle reportu Particle (Particle, 2019) publikovaného během letošního roku respondenti uvedli, že používají IoT ke vzdálenému monitoringu, preventivní údržbě a sledování majetku. Z toho vyplývá, že většina respondentů využívá IoT primárně k tvorbě obchodních hodnot než spotřebitelských hodnot.

Na webu OutGrow (Adithya, 2019) se uvádí, že předpokládané příjmy v odvětví IoT vzrostou z 892 miliard USD z roku 2O15 na 4 biliony USD v roce 2025.

Nárůst počtu IoT zařízení a trendy v oblasti IoT podtrhují významnost IoT zejména v oblastech, které jsou dále popsány v kapitole 3.2.2 a jsou považovány za bezpečnostně kritické oblasti. S takto širokým spektrem IoT aplikací přicházejí otázky bezpečnosti a soukromí, protože bez důvěryhodných a interoperabilních IoT ekosystémů klesá poptávka a potenciál takovýchto IoT systémů (Hassija, 2019).

3.2.2 Bezpečnostně kritické aplikace

Chytrá města

Spojením počítačů, smartphonů a internetu s každodenními aktivitami vzniká automatizovaná společnost. Takovou společnost v rámci měst se označuj chytré město.

[12] GSM (francouzsky „Groupe Spécial Mobile“) je globální systém mobilní komunikace (Vodafone, 2019).

Chytré město zahrnuje rozsáhlé využívání výpočetních a komunikačních zdrojů pro zvýšení celkové kvality života obyvatel. Chytré město se týká automatizace a propojení celého města s možností vzdáleného ovládání prvků města přes internet. Zahrnuje například ovládání dopravy uvnitř města (chytrá správa dopravy), sledovat znečištění ovzduší (chytré prostředí), monitorovat seismické otřesy budov (chytrá detekce katastrof), obsluhovat chytré parkování, zefektivnit nakládání s odpady, a spoustu dalších případů použití (Suresh, 2014).

Ovšem chytrá města s sebou přinášejí větší nároky na udržení bezpečnosti a zajištění soukromí občanů. Při pokusech o útoky na citlivá data občanů může dojít k únikům osobních údajů, lokalizačních dat, finančních záznamů apod (Hassija, 2019).

Životní prostředí

Všechny aplikace IoT v této oblasti jsou úzce spojeny s životy lidí a zvířat. Zahrnují například detekce požárů v lesích a jiných přírodních katastrof, monitorování úrovně sněhu v horských oblastech a ovzduší, prevence sesuvů půdy atd. Je vyžadováno, aby informace poskytované těmito systémy byly důvěryhodné, protože případná narušení mohou mít vážné následky z důvodu závislosti vládních orgánů na těchto informacích. Případné neoprávněné narušení těchto systémů a vysílání falešných zpráv může způsobit finanční ztráty vlády a podniků, nebo dokonce ztráty na životech a majetcích (Hassija, 2019).

Chytré sítě

Chytré sítě měří, monitorují a spravují spotřebu elektrické sítě na spotřebičích nebo elektrických rozvodech, úrovně vody, paliv a plynu v úložných nádržích a cisternách (Hassija, 2019). Chytré sítě umožňují distribučním firmám regulovat a spravovat dodávky energie ke koncovým zákazníkům. Je zde kladen důraz na sběr dat v reálném čase, jejich zpracování a následně pomocí automatizovaného řídícího systému regulovat energetické nároky, uskladňovat nevyužitou energii a dodávat vyprodukovanou energii do rozvodných sítí. Obousměrná digitální komunikace mezi odměrným místem a řídícím datovým centrem na straně správce distribuční soustavy může zkvalitnit služby pro zákazníky a ochránit životní prostředí. Distributor může pružněji reagovat na výkyvy ve spotřebě energie, detekovat pokusy o manipulaci s údaji měřičů a efektivně regulovat dodávku energie a cen (Plchút, c2001-2019).

Chytré sítě jsou ovšem zranitelné vůči fyzickým i kybernetickým útokům. Útočník může vniknout do komunikačních systémů spotřebitele nebo poskytovatele služeb, upravit shromážděná data a tím způsobit finanční ztráty na straně obou stranách (Hassija, 2019).

Útočník by mohl i způsobit přerušení dodávky energie například pro celé město, což by mělo velké následky i při krátké době přerušení (Penttinen, 2017).

Bezpečnostní systémy

Bezpečnostní systémy mohou zahrnovat senzory k monitorování různých událostí jako otevření dveří nebo oken, pohybu v monitorované oblasti, ale i například bezpečnostní kamery v domácnostech nebo průmyslových oblastech atd. Další využití IoT může být při detekci úniku nebezpečných látek nebo radiace. Takovéto bezpečností systémy využívající IoT mohou bránit neoprávněným osobám ke vstupu do chráněných míst, chránit citlivá data a majetek, nebo zdravý a životy lidí například v průmyslových oblastech. Narušení bezpečnostních systémů útočníkem může v takových aplikacích různé závažné důsledky (Penttinen, 2017) (Hassija, 2019).

Logistické a dodavatelské řetězce

IoT lze využít i v maloobchodu, logistických a dodavatelských řetězcích. Lze tak monitorovat zboží po celou dobu jeho přepravy v rámci celého logistického řetězce od výrobce až ke koncovému zákazníkovi v maloobchodech. Zboží má tak zaručenou čerstvost a kvalitu. Monitorují se podmínky a pohyb zboží ve skladech, takže lze optimalizovat doplňování zásob. Pro zákazníky existují nákupní aplikace, které jim asistují při nákupu na základě jejich preferencí, zvyklostem, alergiím atd., a zaručují bezvadnost zboží. Systémy IoT umožňují sledovat veřejnou dopravu, hustotu provozu, kvalitu tras atd. což vede k možnosti zefektivnit a optimalizovat dopravu zboží v logistickém řetězci (Penttinen, 2017) (Suresh, 2014).

Systémy logistických řetězců využívající IoT jsou také náchylné na útoky. Útočník může zmanipulovat data tak, aby znevýhodnil konkurenci či úmyslně napáchal škody. Pokud nejsou dostatečně zajištěny bezpečnostní prvky, může útočník odcizit osobní údaje o zákaznicích, jako jejich kontaktní údaje, údaje o kreditkách a účtech apod (Hassija, 2019).

Chytré zemědělství

Chytré zemědělství zahrnuje monitorování vlhkosti půdy, kontroly mikroklimatických podmínek, selektivní zavlažování a vlhkosti i teploty klima. Tyto možnosti přináší zemědělcům a podnikům vysoké výnosy spolu se snížením finančních ztrát. Kontrolováním environmentálních podmínek zvyšuje kvalitu rostlinné produkce a výnosnost plodin.

Monitorovat lze i živočišnou produkci, například senzory připevněných na zvířatech, které sbírají data o činnosti a zdravotním stavu farmářské zvěře (Hassija, 2019) (Doseděl, 2018).

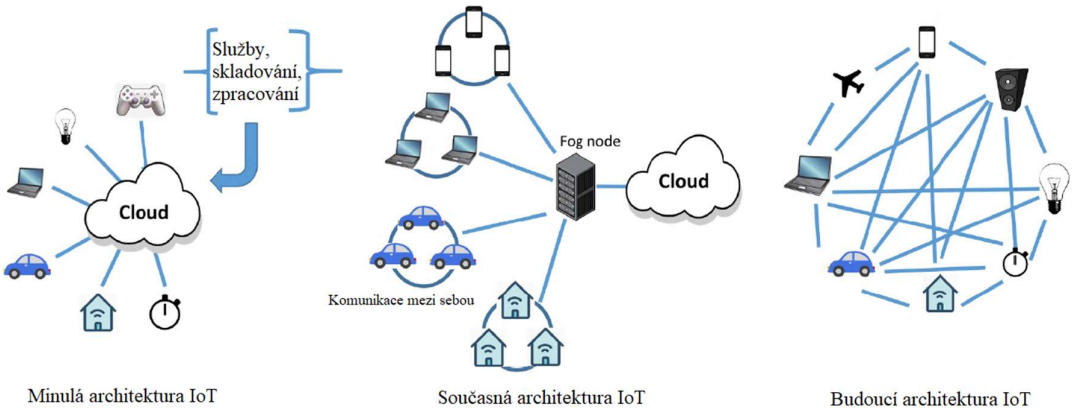
Obdobně jako v případě chytrých logistických řetězců, i zde může dojít k manipulaci dat a tím získání konkurenční výhody útočníka nebo k finančním ztrátám zemědělského podniku. Obzvláště když se na základě těchto dat vytvářejí rozhodování o následných činnostech v produkci, anebo jsou takto řízené autonomní zemědělské stroje (Hassija, 2019).

Další oblasti

Možnosti a oblasti využití IoT rostou velmi rychle a pronikají do většiny stávajících průmyslových, ale nejen průmyslových, odvětví. Mezi další oblasti, kam proniká IoT lze zařadit i domácí využití, tzv. chytré domácnosti. Za chytré domácnosti se označují domy vybavené senzory a zařízeními, které jsou schopné komunikovat mezi sebou, s řídící jednotkou, ale i s majitelem domu například skrze smartphone. Vytvářejí moderní, efektivní a bezpečné prostředí domova (Smolová, 2018).

Další příkladem oblastí může být například zdravotnictví, kde IoT umožňuje sbírat cenná data o pacientech v reálném čase, provádět analýzy, sbírat terénní data atd. IoT nachází uplatnění i v průmyslové automatizaci, kde zrychluje výrobu a napomáhá jí optimalizovat, tím se zvyšuje kvalita produktů a zvyšuje se návratnost investic (Upasana, 2019).

3.2.3 Běžné typy architektury



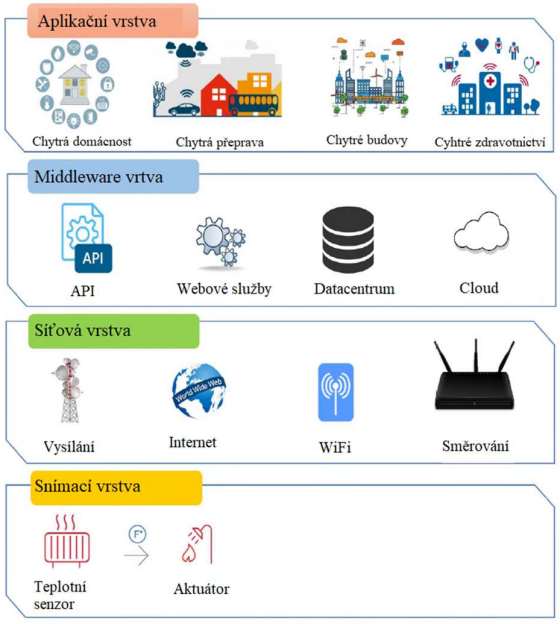
Obrázek 11 - Typy architektur IoT sítí (Hassija, 2019).

Na obrázku 11 jsou znázorněné různé architektury IoT. V minulosti byla většina dat zpracovávaná výpočetním výkonem v centrální jednotce. V takovéto architektuře je středem

architektury cloud, kde dochází k zpracování získaných dat z IoT zařízení. Cloud nabízí služby jako infrastrukturu, platformu, software a uložiště (Jain, 2017).

Později se začínal využívat tzv. fog computing. Zde jsou data částečně zpracovány a analyzovány senzory a síťovými branami. Přibývají zde nové vrstvy, které se starají o monitorování, předzpracovávání a bezpečnost. Dochází tak například k monitorování stavu sítě, filtrování, zpracování a analyzování senzorových dat, dále k replikaci, distribuci a ukládání dat, a dále se zde provádí šifrování a dešifrování a je zajištěna datová integrita a soukromí. Monitorování a předzpracovávání je prováděné na okraji sítě před odesláním dat do cloudu (Jain, 2017) (Bonomi, 2012).

V budoucnu se od IoT zařízení nevyžaduje jen připojení k Internetu a komunikace s ostatními zařízeními v rámci lokální sítě, ale navíc i přímá komunikace s ostatními zařízeními skrze Internet (Hassija, 2019).



Obrázek 12 - Vrstvy IoT systémů (Hassija, 2019).
Překlad a úprava – autor práce.

Obecně se IoT ekosystémy a prostředí skládají ze čtyř vrstev popsaných na obrázku 12. V první nejnižší vrstvě se nacházejí různé senzory a aktuátory, které provádějí různé funkcionality na základě získaných dat či informací. Druhá komunikační vrstva slouží k přenosu nashromážděných dat. Většina IoT řešení implementuje i třetí vrstvu, která obsahuje middleware. Middleware zde slouží jako most mezi síťovou a aplikační vrstvou. Nejvyšší čtvrtá vrstva se skládá z konkrétních end-to-end IoT aplikací jako například chytré sítě, chytrá doprava apod. Všechny tyto vrstvy mají různé specifické bezpečnostní problémy.

Navíc kromě samotných vrstev se zde nacházejí i různé brány (gateway), které se starají o pohyb dat mezi vrstvami. I těmto branám hrozí různé bezpečnostní hrozby (Hassija, 2019).

3.2.4 Bezpečnostní hrozby

V současnosti u různých IoT aplikací chybí framework či standard, který by zajišťoval bezpečnost IoT zařízení, která nyní neobsahují vestavěné bezpečnostní prvky v podobě firewallů, anti-virusů a různých dalších typů ochrany jako je tomu například u smartphonů, osobních počítačů apod. IoT aplikace se skládají z mnoha různých produktů od různých výrobců a společností. Na všech vrstvách zmíněných v kapitole 5.3 se nachází rozmanitá množina produktů a technologií. To zahrnuje mnoho senzorů, aktuátorů a okrajových zařízení (edge uzly), ale i různé komunikační standardy jako jsou mobilní sítě, WiFi, Bluetooth atd. Konektivita je zajištěna na různých vrstvách technologiemi jako jsou například Zigbee, Z-Wave, NFC, RFID, 6LOWPAN atd. V případě aplikační vrstvy není možné použít generický HTTP protokol, protože není určený pro zdrojově omezené prostředí. Proto se v aplikační vrstvě využívají alternativní protokoly určené pro prostředí IoT, mezi které patří například MQTT, SMQTT, CoAP, XMPP, AMQP, M3DA, JavascriptIoT, atd (Hassija, 2019) (Swamy, 2017).

Z důvodu této rozmanitosti protokolů, technologií a zařízení v IoT aplikacích je potřeba nacházet rovnováhu mezi pořizovacími náklady, bezpečnostní, spolehlivostí, soukromím, pokrytím, latencí a spousty dalšími metrikami. Zlepšení jedné metriky může znamenat degradaci jiné metriky. IoT aplikace je tak odolná proti bezpečnostním hrozbám jako její nejslabší článek (Hassija, 2019). Příkladem je možnost v případě IoT aplikace chytré domácnosti odhalit WiFi heslo domácnosti skrze chytrý dveřní zámek (Kumar, 2019). Na druhou stranu implementace velkého počtu bezpečnostních kontrol a protokolů může znamenat zvýšení nákladů a latence aplikace, a tím její nepoužitelnost pro uživatele (Hassija, 2019).

Každá vrstva v IoT architektuře má své specifické technologie, které přinášejí různé problémy a některé možné slabiny zabezpečení (Frustaci, 2018). Dále jsou jednotlivé vrstvy analyzovány vzhledem k jejich možným bezpečnostním slabinám a útokům.

Snímací vrstva

Snímací vrstva obsahuje fyzické IoT senzory a aktuátory. Senzory většinou pracují na technologiích jako identifikace na rádiové frekvenci (RFID), bezdrátové senzorové sítě

(WSN) atd. Nacházejí se zde různé typy senzorů sloužících ke sběru různých typů dat za účelem měření. Z důvodu zdrojově omezených uzlů a distribuované organizační struktuře vycházejí ze snímací vrstvy následující bezpečnostní hrozby (Hassija, 2019) (Frustaci, 2018).

- ***Zachycení uzlu:*** Útočník může zachytit nebo zaměnit uzel se škodlivým uzlem. Uzel se pak tváří jako součást systému, ale je ovládán útočníkem. Je to dáno především komplikovaným procesem autentizace zařízení v distribuovaném prostředí. Zařízení s podvrženou identitou může útočníkovy sloužit ke škodlivým útokům.
- ***Vložení škodlivého kódu a falešných dat:*** Útočník kompromituje uzel fyzickým vložením škodlivého kódu do paměti zařízení, především během vzdálené aktualizace zařízení, kdy je zařízení zranitelné. Poté může útočník skrze zařízení vysílat nesmyslná či chybná data do IoT systému a tím způsobit jeho selhání nebo provést DDoS útok.
- ***Odposlouchávání a rušení:*** Útočník odposlouchává a zachycuje přenášená data během procesu vysílání nebo autentizace zařízení.
- ***Vyčerpání zařízení:*** Útočník může například pomocí vložením a spuštění nekonečné smyčky do zařízení nebo umělého zvýšení spotřeby energie vybít baterii zařízení (Frustaci, 2018) (Hassija, 2019).

Síťová vrstva

Snímací vrstva slouží k přenášení nashromážděných dat získaných ze snímací vrstvy k výpočetní jednotce, která data zpracovává, skrze komunikační síť (3G, WiFi atd.). Hlavní bezpečnostní hrozby síťové vrstvy jsou následující (Hassija, 2019) (Frustaci, 2018).

- ***Směrovací útok:*** Jedná se o útok, kdy škodlivé uzly změní směrovací cesty během procesu shromažďování a přenášení dat.
- ***Útok na přenos dat:*** Data, která jsou přenášená z jedné lokace do druhé, jsou náchylnější na kybernetické útoky a úniky, než je tomu u již uložených dat na lokálních serverech nebo cloudu.
- ***Útok DoS/DDoS:*** Jedná se o útok, kdy útočník přehltí servery velkým počtem nežádoucích požadavků. Tím dojde k nefunkčnosti serveru a znepřístupnění služby ostatním uživatelům. Pokud je k přehlcení serveru využito více zdrojů, tak se používá termín DDoS (distribuované odepření služby). Nedostatečně nakonfigurované IoT zařízení mohou útočníkovi sloužit jako brány ke spuštění těchto cílených útoků.

- ***Phishing:*** Útočník může pomocí podvodné stránky získat uživatelský účet spolu s přístupovým heslem. Tím je IoT prostředí používané uživatelem zranitelné vůči útokům (Hassija, 2019) (Frustaci, 2018).

Middleware vrstva

Middleware v IoT slouží jako abstraktní vrstva mezi síťovou vrstvou a aplikační vrstvou. Může poskytovat silný výpočetní výkon a datové uložiště. Middleware vrstva zahrnuje perzistentní datové uložiště, systémy front, strojové učení atd. Nakažením nebo ovládnutím middleware vrstvy může útočník získat kontrolu nad celou IoT aplikací. Patří sem různé bezpečnostní hrozby (Hassija, 2019).

- ***Man-in-the-middle (MITM):*** Útočník se může ovládnutím zprostředkovatele komunikace mezi klienty stát jejím prostředníkem a získat tak kontrolu nad veškerou komunikací bez znalosti klientů (Swamy, 2017).
- ***Vsunutí malwaru do cloudu:*** Útočník získá kontrolu, vsune škodlivý kód (malware) nebo virtuální stroj do cloudu. Tím může útočník získat přístup k servisním požadavkům služby oběti a zachytit citlivá data, které může upravit.
- ***Přehlcení cloudu:*** Stejně jako v případě DoS útočník zasíláním velkého množství požadavků přehltí cloudové servery a tím je znepřístupní (Hassija, 2019).

Brány (gateways)

Brány poskytují hardwarové a softwarové řešení pro IoT zařízení, které navzájem propojují společně s cloudovými službami a uživateli. Slouží také k šifrování, dešifrování IoT dat a k překladu protokolů pro komunikaci mezi různými vrstvami (Fife, 2019). Brány zahrnují heterogenní IoT systémy jako například LoraWan, ZigBee a Z-Wave. Níže jsou popsány některé bezpečnostní hrozby (Hassija, 2019).

- ***End-to-End šifrování:*** K zajištění důvěrnosti přenášených dat je zapotřebí, aby byla zašifrovaná zpráva dešifrována pouze konkrétním příjemcem. Ovšem v některých IoT systémech brány zprávu dešifrují a znovu zašifrují za účelem překladu mezi různými protokoly.
- ***Update firmwaru:*** Brány, které aplikují aktualizovaný firmware na zdrojově omezených IoT zařízení, by z bezpečnostních důvodů měli vždy ověřit platnost elektronických podpisů zařízení (Hassija, 2019).

Aplikační vrstva

Aplikační vrstva poskytuje služby požadované koncovými uživateli. Patří sem různé IoT aplikace a prostředí jako chytré domácnosti, chytré sítě, chytré zemědělství atd. Níže jsou popsány některé bezpečnostní hrozby týkající se aplikační vrstvy, které mohou specifické dle různých aplikací (Hassija, 2019) (Frustaci, 2018) (Swamy, 2017).

- ***Datové úniky:*** Útočník může snadno odcizit kritická a soukromá data, pokud zná bezpečnostní slabiny služby nebo aplikace. Obranou mohou být techniky nebo protokoly k šifrování dat, správě soukromí, autentizaci atd.
- ***Útok DoS/DDoS:*** Tento typ útoku, který již byl popsán výše, se týká i aplikační vrstvy.
- ***Vložení škodlivého kódu:*** Nedostatečné kontroly kódu aplikace mohou útočníkovi dovolit vložit škodlivý skript například pomocí techniky XSS (cross-site scripting) a neoprávněně se zmocnit účtu k IoT systému (Hassija, 2019) (Frustaci, 2018).
- ***Sniffing:*** Útočník může použít síťový analyzátor k monitorování síťového provozu a získání přístupu k důvěrným datům (Swamy, 2017).

3.2.5 Přístupy k zabezpečení

Bezpečnostní hrozby narůstajícího počtu IoT aplikací vyžaduje zlepšení a posílení jejich struktur a frameworků k dosažení spolehlivosti a bezpečnosti. Dle různých autorů je v tomto ohledu zapotřebí:

- Provádět penetrační testy IoT zařízení ke kvantifikaci rizika spojeného s nasazením těchto zařízení v různých IoT aplikacích.
- Využívat techniky šifrování nejen v rámci jednotlivých vrstev a protokolů, ale v rámci celého systému, tzn. end-to-end.
- Autentizovat zařízení při komunikaci s jiným zařízením například pomocí digitálních certifikátů.
- Zajistit dostatečnou škálovatelnost IoT bezpečnostních frameworků k zamezení bezpečnostních hrozeb při zvyšujícím se počtu uživatelů IoT aplikace po jejím zveřejněním.
- Využívat šifrovací techniky jako RSA, SHA256 nebo hašové řetězce k zabezpečení uživatelských a systémových dat před nepovoleným zásahem. IoT zařízení by měla sesbíraná data vysílat bezpečnou a šifrovanou cestou (Hassija, 2019).

- Změnit paradigma z centralizovaného přístupu na decentralizovaný z důvodu dosahovaní nákladových a kapacitních omezení při stále narůstajícím počtu nasazených IoT aplikací. Při decentralizovaném přístupu mohou zařízení vzájemně komunikovat bezpečně a automaticky. Tím klesnou náklady na správu aplikací a omezit problémy s kapacitními omezeními (Kshetri, 2017).
- Zvážit rizika vyplývající z využívání cloudových služeb při ukládání a načítání dat z cloudu. Protože je cloud veřejná platforma, může dojít k ohrožení uložených dat. Bezpečnost dat uložených na cloudu lze zvýšit jejich zašifrováním a cloud nesmí mít možnost data dešifrovat (Wang, 2018).
- Zejména v případě decentralizovaného přístupu identifikovat mechanismy k ověření datových toků, protože existují různé scénáře, kde senzory v rámci IoT aplikace začnou odesílat nebo sbírat chybná data. Tyto chybná data mohou vést k nežádoucím výsledkům (Suhail, 2016).
- Zvážit využití technik a mechanismů vycházejících z oboru umělé inteligence jako nástroje k zabezpečení IoT zařízení, vytvoření autonomního systému, který potřebuje jen minimální zásah člověka, a ke snížení analytické a komunikační zátěže IoT prostředí (Xiao, 2018).

3.3 Integrace blockchain s IoT technologií

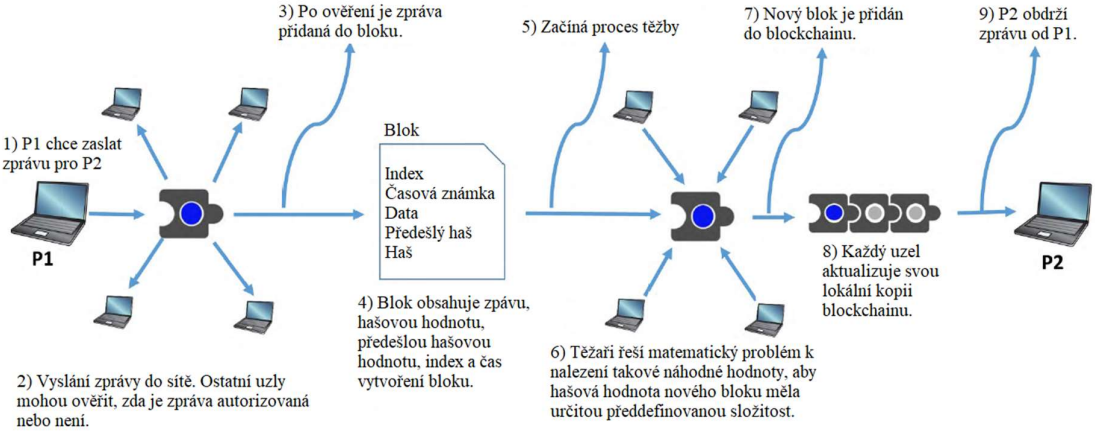
V rámci literatury existuje několik různých přístupů a technik k řešení bezpečnostních hrozeb vyplývajících z IoT aplikací. Jednou z popisovaných možností je využití řešení založeném na technologii blockchain. Mezi další možností patří řešení založené na tzv. fog computing, řešení založené na strojovém učení a řešení založené na tzv. edge computing (Hassija, 2019). V rámci této práce je dále popisováno řešení založené na využití blockchain technologie.

3.3.1 Zabezpečení IoT využitím blockchain technologie

Blockchain i IoT jsou technologie, které se podílejí na celkovém zlepšení transparentnosti, viditelnosti, úrovně pohodlí a důvěryhodnosti pro uživatele. IoT zařízení poskytují v reálné čase data ze senzorů a blockchain těmto datům poskytuje zabezpečení použitím distribuované, decentralizované a sdílené databáze (Miller, 2018). V některé literatuře se tato sdílená databáze označuje výrazem „účetní kniha“. Účetní knihu si lze představit jako chronologicky uspořádanou kolekci všech transakcí v sítě. Je veřejná pro všechny uzly v síti, které zároveň zajišťují její platnost (Panarello, 2018).

Tato databáze obsahuje všechny záznamy v blockchainu. Záznamy jsou v zde chronologicky uspořádané a označené časovým razítkem. Pomocí kryptografického hašování (3.1.8) je každý záznam úzce spojen s předchozím záznamem. Jednotlivé transakce (3.1.6) jsou uloženy ve stromové datové struktuře nazývané Merkleův strom (3.1.5). Kořenový haš stromu je uložen v bloku v rámci blockchainu. K ověření, zda bylo neoprávněně manipulováno s transakcemi, stačí ověřit jejich kořenový haš. Je to dáno tím, že i změna pouze jedné transakce v rámci stromu změní hašové hodnoty. Transakce nebo záznamy ověřuje správce blockchainu nebo těžaři (3.1.11) vygenerováním hašové hodnoty, což umožňuje, aby se poslední transakce stala součástí kompletní databáze (účetní knihy). Z důvodu přítomnosti kryptografických hašových hodnot (klíčů) v každém bloku, je pro útočníka příliš obtížné a časové náročné manipulovat s bloky (Hassija, 2019) (Orman, 2018).

Na obrázku 13 je popsán proces od inicializace transakce až po její zanesení do blockchainu.



Obrázek 13 - Pracovní postup blockchainu (Hassija, 2019). Přeloženo autorem práce.

3.3.2 Benefity integrace Blockchainu s IoT

Blockchain přináší některé benefity při použití v IoT aplikacích. Mezi některé klíčové výhody patří například:

- Blockchain může sloužit jako vhodné řešení k ukládání a přenášení dat vysílaných z IoT zařízení (Hassija, 2019).
- Blockchain svou distribuovanou povahou odstraňuje jediný bod selhání, na rozdíl od IoT aplikací založených na cloudovém řešení (Dinh, 2018).
- Data mohou být zašifrované hašovou funkcí a uložena v cloudu. Hašový klíč pak může být namapován s původními daty. Pokud dojde ke změně dat, změní se tím i hašová hodnota. Tím je dosaženo soukromí a bezpečnost dat. Navíc je každá sada dat uložených v blockchainu ověřena různými těžaři v síti.
- Blockchain může sloužit jako prevence proti ztrátě dat, která vyplývá z vlastnosti IoT zařízení. Protože se většinou jedná o zdrojově omezená zařízení, může dojít ke ztrátě dat jak odesílatelem, tak příjemcem. Ale v případě blockchainu, jakmile je blok přidán do řetězu, není možné jej odstranit.
- Blockchain zabraňuje spoofingu tím, že je každé zařízení nebo uživatel zaregistrován v blockchainu, takže se mohou zařízení navzájem snadno identifikovat a autentizovat bez potřeby centrální autority.

- Komunikace mezi uzly v rámci blockchain sítě probíhá šifrovaně pomocí veřejných a soukromích klíčů. Tím je zamezeno neautorizovanému přístupu ke komunikaci.
- Blockchain odstraňuje potřebu centralizovaného cloudového serveru. Naopak vytváří decentralizovanou P2P síť, kde jsou data zašifrovány kryptografickými hašovými funkcemi a distribuovány mezi všemi uzly v síti (Hassija, 2019).

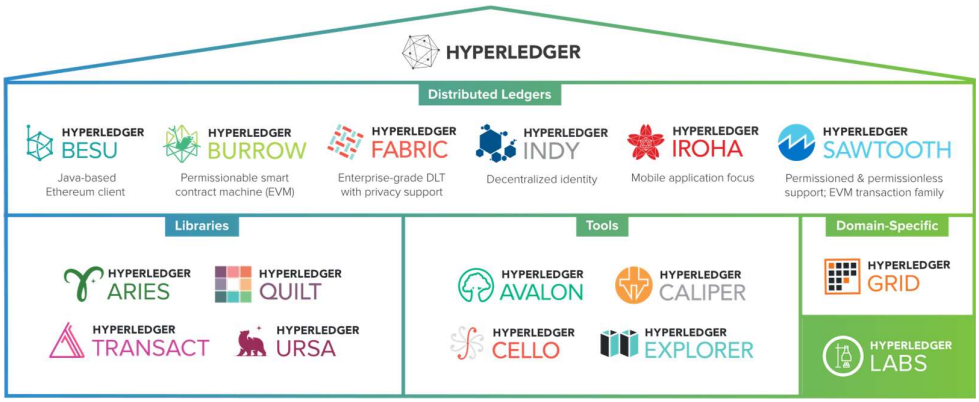
3.3.3 Současný stav výzkumu a vývoje spojujícího blockchain a IoT

Různí autoři se ve svých prací zabývají tímto spojením z různých hledisek. Například Novo (2018) navrhuje novou architekturu pro správu rolí a oprávnění v IoT v plně distribuovaném systému řízení přístupu založeném na technologii blockchain. Autoři Cui, Guin, Skjellum a Umphress (2019) představují různé výzvy, kterým čelí systémy IoT, a shrnují výhody zavedení blockchainu do infrastruktury IoT. Dalším příkladem jsou autoři Qu, Tao a Yuan (2018) navrhující model blockchainu založeném na hypergrafech a jeho případy použití v síti chytré domácnosti. Autoři Shen, Wang, Li a Chen (2018) navrhují schéma využívající blockchain k bezpečnému nahrávání dat v systémech chytrých domácností. Autoři Ourad, Belgacem a Salah (2018) navrhují řešení založené na blockchainu, které umožňuje autentizaci a zabezpečenou komunikaci s IoT zařízeními. Mnoho dalších autorů se věnuje výzkumu, ve kterém se spojují blockchain a IoT technologie.

Integrací blockchainu s IoT za použití blockchain platformy Hyperledger Fabric se věnuje také několik autorů. Například autoři Klaokliang, Teawtim, Aimtongkham a So-In (2018) navrhují distribuovanou architekturu autorizace pro IoT na Hyperledger Fabric. Další příkladem jsou autoři Attia, Khoufi, Laouiti a Adjih (2019) navrhující bezpečnostní architekturu pro aplikaci IoT monitorující zdravotní péči. Architektura využívá Hyperledger Fabric, kam se ukládají získaná citlivá data ze zařízení. Dále autoři Makhdoom, Abolhasan, Abbas a Ni (2019) popisují životaschopnost integrace blockchainu s IoT a související výzvy. V práci uvádějí předpoklad, že Hyperledger Fabric splňuje většinu požadavků IoT na autentizaci, autorizaci, správu identit, důvěrnost dat atd. Mnoho dalších autorů se věnuje výzkumu, ve kterém se spojují blockchain a IoT technologie s použitím blockchain platformy Hyperledger Fabric.

3.3.4 Blockchain platforma Hyperledger Fabric

Hyperledger je open-source kolaborativní úsilí k vytvoření pokroku v blockchain technologiích. Hyperledger vzniká pod záštitou The Linux Foundation od roku 2016 jako kolaborativní projekt několika společností. Patří jsem společnosti jako například IBM, Cisco, Intel, Red Hat, ale i velké společnosti ze sektoru financí, bankovnictví, výroby atd (Hyperledger, 2018).



Obrázek 14 - Zastřešující projekt Hyperledger (Hyperledger, 2018).

Hyperledger zahrnuje a propaguje řadu obchodních blockchain technologií, včetně frameworků distribuovaných účetních knih, chytrých smluv, klientských a uživatelských knihoven, grafických rozhraní a ukázkových aplikací (viz. obrázek 14) (Hyperledger, 2018).

Tato práce se zaměřuje z projektu Hyperledger na blockchain platformu Hyperledger Fabric, s tím související Hyperledger Composer a Hyperledger Playground.

Hyperledger Fabric je privátní blockchain platforma pro řešení distribuované účetní knihy (*ledger*) podpořené modulární architekturou přinášející vysokou míru důvěrnosti, pružnosti, flexibility a škálovatelnosti. Je navržený tak, aby podporoval různé implementace komponent a přizpůsobení komplexnosti v závislosti na odlišné ekonomické ekosystémy (Hyperledger Fabric, 2017).

Vzhledem k tomu, že je Hyperledger Fabric privátní blockchain platforma (3.1.1) určená především pro použití v podnicích, tak je vhodná pro konsorcia s uzavřenou součinností. V případě Hyperledger Fabric se mohou pouze *peer* uzly, které jsou považovány za součást konsorcia, účastnit procesu konsensu, zatímco klienti předkládají své transakce do sítě (Hyperledger Fabric, 2017).

Uzly jsou v privátní blockchain síti nejdříve verifikovány, registrovány a nepředpokládá se jejich velký počet v porovnání s veřejnou blockchain sítí. Proto jsou zde implementovány alternativní konsensuální mechanismy, jako například různé varianty BFT

(3.1.9), Raft, Paxos atd (Baliga, 2019). Tyto konsensuální mechanismy nejsou relativně náročné na výpočetní výkon jako například PoW (3.1.9) ve veřejné Bitcoin síti (Hyperledger Fabric, 2017) (Androulaki, 2018).

Protože má Hyperledger Fabric modulární a konfigurovatelnou architekturu, lze zde zvolit mechanismus konsensu na základě konkrétního případu užití a modelu důvěryhodnosti. V případě, že je blockchain síť implementovaná v rámci jedné organizace, je vhodnější zvolit protokol mechanismu konsensu odolnému proti selhání, tzn. crash fault-tolerant (CFT), který nevytváří nadbytečný odpor vůči výkonnosti a propustnosti sítě, jako je tomu u protokolu byzantské odolnosti vůči chybám (BFT) (Hyperledger Fabric, 2017) (Androulaki, 2018).

Vzhledem k vlastnostem konsensuálních mechanismů, které Hyperledger Fabric využívá, není zapotřebí nativní kryptoměna k motivaci těžení nebo dodržovaní chytrých smluv. Tato absence kryptoměny snižuje rizika některých útoků a implementace platformy nevyžaduje mimořádné provozní náklady, na rozdíl od jiných řešení (Hyperledger Fabric, 2017) (Androulaki, 2018).

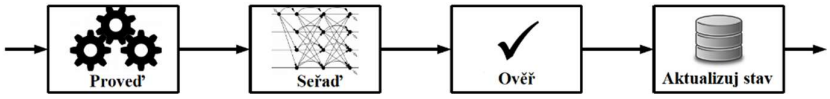
Mezi některé zmiňované výhody blockchain platformy Hyperledger patří následující:

- Jedná se o populární open-source platformu, která je vyvíjená pod The Linux Foundation jako kolaborativní projekt významných společností z různých oblastí, zejména IBM (Baliga, 2019).
- Má modulární a všestrannou architekturu, která vyhovuje širokému spektru případů průmyslového použití (Kuhrt, 2019). Je to dáno modularitou všech komponent jako například konsensuální vrstva, chytré smlouvy, datové uložiště, API a služby identit (Baset, 2018).
- Je flexibilní, protože podporuje několik programovacích jazyků jako Go, Java a JavaScript (Melnik, c2015-2020).
- Jedná se privátní blockchain platformu, takže jsou účastníci sítě předem známí a nikdo jiný nemá přístup k síti (Melnik, c2015-2020).
- Transakce nevyžadují využití kryptoměn a těžení, proto je škálovatelný a může zpracovat velké množství transakcí, což vyžaduje většina podnikových aplikací (Prerna, 2019).

Chytré smlouvy a architektura

Chytré smlouvy se v případě Hyperledger Fabric platformy nazývají *chaincode*. Ten implementuje obchodní logiku blockchain aplikace a funguje jako distribuovaná aplikace, která získává svou důvěru z podstaty blockchain technologie a konsensu mezi uzly v síti, tedy tzn. *peer*. (Hyperledger Fabric, 2017)

Architektura *chaincode* se liší od ostatních chytrých smluv, které podporuje většina známých blockchain platforem jako například Ethereum, Tendermint atd. Tyto platformy využívají architekturu *seřaď-vykonej*. V případě Hyperledger Fabric se využívá architektura *vykonej-seřaď-ověř*, která řeší problémy s odolností, flexibilitou, škálovatelností, výkonem a důvěrností architektury *seřaď-vykonej*. Ta rozděluje tok transakcí na tři kroky (viz. obrázek 15).



Obrázek 15 - Architektura vykonej-seřaď-ověř (Androulaki, 2018).

- ***Vykonání*** transakce a tím ověření její korektnosti. V tomto kroku je navíc odstraněna nedeterminičnost transakcí k zachování konzistentnosti výsledků před krokem *seřazení*. Odstranění nedeterminičnost je zásadní k možnosti vytvoření chytrých smluv v standartních programovacích jazycích (Go, Node.js, Java). V případě nedeterministických transakcí by totiž mohlo dojít k forku blockchainu nebo by bylo zapotřebí využívat doménově specifický jazyk jako například Solidity (Ethereum).
- ***Seřazení*** transakce pomocí protokolu konsensu do posloupnosti schválených transakcí seskupených do bloků a rozeslání bloků *peer* uzlům.
- ***Ověření*** schválených transakcí, které aktualizují stavy, *peer* uzly vůči aplikační politice specifické aplikace před jejím zařazením do účetní knihy (Androulaki, 2018).
- ***Aktualizace stavu*** představuje aktualizaci účetní knihy přidáním bloku transakcí do lokální účetní knihy (*ledger*).

Konsensus a komponenta *Orderer*

Konsensus je v Hyperledger Fabric jako zásuvný modul, který je oddělený od *peerů*, kteří provádějí transakce a aktualizují účetní knihu. Úkolem je vytvořit dohodu o posloupnosti seřazených transakcí a potvrzení správnosti bloku transakcí. S konsensem je

spojená služba tvořící pořadí transakcí, tzn. *ordering service*. Konkrétní komponenta této služby se nazývá *orderer*. *Orderer* lze považovat za důležitou komponentu, protože dosahuje konsensu v síti a distribuuje bloky transakcí všem *peerům* (uzlům) (Androulaki, 2018). Proto je také považována za nejslabší bod a nejpravděpodobnější bod případného útoku na síť.

Vzhledem k modulární architektuře lze volit mezi komponentou řazení podporující vlastnosti CFT (crash fault-tolerant přeloženo jako odolnost proti selhání) a BFT (byzantine fault-tolerant přeloženo jako byzantská odolnost vůči chybám). Služba řazení transakcí s vlastností CFT je implementovaná pomocí distribuované streamovací platformy (Apache) Kafka a centralizované služby (Apache) ZooKeeper. Nicméně i tato implementace je centralizovaná, protože je plně pod kontrolou jedné organizace v síti, která nemusí být důvěryhodná (Thummavet, 2019).

Služba řazení transakcí s vlastností BFT, která je dostupná až od verze Hyperledger Fabric 1.4.1, je implementovaná pomocí konsensuálního protokolu Raft. Tato implementace je již plně decentralizovaná, protože mohou různé organizace přispívat svými uzly k procesu řazení a tím dosáhnout konsensu i v případě škodlivých nebo vadných uzlů (Baset, 2018) (Thummavet, 2019).

Poslední možností je implementace služby řazení pomocí tzn. Solo. Jedná se o centralizovaný *orderer* běžící na jednom uzlu a používaný pro vývoj (Androulaki, 2018). Transakce jsou chronologicky seřazeny do bloku.

Členská služba a certifikační autorita

Další zásadní komponentou v platformě Fabric je *Membership Service Provider* zkráceně *MSP*, což je poskytovatel členských služeb, který poskytuje a udržuje identity všech uzlů v síti (klientů, *peerů* a *ordererů*) k autentizaci a autorizaci. Identita slouží k ověření, zda transakce pochází z validního zdroje. Na každém uzlu v síti je komponenta, která ověřuje transakce a jejich integritu, a podepisuje a ověřuje blockchain operace. Tím je v privátní síti Hyperledger Fabric dosaženo autorizované interakce mezi uzly, typicky s digitálními podpisy (Baset, 2018) (Androulaki, 2018).

Základní implementace MSP poskytuje standardní metody infrastruktury veřejných klíčů (PKI) k autentizaci na základě digitálních podpisů. Certifikační autorita v rámci Fabric se nazývá *Fabric-CA* (Santos, 2019). Architektura sítě s ohledem na konkrétní implementaci Hyperledger Fabric v rámci modelového případu užití je detailně popsaná v kapitole 4.2.2.

4 Praktická část

Praktická část práce se skládá z následujících částí. Nejdříve je navržená aplikace integrující blockchain a IoT technologii. V rámci této části je představena obchodní logika modelového případu užití dodavatelského řetězce. Dále je detailně popsána technická stránka navrhovaného řešení, konfigurace komponent a nástrojů architektury, architektura celého řešení a její procesy, definice modelu sítě a simulace IoT zařízení. V poslední části je popis provedeného experimentu, použitého benchmark nástroje a identifikace sledovaných metriky měření. Dále jsou pomocí měření (testů) vymezeny výkonnostní hranice architektury. Poté je pospán průběh experimentu a získané výsledky experimentu, které jsou poté vyhodnoceny.

4.1 Návrh aplikace k experimentu

Navrhovaná aplikace představuje modelový případ řešení integrující blockchain technologii s IoT. Na základě funkční aplikace je vytvořen experiment k ověření vhodnosti integrace.

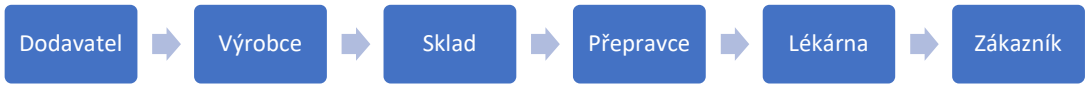
Navržená aplikace vychází z bezpečnostně kritické aplikace – logistického a dodavatelské řetězce popsané v kapitole 3.2.2. IoT lze využít v dodavatelských řetězcích k nepřetržitému monitorování přepravovaného zboží nebo zásob a zaručení jeho bezvadnosti.

4.1.1 Obchodní logika modelového případu užití

V rámci modelového případu užití se vychází z problematiky efektivního řízení dodavatelského řetězce, který je komplexní v každém sektoru. Při řízení dodavatelského řetězce v sektoru zdravotnictví se ke komplexnosti přidává faktor rizika, protože kompromitovaný dodavatelský řetězec může mít přímý dopad na bezpečnost pacientů a zdravotní výsledky. Právě v této oblasti by mohla být blockchain technologie spolu s IoT řešením, které zajistí lepší zabezpečení, integritu, provenienci dat a funkčnost (Clauson, 2018).

Výběr případu užití použitého jako základu pro následný experiment vychází z poznatků získaných z literatury zabývající se touto problematikou. Například (Clauson, 2018) a (Archa, 2018).

Je tedy vytvořena obchodní logika dodavatelského řetězce lékáren v rámci farmaceutického průmyslu. V praxi se jedná o komplexní distribuční síť zahrnující mnoho účastníků, kteří přímo i nepřímo ovlivňují vytvoření a přepravu léků.



Obrázek 16 - Zjednodušený diagram distribuční sítě farmaceutického průmyslu. Vytvořeno autorem práce.

Na obrázku 16 je znázorněn zjednodušený distribuční řetězec dodávající léky zákazníkům lékárny. Během procesu dochází k výrobě léků, které jsou poptávány zákazníky. V první fázi dodavatel dodává potřebné suroviny k výrobě léčiv výrobci, který je následně zpracuje na finální produkt. Léčiva dále putují do skladu, kde se uchovávají před jejich distribucí do lékáren. Poté jsou léčiva přepravcem distribuována v rámci distribuční sítě do lékáren, kde se dále uchovávají a jsou nabízena zákazníkům.

Po celou dobu distribuce léků v rámci distribučního řetězce je zapotřebí dodržovat přísná pravidla a podmínky pro uchovávání a přepravu, aby byla zajištěna jejich bezvadnost, čerstvost, a aby byly splněny podmínky kontrolních úřadů. K tomuto účelu jsou využívány IoT zařízení monitorující podmínky, ve kterých jsou léky uchováváno a přepravováváno. V rámci navrhovaného řešení se monitoruje teplota, vlhkost vzduchu a informace o lokalizaci léků. Nasbíraná data jsou poté ukládána v reálném čase v obchodní blockchain síti. To umožňuje všem účastníkům zpětně dohledat původ a podmínky, při kterých byly léky přepravovány, v nezměnitelné a důvěryhodné podobě.

Sledováním léků od výroby až po prodej zákazníkům lze zabránit nelegální distribuci padělaných léčiv zákazníkům, které představují velké riziko pro spotřebitele a způsobují ztrátu příjmů legitimních výrobců.

Existují obsáhlejší a komplexnější případy distribučního řetězce, než jak je znázorněno na diagramu. Ovšem cílem je navrhnout modelový příklad řešení. Proto je v práci při návrhu řešení zohledněna jen část distribuční sítě s těmito účastníky:

- ***Sklad*:** Účelem tohoto účastníka řetězce je skladování léčiv v nezměněném stavu. Dále udržovat záznamy o lécích a monitorovat teplotu a vlhkost vzduchu, při které jsou léky skladovány společně s lokalizačními informacemi.
- ***Přepravce*:** Úkolem přepravce je distribuce léků ze skladu do lékárny. Během přepravy je monitorována teplota a vlhkost vzduchu společně s lokalizačními informacemi.

- ***Lékárna*:** Účastník, který přijímá léky od přepravce a dále je skladuje a poskytuje zákazníkům. I zde je zapotřebí monitorovat teplotu a vlhkost prostředí k zajištění bezvadnosti léků.

4.2 Technické aspekty řešení

V této kapitole je detailně popsaná technická stránka navrhovaného řešení dodavatelského řetězce. V rámci podkapitol je popsán technická architektura celého řešení a její procesy.

4.2.1 Konfigurace komponent a nástrojů architektury

Architektura, včetně použitých komponent, navrhované aplikace založené na blockchain platformě je prezentována na obrázku 17. V této kapitole jsou detailněji popsány jednotlivé komponenty a nástroje architektury spolu s jejich konfiguracemi. Architektura, na které je experiment proveden, je postavená na prostředí popsaného v kapitole 3.3.4.

Dále jsou popsány konfigurace a použité verze komponent a nástrojů blockchain platformy Hyperledger Fabric nasazených na zvoleném vývojovém prostředí. Součástí Hyperledger Fabric jsou i nástroje Hyperledger Composer a Hyperledger Playground, které jsou blíže popsány v následujících kapitolách.

Konfigurace Hyperledger Fabric

Pro lokální spuštění Hyperledger Fabric slouží Docker Engine určený k běhu prostředí dockeru a Docker-Compose. Ten poskytuje vývojové prostředí k definování a spuštění Docker kontejnerů a obrazů na virtuálním stroji. Aplikační služby se konfigurují v YAML souboru pomoci Docker-Compose. Sada vývojových nástrojů Hyperledger Fabric SDK pro Node.js poskytuje API k interakci s Hyperledger Fabric blockchain sítí.

Ve výchozím stavu obsahuje Ubuntu 16.04 Python ve verzi 3.5.1. Ovšem Hyperledger Fabric SDK pro Node.js vyžaduje k npm operacím Python 2.7. Ve vývojovém prostředí je tedy použitá verze Python 2.7. Sestavení a použité verze knihoven/nástrojů jsou v tabulce 1.

Nástroj	Verze
Docker Engine	19.03.4
Docker-Compose	1.13.0
cURL	7.50.3
Node.js	8.16.2
Npm	6.13.0
Python	2.7.12
Hyperledger Fabric	v1.2

Tabulka 1 - Seznam nainstalovaných nástrojů a jejich verzí.

Konfigurace nástroje Hyperledger Composer

Součástí nástroje je příkazová řádka (CLI) pro Hyperledger Composer, která představuje knihovnu sloužící k administrativním, provozním a vývojovým účelům.

Pro vygenerování kostry frontend aplikace slouží nástroj Yeoman. S tím souvisí generátor knihovny, který slouží jako Yeoman modul pro sadu Yeoman generátorů. Tyto generátory se používají k vytváření šablon určených k použití s Hyperledger Composer.

Další komponenta slouží k lokálnímu spouštění Hyperledger Composer REST Serveru, která vystaví logiku obchodní blockchain sítě jako REST API pro HTTP nebo REST klienty. Tato API komponenta je založena na frameworku LoopBack obsahující funkci k připojení k prostředí Hyperledger Composer a skript k dynamickému shromáždění modelů aktiv, účastníků a transakcí. Sestavení a použité verze knihoven/nástrojů jsou v tabulce 2 přehled. Tabulka navazuje na tabulku 1.

Nástroj	Verze
CLI nástroje	0.20.9
Yeoman	3.1.0
Generátor knihovny	0.20.9
Hyperledger Composer REST Server	0.20.9

Tabulka 2 - Navazující seznam nainstalovaných nástrojů a jejich verzí.

Konfigurace nástroje Hyperledger Playground

Playground je nástroj pro Hyperledger Composer poskytující grafické uživatelské rozhraní sloužící ke konfiguraci, nasazení, testování a definování struktury obchodní sítě v prohlížeči. Ačkoli lze Playground využít k vývoji obchodní sítě, na některé úkony je vhodnější vývojové prostředí jako například Visual Studio Code s rozšířením pro Hyperledger Composer, které poskytuje zvýraznění syntaxe, automatické doplňování, úryvky kódu a kontrolu chyb. Hyperledger Playground se spouští na vývojovém prostředí

lokálně v prohlížeči na adrese *http://localhost:8080* přes příkazovou řádku příkazem *composer-playground*. Sestavení a použité verze knihoven/nástrojů jsou v tabulce 3.

Nástroj	Verze
Hyperledger Playground	0.20.9
Visual Studio (VS) Code	1.39.2
Hyperledger Composer VS Code rozšíření	0.19.12

Tabulka 3 - Verze Hyperledger Playground a dalších nástrojů.

Poznámka k použitým verzím

K provedení experimentu není použita nejnovější verze blockchain platformy Hyperledger Fabric. Nejnovější verze v2.0 je první hlavní vydání Hyperledger Fabric od v1.0 a byla vydána 29. ledna 2020. Tedy nejnovější zásadní verze vyšla v době psaní této práce, a proto již nebylo z časových důvodů možné nejnovější verzi implementovat. Dalším důvodem je, že Hyperledger Fabric v2.0 již nepodporuje framework Hyperledger Composer (Hyperledger Fabric, 2017).

Nejnovější verze Hyperledger Composer 0.20.9 byla v době práce, konkrétně 29. srpna 2019, označena za zastaralou. To znamená, že žádný ze správců již aktivně nevyvíjí nové funkce ani neposkytuje podporu prostřednictvím GitHub. Ovšem stále je funkční a lze tedy implementovat i používat (Hyperledger Composer, 2020).

Pokud je vývojový server Hyperledger Fabric vytvořený pomocí nástroje Hyperledger Composer 0.20.9, tak je ve výchozím nastavení Hyperledger Fabric vytvořen ve verzi 1.2.1. Hyperledger Composer ve verzi 0.20.9 podporuje i novější verze Hyperledger Fabric, konkrétně verze 1.3 a 1.4, ale jedná se pouze o toleranční podporu, což znamená, že je tato kombinace podporována, ale nevyužívá žádné nové funkce těchto verzí Hyperledger Fabric (Kuhrt, 2020) (Hyperledger Fabric, 2017) (Hyperledger Composer, 2020).

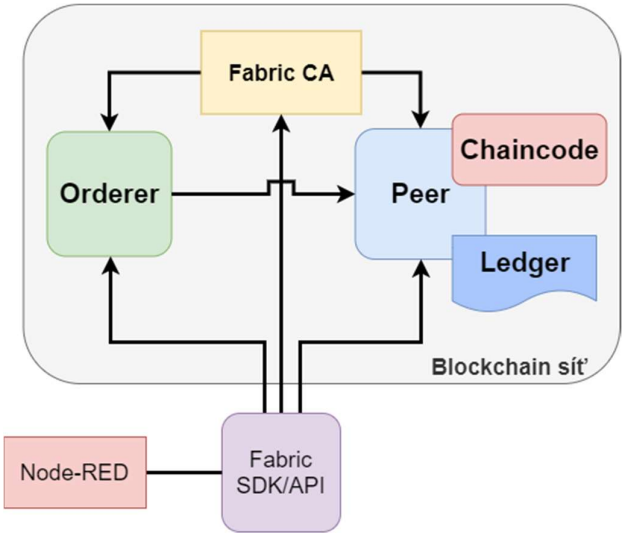
4.2.2 Architektura sítě

V této kapitole je detailněji popsaná architektura sítě, která je prezentována na obrázku 17. Celá architektura sítě se skládá z následujících komponent:

- ***Node-RED*:** Node-RED je open-source programovací nástroj k propojení HW zařízení, API a online služeb (Node-RED, b.r.). Zde komponenta konkrétně slouží k vytvoření simulovaného IoT zařízení, které posílá v časových intervalech náhodná

data o teplotě a vlhkosti v podobě http požadavků směřujících dále ke komponentě *Fabric SDK/API*. Detailně je komponenta *Node-RED* popsaná dále v kapitole 4.2.4.

- ***Fabric SDK/API*:** Tato komponenta je zodpovědná za interakci klienta s blockchain sítí Hyperledger Fabric a voláním chytré smlouvy, tedy *chaincode*. Aplikace používají API k vyvolání *chaincode*. A právě tyto API jsou dostupné skrze SDK. Klient využívá *Fabric SDK/API* k provádění transakcí nad blockchain sítí.
- ***Fabric CA*:** Tato komponenta je implementací MSP (Membership Service Provider – poskytovatel členských služeb, MSP již byl popsán v kapitole 3.3.4), která poskytuje a udržuje digitální identity všech uzlů v síti (klientů, *peerů* a *ordererů*) k autentizaci a autorizaci. Součástí *Fabric CA* je SQL Lite databáze k uchování registrovaných identit spolu s jejich X.509 certifikáty (Baset, 2018). Komponenta *Fabric CA* běží uvnitř Docker kontejneru jako *ca.org1.example.com*.
- ***Orderer*:** Tato komponenta přijímá provedené transakce od *peeru*, zkombinuje je seřazené do bloku transakcí a tyto bloky šíří mezi *peery*. V tomto případě je zde jen jeden *peer*. *Peer* přijme blok transakcí a nejdříve ho validuje před zahrnutím do své účetní knihy *ledger* (Baset, 2018) (Thummavet, 2019). Je důležité zmínit, že v tomto případě je *orderer* implementován v režimu *Solo*, tedy jedná se o centralizovaný *orderer* běžící na jednom uzlu a používaný pro vývoj (Androulaki, 2018)(viz. kapitola 3.3.4). Obdobně jako v předchozích případech, tak i *orderer* běží uvnitř Docker kontejneru jako *orderer.example.com*.
- ***Peer*:** Jedná se o entitu účastnící se blockchain sítě a její identita je registrovaná přes poskytovatele členských služeb (*MSP*), respektive její implementaci *Fabric CA*. Účelem této komponenty je nasazení a instance *chaincode*, aktualizaci účetní knihy (*ledger*) a interakci s ostatními *peery* za účelem sdílení dat souvisejících s transakcemi a interakci s komponentou o*rderer* (Hyperledger Fabric, 2017) (Baset, 2018). Obdobně jako v předchozích případech, tak i *peer* běží uvnitř Docker kontejneru jako *peer0.org1.example.com*. Součástí této komponenty jsou komponenty *chaincode* a *ledger*.



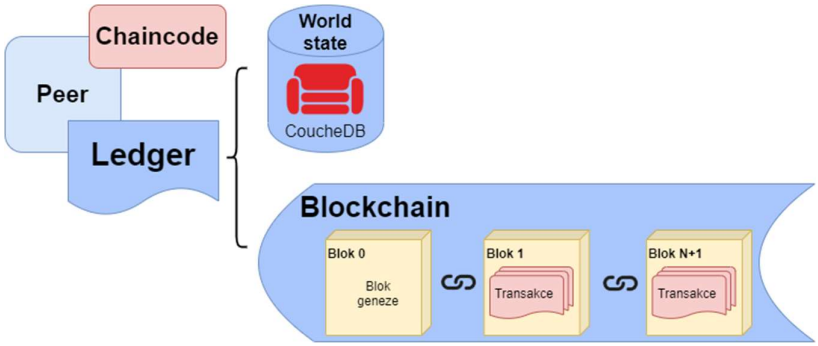
Obrázek 17 - Architektura sítě a komponent. Vytvořeno autorem práce.

Na obrázku 18 lze vidět, že se komponenta *ledger* skládá z dvou dalších komponent. Tyto komponenty se nazývají *world state* a *blockchain*. Digitální účetní kniha (*ledger*) je součástí *peeru* a obsahuje záznam všech finálně zapsaných transakcí. Záznamy jsou zde uloženy jako pár klíč-hodnota. Aktualizace hodnoty stejného klíče změní předešlou hodnotu, ale stará hodnota se neodstraní a v účetní knize zůstává. Za účelem efektivnějšího dotazování je nejnovější hodnota klíče uložena v databázi (Baset, 2018). Databází je zde *CouchDB* a nazývá se *World state*. *World state* je odvozený z *blockchainu*. Tyto komponenty lze tedy shrnout takto:

- ***World state*:** Tato komponenta obsahuje aktuální hodnotu všech objektů (Hyperledger Fabric, 2017). *World state* běží uvnitř Docker kontejneru s názvem *couchdb*. Používá databázi CouchDB podporující operace dotazování pomocí JSON, databázové indexování, replikaci dat, ACID vlastnosti atd (Thummavet, 2019). Alternativou je databáze LevelDB.
- ***Blockchain*:** Tato komponenta uchovává nezměnitelnou historii všech transakcí, které vedly k aktuálnímu stavu *world state*. *Blockchain* je strukturovaný jako sekvence záznamů vzájemně kryptograficky propojených bloků pomocí hašování, kde každý blok obsahuje sekvenci transakcí. Každá transakce reprezentuje dotaz nebo aktualizaci *world state*. Seřazení bloků i transakcí se odehrává v komponentě *Orderer*. Součástí bloku geneze je konfigurační transakce určující počáteční stav sítě (Hyperledger Fabric, 2017).

- ***Chaincode:*** Komponenta je součástí *peer* a slouží k interakci s uloženými daty a *Ledger*. Skrze funkce *chaincode* se provádí dotazování a aktualizovaní *world state* a každá tato změna je zaznamenána do *blockchainu* jako nezměnitelný záznam (Thummavet, 2019). *Chaincode* běží jako Docker kontejner, například s názvem *dev-peer0.org1.example.com-lekarna-0.0.1*, tj. název sítě je *lekarna*, verze je *0.0.1* a navázaný *peer* je *peer0.org1.example.com*.

Na obrázku 18 je prezentován detailnější pohled na komponentu *ledger (L)*, která je součástí komponenty *peer* z obrázku 17.



Obrázek 18 - Detailnější pohled na komponentu Ledger. Vytvořeno autorem práce.

4.2.3 Definice modelu sítě

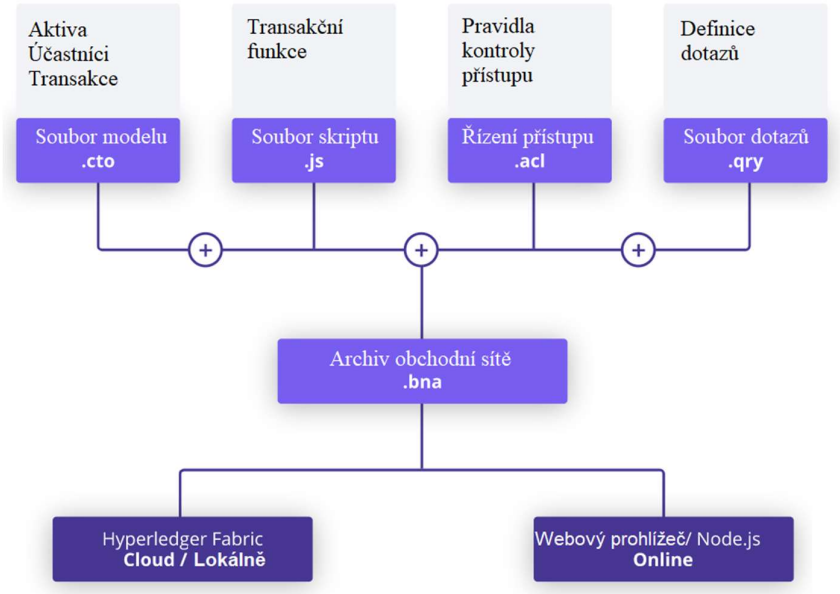
Obchodní logika modelového případu užití, tj. dodavatelského řetězce, je definovaná pomocí nástroje Hyperledger Composer, který podporuje blockchain infrastrukturu a běhové prostředí Hyperledger Fabric. Obsahuje modelovací jazyk a API rozhraní k definici a nasazení sítě, která dovoluje účastníkům provádět transakce vyměňující aktiva (Hyperledger Composer, 2020).

Nástrojem Hyperledger Composer se definují:

- Aktiva, která jsou vyměňována v rámci případu užití založeném na blockchainu.
- Pravidla pro transakce v síti.
- Účastníci a jejich identity společně s přístupovými omezeními určující role při práci s transakcemi (Baset, 2018).

Obrázek 19 reprezentuje architekturu dle oficiální dokumentace nástroje Composer. Archiv obchodní sítě (*.bna*) je balíček obsahující definici sítě, který se poté exportuje a nasadí na Hyperledger Fabric. Součástí je soubor modelu (*.cto*), který definuje strukturu a vztahy mezi elementy modelu. Dále soubor skriptu (*.js*) obsahující funkce transakčního

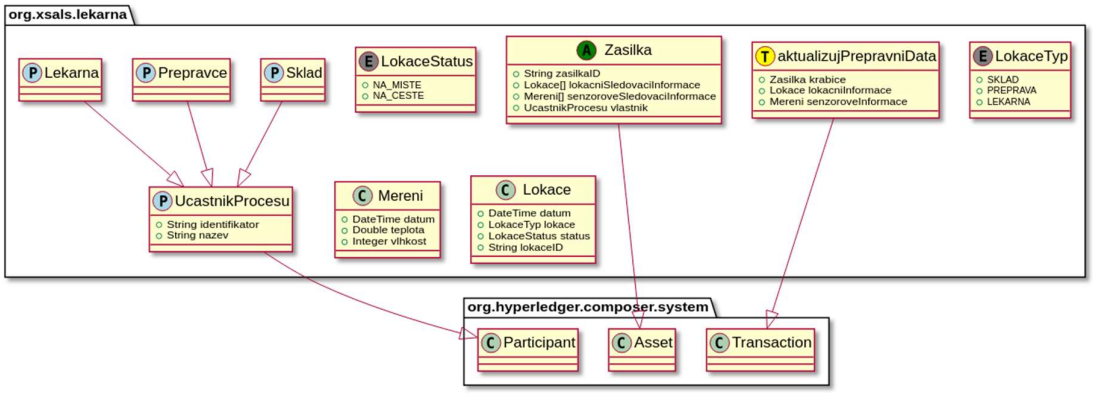
procesoru implementující transakce definované v modelu. Soubor řízení přístupu (*.acl*) deklaruje přístupy účastníků a rolí k prvkům modelu, tj. povolení k jejich vytváření, čtení, změnám a mazaní. Volitelný soubor dotazů (*.qry*) definuje dotazy nad sítí (Hyperledger Composer, 2020).



Obrázek 19 - Oficiální architektura nástroje Hyperledger Composer. Překlad proveden autorem práce. (Hyperledger Composer, 2020).

Soubor modelu

Na obrázku 20 je zobrazen diagram tříd modelu sítě dodavatelského řetězce. Konkrétně se jedná o diagram souboru modelu (*.cto*) definující strukturu a vztahy elementů sítě.



Obrázek 20 - Diagram tříd obchodní blockchain sítě navržený nástrojem PlantUML. Vytvořeno autorem práce.

Jmenný prostor (*namespace*) definice modelu je *org.xsals.lekarna* a všechny vytvořené prostředky jsou implicitně součástí tohoto jmenného prostoru.

V modelu jsou definováni následující účastnící (v diagramu 20 s označením „P“):

- ***UcastnikProcesu:*** Jedná se o abstraktní třídu, takže nemůže mít vytvořenou instanci. Slouží jako základ pro rozšíření tříd *Lekarna, Prepravce* a *Sklad*, které dědí atributy identifikátor a název.
- ***Lekarna, Prepravce, Sklad:*** Tyto třídy rozšiřují abstraktní třídu *UcastnikProcesu*. V modelu lze tedy vytvářet instance jen na tyto tři rozšiřující třídy.
- ***Participant:*** Základní definice abstraktní třídy typu účastník v systémovém jmenném prostoru platformy.

Níže (viz. obrázek 21) je ukázka kódu pro definici třídy účastníka *Lekarna*:

```
abstract participant UcastnikProcesu identified by identifikator {
o String identifikator
o String nazev
}

participant Lekarna extends UcastnikProcesu {
}
```

Obrázek 21 - Definice třídy účastníka Lekarna. Vytvořeno autorem práce.

Zápis definuje abstraktní třídu účastníka *UcastnikProcesu* identifikovanou unikátním polem *identifikator*. Třída *Lekarna* ze třídy *UcastnikProcesu* dědí její vlastnosti *identifikator* a *nazev*.

V modelu je dále definováno aktivum ***Zasilka*** (v diagramu 20 s označením „A“) identifikované atributem *zasilkaID*. Třída dědí z třídy *Asset*, což je základní definice abstraktní třídy typu aktivum v systémovém jmenném prostoru platformy. Zde je kód (viz. obrázek 22), který aktivum zásilku definuje:

```
asset Zasilka identified by zasilkaID {
  o String zasilkaID
  o Lokace[] lokacniSledovaciInformace
  o Mereni[] senzoroveSledovaciInformace
  --> UcastnikProcesu vlastnik
}
```

Obrázek 22 - Definice aktiva zásilka. Vytvořeno autorem práce.

Atributy *Lokace[]* a *Mereni[]* jsou pole lokalizačních a naměřených hodnot uložených v *lokacniSledovaciInformace* a *senzoroveSledovaciInformace*.

Atribut začínající symbolem šipky (-->) je referenční atribut, který ukazuje na třídu *UcastnikProcesu* a jméno elementu je *vlastnik*. Určuje vztah mezi instancí třídy *Zasilka* a instancí třídy *UcastnikProcesu*. Níže (viz. obrázek 23) je tento vztah znázorněn na příkladu zásilky vlastněné lékárnou v JSON formátu:

```
{
  "$class": " org.xsals.lekarna.Zasilka",
  "zasilkaID": "3",
  "lokacniSledovaciInformace": [...],
  "senzoroveSledovaciInformace": [...],
  "vlastnik": "resource:org.xsals.lekarna.Lekarna#3"
}
```

Obrázek 23 - Příklad zásilky vlastněné lékárnou. Vytvořeno autorem práce.

V modelu je dále definována transakce ***aktualizujPrepravniData*** (v diagramu 20 s označením „T“), která je zobrazená na obrázku 24.

```
transaction aktualizujPrepravniData {
  --> Zasilka krabice
  o Lokace lokacniInformace
  o Mereni senzoroveInformace
}
```

Obrázek 24 - Definice transakce aktualizujPrepravniData. Vytvořeno autorem práce.

Referenční atribut ukazuje na třídu *Zasilka* s elementem *krabice*. Tím existuje vztah mezi aktivem *Zasilka* a touto transakcí. Za účelem změny hodnot zásilky jsou dodány nové hodnoty *lokacniInformace* a *senzoroveInformace* z konceptů (elementů) *Lokace* a *Mereni*. Třída dědí z třídy *Transaction*, což je základní definice abstraktní třídy typu transakce v systémovém jmenném prostoru platformy. Ukázka úspěšné transakce, která aktualizuje data aktiva zásilka, lze vidět v příloze č. 1.

Lokace a ***Mereni*** (v diagramu 20 s označením „C“) jsou definované jako třídy typu koncept, které definují strukturální prvky, které jsou v nich obsaženy. Koncepty představují abstraktní třídy, které nejsou aktivy, transakcemi ani účastníky (Hyperledger Composer, 2020). Níže (viz. obrázek 25) je ukázka definice konceptní třídy *Mereni*.

```
concept Mereni {
  o DateTime datum
  o Double teplota
  o Integer vlhkost
}
```

Obrázek 25 - Definice konceptní třídy Mereni. Vytvořeno autorem práce.

V neposlední řadě jsou v modelu definovány výčtové typy ***LokaceStatus*** a ***LokaceTyp*** (v diagramu 20 s označením „E“). Používají se ke specifikování typu, který může mít alespoň jednu možnou hodnotu (Hyperledger Composer, 2020). Níže (viz. obrázek 26) je ukázka definice výčtu *LokaceStatus*, který může nabývat hodnot *NA_MISTE* anebo *NA_CESTE*.

```
enum LokaceStatus {
  o NA_MISTE
  o NA_CESTE
}
```

Obrázek 26 - Definice výčtu LokaceStatus. Vytvořeno autorem práce.

Například tento výčet *LokaceStatus* je součástí definice konceptu *Lokace*, který je součástí definice aktiva *Zasilka*. Takže status lokace zásilky může být „na cestě“ anebo „na místě“.

Soubor skriptu

Skript obsahuje funkci transakčního procesoru, který implementuje transakci definovanou v souboru modelu, v tomto případě tedy transakci *aktualizujPrepravniData*. Funkce transakčního procesoru jsou v běhovém prostředí automaticky vyvolány, když jsou transakce přijaty pomocí rozhraní API (Hyperledger Composer, 2020).

V demonstrovaném případu užití dodavatelského řetězce lékárny aktualizuje transakce informace o sledované zásilce daty poskytnutými simulovaným IoT zařízením. Struktura transakce je definována v modelovém souboru (*.cto*) s názvem *aktualizujPrepravniData*. Definice transakční logiky lze vidět v příloze č. 2.

Funkce je vytvořená v JavaScript ES5 kompatibilním skriptu (*.js*). V hlavičce funkce v poznámkách jsou obsažena metadata potřebná ke zpracování. Definice parametru funkce následuje za značkou *@param* a obsahuje název transakce, která funkci transakčního procesoru spouští.

Řízení přístupu

Pro připojení k síti se používají tzn. *obchodní síťové karty* uložené jako soubor v lokálním souborovém systému. Tato karta obsahuje IP adresu sítě, jméno účastníka a jeho X.509 veřejný klíč. Na základě těchto informací jsou v síti vynucovány práva uživatelů k provádění jen určitých operací s prostředky sítě (Baset, 2018).

Kontrola řízení přístupu se rozlišuje na řízení přístupu ke zdrojům v rámci sítě a na řízení přístupu k administrativním změnám se sítí (Hyperledger Composer, 2020). Z testovacích účelů je všem účastníkům sítě udělen přístup ke všem operacím včetně přístupu ke zdrojům v síti a administrativním změnám sítě. Obsah souboru řízení přístupu (*.acl*) je k vidění v příloze č. 3.

Druhé pravidlo *Default* uděluje všem účastníkům právo provádět všechny operace pod jmenným prostorem *org.xsals.lekarna*. V rámci souboru jsou definovány i další pravidla, ale

z důvodu jejich stejného významu, tedy povolení všech ke všem operacím, tu nejsou tyto pravidla zohledněna. V rámci definice sítě není soubor dotazů (*.qry*) vytvořen, protože nebylo nalezeno jeho praktické uplatnění.

4.2.4 Simulace IoT zařízení

Node-RED v1.0.2 je open-source programovací nástroj k propojení HW zařízení, API a online služeb. Vývoj v nástroji probíhá v prostředí prohlížeče a způsob programování je založený na tzv. *flow-based* programování. Jednotlivé uzly představují programové celky, které se navzájem propojují k dosažení výsledného algoritmu programu. Každý uzel má přesně definovaný účel a způsob, kterým zpracovává vstup na výstup (Node-RED, b.r.).

Node-RED je Node.js aplikace nainstalovaná a běžící ve vývojovém prostředí. Uživatelské rozhraní aplikace běží lokálně na *http://127.0.0.1:1880/*.

V rámci práce je Node-RED použit konkrétně k vytvoření simulovaného zařízení, které posílá v časových intervalech simulovaná data o teplotě a vlhkosti v podobě http požadavku směřující na Composer REST API, sloužící ke komunikaci s blockchain sítí. Na obrázku níže (č. 27) je celý výsledný algoritmus zachycen přímo v nástroji Node-RED.

Obrázek 27 - Workflow v Node-RED – simulované zařízení a odeslání http požadavku. Vytvořeno autorem práce.

Algoritmus (tok) programu v Node-RED je složený z následujících uzlů:

Inject node *(Spustit)*

Jedná se o uzel, který slouží ke spuštění toku manuálně nebo v pravidelných časových intervalech. Obsahem zaslané zprávy je náklad, kterým může být datový typ (například řetězec, číslo, boolean atd.), časové razítko (timestamp) v milisekundách od 1. ledna 1970., anebo toková a globální hodnota vlastnosti kontextu (Node-RED, b.r.). V případě sestaveného algortimu je uzel nastaven na spuštění toku v pravidelných intervalech každou 1 minutu. Obsahem zaslané zprávy je časové razítko.

Function node (*Simulované zařízení*)

Následuje uzel vykonávající JavaScript kód proti procházejícím zprávám z předchozího uzlu. Zpráva je předána jako objekt nazvaný *msg*. Tělo zprávy je předáváno jako *msg.payload* (Node-RED, b.r.). Kód uvnitř uzlu lze vidět v příloze č. 4, kód je zde dále popsán.

Kód simuluje aplikaci IoT zařízení aktualizující ledger (účetní knihu) blockchainu použitím API, které je vystavené Composer REST serverem. Nejdříve jsou vytvořená pole pro teplotu a relativní vlhkost, která obsahují pseudonáhodné hodnoty. Poté jsem hodnoty z polí vybírány počítadlem. Hodnoty představují vlastnosti prostředí, ve kterém je zásilka s léky přepravována. Tedy naměřené hodnoty ze zařízení kontrolující podmínky působící na zásilku během přepravy.

Následuje tělo zprávy *msg.payload* ve formátu JSON. Obsahem zprávy je tělo požadavku, které volá definovanou transakci *aktualizujPrepravdniData* v síti blockchain prostřednictvím serveru Composer REST. Nová transakce aktualizuju zásilku s identifikačním číslem 3 (*Zasilka#3*) lokalizačními a senzorovými informacemi.

Data v JSON formátu jsou poslána na URL lokálního Composer REST serveru. Konkrétně na HTTP POST požadavek používající kódování obsahu *appliaction/json* transakce *aktualizujPrepravdniData*.

Composer REST server založený na LoopBack frameworku spustí loopback aplikaci, která využívá Hyperledger Composer LoopBack Connector k připojení k definované blockchain síti. Poté extrahuje modely aktiv, účastníků a transakcí k jejich prezentaci na stránku obsahující REST API. API slouží ke komunikaci s blockchain sítí a jejím modelem.

Obrázek 28 je snímek obrazovky uživatelského rozhraní Composer REST serveru. Konkrétně dedikovaná instance REST serveru účastníka *Přepravce* s identifikačním číslem *2*. Webový server naslouchá na *http://localhost:3002*. Na obrázku lze vidět transakce *aktualizujPrepravdniData* s HTTP metodou POST, která je odkazována v kódu (příloha č. 4) v *msg.url* společně daty. Celá zpráva je poté předána jako objekt *msg*.

Obrázek 28 - Instance Composer REST serveru účastníka „Přepravce“ s číslem 2 naslouchající na portu 3002. Vytvořeno autorem práce.

Http request node *(HTTP požadavek)*

Další uzel v pořadí, který odešle HTTP požadavek na Composer REST server API a vrátí odpověď. HTTP metoda je typu POST a URL je již nastavená v přechozím uzlu. Odpověď serveru je nastavená na rozebraný JSON objekt.

V příloze č. 5 lze vidět příklad těla úspěšného HTTP požadavku na aktualizaci zásilky s identifikačním číslem 3 (*Zasilka#3*) lokalizačními a senzorovými informacemi přes API transakce *aktualizujPrepravniData*.

Debug node *(Response)*

Ladící uzel, který slouží k zobrazení struktury předávaných zpráv v rámci toku. V uživatelském rozhraní se při nastavení ladícího uzlu zobrazují předávané zprávy mezi uzly společně s časovým razítkem, kdy k vytvoření zprávy došlo, a také identifikaci ladícího uzlu, který zprávu zachytil. Lze tak lépe pochopit zdroj chyb nebo strukturu zprávy, která je předávána v rámci toku (Node-RED, b.r.).

Například v případě vytvořeného algoritmu slouží ladící uzel, s názvem *Response*, k zachycení těla zprávy *msg.payload* ve formátu JSON vytvořeného v uzlu *Device Payload*

a dále odeslaného jako HTTP požadavek v uzlu *HTTP požadavek*. Dále slouží k zachycení chybových stavů požadavku ze serveru.

Níže je snímek obrazovky z rozhraní Node-RED, konkrétně zachycení těla zprávy *msg.payload* pomocí ladícího uzlu *Response*.

Obrázek 29 - Snímek obrazovky z ladícího režimu v Node-RED – zachycení těla zprávy. Vytvořeno autorem práce.

V horní části obrázku 29 lze vidět, že zpráva byla zachycena v uzlu *Response*. Součástí je časový záznam.

4.3 Experiment

Stanoveným cílem experimentu bylo ověření bezpečnosti vytvořeného řešení a zhodnocení vhodnosti integrace technologie blockchain s IoT v kontextu bezpečnosti, implementované při použití blockchain platformy Hyperledger Fabric v1.2 a Hyperledger Composer 0.20.

V této kapitole je popsán experiment, použitý benchmark nástroj a identifikace sledovaných metriky měření. Dále jsou pomocí měření (testů) vymezeny výkonnostní hranice. Poté je popsán průběh experimentu.

4.3.1 Prostředí experimentu

Předpokladem pro běh blockchain platformy Hyperledger Fabric a nástroje Hyperledger Composer je unixový operační systém. Možnosti jsou nainstalovat Linuxový subsystém na Windows 10, MacOS 10.12 nebo Ubuntu Linux 14.04 / 16.04 LTS (64-bit). Jako vývojové prostředí řešení byl zvolen virtuální stroj s nainstalovaným 64bitovým operačním systémem Ubuntu 16.04 LTS běžící na Windows 10. V tabulce 4 jsou specifikovány parametry prostředí experimentu provozovaném na virtuálním stroji.

Komponenta	Parametr
Lokální stroj:	
Operační systém	Windows 10 Home (64-bit)
CPU	Intel Core i5-8265U @ 1.60GHz
Operační paměť	8 GB
Virtuální stroj (vývojové prostředí):	
Virtualizační nástroj	Oracle VM VirtualBox 6.0.14
Operační systém	Ubuntu Linux 16.04.6 LTS (64-bit)
Operační paměť	4 GB
Virtuální velikost uložiště	40,04 GB

Tabulka 4 - Specifikace vývojového prostředí. Vytvořeno autorem práce.

Na vývojovém prostředí je lokálně nainstalován framework Hyperledger Fabric k lokálnímu nasazení / spuštění vytvořené obchodní sítě. Poté je nainstalován vývojářský nástroj Hyperledger Compose určený primárně k vytvoření obchodních sítě (blockchain sítí).

4.3.2 Benchmark nástroj

Jako měřící nástroj byl zvolen open-source benchmark nástroj Apache JMeter ve verzi 5.2.1. Jedná se o aplikaci napsanou v jazyce Java, která slouží k provádění zátěžových testů a testování výkonu aplikací, serverů a protokolů různých typů. Více-vláknová struktura umožňuje souběžné vzorkování mnoha vlákny a současné vzorkování různých funkcí samostatnými skupinami vláken (Apache JMeter, c1999-2019).

Implementované řešení Hyperledger Fabric blockchain sítě s použitím nástroje Hyperledger Composer disponuje komponentou Composer REST Server (4.2.1), která vystaví logiku obchodní blockchain sítě jako REST API pro HTTP nebo REST klienty. Nástroj podporuje testování výkonu skrze HTTP dotazy, a proto jej lze využít při testování výkonnosti modelové blockchain sítě a k měření v rámci experimentu (4.1).

4.3.3 Sledované metriky a testovací případy

Základním prvkem testovacího plánu v benchmark nástroji JMeter je *skupina vláken* (Thread Group). *Skupina vláken* představuje sadu vláken, které JMeter použije k provedení testovacího scénáře (Baeckstrom). Mezi hlavní ovládací prvky *skupiny vláken* patří:

- ***Počet vláken (Number of Threads):*** Počet vláken představuje celkový počet virtuálních uživatelů provádějících testovací skript. Virtuální uživatele lze v tomto testovaném řešení interpretovat jako IoT zařízení.
- ***Doba náběhu (Ramp-up period):*** Určuje, za jakou dobu naběhne celkový počet všech zvolených vláken. Jednotkou jsou sekundy. Pokud je například počet vláken 120 a doba náběhu je 60 sekund, tak celkem potrvá 60 sekund ke spuštění všech 120 vláken přidáváním 2 vláken za sekundu.
- ***Počet smyček (Loop Count):*** Určuje kolikrát dojde ke spuštění skriptu (Apache JMeter, c1999-2019).

Mezi nejdůležitější sledované metriky v rámci JMeter, pospané dle (Loisel, c2014-2020), patří:

- ***Datová propustnost (Throughput):*** Propustnost je vyjádřena jako podíl počtu požadavků za celkový čas. Čas se počítá od začátku prvního požadavku do konce posledního požadavku.
- ***Doba odezvy (Response Time):*** Uplynulý čas mezi odesláním požadavku a přijetím celé odpovědi.

4.3.4 Identifikace výkonnostní hranice architektury

Ke správnému provedení experimentu je nejdříve zapotřebí identifikovat výkonnostní hranici řešení, tedy způsobu, kterým je testovaná blockchain platforma implementovaná. Předpokládá se, že testovaný systém má softwarové limity dané technologií a tím, jak je naprogramován, ale také hardwarovými limity dané prostředím a způsobem, kterým je systém provozován.

Test funkčnosti

První provedený test slouží k zjištění, zda je správně nastavená konfigurace testovacího plánu. Cílem je zjistit, zda Composer REST server přijímá odesílané HTTP požadavky metodou POST. Dále zda tyto požadavky správným způsobem vyvolávají definovanou transakci *aktualizujPrepravniData* (4.2.3), která aktualizuje lokalizační a senzorové hodnoty definované zásilky uvnitř blockchain sítě Fabric.

K provedení tohoto testu je nastavena skupina vláken tímto způsobem:

- Hodnota *počtu vláken* (4.3.3) je nastavena na hodnotu 1. Takže pouze jedno vlákno (virtuální zařízení/uživatel) odesílá požadavek na REST server.

- *Doba náběhu* vláken je nastavena na 0 sekund, což znamená, že JMeter okamžitě spustí všechny uživatele.
- *Počet smyček* je nastaven na 1, takže odeslání HTTP požadavku se provede pouze jednou.

HTTP požadavek se odešle na lokální IP adresu 127.0.0.1 s portem 3000, kde na této adrese a s tímto portem běží a naslouchá Composer REST server. Metoda je typu POST a konkrétní cesta k cílené transakci je api/aktualizujPrepravniData. Kódování obsahu je nastaveno na UTF-8. Hlavička požadavku obsahuje hodnoty typ obsahu (Content-Type) a přijatelný typ odpovědi (Accept), tyto hodnoty jsou na staveny na application/json.

Definice těla odesílaného HTTP požadavku je následující (viz. obrázek 30):

```
{
    "$class": "org.xsals.lekarna.aktualizujPrepravniData",
    "krabice": "org.xsals.lekarna.Zasilka#6",
    "lokacniInformace": {
        "$class": "org.xsals.lekarna.Lokace",
        "datum": ${__javaScript(Date.now())},
        "lokace": "PREPRAVA",
        "status": "NA_CESTE",
        "lokaceID": ${__RandomString(4,123456789)}
    },
    "senzoroveInformace": {
        "$class": "org.xsals.lekarna.Mereni",
        "datum": ${__javaScript(Date.now())},
        "teplota": ${__Random(0,50)},
        "vlhkost": ${__Random(0,100)}
    }
}
```

Obrázek 30 - Definice těla odesílaného HTTP požadavku. Vytvořeno autorem práce.

Součástí těla požadavku jsou funkce JMeter. První funkce *javaScript()* použitá pro naplnění atributu datumu spustí část kódu JavaScript, který vrátí hodnotu aktuálního datumu. Funkce *RandomString()* vrací náhodný řetězec definované délky k naplnění atributu lokaceID náhodným řetězcem čísel. Funkce *Random()* vrací náhodné číslo, které leží mezi danou minimální a maximální hodnotou, k naplnění atributu teploty a vlhkosti.

Po spuštění testu byla zjištěna fungující komunikace s REST serverem, úspěšné vyvolání transakce a zapsaní lokalizačních i senzorových dat zásilky. Historie provedených transakcí se úspěšně zaznamenala na blockchain.

V příloze č. 6 je k vidění příklad těla odeslaného HTTP požadavku. Dále je v příloze č. 7 k vidění kód uložené zásilky v databázi sítě s naplněnými hodnotami transakce a v příloze č. 8 lze vidět nezměnitelně uloženou historii transakce v blockchainu.

Test odeslání více požadavků od více virtuálních zařízení současně

Další výkonnostní test slouží k vymezení výkonnostní hranice testované blockchain sítě. Podstatou tohoto testu je ověřit chování sítě při odeslání více požadavků na blockchain síť od více virtuálních zařízení současně. Jedná se o vyvolání stejné transakce jako v předchozím testu, tedy *aktualizujPrepravniData*.

K provedení tohoto testu je nastavena skupina vláken tímto způsobem:

- Hodnota *počtu vláken* je nastavena na hodnotu 2. Takže požadavky jsou odesílány na REST server dvěmi vlákny (virtuálními zařízeními/uživateli) současně.
- *Doba náběhu* vláken je nastavena na 0 sekund, což znamená, že JMeter okamžitě spustí všechny uživatele.
- *Počet smyček* je nastaven na 2, takže odeslání HTTP požadavku od obou vláken se provede celkem dvakrát.
- Dále je nastavena komponenta *konstantní časovač (Constant Timer)* na hodnotu 3000 milisekund, což znamená, že dojde ke konstantně dlouhým pauzám mezi smyčkami.

Dle výše uvedeného nastavení tedy dochází k odeslání celkem čtyř požadavků s pauzou tří sekund mezi dvojicemi odeslaných požadavků.

Po spuštění testu byl zjištěn neočekávaný výsledek. Každý druhý odeslaný požadavek vyvolal na serveru chybu a vrátil odpověď s interní chybou serveru se status kódem číslo 500. Níže (viz. obrázek 31) je zobrazena chybová zpráva obsažena v těle odpovědi ze serveru:


```
"message": "Error trying invoke business network with transaction id
e41804b5e45582d4ccce4b65dd5da3e621e6df9ee7dc32b714ec37835588784d.
Error: Peer localhost:7051 has rejected transaction
'e41804b5e45582d4ccce4b65dd5da3e621e6df9ee7dc32b714ec37835588784d'
with code MVCC_READ_CONFLICT"
```

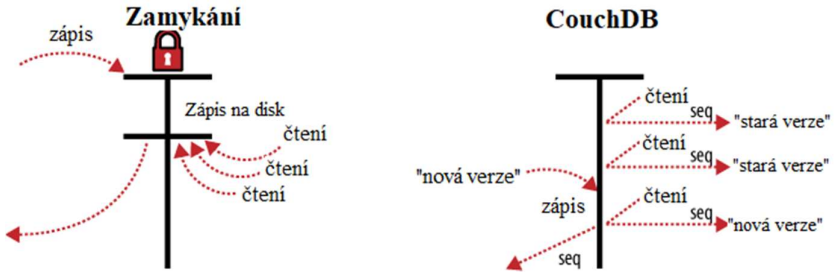

Obrázek 31 - Chybová zpráva uvnitř těla HTTP odpovědi. Vytvořeno autorem práce.

Název vlákna	Čas začátku	Doba vzorkování [ms]	Status	Bajty	Odeslané bajty	Latence [ms]	Přijaté bajty
1-1	16:29:18.928	2307	Úspěch	826	710	2307	71
1-2	16:29:18.933	2306	Selhání	1259	711	2306	73
1-2	16:29:24.240	2265	Úspěch	825	710	2265	46
1-1	16:29:24.237	2271	Selhání	1259	711	2271	54

Tabulka 5 - Výsledky testu při odesílání více požadavku současně. Vytvořeno autorem práce.

Výše jsou zobrazeny výsledky testu v tabulce 5. Z tabulky lze pozorovat, že vždy selhal jeden požadavek z dvojice odeslaných požadavků. V čase 16:29:18 byla odeslána první dvojice požadavků, kde selhalo zpracování požadavku s označením 1-2, respektive druhý požadavek z dvojice s rozdílem v řádu milisekund oproti prvnímu požadavku 1-1. V čase 16:29:24 byla odeslána druhá dvojice požadavků. Zde opět selhal druhý požadavek z dvojice (1-2) odeslaný později.

Výše zmíněné chyby souvisejí s vlastností databáze CouchDB, která je součástí komponenty *world state*. Komponenta *world state* je součástí komponenty *ledger*, která je součástí komponenty *peer*. Komponenta *world state* a její nadřazené komponenty jsou popsány v kapitole 4.2.2 (architektura sítě). Úkolem databáze CouchDB je udržovat záznamy o všech finálně zapsaných transakcí. Tyto záznamy jsou zde uloženy jako pár klíč-hodnota (Baset, 2018). Databáze CouchDB používá metodu známou jako *Multi-Version Concurrency Control (MVCC)* k řízení souběžného přístup k databázi. Jedná se o alternativní přístup oproti metodě *zamykání* databáze (viz. obrázek 32), kdy databázový systém uzamkne tabulku, kterou momentálně upravuje jiný uživatel. Tím se zajistí, že se nikdo nepokusí o změnu záznamu nebo čtení záznamu tabulky během její aktualizace, čímž by došlo k nekonzistenci dat. Tento způsob ovšem plýtvá značné množství výpočetní kapacity serveru (Anderson, 2010). V případě metody *MVCC* jsou záznamy verzovány. To znamená, že pokud dojde k požadavku změny hodnoty záznamu, tak se vytvoří nová verze záznamu (Anderson, 2010).



Obrázek 32 - Metoda zamykání oproti metodě MVCC v CouchDB (Anderson, 2010). Překlad obrázku autorem práce.

Tato vlastnost souvisí s architekturou blockchain platformy Hyperledger využívající model *vykonej-seřaď-ověř* pro řízení transakčního toku. Tato architektura transakčního toku je stručně pospaná v kapitole 3.3.4. Dále zmíněné komponenty (*peer, chaincode,* ledger atd.) jsou popsané v kapitole 3.3.4. Jednotlivé kroky transakčního toku jsou následující:

1) Klient (SDK) odešle návrh transakce *peeru*, který je součástí schvalovací politiky. Schvalovací politika specifikuje množinu *peer* uzlů, která je nezbytná při procesu

schválení transakce (Androulaki, 2018). V tomto případě je použitý *peer* součástí této politiky.

2) *Peer* vykoná (simuluje) transakci s použitím *chaincode* nad současným stavem *ledger*. Tím jsou získány výsledky transakce včetně hodnoty odpovědi a sady čtení/zápisů. V tomto okamžiku nejsou provedeny žádné aktualizace *ledger* (Kuhrt, 2019).
3) Výsledky, množiny hodnot a podpis *peeru* jsou předány zpět na klienta. (Hyperledger Fabric, 2017).
4) Klient ověří podpis schvalovacího *peeru* a odešle transakci na *orderer* (3.3.4 a 4.2.2) společně se svým podpisem a sadou čtení/zápisů. *Orderer* vytvoří bloky transakcí.
5) Bloky jsou doručeny *peeru* (případně množině *peerů*). Jsou ověřeni zásady schvalovací politiky a transakce v bloku jsou ověřeny, zda nedošlo ke změně stavu *ledger* pro proměnné sady čtení od jejich vzniku během vykonání v kroku 2.
6) *Peer* přidá blok do svého řetězce a pro každou validní transakci jsou provedeny změny ve *world state* databázi dle sady zápisů. Klient je poté informován o úspěšnosti či neúspěšnosti transakce (Hyperledger Fabric, 2017) (Androulaki, 2018).

Zásadní pro určení zdroje chyby je druhý a pátý krok. V druhém kroku během simulace transakce vzniká sada čtení a zápisů nad aktuálním stavem *ledger*. Sada obsahuje název proměnné, na které má být provedeno čtení/zápis, a její verzi v průběhu čtení. V pátém kroku je provedena fáze *ověř*, kde je ověřen blok transakcí ve třech krocích. Jedním z těchto kroků je kontrola konfliktu čtení a zápisu pro všechny transakce v bloku. Pro každou transakci se porovná verze klíčů s verzemi v aktuálním stavu *ledger* a kontroluje se, zda jsou stále stejné. Pokud se verze neshodují, transakce je označena jako neplatná a její účinky se neberou v úvahu (Androulaki, 2018). Dojde-li, mezi vytvořením této sady ve fázi *vykonej* (krok 2) a její opětovnou kontrolou ve fázi *ověř* (krok 5), k další transakci, která mění klíč, a tedy i verzi stejné proměnné uvnitř sady čtení, tak je tato transakce později odmítnuta, protože verze sady čtení při *ověření* není aktuální a nesouhlasí s verzi sady čtení při *vykonání* (simulaci) transakce *peerem* (Hyperledger Fabric, 2017).

Metoda *MVCC* je v blockchain platformě Hyperledger Fabric implementovaná jako ochrana a prevence proti útoku dvojité útraty, popsané v kapitole 3.1.12 (Vankov, 2018). V tomto stavu to ovšem představuje problém při implementaci v blockchain síti obsluhující velké množství požadavků na zápis (časté změny záznamů), jako je tomu právě při integraci s IoT zařízeními. Pokud zde dochází k požadavkům na zápis dat v rámci Hyperledger Fabric

blockchainu rychleji, než se dokončí transakční tok *vykonej-seřaď-ověř*, tak jsou požadavky zamítnuty.

Z této vlastnosti ovšem vyplývá i zranitelnost blockchain sítě Hyperledger Fabric vůči DoS (Denial of Service) útoku, kdy útočník přehltí REST server velkým počtem nežádoucích požadavků. Takovým množstvím požadavků, kdy blockchain síť Fabric přestane stíhat transakce zpracovávat a začne tedy transakce označovat za neplatné. Tím dojde k nefunkčnosti služby a její znepřístupnění ostatním klientům.

4.3.5 Průběh experimentu

Na základě zjištěných výkonnostních hranic zkoumané blockchain sítě byl dále proveden experiment ověřující řešení v kontextu bezpečnosti. Náplní experimentu bylo provedení DoS (Denial of Service) útoku na komponentu *world state* (4.2.2) používající databázi CouchDB, která je součástí komponenty *peer* uvnitř blockchain sítě Hyperledger Fabric, a sledování vlivu tohoto útoku na ukazatele *doby odezvy* a *datové propustnosti* sítě. Komponenta *world state* je důležitým prvkem v procesu zpracování transakcí, a je tedy potenciálním terčem případného útoku na síť. Prostředí, na kterém byl experiment proveden je popsané v kapitole 4.3.1. Výsledky experimentu budou poté vyhodnoceny a ze získaných poznatků formulovány závěry.

Během testu odeslání více požadavků od více virtuálních zařízení současně bylo zjištěno omezení v podobě MVCC (viz. předchozí kapitola 4.3.4), na síť je tedy v jeden okamžik odeslán pouze jeden požadavek od jednoho virtuálního zařízení (vlákna). Zkoumanými ukazateli jsou *doba odezvy* a *datová propustnost*. Popis ukazatelů se nachází v kapitole 4.3.3.

Průběh experimentu se skládá ze dvou fází. V první fázi bylo odesláno 60 HTTP požadavků s metodou POST na Composer REST server k vyvolání transakce *aktualizujPrepravniData* (4.2.3), která aktualizuje lokalizační a senzorové hodnoty definované zásilky uvnitř blockchain sítě Fabric. Definice těla odesílaného HTTP požadavku je stejná jako v případě testu funkčnosti v kapitole 4.3.4. Výsledkem měření byl zjištěn průběh ukazatele *doba odezvy* a hodnota pro *datovou propustnost*. V druhé fázi byl spuštěn DoS útok na komponentu *world state* a byl opět byl zjištěn průběh ukazatele *doba odezvy* a hodnota pro *datovou propustnost*. Nástroj použitý k provedení útoku DoS je hping3, pomocí kterého je cílená komponenta zaplavena syn pakety z náhodných neplatných IP adres. Výsledky experimentu jsou zobrazeny a popsány v další kapitole.

5 Výsledky a diskuze

V praktické části práce je vytvořena obchodní logika modelové aplikace integrující IoT a blockchain technologie. Vytvořená obchodní logika řeší problematiku dodavatelského řetězce ve zdravotnictví, což je jedna z bezpečnostně kritických aplikací vyžadující přísnější zabezpečení. Definice modelu blockchain sítě je vytvořena pomocí nástroje Hyperledger Composer 0.20.9 a nasazena v běhovém prostředí open-source blockchain platformy Hyperledger Fabric v1.2.

Dále je provedena simulace IoT zařízení v nástroji Node-RED v1.0.2 odesílající údaje o teplotě a vlhkosti prostředí sledované zásilky v rámci dodavatelského řetězce. Toto zařízení je propojeno s vytvořenou blockchain sítí k ověření vhodnosti integrace IoT s blockchain technologií. Komunikace IoT zařízení s blockchain sítí Fabric probíhá pomocí HTTP požadavků skrze Composer REST Server 0.20.9. Tato komunikace se prokazuje jako funkční, protož dochází k záznamu hodnot v rámci blockchain sítě.

V další fázi je použit benchmark nástroj k provedení testů identifikujících výkonnostní hranice použité architektury, a k následnému měření v rámci experimentu. K měření je použit open-source benchmark nástroj Apache JMeter 5.2.1, který podporuje testování výkonu skrze HTTP dotazy.

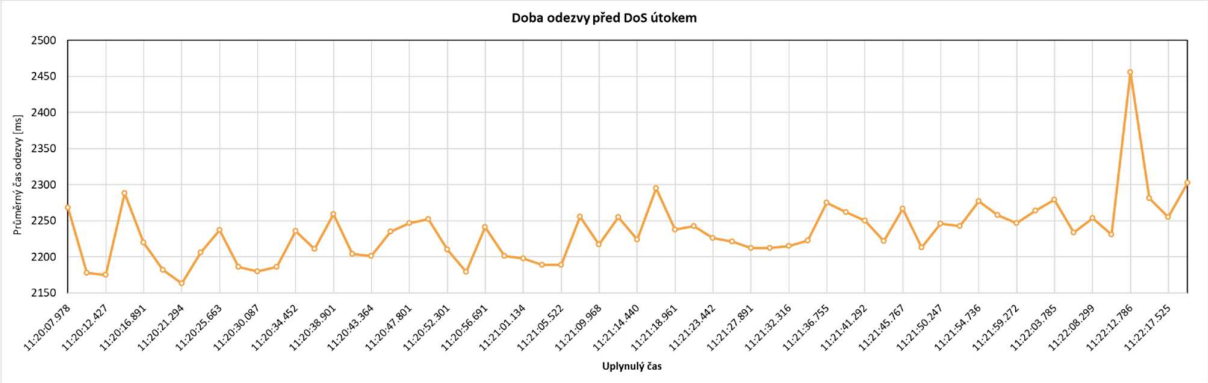
Během testování výkonnostních hranic řešené architektury je nejdříve ověřeno, zda je korektně nakonfigurovaný měřící nástroj JMeter a zda probíhá bezchybná komunikace s blockchain sítí Fabric. Výsledkem je zjištění, že je vše správně nastaveno. Později je proveden další test, který ověřuje chování sítě při odeslání více požadavků na blockchain síť od více virtuálních zařízení současně. Test ověřuje chování sítě při větší zátěži, což je při integraci s IoT předpokládáno. Výsledkem je neočekáváné zjištění, že blockchain síť není schopná zpracovávat požadavky od dvou a více zařízení současně, pokud dochází k vyvolávání transakce zaznamenávající hodnoty ke stejnému aktivu (zásilce). Při zjišťování příčiny chyby je identifikován problém s metodou řízení souběžnosti MVCC (Multi-Version Concurrency Control) vycházející z architektury řízení transakčního toku blockchain sítě Hyperledger Fabric v1.2 a ze související vlastnosti databáze CouchDB, která je součástí komponenty architektury blockchain sítě. Metoda *MVCC* slouží jako ochrana a prevence proti útoku dvojité útraty, ale zároveň poukazuje na náchylnost vůči DoS (Denial of Service) útoku, kdy útočník přehltí REST server velkým počtem nežádoucích požadavků. Ke stejnému zjištění došel během testování Hyperledger Fabric autor (Fuentes Contreras, 2019),

který zároveň navrhuje řešení problému generováním unikátních identifikátorů aktiv pomocí modulu uuidv4. Ovšem v případě autorova testování dochází k vytváření nových aktiv v síti, narozdíl od provádění aktualizací hodnot aktiva (zásilky). Problém má tedy obdobný zdroj, ale při jiném případu užití.

Na základě předchozích zjištění je dále proveden experiment vyhodnocující dopad DoS (Denial of Service) útoku na jednu z komponent blockchain sítě. Během experimentu je sledován vlivu útoku na ukazatele *doby odezvy* a *datové propustnosti* sítě. Terčem útoku je komponenta *world state* s databází CouchDB, která je skrze IP adresu a port zahlcena syn pakety z náhodných neplatných IP adres. K provedení útoku je použit nástroj hping3. Dále jsou popsány jednotlivé výsledky experimentu.

Měření před provedením DoS útoku

Na grafu č. 1 je zobrazena změna průměrné *doby odezvy* v čase. Osa *x* reprezentuje uplynulý čas měření. Osa *y* reprezentuje průměrnou *dobu odezvy* v milisekundách.

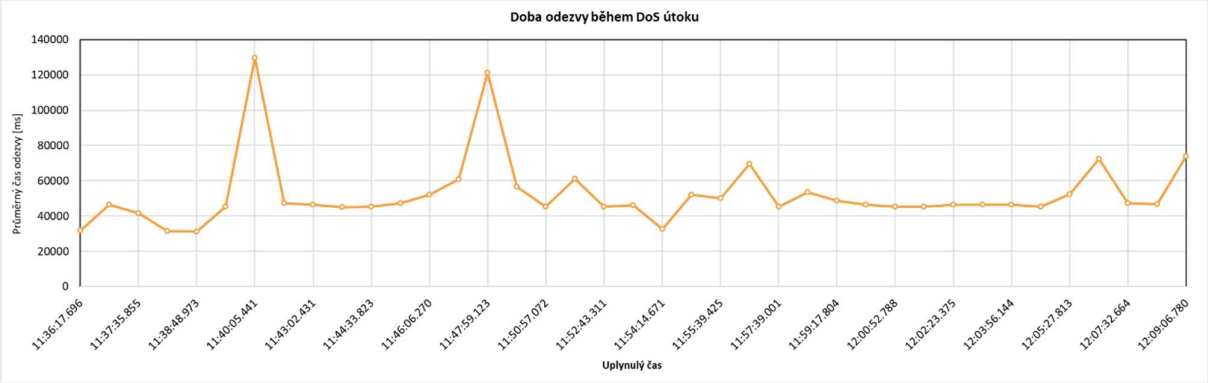


Graf 1 - Graf průměrné doby odezvy v čase před provedením útoku. Vytvořeno autorem práce.

V průběhu měření bylo odesláno 60 požadavků na server. Všech 60 požadavků bylo úspěšných, takže procento neúspěšnosti je 0 %. Průměrná ($\bar{x}$) *doba odezvy* je 2234 milisekund se směrodatnou odchylkou (σ) 43 milisekund. Hodnota mediánu ($\tilde{x}$) je stejná jako hodnota průměru, tedy 2234 milisekund. Minimální hodnota činí 2163 milisekund a maximální 2456 milisekund. Až na jeden významnější výkyv, v průběhu *doby odezvy* u požadavku číslo #57 s hodnotou 2456 milisekund, je průběh stálý. Výsledná *datová propustnost* činí 0,45 transakcí za sekundu.

Měření během DoS útoku

Během tohoto měření byl na komponentu *world state*, tedy na IP adresu a port databáze CouchDB, spuštěn DoS útok pomocí nástroje hping3, který zaplavil komponentu syn pakety z náhodných neplatných IP adres. Následkem útoku došlo k nedostupnosti sítě, proto bylo odesláno pouze 39 požadavků ze zamýšlených 60 požadavků. Odpovědí na všech 39 požadavků byla chyba, takže procento neúspěšnosti je 100 %. Průměrná *doba odezvy* je zobrazena na grafu č. 2. Průměrná ($\bar{x}$) *doba odezvy* pro 39 požadavků je 52380 milisekund s vysokou směrodatnou odchylkou (σ) 19403 milisekund. Hodnota mediánu ($\tilde{x}$) je 46372 milisekund. Minimální hodnota činí 31201 milisekund a maximální 129592 milisekund. Výsledná *datová propustnost* pro 39 požadavků činí 0,02 transakcí za sekundu.



Graf 2 - Graf průměrné doby odezvy v čase během DoS útoku. Vytvořeno autorem práce.

Během útoku došlo k znepřístupnění některých komponent sítě, následkem čehož jsou HTTP požadavky na vyvolání transakce *aktualizujPrepravniData* neúspěšné. Po skončení útoku je blockchain síť i nadále nedostupná, takže následkem útoku došlo k dlouhodobé nedostupnosti sítě. Nedostupnost je ověřena pomocí příkazu k ověření připojení k blockchain síti nasazené na Hyperledger Fabric. Níže (viz. obrázek 33) je zobrazen výstup tohoto příkazu, který popisuje nedostupnost komponenty *peer*.

```
Error: Error trying to ping. Error: No peers available to query. last error was
Error: Failed to connect before the deadline
```

Obrázek 33 - Výstup příkazu ping – chybová hláška. Vytvořeno autorem práce.

Během útoku dále došlo k vypnutí docker kontejnerů pro komponenty *peer* (peer0.org1.example.com) a *chaincode* (dev-peer0.org1.example.com-lekarna-0.0.1). Ze záznamů platformy lze potvrdit, že došlo během útoku k znepřístupnění komponenty *peer* na portu 7051, která je zcela zásadní v procesu zpracování transakcí. Z nedostupnosti komponenty *peer* lze odvodit i nedostupnost dílčích komponent *chaincode* a *ledger*, a tedy

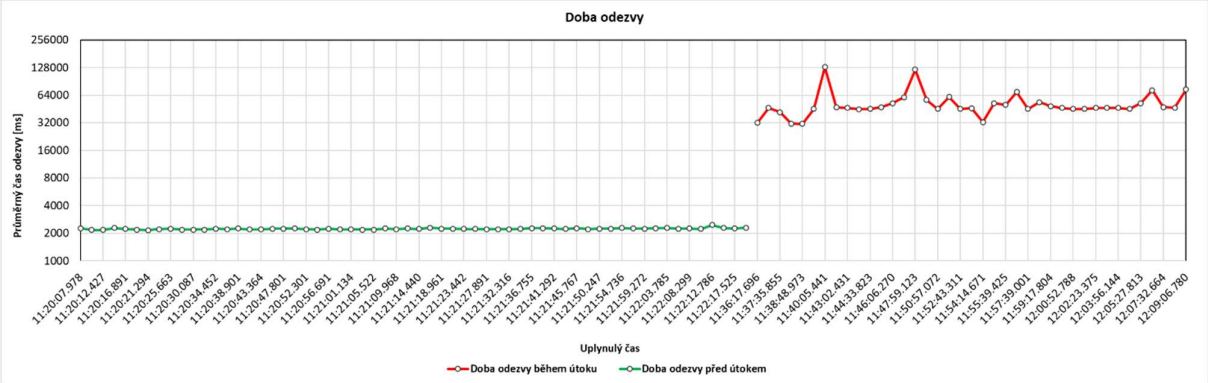
i dalších dílčích komponent *world state* a *blockchain*. Níže (viz. obrázek 34) je zobrazen záznam chyby.

```
2020-02-17T11:53:01.193Z WARN :HLFConnection :_connectToEventHubs() event hub lo-
calhost:7051 failed to connect: 14 UNAVAILABLE: Connect Failed {}$
2020-02-17T11:53:01.196Z WARN :HLFConnection :_registerForChaincodeE-
ven could not find any connected event hubs out of 1 defined hubs to listen on for chain-
code events {}$
2020-02-17T11:53:04.207Z ERROR :HLFConnectionManager :fabri client() [Remote.js]: Er-
ror: Failed to connect before the deadline {}$
```

Obrázek 34 - Chybové záznamy během útoku. Vytvořeno autorem práce.

Během sledování vlivu útoku na ukazatele *doby odezvy* a *datové propustnosti* sítě došlo k nedostupnosti sítě, proto bylo během útoku odesláno pouze 39 požadavků z původních 60 požadavků. Vliv na ukazatele byl významný, průměrná ($\bar{x}$) *doba odezvy* se z 2234 milisekund se směrodatnou odchylkou (σ) 43 milisekund zvýšila na 52380 milisekund se směrodatnou odchylkou (σ) 19403 milisekund. *Datová propustnost* se z původních 0,45 transakcí za sekundu se snížila na 0,02 transakcí za sekundu. Ovšem je nutno dodat, že na ukazatel *datové propustnosti* má vliv rozdílný počet odeslaných požadavků. Počet neúspěšných požadavků se během útoku zvýšil o 100 %.

Níže je zobrazen graf č. 3 průměrné *doby odezvy* kombinující předchozí dva grafy (č. 1 a č. 2) za účelem porovnání výsledků experimentu.



Graf 3 - Graf průměrné doby odezvy před útokem a během DoS útoku. Vytvořeno autorem práce.

Výsledkem experimentu je tedy zjištění náchylnosti blockchain sítě, implementovaného pomocí Hyperledger Fabric v1.2 a Hyperledger Composer 0.20.9, při DoS útok na komponentu architektury. Integrace IoT s blockchain technologií implementovanou pomocí této platformy, při konkrétním prostředí a za konkrétních kritérií, je z hlediska bezpečnosti ke zvážení. Především je potřeba zvážit použití hardwarových či softwarových ochranných prvků sítě k zabránění zahlcení sítě nebo adaptovat architekturu k eliminaci nebo snížení konfliktních transakcí. Několika návrhům na adaptaci sítě se zabývá

Vankov (2018) popisující některé techniky k vyřešení konfliktních transakcí, které ale přinášejí různé nevýhody. Autor zde například popisuje techniku obcházející MVCC metodu, což ale přináší riziko útoku dvojité útraty a komplexnost správy klíčů. Dalším příkladem je použití fronty pro požadavky, která může být implementovaná v aplikační vrstvě nebo síťovém zařízení (například síťová brána). To ovšem přináší nevýhody v podobě nižší datové propustnosti a potenciálně komplexní správy fronty.

Hodnocením dopadu DoS útoku na komponentu blockchain sítě Hyperledger Fabric se zabývali Andola, Raghav, Gogoi a Venkatesan (2019). Jejich výsledky také naznačují, že útok DoS má významný vliv na účinnost sítě dle sledovaných ukazatelů datové propustnosti a latence. Tato práce se liší použitým postupem měření, použitým měřícím nástrojem a cílenou komponentou sítě.

6 Závěr

Prvním dílčím cílem je teoretické vymezení pojmů IoT a blockchain. Nejdříve je vymezen pojem blockchain. Je zde obecně charakterizováno, co tento termín znamená a jaké jsou výhody vyplývající z vlastností blockchain technologie. Dále jsou zde vymezeny a popsány technické aspekty blockchain technologie. Protože blockchain přináší i některé nevýhody, jsou zde popsány některé známé útoky a bezpečnost technologie. V případě vymezení pojmu IoT je postupováno obdobně, takže jsou popsány technické aspekty, architektury a bezpečnost technologie.

V rámci druhého dílčího cíle je analyzován stávající stav výzkumu a vývoje integrujícího IoT a blockchain technologie. Řada autorů zmiňuje výhody integrace těchto dvou technologií k celkovém zlepšení důvěryhodnosti a bezpečnosti dat v IoT. Z analýzy je zjištěno, že tato integrace skutečně nachází svá uplatnění, ale vždy je potřeba brát v úvahu různá rizika vyplývající ze způsobu implementace. Jedním ze způsobu implementace je právě použití blockchain platformy Hyperledger Fabric a souvisejícího frameworku Hyperledger Composer.

Na základě teoretických poznatků je v rámci dalšího dílčího cíle navrženo modelové řešení bezpečnostně kritické aplikace integrující IoT s blockchain technologií. Na základě syntézy poznatků získaných z literatury je vytvořena obchodní logika dodavatelského řetězce léků, která představuje kritickou a rizikovou oblast, kde by integrace blockchain s IoT mohla představovat vhodné řešení. Proto je zvolena jako tématika blockchain sítě, na které je později experiment proveden. Během psaní práce došlo k vydání novější zásadní verze blockchain platformy Hyperledger Fabric, konkrétně v2.0. Protože k vydání verze došlo během psaní práce, nebylo již z časových důvodů možné nejnovější verzi implementovat. Během simulace IoT zařízení odesílajícího data do implementované blockchain sítě nedochází k žádným komplikacím. Data se ukládají na blockchain a komunikace tedy probíhá v pořádku.

V rámci posledního dílčího cíle je proveden experiment nad vytvořeným řešením ověřující vhodnost integrace blockchain technologie s IoT technologií z hlediska bezpečnosti. V rámci experimentu je proveden DoS (Denial o Service) útok, který zahlcuje komponentu blockchain sítě syn pakety. K měření během experimentu je použit nástroj Apache JMeter. Je zjištěn významný vliv útoku na funkčnost sítě a na sledované ukazatele doby odezvy a datové propustnosti sítě. Blockchain síť během útoku přestala správně

pracovat a všechny odeslané požadavky jsou neúspěšné. Odeslaná data se tedy nezapisují na blockchain uvnitř sítě.

Na základě vyhodnocení experimentu ověřujícího vhodnost integrace technologií blockchain a IoT v kontextu bezpečnosti je zjištěna náchylnost konkrétní implementace blockchain sítě Hyperledger Fabric na DoS útok. V tomto stavu tedy není integrace blockchain s IoT technologií pomocí Hyperledger Fabric zcela vhodná a obsahuje bezpečnostní rizika. Především je potřeba zvážit implementaci dodatečných softwarových nebo hardwarových prvků k ochraně sítě. To vede k dalším možným rozšířením této práce o implementaci těchto ochranných prvků a následnému sledování vlivu na výkonnostní ukazatele. Dalším možným rozšířením je porovnání s jinými blockchain platformami nebo rozšíření experimentu o další typy útoků.

Práce uplatňuje teoretické poznatky k ověření vhodnosti integrace technologií blockchain a IoT v kontextu bezpečnosti. Přináší kritický pohled na bezpečnost vyplývající z této integrace vytvořením konkrétní implementace a následným provedením experimentu. Identifikuje nedostatky a bezpečnostní riziko řešení spojujícího blockchain platformu Hyperledger Fabric 1.2 s IoT. Dále vytváří základ pro metodiku k ověření bezpečnosti blockchain sítě.

Seznam použitých zdrojů

ADITHYA, , 2019. Top B2B Content Marketing Trends of 2018: Internet of Things. *The Interactive Marketing Blog by Outgrow* [online]. OutGrow [cit. 2019-10-03]. Dostupné z: https://outgrow.co/blog/content-marketing-trends-2018

ANDERSON, J., Jan LEHNARDT a Noah SLATER, 2010. *CouchDB: the definitive guide*. Sebastopol, CA: O'Reilly. ISBN 978-0596155896.

ANDOLA, Nitish RAGHAV, Manas GOGOI, S. VENKATESAN a Shekhar VERMA, 2019. Vulnerabilities on Hyperledger Fabric. *Pervasive and Mobile Computing* [online]. vol. 59. **59** [cit. 2020-02-19]. DOI: 10.1016/j.pmcj.2019.101050. ISSN 15741192. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S157411921830720X

ANDROULAKI, Elli, Yacov MANEVICH, Srinivasan MURALIDHARAN et al., 2018. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. *Proceedings of the Thirteenth EuroSys Conference on - EuroSys '18* [online]. New York, New York, USA: ACM Press, 1-15 [cit. 2019-12-04]. DOI: 10.1145/3190508.3190538. ISBN 9781450355841. Dostupné z: http://dl.acm.org/citation.cfm?doid=3190508.3190538

APACHE JMETER, , c1999-2019. *Apache JMeter* [online]. Apache Software Foundation [cit. 2020-02-05]. Dostupné z: https://jmeter.apache.org/

ARCHA, , Bithin ALANGOT a Krishnashree ACHUTHAN, 2018. Trace and Track: Enhanced Pharma Supply Chain Infrastructure to Prevent Fraud. *Ubiquitous Communications and Network Computing* [online]. Cham: Springer International Publishing, 189-195 [cit. 2020-01-09]. DOI: 10.1007/978-3-319-73423-1_17. ISBN 978-3-319-73422-4. Dostupné z: http://link.springer.com/10.1007/978-3-319-73423-1_17

ASOLO, Bisade, 2018b. Double-Spending Explained. *Mycryptopedia: Learn About Cryptocurrency & Blockchain Technology* [online]. Mycryptopedia [cit. 2019-08-23]. Dostupné z: https://www.mycryptopedia.com/double-spending-explained/

ASOLO, Bisola, 2018a. What Is SHA-256 And How Is It Related to Bitcoin?. *Mycryptopedia: Learn About Cryptocurrency & Blockchain Technology* [online]. Mycryptopedia [cit. 2019-08-21]. Dostupné z: https://www.mycryptopedia.com/sha-256-related-bitcoin/

ATTIA, Oumaima, Ines KHOUFI, Anis LAOUITI a Cedric ADJIH, 2019. An IoT-Blockchain Architecture Based on Hyperledger Framework for Healthcare Monitoring

Application. *2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS)* [online]. IEEE, 1-5 [cit. 2020-02-20]. DOI: 10.1109/NTMS.2019.8763849. ISBN 978-1-7281-1542-9. Dostupné z: https://ieeexplore.ieee.org/document/8763849/

BAECKSTROM, Chris, Guide to JMeter Thread Groups. *RedLine13: (Almost) Free Load Testing in the Cloud* [online]. RedLine13 [cit. 2020-02-06]. Dostupné z: https://www.redline13.com/blog/2018/05/guide-jmeter-thread-groups/

BAGCHI, Rupsha, 2017. *Using Blockchain Technology and Smart Contracts for Access Management in IoT devices* [online]. University of Helsinki [cit. 2019-08-15]. Dostupné z: http://urn.fi/URN:NBN:fi:hulib-201711135685. Diplomová práce. University of Helsinki. Vedoucí práce Prof. Sasu Tarkoma.

BALIGA, Arati, 2019. *Understanding blockchain consensus models* [online]. Persistent Systems [cit. 2019-12-04]. Dostupné z: https://www.persistent.com/wp-content/uploads/2018/02/wp-understanding-blockchain-consensus-models.pdf

BANO, Shehar, Mustafa AL-BASSAM a George DANEZIS, 2017. The Road to Scalable Blockchain Designs. *USENIX: The Advanced Computing Systems Association* [online]. ;login: [cit. 2019-08-12]. Dostupné z: https://www.usenix.org/publications/login/winter2017/bano

BASET, Salman, Luc DESROSIERS, Nitin GAUR, Petr NOVOTNY, Anthony O'DOWD a Venkatraman RAMAKRISHNA, 2018. *Hands-On Blockchain with Hyperledger: Building Decentralized Applications with Hyperledger Fabric and Composer*. Birmingham: Packt Publishing, Limited. ISBN 9781788994521.

BINANCE, , 2019a. What Is a Blockchain Consensus Algorithm?. *Binance Academy* [online]. Binance.com [cit. 2019-08-21]. Dostupné z: https://www.binance.vision/blockchain/what-is-a-blockchain-consensus-algorithm

BINANCE, , 2019b. Byzantine Fault Tolerance Explained. *What Makes a Blockchain Secure?* [online]. Binance.com [cit. 2019-08-21]. Dostupné z: https://www.binance.vision/blockchain/byzantine-fault-tolerance-explained

BINANCE, , 2019c. Proof of Stake Explained. *Binance Academy* [online]. Binance.com [cit. 2019-08-21]. Dostupné z: https://www.binance.vision/blockchain/proof-of-stake-explained

BINANCE, , 2019d. Delegated Proof of Stake Explained. *Binance Academy* [online]. Binance.com [cit. 2019-08-22]. Dostupné z: https://www.binance.vision/blockchain/delegated-proof-of-stake-explained

BINANCE, , 2019e. What Is Cryptocurrency Mining?. *Binance Academy* [online]. Binance.com [cit. 2019-08-22]. Dostupné z: https://www.binance.vision/blockchain/what-is-cryptocurrency-mining

BINANCE, , c2017-2019. What Makes a Blockchain Secure?. *Binance Academy: Crypto & Blockchain Education* [online]. Binance.com [cit. 2020-01-09]. Dostupné z: https://www.binance.vision/blockchain/what-makes-a-blockchain-secure

BONOMI, Flavio, Rodolfo MILITO, Jiang ZHU a Sateesh ADDEPALLI, 2012. Fog computing and its role in the internet of things. *Proceedings of the first edition of the MCC workshop on Mobile cloud computing - MCC '12* [online]. New York, New York, USA: ACM Press [cit. 2020-03-06]. DOI: 10.1145/2342509.2342513. ISBN 9781450315197.

BUTERIN, Vitalik, 2015. On Public and Private Blockchains. *Ethereum blog* [online]. Ethereum Foundation [cit. 2019-08-15]. Dostupné z: https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/

CLAUSON, Kevin, Elizabeth BREEDEN, Cameron DAVIDSON a Timothy MACKEY, 2018. Leveraging Blockchain Technology to Enhance Supply Chain Management in Healthcare. *Blockchain in Healthcare Today* [online]. [cit. 2020-01-09]. DOI: 10.30953/bhty.v1.20. ISSN 2573-8240. Dostupné z: https://blockchainhealthcaretoday.com/index.php/journal/article/view/20

CUI, Pinchen, Ujjwal GUIN, Anthony SKJELLUM a David UMPHRESS, 2019. Blockchain in IoT: Current Trends, Challenges, and Future Roadmap. *Journal of Hardware and Systems Security* [online]. vol. 3. **3**(4), 338-364 [cit. 2020-02-20]. DOI: 10.1007/s41635-019-00079-5. ISSN 2509-3428. Dostupné z: http://link.springer.com/10.1007/s41635-019-00079-5

DASGUPTA, Dipankar, John M. SHREIN a Kishor Datta GUPTA, 2019. A survey of blockchain from security perspective. *Journal of Banking and Financial Technology* [online]. vol. 3. **3**(1), 1-17 [cit. 2020-03-06]. DOI: 10.1007/s42786-018-00002-6. ISSN 2524-7956. Dostupné z: http://link.springer.com/10.1007/s42786-018-00002-6

DINH, Tien, Rui LIU, Meihui ZHANG, Gang CHEN, Beng OOI a Ji WANG, 2018. Untangling Blockchain: A Data Processing View of Blockchain Systems. *IEEE Transactions on Knowledge and Data Engineering* [online]. vol. 30. **30**(7), 1366-1385 [cit.

2019-10-14]. DOI: 10.1109/TKDE.2017.2781227. ISSN 1041-4347. Dostupné z: https://ieeexplore.ieee.org/document/8246573/

DOSEDĚL, Tomáš, 2018. Chytré zemědělství: Počítač místo vidlí. *Světchytře.cz: Píšeme o technologiích, které lidem usnadňují život.* [online]. [cit. 2019-08-29]. Dostupné z: https://www.svetchytre.cz/a/iCPP6/chytre-zemedelstvi-pocitac-misto-vidli

EVANS, John, 2019. Blockchain Nodes: An In-Depth Guide. *Nodes.com* [online]. Nodes.com [cit. 2019-08-16]. Dostupné z: https://nodes.com/#lightning-nodes

FIFE, Christopher, 2019. Securing the IoT Gateway. *Citrix: People-centric solutions for a better way to work* [online]. Citrix [cit. 2019-10-23]. Dostupné z: https://www.citrix.com/blogs/2015/07/24/securing-the-iot-gateway/

FRANCO, Pedro, 2014. *Understanding Bitcoin: Cryptography, Engineering and Economics*. John Wiley & Sons, Incorporated. ISBN 9781119019152.

FRANKENFIELD, Jake, 2019a. Hard Fork. *Investopedia* [online]. Dotdash [cit. 2019-08-16]. Dostupné z: https://www.investopedia.com/terms/h/hard-fork.asp

FRANKENFIELD, Jake, 2019b. Soft Fork. *Investopedia* [online]. Dotdash [cit. 2019-08-17]. Dostupné z: https://www.investopedia.com/terms/s/soft-fork.asp

FRUSTACI, Mario, Pasquale PACE a ALOI, 2018. Evaluating Critical Security Issues of the IoT World: Present and Future Challenges. *IEEE Internet of Things Journal* [online]. vol. 5. **5**(4), 2483-2495 [cit. 2019-10-16]. DOI: 10.1109/JIOT.2017.2767291. ISSN 2327-4662. Dostupné z: https://ieeexplore.ieee.org/document/8086136

FUENTES CONTRERAS, Alberto, 2019. *Benchmarking of blockchain technologies used in a decentralized data marketplace*. Madrid, España. oai:oa.upm.es:55775.. Final Project. E.T.S. de Ingenieros Informáticos (UPM).

HASSIJA, Vikas, Vinay CHAMOLA, Vikas SAXENA, Divyansh JAIN, Pranav GOYAL a Biplab SIKDAR, 2019. A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures. *IEEE Access* [online]. vol. 7. **7**, 82721-82743 [cit. 2019-08-28]. DOI: 10.1109/ACCESS.2019.2924045. ISSN 2169-3536. Dostupné z: https://ieeexplore.ieee.org/document/8742551/

HLŮŠKOVÁ, Kateřina, 2020. Co je Masternode a jak je užitečný pro investory?. *Investree* [online]. [cit. 2020-03-06]. Dostupné z: https://investree.cz/co-je-masternode/

HONG, Zhen, Zehua WANG, Wei CAI a Victor LEUNG, 2017. Blockchain-Empowered Fair Computational Resource Sharing System in the D2D Network. *Future*

Internet [online]. 9. **9**(4) [cit. 2019-08-18]. DOI: 10.3390/fi9040085. ISSN 1999-5903. Dostupné z: http://www.mdpi.com/1999-5903/9/4/85

HORDĚJČUK, Vojtěch, c2008-2019. Blockchain. *Vojta Hordějčuk aka voho: Software Engineer and Bedroom Music Producer* [online]. [cit. 2019-08-19]. Dostupné z: http://voho.eu/wiki/blockchain/

HYPERLEDGER, , 2018. *Hyperledger* [online]. The Linux Foundation [cit. 2019-12-03]. Dostupné z: https://www.hyperledger.org/

HYPERLEDGER COMPOSER, , 2020. *Hyperledger Composer* [online]. [cit. 2020-01-14]. Dostupné z: https://hyperledger.github.io/composer/v0.19/introduction/introduction.html

HYPERLEDGER FABRIC, , 2017. *Hyperledger Fabric: A Blockchain Platform for the Enterprise* [online]. Hyperledger [cit. 2019-12-03]. Dostupné z: https://hyperledger-fabric.readthedocs.io/en/release-1.2/index.html

IDC, , 2020. The Growth in Connected IoT Devices Is Expected to Generate 79.4ZB of Data in 2025, According to a New IDC Forecast. *IDC: The premier global market intelligence firm.* [online]. Framingham: IDC Corporate USA [cit. 2020-02-24]. Dostupné z: https://www.idc.com/getdoc.jsp?containerId=prUS45213219

JAIN, Prachi, Patwa SANJIVANI a Kancha JHA, 2017. *Architecture of Internet of Things (IoT)* [online]. 5. IJSRD - International Journal for Scientific Research & Development [cit. 2019-10-08]. ISSN 2321-0613. Dostupné z: http://www.ijsrd.com/articles/IJSRDV5I90212.pdf

KLAOKLIANG, Nuttapong, Padungpol TEAWTIM, Phet AIMTONGKHAM, Chakchai SO-IN a Aimaschana NIRUNTASUKRAT, 2018. A Novel IoT Authorization Architecture on Hyperledger Fabric With Optimal Consensus Using Genetic Algorithm. *2018 Seventh ICT International Student Project Conference (ICT-ISPC)* [online]. IEEE, 1-5 [cit. 2020-02-20]. DOI: 10.1109/ICT-ISPC.2018.8523942. ISBN 978-1-5386-7804-6. Dostupné z: https://ieeexplore.ieee.org/document/8523942/

KSHETRI, Nir, 2017. Can Blockchain Strengthen the Internet of Things?. *IT Professional* [online]. vol. 19. **19**(4), 68-72 [cit. 2019-10-09]. DOI: 10.1109/MITP.2017.3051335. ISSN 1520-9202. Dostupné z: http://ieeexplore.ieee.org/document/8012302/

KUHRT, Tracy, 2019. Hyperledger Fabric. *Hyperledger* [online]. [cit. 2020-01-09]. Dostupné z: https://wiki.hyperledger.org/display/fabric/Hyperledger+Fabric

KUHRT, Tracy, 2020. *Welcome to the Hyperledger Composer wiki!* [online]. GitHub [cit. 2020-02-06]. Dostupné z: https://github.com/hyperledger/composer/wiki

KUMAR, Mohit, 2019. How to Hack WiFi Password from Smart Doorbells. *The Hacker News: Latest Cyber Security News* [online]. The Hacker News [cit. 2019-10-09]. Dostupné z: https://thehackernews.com/2016/01/doorbell-hacking-wifi-pasword.html

LASTOVETSKA, Anastasiia, 2019. Blockchain Architecture Basics: Components, Structure, Benefits & Creation. *MLSDev* [online]. MLSDev [cit. 2019-08-15]. Dostupné z: https://mlsdev.com/blog/156-how-to-build-your-own-blockchain-architecture

LISK, , 2019. Blockchain Basics: How Blockchain Works. *Lisk* [online]. Lisk [cit. 2019-08-15]. Dostupné z: https://lisk.io/academy/welcome-to-the-lisk-academy

LI, Wenting, Sébastien ANDREINA, Jens-Matthias BOHLI a Ghassan KARAME, 2017a. Securing Proof-of-Stake Blockchain Protocols. *Data Privacy Management, Cryptocurrencies and Blockchain Technology* [online]. Cham: Springer International Publishing, 297-315 [cit. 2019-08-22]. Lecture Notes in Computer Science. DOI: 10.1007/978-3-319-67816-0_17. ISBN 978-3-319-67815-3. Dostupné z: http://link.springer.com/10.1007/978-3-319-67816-0_17

LI, Wenting, Sébastien ANDREINA, Jens-Matthias BOHLI a Ghassan KARAME, 2017b. Securing Proof-of-Stake Blockchain Protocols. *Data Privacy Management, Cryptocurrencies and Blockchain Technology* [online]. Cham: Springer International Publishing, 297-315 [cit. 2019-08-23]. Lecture Notes in Computer Science. DOI: 10.1007/978-3-319-67816-0_17. ISBN 978-3-319-67815-3. Dostupné z: http://link.springer.com/10.1007/978-3-319-67816-0_17

LOISEL, Jérôme, c2014-2020. JMeter result analysis: the ultimate guide. *OctoPerf* [online]. OctoPerf [cit. 2020-02-13]. Dostupné z: https://octoperf.com/blog/2017/10/19/how-to-analyze-jmeter-results/

LUETH, Knud, 2019. State of the IoT 2018: Number of IoT devices now at 7B – Market accelerating. *IoT Analytics: Market insights for the Internet of Things* [online]. IoT Analytics GmbH [cit. 2019-10-03]. Dostupné z: https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/

LUU, Loi, Viswesh NARAYANAN, Chaodong ZHENG, Kunal BAWEJA, Seth GILBERT a Prateek SAXENA, 2016. A Secure Sharding Protocol For Open Blockchains. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security - CCS'16* [online]. New York, New York, USA: ACM Press, 17-30 [cit. 2019-08-

12]. DOI: 10.1145/2976749.2978389. ISBN 9781450341394. Dostupné z: http://dl.acm.org/citation.cfm?doid=2976749.2978389

MAKHDOOM, Imran, Mehran ABOLHASAN, Haider ABBAS a Wei NI, 2019. Blockchain's adoption in IoT: The challenges, and a way forward. *Journal of Network and Computer Applications* [online]. vol. 125. **125**, 251-279 [cit. 2020-02-20]. DOI: 10.1016/j.jnca.2018.10.019. ISSN 10848045. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S1084804518303473

MARTINEZ, Julian, 2018a. Dispelling Myths: How a Pruned Ethereum Node Can Fully Verify the Blockchain. *Medium* [online]. [cit. 2019-08-16]. Dostupné z: https://medium.com/coinmonks/how-a-pruned-ethereum-node-can-fully-verify-the-blockchain-bbe9f29663ed

MARTINEZ, Julian, 2018b. Understanding Proof of Stake: The Nothing at Stake Theory. *Medium* [online]. Medium [cit. 2019-08-23]. Dostupné z: https://medium.com/coinmonks/understanding-proof-of-stake-the-nothing-at-stake-theory-1f0d71bc027

MELNIK, Ian, c2015-2020. Comparison: Ethereum vs Hyperledger Fabric vs R3 Corda. *Merehead* [online]. Merehead [cit. 2020-01-09]. Dostupné z: https://merehead.com/blog/comparison-ethereum-hyperledger-fabric-r3-corda/

MILLER, Dennis, 2018. Blockchain and the Internet of Things in the Industrial Sector. *IT Professional* [online]. vol. 20. **20**(3), 15-18 [cit. 2019-10-14]. DOI: 10.1109/MITP.2018.032501742. ISSN 1520-9202. Dostupné z: https://ieeexplore.ieee.org/document/8378971/

MOUGAYAR, William a Vitalik BUTERIN, 2016. *The business blockchain: Promise, practice, and application of the next Internet technology*. Hoboken, New Jersey: John Wiley & Sons, Incorporated. ISBN 978-111-9300-335.

NAKAMOTO, Satoshi, 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. *Bitcoin: Open source P2P money* [online]. [cit. 2019-08-12]. Dostupné z: https://bitcoin.org/bitcoin.pdf

NODE-RED, , b.r. *Node-RED: Documentation* [online]. [cit. 2019-12-02]. Dostupné z: https://nodered.org/docs/

NOFER, Michael, Peter GOMBER, Oliver HINZ a Dirk SCHIERECK, 2017. *Blockchain* [online]. vol. 59. **59**(3), 183-187 [cit. 2020-03-06]. DOI: 10.1007/s12599-017-

0467-3. ISSN 2363-7005. Dostupné z: http://link.springer.com/10.1007/s12599-017-0467-3

NOVO, Oscar, 2018. Blockchain Meets IoT: An Architecture for Scalable Access Management in IoT. *IEEE Internet of Things Journal* [online]. 5. **5**(2), 1184-1195 [cit. 2019-08-12]. DOI: 10.1109/JIOT.2018.2812239. ISSN 2327-4662. Dostupné z: https://ieeexplore.ieee.org/document/8306880/

ORMAN, Hilarie, 2018. Blockchain: the Emperors New PKI?. *IEEE Internet Computing* [online]. vol. 22. **22**(2), 23-28 [cit. 2019-10-14]. DOI: 10.1109/MIC.2018.022021659. ISSN 1089-7801. Dostupné z: https://ieeexplore.ieee.org/document/8345567/

OURAD, Abdallah, Boutheyna BELGACEM a Khaled SALAH, 2018. Using Blockchain for IOT Access Control and Authentication Management. *Internet of Things – ICIOT 2018* [online]. Cham: Springer International Publishing, 150-164 [cit. 2020-02-20]. Lecture Notes in Computer Science. DOI: 10.1007/978-3-319-94370-1_11. ISBN 978-3-319-94369-5. Dostupné z: http://link.springer.com/10.1007/978-3-319-94370-1_11

PANARELLO, Alfonso, Nachiket TAPAS, Giovanni MERLINO, Francesco LONGO a Antonio PULIAFITO, 2018. Blockchain and IoT Integration. *Sensors* [online]. vol. 18. **18**(8) [cit. 2019-10-14]. DOI: 10.3390/s18082575. ISSN 1424-8220.

PARTICLE, , 2019. The 2019 state of IoT report. *Particle: Welcome to real IoT* [online]. Particle [cit. 2019-10-03]. Dostupné z: https://www.particle.io/solutions/2019-state-of-iot-report/

PENTTINEN, Jyrki, 2017. *Wireless communications security: Solutions for the internet of things*. The Atrium, Southern Gates, Chichester, West Sussex, United Kingdom: John Wiley & Sons, Incorporated. ISBN 9781119084419.

PLCHÚT, Martin, c2001-2019. Co je Smart Grid?. *TZB-info: Stavebnictví, úspory energií, technická zařízení budov* [online]. Siemens [cit. 2019-08-29]. Dostupné z: https://elektro.tzb-info.cz/12544-co-je-smart-grid

PRERNA, , 2019. Hyperledger vs Ethereum: Which Blockchain Platform Will Benefit Your Business?. *Edureka* [online]. Brain4ce Education Solutions Pvt. Ltd. [cit. 2020-01-09]. Dostupné z: https://www.edureka.co/blog/hyperledger-vs-ethereum/

QU, Chao, Ming TAO a Ruifen YUAN, 2018. A Hypergraph-Based Blockchain Model and Application in Internet of Things-Enabled Smart Homes. *Sensors* [online]. vol. 18.

18(9) [cit. 2020-02-20]. DOI: 10.3390/s18092784. ISSN 1424-8220. Dostupné z: http://www.mdpi.com/1424-8220/18/9/2784

RAY, James, 2018. Problems. *Ethereum/wiki* [online]. [cit. 2019-08-23]. Dostupné z: https://github.com/ethereum/wiki/wiki/Problems

SALEH, Fahad, 2018. Blockchain Without Waste: Proof-of-Stake. In: *SSRN Electronic Journal* [online]. [cit. 2020-03-06]. DOI: 10.2139/ssrn.3183935. ISSN 1556-5068. Dostupné z: https://www.ssrn.com/abstract=3183935

SALMAN, Tara, Maede ZOLANVARI, Aiman ERBAD, Raj JAIN a Mohammed SAMAKA, 2019. *Security Services Using Blockchains: A State of the Art Survey* [online]. In: . vol. 21. [cit. 2019-10-16]. DOI: 10.1109/COMST.2018.2863956. ISSN 1553-877X. Dostupné z: https://ieeexplore.ieee.org/document/8428402/

SANTOS, Maximiliano a Enio MOURA, 2019. *Hands-On IoT Solutions with Blockchain: Discover how converging IoT and blockchain can help you build effective solutions.* Birmingham: Packt Publishing, 206 s. ISBN 978-1-78913-224-3.

SHARMA, Abhishek, 2018. Understanding Proof of Stake through it's Flaws: Part 3 - 'Long Range Attacks'. *Medium* [online]. Medium [cit. 2019-08-28]. Dostupné z: https://medium.com/@abhisharm/understanding-proof-of-stake-through-its-flaws-part-3-long-range-attacks-672a3d413501

SHEN, Jian, Chen WANG, Tong LI, Xiaofeng CHEN, Xinyi HUANG a Zhi-Hui ZHAN, 2018. Secure data uploading scheme for a smart home system. *Information Sciences* [online]. vol. 453. **453**, 186-197 [cit. 2020-02-20]. DOI: 10.1016/j.ins.2018.04.048. ISSN 00200255. Dostupné z: https://linkinghub.elsevier.com/retrieve/pii/S0020025518303001

SMOLOVÁ, Veronika, 2018. Jak na chytrou domácnost?. *Blog ZOOCO: Žijte chytře* [online]. [cit. 2019-08-29]. Dostupné z: https://www.zooco.io/blog/jak-na-chytrou-domacnost/

SUHAIL, Sabah, Choon HONG, Zuhaib AHMAD, Faheem ZAFAR a Abid KHAN, 2016. Introducing Secure Provenance in IoT: Requirements and Challenges. *2016 International Workshop on Secure Internet of Things (SIoT)* [online]. IEEE, 39-46 [cit. 2019-10-09]. DOI: 10.1109/SIoT.2016.011. ISBN 978-1-5090-5091-8. Dostupné z: http://ieeexplore.ieee.org/document/7913564/

SURESH, P., J. DANIEL, V. PARTHASARATHY a R. ASWATHY, 2014. A state of the art review on the Internet of Things (IoT) history, technology and fields of deployment.

2014 International Conference on Science Engineering and Management Research (ICSEMR) [online]. IEEE, 1-8 [cit. 2019-08-28]. DOI: 10.1109/ICSEMR.2014.7043637. ISBN 978-1-4799-7613-3. Dostupné z: http://ieeexplore.ieee.org/document/7043637/

SWAMY, Sowmya, Dipti JADHAV a Nikita KULKARNI, 2017. Security threats in the application layer in IOT applications. *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)* [online]. IEEE, 477-480 [cit. 2019-10-23]. DOI: 10.1109/I-SMAC.2017.8058395. ISBN 978-1-5090-3242-6. Dostupné z: http://ieeexplore.ieee.org/document/8058395/

SZABO, Nick, 1997. Formalizing and Securing Relationships on Public Networks. *First Monday* [online]. 2. **2**(9) [cit. 2019-08-15]. DOI: 10.5210/fm.v2i9.548. ISSN 13960466. Dostupné z: http://journals.uic.edu/ojs/index.php/fm/article/view/548

THUMMAVET, Phuwanai, 2019. Demystifying Hyperledger Fabric: Fabric Architecture. *Medium* [online]. Medium [cit. 2019-12-11]. Dostupné z: https://medium.com/coinmonks/demystifying-hyperledger-fabric-1-3-fabric-architecture-a2fdb587f6cb

UPASANA, , 2019. Real World IoT Applications in Different Domains. *Edureka* [online]. Brain4ce Education Solutions [cit. 2019-08-29]. Dostupné z: https://www.edureka.co/blog/iot-applications/#industrial%20automation

VANKOV, Ivan, 2018. How to prevent key collisions in Hyperledger Fabric chaincode. *Medium* [online]. Medium [cit. 2020-02-10]. Dostupné z: https://medium.com/@gatakka/how-to-prevent-key-collisions-in-hyperledger-fabric-chaincode-303700716733

VODAFONE, , 2019. Slovník pojmů. *Vodafone* [online]. Vodafone Czech Republic a.s. [cit. 2019-08-28]. Dostupné z: https://www.vodafone.cz/uzitecne-odkazy/slovnik-pojmu/gsm/

WANG, Wei, Peng XU a Laurence YANG, 2018. Secure Data Collection, Storage and Access in Cloud-Assisted IoT. *IEEE Cloud Computing* [online]. vol. 5. **5**(4), 77-88 [cit. 2019-10-09]. DOI: 10.1109/MCC.2018.111122026. ISSN 2325-6095. Dostupné z: https://ieeexplore.ieee.org/document/8255790/

WANG, Wenbo, Dinh Thai HOANG, Peizhao HU, Zehui XIONG, Dusit NIYATO, Ping WANG, Yonggang WEN a Dong In KIM, 2019. A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks. *IEEE Access*

[online]. vol. 7. **7**, 22328-22370 [cit. 2020-03-06]. DOI: 10.1109/ACCESS.2019.2896108. ISSN 2169-3536. Dostupné z: https://ieeexplore.ieee.org/document/8629877/

XIAO, Liang, Xiaoyue WAN, Xiaozhen LU, Yanyong ZHANG a Di WU, 2018. IoT Security Techniques Based on Machine Learning: How Do IoT Devices Use AI to Enhance Security?. *IEEE Signal Processing Magazine* [online]. vol. 35. **35**(5), 41-49 [cit. 2019-10-09]. DOI: 10.1109/MSP.2018.2825478. ISSN 1053-5888. Dostupné z: https://ieeexplore.ieee.org/document/8454402/

YANG, Fan, Wei ZHOU, QingQing WU, Rui LONG, Neal N. XIONG a ZHOU, 2019. Delegated Proof of Stake With Downgrade: A Secure and Efficient Blockchain Consensus Algorithm With Downgrade Mechanism. *IEEE Access* [online]. vol. 7. **7** [cit. 2020-03-06]. DOI: 10.1109/ACCESS.2019.2935149. ISSN 2169-3536. Dostupné z: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8798621&isnumber=8600701

ZHENG, Zibin, Shaoan XIE, Hongning DAI, Xiangping CHEN a Huaimin WANG, 2017. An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. *2017 IEEE International Congress on Big Data (BigData Congress)* [online]. IEEE, 557-564 [cit. 2020-03-06]. DOI: 10.1109/BigDataCongress.2017.85. ISBN 978-1-5386-1996-4. Dostupné z: http://ieeexplore.ieee.org/document/8029379/

Seznam použitých zkratek

1G	1. generation	1. generace bezdrátové telefonní technologie
2G	2. generation	2. generace bezdrátové telefonní technologie
ASIC	Application Specific Integrated Circuit	Integrované čipy se specifickým aplikačním určením
BFT	Byzantine Fault Tolerance	Volně přeloženo jako „Byzantská odolnost vůči chybám“, která vychází z problém dvou armád
CFT	Crash fault-tolerant	Odolnost proti selhání
CPU	Central Processing Unit	Centrální procesorová jednotka
cURL	Client for URLs	Klient pro URL adresy
DDoS	Distributed Denial of Service	Distribuované odepření služby
DoS	Denial of Service	Odepření služby
DPoS	Delegated Proof of Stake	Delegovaný důkaz o vkladu
GPU	Graphics Processing Unit	Grafický procesor
GSM	Groupe Spécial Mobile	Globální systém mobilní komunikace
IoT	Internet of Things	Internet věcí
MITM	Man-in-the-middle	„prostředník“ / „člověk uprostřed“
MSP	Membership Service Provider	Poskytovatel členských služeb
P2P	Peer-to-peer	Decentralizovaný komunikační model
PBFT	Practical Byzantine Fault Tolerance	Volně přeloženo jako „Praktická byzantská odolnost vůči chybám“
PKI	Public Key Infrastructure	Infrastruktura veřejného klíče
PoA	Proof of Autority	Důkaz autority
PoS	Proof of Stake	Důkaz o vkladu (mechanismus konsensu)
PoW	Proof of Work	Důkaz o práci (mechanismus konsensu)

REST	Representational State Transfer	Rozhraní pro distribuované prostředí orientované na data
SDK	Software development kit	Sada vývojových nástrojů
SHA	Secure Hash Algorithm	Kryptografická hašovací funkce
SPV	Simple Payment Verification	Jednoduché ověření platby
TXID	Transaction ID	Identifikační kód transakce
TxIn	Transaction inputs	Transakční vstupy
TxOut	Transaction outputs	Transakční výstupy
UX	User Experience	Uživatelská přívětivost

Seznam obrázků

Seznam tabulek

Seznam grafů

7 Přílohy

Příloha 1 - Kód úspěšné transakce aktualizující hodnoty zásilky


```
{
    "$class": "org.xsals.lekarna.aktualizujPrepravniData",
    "krabice": "resource:com.xsals.businessnetwork.lekarna.Zasilka#3",
    "lokacniInformace": {
        "$class": "org.xsals.lekarna.Lokace",
        "datum": "2019-11-27T20:33:22.090Z",
        "lokace": "PREPRAVA",
        "status": "NA_CESTE",
        "lokaceID": "100"
    },
    "senzoroveInformace": {
        "$class": "org.xsals.lekarna.Mereni",
        "datum": "2019-11-27T20:33:22.090Z",
        "teplota": 23,
        "vlhkost": 60
    },
    "transactionId": "b0ff5dbc1170bca7535ca2911e0e2b5f95baa655f85c427a71e7ce7fdfb72d80",
    "timestamp": "2019-11-27T20:33:22.095Z"
}
```


Příloha 1 - Kód úspěšné transakce aktualizující hodnoty zásilky.

Příloha 2 - Definice transakční logiky.

```
/**
* Funkce aktualizuje zásilku poskytnutými daty.
* @param {com.xsals.businessnetwork.lekarna.aktualizujPrepravniData} tx Aktualizuje zásilku poskytnu-
tými daty.
* @transaction
*/

async function aktualizujPrepravniData(tx) {
    // Získání parametrů transakce.
    let novaHodnota = tx.krabice;
    let lokace = tx.lokacniInformace;
    let mereni = tx.senzoroveInformace;

    // Aktualizuje zásilku poskytnutými daty.
    if (!novaHodnota.lokacniSledovaciInformace || novaHodnota.lokacniSledovaciInformace == undefined)
        novaHodnota.lokacniSledovaciInformace = [];

    if (!novaHodnota.senzoroveSledovaciInformace || novaHodnota.senzoroveSledovaciInformace == undefined)
        novaHodnota.senzoroveSledovaciInformace = [];

        novaHodnota.lokacniSledovaciInformace.push(lokace);
        novaHodnota.senzoroveSledovaciInformace.push(mereni);

    // Získání registru aktiva.
    let registrAktiv = await
        getAssetRegistry('org.xsals.lekarna.Zasilka');

    // Aktualizace aktivum v registru aktiv.
    await registrAktiv.update(novaHodnota);
}
```

Příloha 2 - Definice transakční logiky.

Příloha 3 - Definice kontroly řízení přístupu.

```
rule SystemACL {
  description:  "System ACL to permit all access"
  participant: "ANY"
  operation: ALL
  resource: "org.hyperledger.composer.system.**"
  action: ALLOW
}

rule Default {
    description: "Allow all participants access to all resources"
    participant: "ANY"
    operation: ALL
    resource: "org.xsals.lekarna.*"
    action: ALLOW
}
```

Příloha 3 - Definice kontroly řízení přístupu.

Příloha 4 - JavaScript kód uvnitř funkčního uzlu v Node-RED

```
// Pole pseudonáhodných teplot.
var temp1 = [15,17,18.5,20,21.5,23,24,22.2,19,32];

// Pole pseudonáhodných relativních vlhkostí.
var humidity1 = [50,55,61,68,65,60,53,49,45,47];

// Počítadlo pro výběr z pole.
var counter1 = context.get('counter1')||0;
counter1 = counter1+1;
if(counter1 > 9) counter1 = 0;
context.set('counter1',counter1);

// Tělo zprávy v JSON formátu
msg.payload = {
    "$class":
        "org.xsals.lekarna.aktualizujPrepravniData",
    "krabice":
        "resource: org.xsals.lekarna.Zasilka#3",
    "lokacniInformace": {
        "$class": " org.xsals.lekarna.Lokace",
        "datum": Date.now(),
        "lokace": "PREPRAVA",
        "status": "NA_CESTE",
        "lokaceID": "100"
    },
    "senzoroveInformace": {
        "$class": " org.xsals.lekarna.Mereni",
        "datum": Date.now(),
        "teplota": temp1[counter1],
        "vlhkost": humidity1[counter1]
    }
};

// Převod JavaScript hodnoty na JavaScript objekt (JSON).
devicedata = JSON.stringify(msg.payload);

// URL na REST API blockchain sítě.
msg.url = "http://localhost:3000/api/aktualizujPrepravniData?data="+devicedata;
msg.headers = {
    "Content-Type": "application/json",
    "Accept": "application/json"
};

return msg;
```

Příloha 4 - JavaScript kód uvnitř funkčního uzlu v Node-RED.

Příloha 5 - Tělo HTTP požadavku na aktualizaci zásilky metodou POST.


```
{
    "$class": "org.xsals.lekarna.aktualizujPrepravniData",
    "krabice": "org.xsals.lekarna.Zasilka#3",
    "lokacniInformace": {
        "$class": "org.xsals.lekarna.Lokace",
        "datum": "2019-11-27T20:31:25.617Z",
        "lokace": "PREPRAVA",
        "status": "NA_CESTE",
        "lokaceID": "100",
        "id": "string"
    },
    "senzoroveInformace": {
        "$class": "org.xsals.lekarna.Mereni",
        "datum": "2019-11-27T20:31:25.617Z",
        "teplota": 20,
        "vlhkost": 20,
        "id": "string"
    },
    "transactionId": "0f2c2a6c2a75d7650ec8d8bc526dbc030b5fdbf6b11b95b116d096a7e2e0284b",
    "timestamp": "2019-11-27T20:31:25.734Z"
}
```


Příloha 5 - Tělo HTTP požadavku na aktualizaci zásilky metodou POST.

Příloha 6 - Tělo odesílaného HTTP požadavku.


```
POST http://127.0.0.1:3000/api/aktualizujPrepravniData

POST data:
{
    "$class": "org.xsals.lekarna.aktualizujPrepravniData",
    "krabice": "org.xsals.lekarna.Zasilka#6",
    "lokacniInformace": {
        "$class": "org.xsals.lekarna.Lokace",
        "datum": 1.581081041028E12,
        "lokace": "PREPRAVA",
        "status": "NA_CESTE",
        "lokaceID": 9384
    },
    "senzoroveInformace": {
        "$class": "org.xsals.lekarna.Mereni",
        "datum": 1.581081041041E12,
        "teplota": 14,
        "vlhkost": 42
    }
}
[no cookies]
```


Příloha 6 - Tělo odesílaného HTTP požadavku.

Příloha 7 - Kód uložené zásilky v databázi.


```
{
  "$class": "org.xsals.lekarna.Zasilka",
  "zasilkaID": "6",
  "lokacniSledovaciInformace": [
    {
      "$class": "org.xsals.lekarna.Lokace",
      "datum": "2020-02-07T13:10:41.028Z",
      "lokace": "PREPRAVA",
      "status": "NA_CESTE",
      "lokaceID": "9384"
    }
  ],
  "senzoroveSledovaciInformace": [
    {
      "$class": "org.xsals.lekarna.Mereni",
      "datum": "2020-02-07T13:10:41.041Z",
      "teplota": 14,
      "vlhkost": 42
    }
  ],
  "vlastnik": "resource:org.xsals.lekarna.Prepravce#2"
}
```


Příloha 7 - Kód uložené zásilky v databázi.

Příloha 8 - Kód transakce uložené na blockchainu.


```
{
 "$class": "org.xsals.lekarna.aktualizujPrepravniData",
 "krabice": "resource:org.xsals.lekarna.Zasilka#6",
 "lokacniInformace": {
  "$class": "org.xsals.lekarna.Lokace",
  "datum": "2020-02-07T13:10:41.028Z",
  "lokace": "PREPRAVA",
  "status": "NA_CESTE",
  "lokaceID": "9384"
 },
 "senzoroveInformace": {
  "$class": "org.xsals.lekarna.Mereni",
  "datum": "2020-02-07T13:10:41.041Z",
  "teplota": 14,
  "vlhkost": 42
 },
 "transactionId": "fab20390253350c527b1ad0e0485a74fe39d1c378fe359d70af9f9b52b4fdc2b",
 "timestamp": "2020-02-07T13:10:41.043Z"
}
```


Příloha 8 - Kód transakce uložené na blockchainu.