FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Aplikace technologie Blockchain v chytrých konktraktech
Student:	Jaroslav Pešek
Vedoucí:	doc. Ing. Štěpán Starosta, Ph.D.
Studijní program:	Informatika
Studijní obor:	Bezpečnost a informační technologie
Katedra:	Katedra počítačových systémů
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

1. Proveďte rešerši využití technologie Blockchain. Zaměřte se na její bezpečnostní aspekty.
2. Proveďte rešerši chytrých kontraktů (Smart contracts) a existujících implementací, které chytré kontrakty umožňují.
3. Navrhněte modelové využití chytrých kontraktů v prostředí FIT. Pro svůj návrh vyjděte z vámi vybraného řešení využívající technologii Blockchain. Zvláštní pozornost věnujte bezpečnosti vašeho návrhu.
4. Demonstruje možnost reálného využití vašeho návrhu (proof of concept).

Seznam odborné literatury

Dodá vedoucí práce.

prof. Ing. Róbert Lórencz, CSc.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 10. prosince 2017

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Aplikace technologie Blockchain v chytrých kontraktech

Jaroslav Pešek

Katedra počítačových systémů
Vedoucí práce: doc. Ing. Štěpán Starosta, Ph.D.

15. května 2019

Poděkování

Děkuji vedoucímu práce doc. Ing. Štěpánu Starostovi, Ph.D. za výběr zajímavého tématu a jeho rady, stejně tak jako celé instituci, která mi pomohla získat cenné vědomosti.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo"), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 15. května 2019 . . . . . .

Odkaz na tuto práci

Pešek, Jaroslav. *Aplikace technologie Blockchain v chytrých kontraktech*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2019.

Abstrakt

Tématem práce je představení technologie blockchain jako důvěryhodné výpočetní distribuované platformy. Práce obsahuje stručný úvod do použité kryptografie v existujících blockchainech. Dále rozebírá architekturu blockchainu, způsob fungování a bezpečnostní rizika. Představuje celý koncept chytrých kontraktů. Výstupem práce je analýza a model využití těchto nových technologií v instituci Fakulty informačních technologií při volbách do akademického senátu.

Klíčová slova proof of concept, blockchain, chytré kontrakty, aplikovaná kryptografie, peer-to-peer síť

Abstract

The subject of this thesis is an insight into blockchain technology as a reliable computing distributed platform. A short introduction to cryptography used in existing blockchains is provided, as well as architecture, principle and potential security problems. It describes the concept of smart contracts. The output of this thesis is the analysis and model of application of these new technologies in the institution of the Faculty of Information Technology during election to academic senate.

Keywords proof of concept, blockchain, smart contracts, applied cryptography, peer-to-peer network

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Decentralizované aplikace v rámci internetu zažívají v posledních letech mimořádný rozmach. Již dávno nejsou pouze záležitostí hrstky nadšenců a velké množství subjektů středního proudu bere existenci takových aplikací na vědomí.

Jedna z nejznámějších technologií, která absolutní decentralizaci umožňuje, se nazývá blockchain, kde každý účastník sítě poskytuje výpočetní výkon svého zařízení do sdílené sítě. O oblíbenosti této technologie svědčí kapitalizace nejznámější kryptoměny založené na blockchainu, Bitcoinu, která na svém vrcholu dosáhla 50 miliard dolarů.

Zajímavý koncept, který dokáže potenciál blockchainu jakožto výpočetní platformy využít, jsou chytré kontrakty. Ty umožňují tvorbu dohod, kontraktů, mezi dvěma subjekty. Díky blockchainu není potřeba centrálních autorit jako dozorčího nebo exekutivního objektu, protože každý účastník kontroluje průběh a užívaný protokol postihuje či odměňuje aktéry kontraktu.

Výstupem práce je návrh na využití chytrých kontraktů na platformě blockchain na půdě Fakulty informačních technologií a proof of concept takového modelu.

Téma jsem si zvolil, neboť technologie blockchain je velmi mladá a nabízí široké spektrum aplikací, které zcela jistě nejsou uspokojivě prozkoumány. Jako aplikaci provozovanou na blockchainu jsem zvolil chytré kontrakty, které nabízejí zobecnění dohod mezi subjekty, jsou sepsané ve formálním jazyce a právě díky architektuře blockchainu jsou i bez centrální autority.

Kapitola 1 je věnována kryptografickému minimu, jež je používáno ve zbytku práce – hashovací funkci, asymetrickému šifrování a digitálnímu podpisu. V následující kapitole 2 je definován blockchain se všemi jeho principy a nutnými součástmi. Jsou představena bezpečnostní rizika a využití. Kapi-

tola 3 se věnuje chytrým kontraktům, což je již starší koncept a blockchain jim poskytl novou platformu. V další, 4. kapitole, rozebírá práce možnosti využití představených technologií v prostředí fakulty a v následující, poslední kapitole, je představen proof of concept zvolené možnosti.

Práce nepřímo navazuje na vzniklé bakalářské práce při Fakultě informačních technologií věnující se blockchainu (projekt FITcoin) a rozšiřuje jeho využití.

Cílem rešeršních prvních částí bakalářské práce je seznámení se základními pojmy, jež jsou potřeba k vybudování jednoduchého modelu blockchainu, a zároveň s existujícími implementacemi. Věnují se i takzvaným chytrým kontraktům (*smart contracts*), jejich významem a využitím. Důležité je jejich propojení s blockchainem jakožto bezpečnou a výkonnou platformou.

Praktická část se zabývá návrhem modelového využití chytrých kontraktů v prostředí Fakulty informačních technologií ČVUT. Řešení bude vycházet z poznatků z rešerše a důraz bude kladen na kryptografickou bezpečnost návrhu. Proveditelnost modelu bude dokázána pomocí *proof of concept*.

Kapitola 1

Použité kryptografické znalosti

Do širšího povědomí se pojem blockchain dostal díky digitální měně *Bitcoin*. Již před ním se však vyskytly pokusy o digitální měnu, avšak drtivá většina jich skončila neúspěchem a blockchain v podobě, jak jej známe dnes, přinesl právě Bitcoin [1]. V této kapitole bude představen celý koncept současného blockchainu a budou popsány všechny jeho nedílné součásti včetně definicí a budou popsány slabiny z hlediska bezpečnosti a z toho vyplývající známé vektory útoku.

Přestože blockchain v tom nejobecnějším smyslu je relativně jednoduchá struktura, je založena na některých netriviálních kryptografických principech.

1.1 Hashovací funkce a hash

Ústředním pojmem je takzvaná kryptografická hashovací[1] funkce, jejíž produktem je hash nebo též fingerprint. Ta slouží skutečně podobně jako otisk prstů v kriminalistice, jelikož v ideálním případě požadujeme, aby každý kus informace (obecně posloupnost n bitů) se hashovací funkcí transformoval do unikátní posloupnosti m bitů a zároveň, aby proces byl neinverzibilní.

1.1.1 Kryptografická hashovací funkce

Mějme množiny textových řetězců A, B. Kryptografická hashovací funkce Hs je nějaké zobrazení $Hs : A \rightarrow B$, které má následující sadu vlastností:

bezkoliznost I $x, y \in A, x \neq y \Longrightarrow Hs(x) \neq Hs(y)$

bezkoliznost II pro každé x neexistuje y tak, že $Hs(x) = Hs(y)$

[1] též jednosměrná nebo hešovací

totálnost definiční obor je A

jednosměrnost pro nějaký známý výstup zobrazení Hs r, $Hs(x) = r$ je velmi těžko spočítatelné x

V praxi však vyžadujeme, aby množina B byla tvořena pouze posloupnostmi nějaké délky m, ale zároveň množina A je tvořena úplně všemi řetězci. To implikuje, že bezkoliznost nebude tak dobře možná. Proto cílem hashovacích funkcí je spíše udělat nalezení kolizí či vzoru velmi těžkou úlohou. Příkladem nějaké reálné funkce je například (nechvalně [2]) proslulá MD5 s délkou výstupu 128 bitů. Příklad některých vstupů a výstupů je v tabulce 1.1.

Nejnovější standard v rodině SHA, která je specifikována Národním institutem standardů a technologií v USA, je SHA-3 [3], která se aktivně využívá [4] a zároveň se aktivně využívají i starší standardy, jako třeba SHA-256 [5].

Tabulka 1.1: Příklad některých výstupů funkce MD5

Vstupní řetězec	Výstup funkce MD5
Jaroslav Pešek	CD230B74DEFF484CA07B694B3F7F06B9
21-7-1977	40B8BFF073B6F15EE3B11564BE382B9D
10110110	6855267DF61E69E8ABB6797D74EF42B3
int main(){}	B6918CDED4F4CDE51516F93C7C6C0960

1.2 Kryptografie s veřejným klíčem a digitální podpis

Nacházíme-li se v prostoru, kterému nelze věřit, například velká počítačová síť jako je internet, je často bezpodmínečně nutné šifrovat komunikaci ať už s jiným člověkem, nebo serverem. Slabina symetrického šifrování je v tom, že je potřeba znát dešifrovací klíč pro dešifrování zprávy, který je snadno vypočitatelný z šifrovacího[2] [6] , popřípadě se zcela shodují. Šifrování s veřejným klíčem[3] řeší tento problém a dovoluje komukoliv zašifrovat veřejným klíčem zprávu, kterou může dešifrovat pouze entita[4] vlastnící klíč soukromý.

[2] a vice versa
[3] též asymetrické šifrování
[4] člověk nebo stroj

1.2.1 Asymetrická kryptografie nad eliptickými křivkami

V nejmasovějších veřejných blockchainech, což je Bitcoin a *Ethereum* [7] se pro asymetrickou kryptografii používají eliptické křivky [8, 4]. Kryptografie nad eliptickými křivkami je způsob asymetrického šifrování založený na problému diskrétního logaritmu přeneseného nad množinu bodů dané eliptické křivky.

Základem toho je eliptická křivka nad tělesem $\mathbb{R}$, což je množina bodů $E \subset \mathbb{R}^2$, která vyhovuje řešení rovnice 1.1 se splněním podmínky 1.2 [9] a dodefinovaný bod $\mathcal{O}$, který bude popsán dále v textu. Tento text se bude zabývat právě tímto tvarem, kterému se také říká Weierstrassova rovnice [9].

$$y^2 = x^3 + ax + b, (x, y) \in \mathbb{R}^2 \tag{1.1}$$

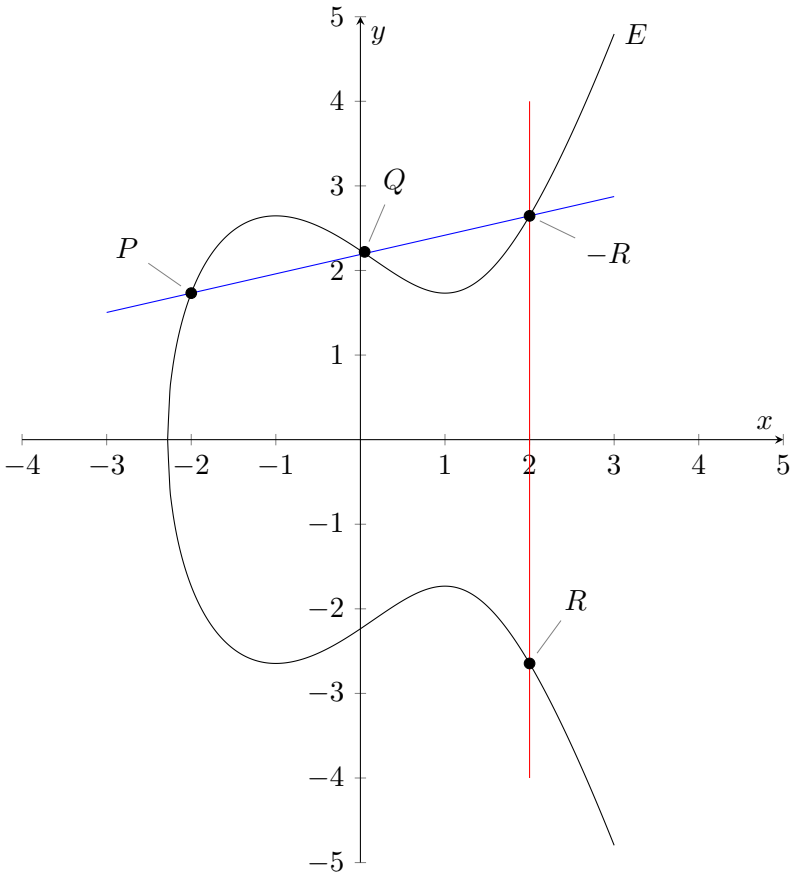
$$4a^3 + 27b^2 \neq 0 \tag{1.2}$$

Číslo 1.2 se nazývá diskriminantem [9] kubického polynomu $x^3 + ax + b$. Nulovost diskriminantu má za důsledek existenci průsečíků křivky se sebou samou a tzv. ostrých zlomů [9]. Tyto body znemožňují zavést základní operace, které budou v textu dále rozvedeny.

V rámci E ještě zavedeme pojem opačný bod. Ať $P = (p_1, p_2)$ a $P \in E$, pak opačný bod k P značíme $-P$ a platí $-P \in E$ a $-P = (p_1, -p_2)$. Platí tedy, že opačný bod $-P$ je k bodu P symetrický podle osy x v kartézské soustavě souřadnic v rovině.

Mezi body křivky je zavedena binární operaci $\oplus : E \times E \to E$. Tato operace může být definována geometricky. Mějme body $P = (p_1, p_2)$ a $Q = (q_1, q_2)$, $P \neq Q$, $p_1 \neq q_1$ a jistě $P, Q \in E$. Potom bod R, $R = P \oplus Q$ lze získat tak, že je sestrojena přímka t, která protne body P a Q a je nalezen třetí průsečík přímky t a křivky E. Nalezený průsečík odpovídá bodu $-R$. Operace je znázorněná na obrázku 1.1. Jestliže $P = Q$, pak přímka t je tečna křivky E v bodě P a jediný průnik s E bude opět $-R$. Jestliže $p_1 = q_1$ a $p_2 \neq q_2$ (jedná se o navzájem opačné body), zavedeme bod $\mathcal{O}$ tak, že $P + Q = \mathcal{O}$. Je to z toho důvodu, že v takovém případě přímka t neprotíná křivky v žádném dalším bodě. Bod $\mathcal{O}$ je tedy dodefinovaný a pro $P \in E$ platí $\mathcal{O} \oplus \mathcal{O} = \mathcal{O}$, $P \oplus \mathcal{O} = P$ a $\mathcal{O} \oplus P = P$ [9, 10].

Z toho vyplývá, že $E \cup \mathcal{O}$ s takto definovanou operací $\oplus$ tvoří komutativní grupu [9] a proto se pro operaci $\oplus$ používá aditivní notace.



Obrázek 1.1: Geometrická interpretace operace $P \oplus Q = R$

Problém diskrétního logaritmu na eliptické křivce

Mějme body $P, Q \in E$. Určení $n \in \mathbb{N}$, které splňuje rovnici 1.3 je problém diskrétního logaritmu na eliptické křivce, který je obecně těžko řešitelný, nicméně konkrétní složitost závisí na konkrétní křivce [9].

$$Q = \overbrace{P \oplus P \oplus \ldots \oplus P}^{\text{n-1}} = n \cdot P \tag{1.3}$$

Zároveň zaveďme operaci $\cdot : \mathbb{N} \times E \to E$, která je definována právě jako v rovnici 1.3. Tečku lze vynechat a zápis nP a $n \cdot P$ je tedy ekvivalentní.

Použití v asymetrické kryptografii

V praxi se problém diskrétního logaritmu na eliptické křivce využívá v asymetrické kryptografii.

Soukromý klíč $k_{private}$ je generován jako náhodné číslo [5] z množiny binárních řetězců s délkou n bitů, $n \in \mathbb{N}$, tj. $k_{private} \in \{0,1\}^n$ a je tajemstvím majitele.

Veřejný klíč K_{public} je vypočítaný ze soukromého klíče jako bod eliptické křivky E podle vztahu $K_{public} = k_{private} \cdot G$, kde G je bod eliptické křivky E nazývaný generátor [5].

Příklad

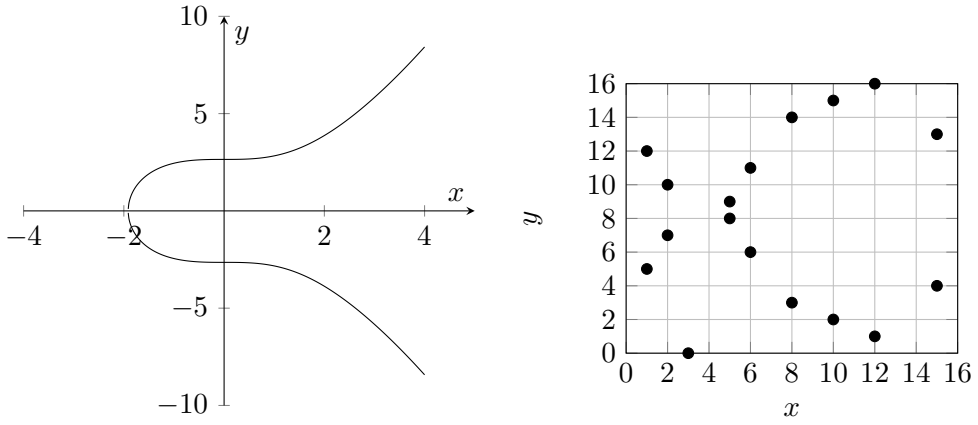
Konkrétní křivka použitá ve výše zmíněných blockchainech, tedy v Bitcoinu a Ethereu, se nazývá *secp256k1* a po dosazení doporučených parametrů [11] má v oboru reálných čísel tvar 1.4.

$$y^2 = x^3 + 7 \tag{1.4}$$

nicméně je třeba brát v úvahu, že ve skutečnosti je *secp256k1* nad tělesem $\mathbb{Z}_p$, kde dle doporučení [11] je p prvočíslo 1.5.

$$p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1 \tag{1.5}$$

Je zřejmé, že prvočíslo p je velmi velké, proto ve vizualizaci 1.2 je použito mnohem menší.



(a) Nad reálnými čísly

(b) Nad Z_{17}

Obrázek 1.2: Vizualizace křivky *secp256k1*

1.2.2 Digitální podpis

Nějaký blok dat lze podepsat (tj. zašifrovat) soukromým klíčem a ověřit (tj. rozšifrovat) klíčem veřejným. To zaručuje několik zajímavých vlastností:

autentizaci podpis se nedá napodobit nikým jiným, než podepisujícím

integritu podepsaný blok dat nemůže být pozměněn bez změny podpisu

nepopiratelnost podepisující nemá možnost popřít, že blok dat podepsal

ECDSA

Digitální podepisování nad eliptickými křivkami je prováděno pomocí algoritmu, který se nazývá ECDSA a je to hlavní [8, 4] využití asymetrické kryptografie v blockchainech. Jeho fungování je přiblíženo v algoritmu 1, respektive 2 [12]. Algoritmus ECDSA slouží jako alternativa ke známému podpisovému schématu DSA, který ale místo problému diskrétního logaritmu nad výše definovanou grupou funguje nad určitou celočíselnou grupou.

Algoritmus 1: Podepsání zprávy

Data: Eliptická křivka E, zpráva m, generátor G, prvočíslo p, soukromý klíč k
Result: Podepsaná zpráva

1 Vyber náhodné číslo b, $1 \leq b \leq p-1$ a ať existuje inverze b^{-1} v modulu p;
2 $b \cdot G = (x, y)$ a polož $r = x \ mod \ p$;
3 **if** $r == 0$ **then**
4 | Vrať se na řádek 1;
5 **end**
6 $e = \text{Hash}(m)$;
7 $s = b^{-1}(e + kr) \ mod \ p$;
8 **if** $s == 0$ **then**
9 | Vrať se na řádek 1;
10 **end**
11 **return** (r, s)

Takto představený algoritmus ECDSA funguje správně [12]. Ať Alice vygeneruje podle algoritmu 1 podpis (r, s). Potom $s = b^{-1}(e+kr) \ mod \ p$ a platí 1.6. Veškeré vztahy jsou přebírány z výše uvedených algoritmů a zůstává i značení.

$$b \equiv s^{-1}(e + kr) \equiv s^{-1}e + s^{-1}kr \equiv we + wkr \equiv u_1 + u_2k \ mod \ p \qquad (1.6)$$

Algoritmus 2: Ověření podpisu

Data: Eliptická křivka E, zpráva m, generátor G, prvočíslo p, veřejný klíč K, podpis (r, s)
Result: Platnost podpisu (r, s) na zprávě m

if $r \notin [1, p-1]$ *nebo* $s \notin [1, p-1]$ **then**
  return Neplatný
end
$e = \text{Hash(m)}$;
$w = s^{-1}\ mod\ p$;
$u_1 = e \cdot w\ mod\ p$;
$u_2 = r \cdot w\ mod\ p$;
$X = u_1 \cdot G + u_2 \cdot K$;
$X = (x, y)$;
if $X == \mathcal{O}$ **then**
  return Neplatný
end
$v = x\ mod\ p$;
if $v == r$ **then**
  return Platný
end
return Neplatný

Dále platí 1.7, ze kterého plyne, že je potřeba, aby $v = r$.

$$u_1 G + u_2 K = (u_1 + u_2 k)G = bG \tag{1.7}$$

Kapitola 2

Blockchain a jeho architektura

V této kapitole se zabýváme blockchainem a jeho jednotlivým částmi a přestože tento pojem není nijak standardizován, většina veřejných blockchainů odvozuje svoji strukturu od toho bitcoinového, jak jej představil Satoshi Nakamoto ve své práci *Bitcoin: A Peer-to-Peer Electronic Cash System* v roce 2009, přestože Nakamoto ani jednou pojem blockchain nepoužil [13]. V nejobecnějším smyslu je blockchain sdílená struktura dat sdružující informace o výměnách dat, která se dělí na jednotlivé bloky, které jsou stromově propojeny ukazateli z listů do kořene.

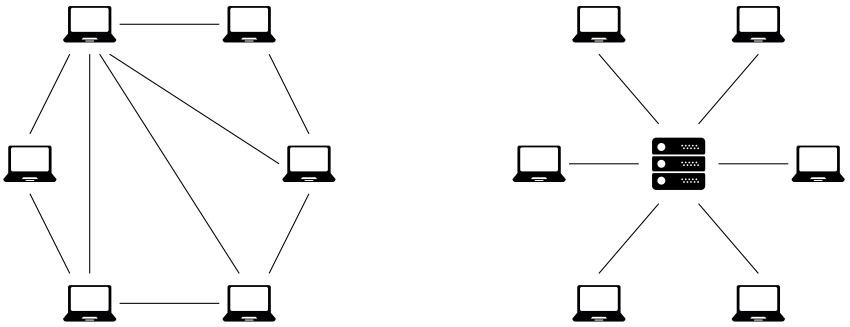
Na úvod je vhodné vymezit určité veřejně známé pojmy. Blockchain je souhrnné označení pro sdílenou datovou strukturu určitých vlastností, jež budou popsány v této kapitole. Kryptoměna je pojmenování pro specifický druh měny, kdy veškeré transakce touto měnou jsou zaznamenány do blockchainu, který bude v následující kapitole popsán. Pokud se v následujícím textu bude hovořit o některém z existujících blockchainů, vždy bude myšlena síť a technologie, nikoliv konkrétní obchodovatelná kryptoměna, pokud nebude řečeno jinak.

2.1 Peer-to-peer síť

V překladu název podkapitoly znamená *rovný s rovným*, což vcelku přesně vystihuje skutečnost, jakým způsobem jsou organizovány peer-to-peer (běžně známé jako P2P) sítě. P2P aplikace nebo protokoly jsou typy distribuované nestrukturizované[5] architektury, kde každý účastník je rovný s jiným účastníkem a sdílí vlastní prostředky (paměť, výkon, ...) [14]. Oproti tomu v klasické server-klient architektuře je server výhradním poskytovatelem prostředků a

[5] existují i strukturizované, ale ty pro nás nejsou příliš zajímavé

informací. Grafické porovnání těchto dvou základních topologií lze vidět na obrázku 2.1.



(a) P2P (b) Server-klient

Obrázek 2.1: Porovnání topologie

Mimo klasické blockchainy jsou P2P sítě užívány například ke sdílení souborů[6]. Mezi obrovské výhody těchto sítí patří neexistence centrální autority. Velké nevýhody a nebezpečí se skrývají právě v obrovské propojenosti jednotlivých účastníků, například:

DoS útoky neúčelné přetěžování skupinou nebo jedincem s velkým výpočetním výkonem. P2P jsou na tyto útoky ještě více náchylné než klasické server-klient architektury[7]

injektování záměna sdílených dat za neočekávané, včetně sdílení virů

neužiteční jedinci uzly v síti, které neposkytují prostředky

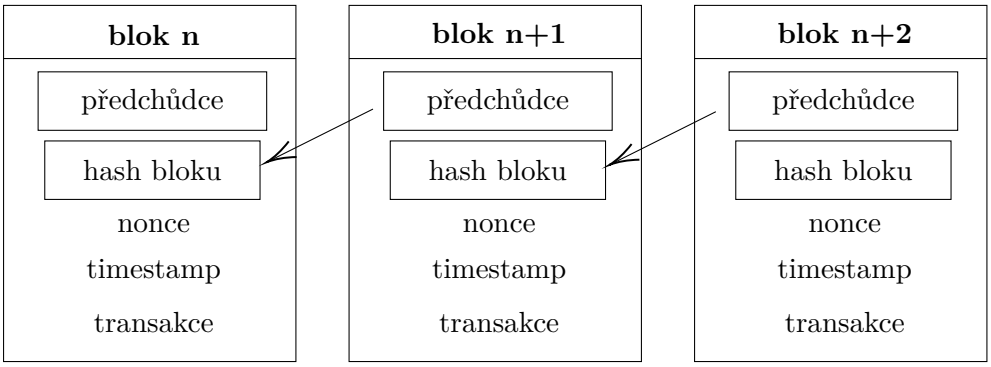
narušení soukromí účastníci jsou více propojeni s ostatními a mohou být nechtěně sdílena osobní data

2.2 Časové razítkování

Blockchain může být vnímán jako účetní kniha nebo neměnitelná, stále rostoucí databáze záznamů. Myšlenka schovaná za bloky a blockchainy je poměrně stará a její původ lze vystopovat už do roku 1991 [1], kdy byla představena metoda pro bezpečné časové orazítkování digitálních dokumentů [15].

[6] např. síť Bittorrent, síť Napster nebo síť Gnutella

[7] Server jakožto autorita může zablokovat komunikaci od pachatele útoku, účastník P2P sítě se spíše odpojí a tím se síť více rozpadá a zatěžuje



Obrázek 2.2: Zřetězení bloků spolu s minimálními daty přítomnými v blocích

Toto orazítkování, kromě důkazu existence dat v nějakém čase, určovalo přesnou posloupnost, v jakou byly dokumenty vytvářeny a tedy který dokument předcházel jaký. Toto schéma samozřejmě požadovalo, aby tento řetězec dat nemohl být měněn, čehož se přirozeně dosáhlo zakomponováním hashovací funkce. V tomto schématu byla však nějaká služba jakožto centrální autorita, která časová razítka poskytovala.

2.3 Architektura blockchainu

Základní stavební prvek blockchainu je blok, který je navázán do celého řetězu pomocí hashových ukazatelů. Blok je jednoznačně identifikovatelná datová struktura, která obsahuje povinně minimálně následující data:

Odkaz na předchozí blok je hash předchozího bloku a funguje jako ukazatel

Timestamp časové razítko, kdy byl blok vytvořen

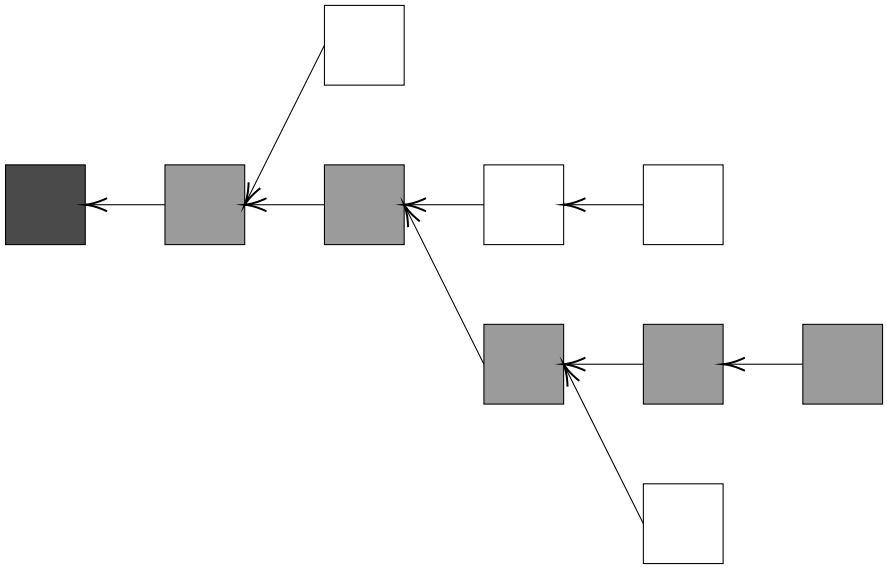
Transakce tradičně se v blockchainu o datech, která jsou součástí bloku, hovoří jako o transakcích

Nonce je proměnlivá číselná hodnota

Obvykle je do bloku navíc přidána jeho hash, která usnadňuje orientaci. Schéma, jakým jsou bloky vázány zobrazuje obrázek 2.2.

2.3.1 Tvorba blockchainu

Blockchain je kolektivně vytvářen všemi účastníky sítě, kteří tvoří bloky a napojují je do existujícího blockchainu. Tradičně se procesu vytváření bloků říká



Obrázek 2.3: Stromová podstata blockchainu, kde kořen je *genesis* blok a validní cesta je právě ta nejdelší; v obrázku vybarvena šedě

těžba[8]. Celý řetězec začíná počátečním blokem, který se nazývá *genesis*. Další bloky jsou napojeny na něj. Každý účastník může blok napojit na libovolné místo. Díky tomu celý blockchain je možné vidět jako orientovaný strom a počáteční blok jako jeho kořen, viz obrázek 2.3. Z vlastnosti stromu plyne, že z každého listu[9] existuje jedna cesta do kořene. Platí, že za platná jsou považována ta data, která se nacházejí na nejdelší cestě a účastníci jsou tedy motivováni napojovat bloky v takové cestě.

2.3.2 Řetězení bloků

Jako ochrana proti záměrnému zahlcování bloky a k vytvoření konsenzu na podobě struktury slouží mnohé koncepty. Nejužívanější je zřejmě algoritmus *proof of work* [13], který vyžaduje, aby hash bloku, který by měl být přidán, měl nějakou určitou formu danou protokolem blockchainu. Například v síti Bitcoin je vyžadováno, aby určitý počet prvních bitů měl hodnotu 0 [13]. Samozřejmě čím větší požadovaný počet, tím více práce dá takový blok najít, přesněji řečeno se pravděpodobnost nalezení validního bloku snižuje exponenciálně a pravděpodobně bude nalezen tím, kdo vlastní nejvíce výpočetních prostředků. A proto je potřeba, aby v bloku byla zcela proměnlivá část – celé číslo zvané nonce, které je měněno, dokud není nalezen validní blok. Jiné algo-

[8] Obyčejně je ten, kdo přidá blok do sítě odměněn v rámci sítě

[9] tedy bloku, na který není odkazováno

ritmy k hledání konsenzu budou v případě nutnosti představeny dále v práci, ale důvod jejich existence je vždy stejný.

Tento princip také řeší problém validnosti dat. Vždy se považují za validní ta data, které se nachází na nejdelší cestě od genesis bloku, protože bylo vyloženo nejvíce energie pro jejich umístění do blockchainu.

Takovýto princip hledání konsenzu mimochodem řeší problém v distribuovaných sítích, který se jmenuje Problém byzantských generálů[10]. Ve zkratce, jde o problém dohody dvou a více stran (generálů) jakým směrem pokračovat (jakým směrem má armáda zaútočit), jestliže je k dispozici pouze nespolehlivý a potenciálně kompromitovaný kanál [5].

2.4 Transakce

Již máme dobře definovanou datovou strukturu nazvanou blok, která obaluje jednotlivé transakce, proto je možné si přiblížit, co to transakce jsou. Jak již bylo řečeno, jsou to v podstatě data, jež jsou vyměňována v blockchainové síti jednotlivými účastníky. Tradičně se jedná o finanční data, tedy informace typu *Bob poslal Alici 5 jednotek peněz*, ale netřeba se omezovat pouze na to; může se jednat prakticky o cokoliv. Záleží na konkrétním blockchainu.

Transakce je vždy kryptograficky podepsaná instrukce [4] zkonstruovaná daným podepisovaným účastníkem. Podepisování se samozřejmě odehrává mimo síť a používá se k němu především algoritmus ECDSA.

2.5 Adresa a účet

Každý účastník v blockchainové síti musí být nějakým způsobem identifikován. K tomuto účelu slouží adresa, která je obvykle vygenerována z veřejného klíče tak, že je veřejný klíč zahashován. Tak je zaručeno, že adresa patří určitému člověku, jenž disponuje příslušným soukromým klíčem. Na adresu lze tedy nahlížet jako na mapování mezi nějakým identifikátorem a stavem daného účtu. Účtem nazýváme konkrétní stav, obvykle se jedná o finanční zůstatek nebo jiná nabytá data získaná transakcemi. Stav každého účtu je samozřejmě zpětně zjistitelný z jednotlivých transakcí z nejdelší cesty.

[10] angl. Byzantine Generals' Problem

2.6 Decentralizace

Základní principy jsou již jasné a zbývá je jen spojit. Každý účastník sítě má k dispozici celou historii blockchainu, tedy veškerá data o transakcích a blocích, zároveň každý účastník může participovat na hledání konsenzu. Všichni účastníci jsou si rovni a jediný zdroj pravdy jsou kryptografické a matematické principy. Důkaz existence transakce je dán přítomností na validním bloku v nejdelší cestě.

2.7 Bezpečnost

Přestože každý blockchain může být jiný, lze vystihnout několik bezpečnostních vlastností, které jsou společné [16]:

Neměnnost Jakmile je přidán blok do blockchainu, nelze jej odstranit ani měnit.

Transparentnost Vše v blockchainu je viditelné pro každého účastníka, neboť v síti jsou si všichni rovni.

Integrita Pro ukládaná data i samotné bloky jsou používány hashovací funkce, které teoreticky zaručují integritu daných dat.

Soukromí Klíče každého účastníka jsou jeho tajemstvím.

Dostupnost Neexistuje centrální účastník, pokud se kdokoliv odpojí, síť stále existuje.

Stejně tak z povahy věcí vyplývají určité problémy, které mohou vést k různým útokům.

Nejznámější je tzv. **51% attack**, kdy útočník či skupina útočníků shromáždí více než polovinu výpočetního výkonu v síti, čímž mohou kompromitovat celý blockchain tím, že mohou vytvářet bloky o kterých budou tvrdit, že jsou validní, nicméně díky jejich převaze budou mít nárok na pravdu, neboť dokáží vytvořit nejdelší cestu.

Další útok také vyplývá z povahy blockchainu a nazvývá se **spam attack**. Spočívá v tom, že jsou útočníkem či útočníky spamovány transakce, čímž se zahlcuje celá síť. V reálných blockchainech existují poplatky za využívání sítě, což slouží jako účinná ochrana.

Další zranitelnost vyplývá z povahy počítačových sítí a je to existence metadat, se kterými přistupuje každý účastník. Je to například IP adresa. Tato

vlastnost může být obejita pomocí anonymizačních nástrojů, jako je například TOR[11] [17]. Podobné prozrazení identity může být zaviněno i používáním stále stejné blockchainové adresy; jednoduché řešení je používání adresy právě jednou.

Z povahy P2P sítě ještě hrozí DoS nebo DDoS útok při nashromáždění dostatečného množství výpočetního výkonu v síti.

2.7.1 Dvojité utrácení

V blockchainech je nejzásadnější útok dvojitého utrácení. Ten je úspěšný právě tehdy, když útočník utratí prostředky, které má připsány, vícekrát, než jednou. Pokud se budeme snažit tento útok vztáhnout na obecný blockchain, jde vlastně o takový útok, kdy útočník tvrdí, že má ve vlastnictví nějaké informace, na které však již nemá nárok; vytváří takové informace považované za validní jak se mu hodí. Tento útok je založen na možnosti větvit blockchain (obrázek 2.3). Řekněme, že útočnice Mallory uplatnila prostředky a tato transakce je zanesena v bloku n. Ovšem Mallory tajně těžila validní bloky, které jsou navěšeny na blok $n-1$ a v bloku n', který je sourozenec s n uplatnila tytéž prostředky. Poté dokázala vygenerovat delší cestu, než byla ta s blokem n a bloky zveřejnila. Ostatní účastníci bloky na kratší cestě zahodí a transakce čekají na opětovné zanešení do nových bloků (pokud již nebyly zanešeny Mallory). Problém je, že prostředky má Mallory jen na jednu z transakcí a to na tu, která se nachází na jejím bloku. Tedy první transakce je zamítnuta a příjemce nedostane nic.

Tato zranitelnost vychází z povahy blockchainu a principu konsenzu, že správná cesta je ta nejdelší. Byla popsána již Nakamotem [13].

Úspěšnost útoku

Hashrate je množství hashů, které je systém schopen vygenerovat za 1 sekundu. Z povahy hashovací funkce lze předpokládat, že hash je náhodný řetězec a tedy lze předpokládat, že nalezení konkrétního hashe je nezávislý náhodný jev. RH je celková *hash rate* sítě, RH_m je *hashrate* útočnice Mallory a RH_q upřímného zbytku. Pro pravděpodobnosti nalezení bloku upřímnými účastníky platí $q = \frac{RH_q}{RH}$ a Mallory $m = \frac{RH_m}{RH}$ a tedy $q + m = 1$. Předpokládáme, že náročnost těžby a veškerý *hashrate* jsou konstantní. Celý problém si lze vymodelovat jako analogii k *Gambler's ruin problem* s tím, že hráči mají k dispozici nekonečné zdroje [13]. Mějme z definováno jako rozdíl mezi délkami cest mezi upřímnou

[11] Anonymizační software, který dokáže skýt adresu

cestou a Mallory cestou. Dle [13] platí, že pravděpodobnost m_z, že Mallory předběhne upřímnou cestu o z bloků odpovídá vztahu 2.1.

$$m_z = \begin{Bmatrix} 1 & \text{jestliže } q \leq m \\ (\frac{m}{q})^z & \text{jestliže } q > m \end{Bmatrix} \tag{2.1}$$

Důležitá je tedy otázka, jak dlouho bychom měli čekat[12], abychom blok n mohli považovat za validní s vysokou jistotou. Počet bloků za n se nazývá potvrzení. Tedy rozdíl délky cesty z listu blockchainového stromu do genesis bloku a délky cesty z bloku n do genesis se nazývá právě *rozdíl* počet potvrzení. Pokud RH_m a RH_q jsou konstantní, pak Mallorynin postup lze charakterizovat jako $\lambda = z\ \frac{m}{q}$. Pravděpodobnost, že útočník dokáže vygenerovat delší cestu a zvalidnit svoje data odpovídá pak [13] vztahu 2.2. Závislost pravděpodobnosti úspěšného útoku pro různé poměry výpočetních sil je na obrázku 2.4 a nutný počet potvrzení pro různá m, aby pravděpodobnost úspěšnosti byla velmi nízká v tabulce 2.1.

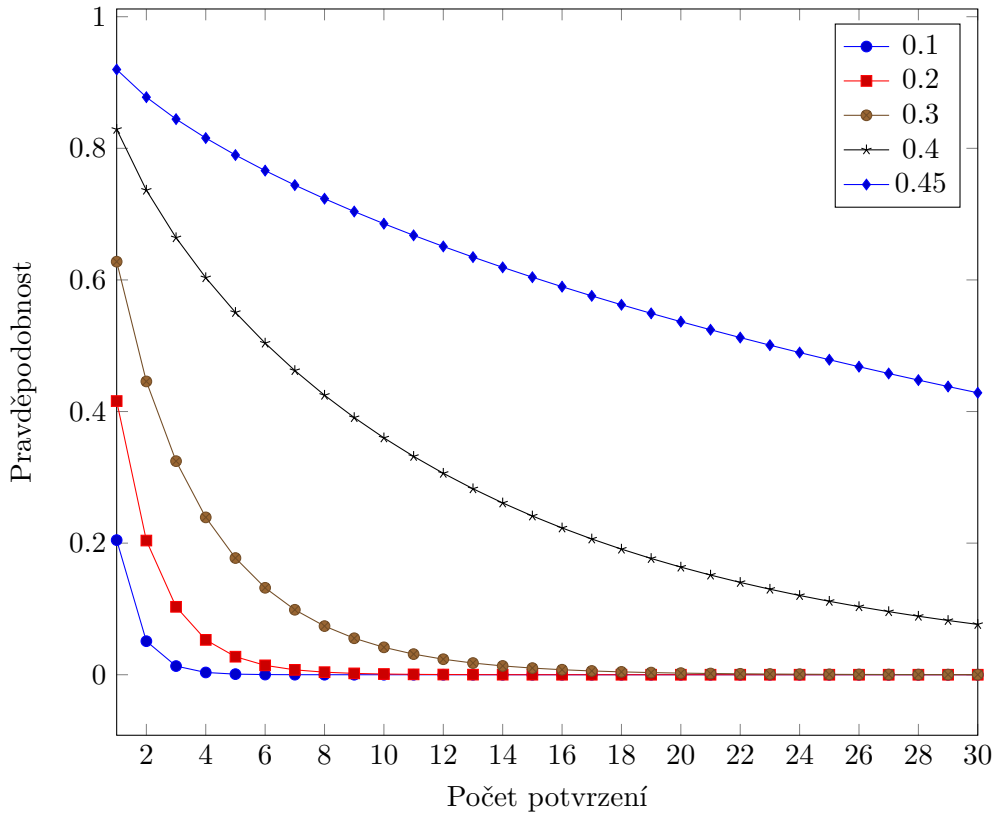
$$\sum_{k=0}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} \cdot \begin{Bmatrix} (\frac{m}{q})^{(z-k)} & \text{jestliže } k \leq z \\ 1 & \text{jestliže } k\ > z \end{Bmatrix} \tag{2.2}$$

Tabulka 2.1: Řešení m a počet potvrzení pro pravděpodobnost úspěšného útoku 0,01 %

m	počet potvrzení
0,10	5
0,15	8
0,20	11
0,30	24
0,40	89
0,45	340

Nikdy tedy nenastane situace, kdy bychom si mohli být zcela jisti, že námi považovaná cesta bude považována za validní napořád, ale můžeme si být takřka jistí, neboť pravděpodobnost úspěšnosti útoku bude mizivá.

[12]předpokládáme, že *hashrate* je konstantní



Obrázek 2.4: Závislost pravděpodobnosti úspěšného útoku na počtu potvrzení s danou Mallory silou

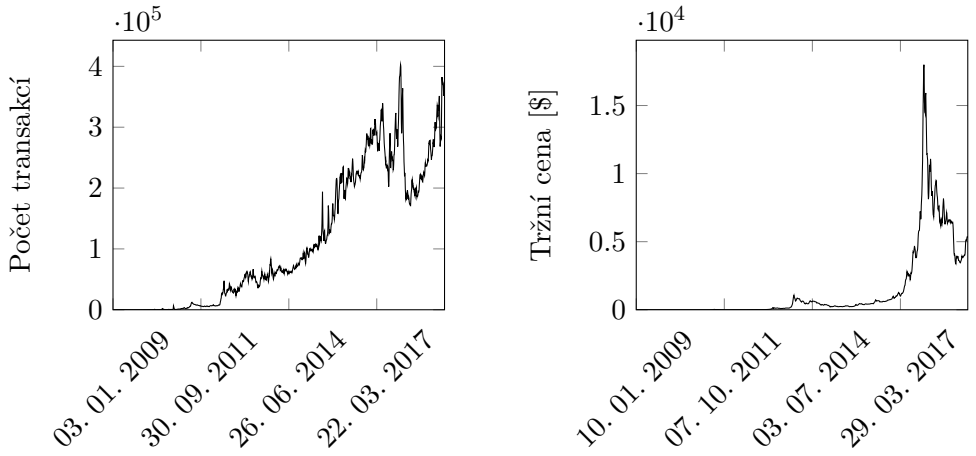
2.8 Současné využití technologie blockchain

Současný blockchain byl popsán poprvé v Nakamotově [13] práci, která dala vzniknout první blockchainové *kryptoměně*, Bitcoinu. Později se objevili další blockchainy, ať už více či méně inovativní. Využití blockchainu je široké a následující odstavce popíšou některé zajímavé z nich. Využití blockchainu jako výpočetní platformy je věnována celá následující kapitola o chytrých kontraktech.

2.8.1 Kryptoměny

Bylo již řečeno, že první veřejná a masová blockchainová síť založena Nakamotem v roce 2009 je Bitcoin [13]. Slouží přímo k účelu elektronického platebního systému a zároveň k vedení finančních účtů. V současné době má nejvíce uživatelů a největší tržní kapitalizaci [7] a mnoho principů, jako získávání kryptoměn a transakce jsou odvozeny právě ze sítě Bitcoin. Zajímavé

statistiky o používání jsou zobrazeny na grafech v obrázcích 2.5a a 2.5b.



(a) Počet potvrzených transakcí poskytuje dobrou představu o aktivitě [18]

(b) Průměrná tržní cena kryptoměny dává dobrou představu o zájmu [19]

Obrázek 2.5: Statistiky sítě Bitcoin po celou dobu její existence; hodnoty jsou týdenní průměry

Bitcoiny se těží již uvedeným způsobem, tedy tak, že účastník sítě (těžař) nalezne validní blok a ostatním tuto skutečnost sdělí. Ověření jeho pravdy je snadné. Jestliže většina ostatních účastníků uzná blok jako validní, potom je přidán a těžaři náleží odměna.

Každá transakce v této síti obsahuje nějaký počet vstupů, což jsou přesuny z adres a výstupy, což jsou přesuny na adresy. Každý, kdo chce uskutečnit transakci, tak může přidat poplatek[13], což má motivovat těžaře, aby tyto transakce zanesli na blok. Tyto poplatky pak, stejně jako odměna za nalezení bloku, náleží těžaři.

2.8.2 Append-only databáze

Výborné použití má blockchain v případech, kdy je potřeba append-only databáze s historií všech změn. Tyto blockchainy jsou často privátní a centralizované a neanonymní. Takové řešení může být prodáváno i jako služba, podobně, jako například webové úložitě [20].

Příkladem může být katastr pozemků, kde blockchain může nahradit komplikované relační databáze a navíc má potenciál proces převodu pozemku urychlit [21]. Navíc převod může být sledován všemi zainteresovanými stranami (státem, bankou), což by mohlo ušetřit byrokratickou zátěž [21].

[13] angl. *fee*

2.8.3 Důkaz pravosti

Díky časovým razítkům a neměnnosti blockchainu může být tato technologie využívána i pro důkaz existence, pravosti nebo vlastnictví, podobně, jako výše zmíněné časové razítkování. Komerční řešení nabízí například firma Kodak [22]. Ve zkratce to funguje na podobném principu jako časové razítkování zmíněné výše – tedy originální dílo patří tomu, kdo jej do blockchainu zanesl jako první.

2.8.4 Volby

S použitím vhodného protokolu může být blockchain využit také k volbám a dalším rozhodovacím procesům v elektronické demokracii. Elektronická demokracie[14] je forma vlády, kdy je předpokládáno, že voliči se budou více podílet na chodu společnosti pomocí nových technologií, jako je například internet [23, 24]. První velký subjekt, který využil možností blockchainu jako volební platformy byla dánská politická strana *Liberal Alliance* v roce 2015 k vlastním interním volbám [25]. V současnosti existuje několik aplikací, které blockchain aktivně využívají. Mezi nejvýznamnější [26] patří otevřený software *Follow My Vote* [27].

V této práci bude představen vlastní model volebního systému, který kromě technologie blockchain využívá i koncept chytrých kontraktů, který bude detailněji popsán v další kapitole.

[14] angl. e-democracy

Kapitola 3

Chytré kontrakty

Blockchainová síť nabízí kromě zajímavých vlastností uvedených v předchozí kapitole navíc sílu distribuované počítačové sítě. Jedním z konceptů, které tuto sílu využívají, jsou chytré kontrakty[15]. Tato kapitola se zabývá chytrými kontrakty nejprve obecně jakožto konceptem a následně i konkrétní implementací v síti Ethereum. Zmíněn bude i jazyk, určený k formálnímu zápisu chytrých kontraktů, *Solidity*. V poslední části budou ve zkratce představeny další blockchainy s implementovanou možností provádění chytrých kontraktů.

Poznámka k terminologii: napříč prací bude používán termín *chytrý kontrakt* jako doslovný překlad z anglického originálu *smart contract* a zároveň termín *kontrakt*. Oba termíny jsou zde považovány za rovnocenné.

3.1 Počátky

První články věnující se formalizování dohod do počítačového kódu byly napsány Nickem Szabem [28, 29], který se zabývá využitím moderních technologií v dohodách a smlouvách mezi lidmi; formální jazyk počítačového kódu je transparentní a předvídatelný, na rozdíl od přirozeného jazyka, ve kterém jsou tyto listiny obvykle psány. Szabo navrhuje databázi vlastníků a jejich majetků[16], která by měla být veřejná a měla by mít následující vlastnosti:

- Vlastník může svůj majetek prodat právě jednomu příjemci
- S majetkem může disponovat pouze vlastník, tj. žádná transakce se neobejde bez souhlasu vlastníka

[15] angl. smart contracts

[16] Majetek v tomto případě je něco, k čemu má vlastník vlastnická práva a co může být předmětem prodeje, výměny, darování a podobně

- Nikdo nemůže majiteli upřít vlastnická práva z jakéhokoliv důvodu

V [28] je chytrý kontrakt považován za technologického následníka obyčejného prodejního automatu, který de-facto funguje podobně; zákazník, tedy jedna strana by chtěla koupit limonádu v klasickém prodejním automatu, který je vlastně pouze prostředkem, jak onu limonádu zákazníkovi prodat. Automat si lze představit jako onen formální počítačový program. Je jednoduchý – uvnitř je nějaký konečný automat, který přijímá mince a v případě, že se dostane do stavu, kdy zákazník vloží dostatečný obnos, je mu ona limonáda vydána. V ideálním případě chrání tento prodejní automat jak prodejce, jehož chrání od zlodějů, tak zákazníka, který nebude ošizen nebo nějak diskriminován.

Právě tyto výše uvedené vlastnosti mohou být přirozeně uspokojeny právě takovou transparentní a distribuovanou databází, jako je blockchain, který jde ještě dál, neboť ten neobsahuje žádnou autoritu a v případě potenciálního vykonávání těchto chytrých kontraktů není potřeba svěřovat důvěru třetí straně; ale pouze technologii.

3.2 Existující implementace a využití

V této podkapitole budou představeny existující implementace výše uvedené ideje a technologie, které ji využívají.

3.2.1 Jazyk pro zápis kontraktů

Formální návrh jazyka *Formal Language for Analyzing Contracts* pro zápis chytrých kontraktů představil Szabo v roce 2002 [30]. Účel tohoto jazyka bylo naplnit podstatu výše uvedených nápadů pro chytré kontrakty. Tento formální návrh nenašel cestu do běžného používání a nebyl myšlen jako jazyk interpretovatelný počítačem, nicméně může sloužit jako inspirace pro další jazyky a proto je jeho zmínka v této podkapitole pro úplnost potřeba. Příklad lze nalézt ve výpisu kódu 1.

3.2.2 Ethereum

Úspěchu dosáhla později síť Ethereum. Ethereum je open source projekt postavený na blockchainu, který poskytuje distribuovanou síť s možností tvorby chytrých kontraktů. Častý omyl je, že Ethereum je měna; není to pravda, token generovaný těžbou bloků se nazývá *ether*. Provádění chytrých kontraktů se odehrává v turingovsky úplném virtuálním stroji EVM (který bude blíže popsán v následujícím odstavci), který funguje v odděleném běhovém prostředí

```
future(rightA="1 kg of pork bellies",
rightB="$10",
p = "for delivery in July 2002") =

when withinPeriod(p)
to Holder rightA   with   to Counterparty rightB
then terminate
```

Výpis kódu 1: Kontrakt v navrhovaném Szabově jazyce [30]. Výměna 10 dolarů za 1 kilogram vepřového, jestliže bude dodáno během července 2002

a nemá tak přístup k procesům nebo datům v operačním systému, na kterém je spuštěn. Hlavní idea za vznikem Etherea je tvorba univerzálního distribuovaného výpočetního stroje, který dokáže provozovat decentralizované aplikace [4], což je považováno za ekvivalentní pojem s chytrými kontrakty.

Ethereum virtual machine

Chytré kontrakty v síti Ethereum provádí tedy virtuální stroj. EVM je jednoduchý zásobníkový stroj s délkou slova 256 bitů, což odpovídá délce hashe funkce Keccak-256. Dále má EVM k dispozici nezávislé úložitě typu klíč-hodnota [31].

Kontrakty jsou zapisovány nízkoúrovňovým bytekódovým jazykem nazývaným *EVM code* [31]. Program v tomto jazyce je do blockchainu nasazen transakcí kódu na speciální adresu $0x0$. Každý kontrakt je pak charakterizován svoji adresou která je touto transakcí vygenerována, kterou pak může být kontrakt kýmkoliv volán. K této adrese, na které existuje kontrakt, neexistuje soukromý klíč a takovou adresu nikdo nevlastní a ani tvůrce k němu nemá speciální práva na úrovni protokolu; nicméně v kódu kontraktu lze pochopitelně speciální práva přiznat [31].

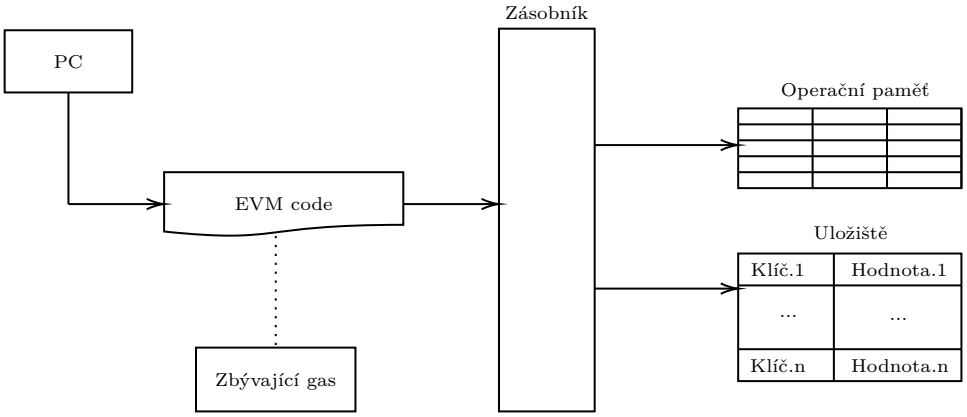
Chytré kontrakty lze volat pouze v transakci, nikdy se nespustí samy, ani neběží na pozadí a nelze provádět jakékoliv paralelní výpočty; EVM je jednovláknový stroj [31].

Architektura stroje není zcela dle von Neumanna, neboť kód a data jsou přísně odděleny. Není zde přítomen koncept registrů. Schéma stroje je na obrázku 3.1. K dispozici je

zásobník přes který jsou vykonávány veškeré příkazy s maximální hloubkou 1024 slov o 256 bitech,

(operační) paměť adresovaná po bytech, lineární, formálně nekonečná (omezena pouze fyzickými možnostmi), platná pro dobu výpočtu,

uložiště je persistentní, platné pro celou síť, je typu klíč-hodnota – mapuje 256 bitů na 256 bitů.



Obrázek 3.1: Zjednodušené schéma výpočetního modelu EVM, kde je jasně znázorněna zásobníková podstata virtuálního stroje. Kromě zásobníku je přítomna operační paměť, která je lineární, neomezená a adresovaná po bytech, je platná pro danou instanci a defaultně vynulována. Úložiště je typu klíč-hodnota a je persistentní a platné pro celý EVM, tedy blockchain

Gas

Jelikož EVM je turingovky úplný, mohou být prováděny nekonečné programy, které navíc nelze algoritmicky odhalit, což je skutečnost, která se obecně označuje jako problém zastavení. V Ethereu a jiných blockchainech jsou dvě možnosti potlačení tohoto problému. Buď časové omezení běhu, nebo zpoplatnění v rámci sítě. V Ethereu se používá druhá možnost – vykonávání kódu je potřeba zaplatit. Cena je dána veličinou *gas* [4], jejíž jednotkou je právě ether, generovaný těžbou.

Solidity

Protože zápis v bytekódu je vždy zdlouhavý, v obvyklých situacích preferují programátoři zápis v nějakém vyšším jazyku, který se kompiluje do bytekódu; ať už se jedná o bytekód nebo přímo nativní kód procesoru. Jazyků, pro které existují kompilátory do EVM code, je hned několik [31] s různou syntaxí. Nejvíce se rozšířil jazyk Solidity, který je de-facto standardem [31]. Ukázka

jednoduchého programu je ve výpisu kódu 2 a transakce k jeho zavolání je k nahlédnutí v čitelném formátu ve výpisu kódu 3.

Solidity je *contract-oriented* jazyk s C-like syntaxí podobnou JavaScriptu. Je staticky typovaný, podporuje dědičnost, kód lze rozšiřovat knihovnami a programátor může definovat vlastní typy[17] [32].

```
pragma solidity ^0.4.0;

contract SimpleStorage {
        uint storedData;

        function set(uint x) {
                storedData = x;
        }

        function get() constant returns (uint) {
                return storedData;
        }
}
```

Výpis kódu 2: Ukázka jednoduchého chytrého kontraktu, který je zapsaný v jazyce Solidity; jde o jednoduché úložitě, kde každý může změnit hodnotu typu `uint` nebo ji přečíst; klíčové slovo `pragma` říká kompilátoru, jakou verzi jazyka použít

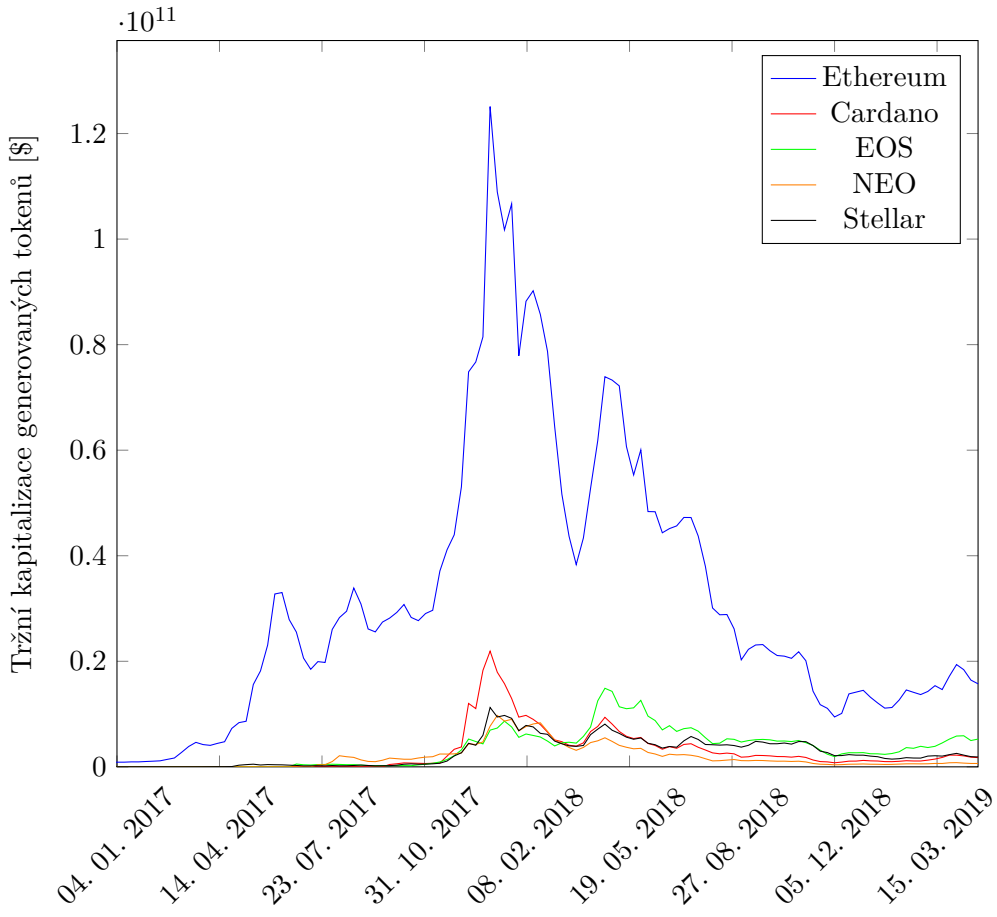
```
{
 "from": "0xca35b7d915458ef540ade6068dfe2f44e8fa733c",
 "to": "0x692a70d2e424a56d2c6c27aa97d1a86395877b3a",
 "gas": "3000000",
 "data": "0x60fe47b1000000000000000000000000000000000000000000000000000000000000002a"
}
```

Výpis kódu 3: Ukázka transakce pro zavolání metody `set` s argumentem `42` z kódu 2; `from` je adresa volajícího, `to` je adresa kontraktu, `gas` je hodnota, kterou je volající ochoten maximálně zaplatit za kontrakt (nevyužitý zbytek se mu vrací [4]), `data` je řetězec se zakódovanými parametry

[17] prakticky se jedná o struktury známé z jazyka C

3.2.3 Další platformy

Přestože hegemonem mezi veřejnými blockchainy umožňující chytré kontrakty je Ethereum (viz obrázek 3.2), samozřejmě existují další. Jejich stručný přehled je k dispozici v této podkapitole.



Obrázek 3.2: Tržní kapitalizace je dobrý ukazatel oblíbenosti daného blockchainu

Cardano

První koncept sítě Cardano [33] pochází od jednoho ze spoluzakladatelů Ethereа. Cardano bylo spuštěno v roce 2017. Obchodovatelný token v rámci sítě se nazývá *Cardano coin*. Síť je podobně jako Ethereum založena jako výpočetní platforma. Na rozdíl od všech předchozích blockchainů nepoužívá k hledání konsenzu algoritmus proof of work, ale proof of stake.

Na rozdíl od (již) klasického proof of work algoritmu, objevitel bloku není ten, kdo nejrychleji vyřeší početní úlohu, ale je vybrán na základě náhody a bohatství nebo stáří; lze říci, že preferováni jsou účastníci s delší historií účasti.

EOS

Na síti EOS [34] je zajímavé použití delegovaného proof of stake algoritmu, který k dosažení konsenzu používá systém reputací jednotlivých účastníků spolu s hlasováním. Zvolen je delegát, který zařazuje další bloky, ale nemůže měnit transakce a musí složit finanční depozit.

3.2.4 Shrnutí

Principiálně jsou představené sítě velmi podobné a zásadní rozdíl je v použití algoritmů k dosažení konsenzu. Tyto blockchainy, které umožňují provádění kódu velmi často nepoužívají algoritmus proof of work, protože tento algoritmus konverguje k vyšší složitosti těžení a tím pádem se omezuje škálovatelnost a použití sítě je těžkopádné pro hojné používání decentralizovaných aplikací. Ethereum v současné době používá proof of work, nicméně v budoucnosti dojde k přechodu na proof of stake [4].

Kapitola 4

Modelové využití ve volbách do akademického senátu FIT

Jako modelovému využití technologie chytrých kontraktů na platformě blockchainu se tato práce věnuje implementaci elektronických voleb do Akademického senátu Fakulty informačních technologií při ČVUT v Praze. Tato kapitola nejdříve zanalyzuje současný stav a přihlédne k existujícím zákonům a předpisům. Budou vystiženy silné a slabé stránky existujícího řešení, stejně tak navrženého. Důvody, proč byly zvoleny právě volby zčásti vyplývají z nedostatků současného řešení (viz dále). Dále je nutné vzít v úvahu, že elektronické volby řešeny centrálně vždy vyžadují důvěru v autoritu – ale právě ta autorita, která volby vyhlašuje je velmi často úzce spjata s institucí, do které se volí.

4.1 Akademický senát FIT

Akademický senát FIT je nejvyšší samosprávný zastupitelský orgán akademické obce naší fakulty. Jeho členy volí ze svých řad členové akademické obce. Má celkem deset členů, šest z nich jsou akademičtí pracovníci a čtyři studenti. Senát schvaluje důležité fakultní dokumenty, rozpočet fakulty a usnáší se o návrhu na jmenování děkana. V čele fakultního senátu stojí předseda a dále předsedající a tajemník [35]. Jeho existence vyplývá z vysokoškolského zákona a je povinná pro každou fakultu veřejné vysoké školy [36].

4.1.1 Volby do Akademického senátu FIT

Voleb se mohou účastnit členové akademické obce FIT ČVUT s aktivním i pasivním právem, nicméně funkce senátora je neslučitelná s některými funkcemi,

například funkcí rektora, děkana, jejich zástupců a podobně [37].

Volby do AS jsou tajné a přímé. Konají se ve dvou obvodech a to v obvodu akademických pracovníků a studentů. Každý příslušník svého obvodu volí ve svém obvodu. Příslušník obou obvodů se rozhodne, ve kterým chce kandidovat nebo uplatnit svůj hlas. Funkční období je tříleté a nové volby vyhlašuje AS s dostatečným předstihem před koncem funkčního období; jestli se tak nestane, po jeho konci volby vyhlašuje děkan fakulty, který ustanoví i volební komisi [37].

4.1.2 Organizace voleb

Kandidáta může navrhnout jakýkoliv člen obce, nicméně navržená osoba musí s kandidaturou souhlasit. Návrh musí obsahovat

- jméno a příjmení kandidáta,
- charakteristiku kandidáta,
- fakultní e-mail,
- fotografii,
- souhlas s kandidaturou a podpis,

nicméně může obsahovat další informace, jako například program. Tento návrh se odevzdává ve fyzické podobě obvykle na sekretariát děkana. K volbám je k dispozci seznam oprávněných volitelů [37].

Volby probíhají elektronicky, kdy se volič autentizuje v systému, zaškrtne možnosti volby a svoji volbu odešle. Software vyhodnotí výsledky v každém obvodu zvlášť. Informace o volebních výsledcích jednotlivých kandidátů má k dispozici ovšem pouze volební komise. Software také provede anonymizaci, aby nebylo jasné, kdo jak hlasoval, ale zároveň, aby bylo zaručeno, že každý hlasuje pouze jedenkrát. Datum a čas voleb určuje volební komise [37].

4.1.3 Slabiny současné implementace

Z hlediska předpisů zřejmě volební aplikace netrpí žádným nedostatkem, což je zaručeno díky tomu, že předpisy jsou psány na míru současnému řešení. Nicméně již z podstaty návrhu centralizovaného řešení vyplývají 2 zásadní vlastnostní problémy, což je:

tajnost je potřeba zcela odevzdat důvěru ve kvalitu anonymizace uzavřenému systému

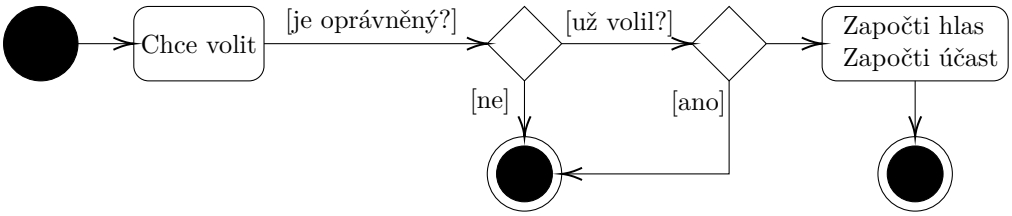
transparentnost stejně jako v předchozím bodě je nutné věřit, že systém započítává hlasy korektně, či dokonce, že hlasy nefalšuje

4.2 Volební systém na blockchainu

Oba problémy nastíněné v předchozím odstavci řeší elegantně decentralizovaná aplikace postavená na blockchainu. V této kapitole bude návrh, jak takovouto aplikaci implementovat a v následující kapitole bude představen *proof of concept* takového řešení.

Je potřeba, aby systém byl důvěryhodnější než ten současný, ale zároveň by neměl trpět bezpečnostními slabinami nebo poskytnout horší služby. Měl by mít následující vlastnosti:

- zaručení tajnosti volby tak, že reálná identita hlasujícího bude od počátku volebnímu systému neznámá
- neexistuje centrální autorita, která je jediným zdrojem pravdy (tedy hlasy jsou sčítány bez autority)
- lze volit více možností v souladu s volením řádem
- přesto, že by volby v systému měly být spouštěny pouze volební komisí, tak by měly být kontrolovatelné kterýmkoliv členem akademické obce FIT ČVUT



Obrázek 4.1: Diagram volebního systému

Samotný proces voleb, vizualizovaný na obrázku 4.1 se prakticky neliší od konvenčního elektronického systému, který je v současnosti provozován. S rezervou takovýto diagram dokonce odpovídá klasickému, neelektronickému systému. Hlavní rozdíl oproti oběma způsobům je však v míře anonymizace; v navrhovaném modelu je požadované, aby byl volič od počátku volby anonymní. Je tak třeba navrhnout systém, který nezná identity potenciálních voličů, ale zároveň je schopný říct, zda tento volit může a zda již nevolil.

Zároveň by měl být natolik transparentní, aby mohl být průběh kontrolován nejen autoritou[18], ale zároveň každým členem akademické obce, který má zájem kontrolovat hladký a spravedlivý průběh voleb.

4.2.1 Možnosti ověření a anonymizace voliče

V námi uvažovaném systému jsou potřeba u každého voliče ověřit 2 věci a to

- zda-li je oprávněný k volbě,
- zda-li již nevolil.

Obě tyto věci je ideální ověřovat před samotnou volbou bez toho, aby volební systém samotný znal identitu voliče. Proto se nelze vyhnout registrační fázi, kdy se volič bude muset sám zaregistrovat. Způsobů, jakým dosáhnout anonymizace registrovaných voličů je více.

Open Vote Network protokol

Tento protokol je dvoukolový a přirozeně decentralizovaný [38]. Jeho nevýhoda je, že jeho základní verze nepodporuje jiný tip otázek, než binární. To nelze však považovat za významný problém, neboť výběr z n kandidátů lze transformovat na n pro/proti otázek.

Před samotnou volbou se zvolí konečná cyklická multiplikativní grupa G s prvočíselným modulem q generátorem g. Každý účastník si náhodně zvolí tajemství $x_i \in \mathbb{Z}_q^\times$. Předpokládejme binární otázku. Poté každý účastník zveřejní g^{x_i}. Nechť účatník má přiděleno i, $i \in [1, n]$, kde n je počet účastníků a x_i náleží i-tému účastníkovi a každý spočítá g^{y_i} dle 4.1.

$$g^{y_i} = \frac{\prod_{j=1}^{i-1} g^{x_j}}{\prod_{j=i+1}^{n} g^{x_j}} \tag{4.1}$$

a následně zveřejní $g^{x_i y_i} g^{v_i}$, kde $v_i \in \{0, 1\}$. Hlasy kódované jako 1 se následně spočítají pomocí vztahu $\prod_{k=1}^{n} g^{x_k y_k} g^{v_k} = g^{\sum_{k=1}^{n} v_k}$ [38]. Zjistit sumu hlasů je samozřejmě problém diskrétního logaritmu, nicméně exponent není obvykle velký a vyřešit jej tedy není výpočetně náročné.

Významný problém je, že k úspěšné volbě je třeba, aby všichni, kteří zveřejní své g^{x_i} odevzdali svůj hlas. To nemusí být problém při hlasování v menším počtu lidí, tedy například během zasedání nějaké komise nebo Senátu. Tento protokol dokonce poskytne informaci, kdo nevolil a celou volbu zkazil.

[18] v tomto případě volební komisí

Nicméně při tak poměrně objemném počtu lidí, jako je počet členů Akademické obce FIT, je poměrně vysoká šance, že registrovaný volič neodvolí, což zkazí celou volbu.

Protokol s jednorázovým prstencovým podpisem

Mnohem větší flexibilitu dokáže poskytnout takový protokol, kde svoji volbu volič podepíše svým jednorázovým klíčem vygenerovaným ze svého veřejného, ale tak, aby jednorázový nebyl spojitelný s veřejným (tedy vlastní identitou voliče). Takový protokol vychází z prstencového podpisu a byl představen Nicolasem von Saberhagenem v roce 2013 jako protokol pro anonymní transakce v bitcoinové síti [39].

Protokol umožňuje vytvořit jednorázový podpis, který je ale ověřován množinou veřejných klíčů. V konvenčním podpisovém schématu je podpis ověřován pouze jedním veřejným klíčem, který samozřejmě koresponduje s identitou podepisovaného. Protokol s jednorázovým prstencovým podpisem umožňuje podepisovanému skrýt vlastní identitu, ale zároveň ostatní účastníky ubezpečit, že podepisovaný je jeden z nich [39]; tedy je ověřen, v tomto případě, k volbě. Vizualizace takového podpisového schématu je na obrázku 4.2. Tento protokol je narozdíl od Open Vote Network protokolu uvedeného výše o něco složitější a tedy náchylnější na implementační chyby. Vyžaduje také od voličů poněkud větší zapojení.

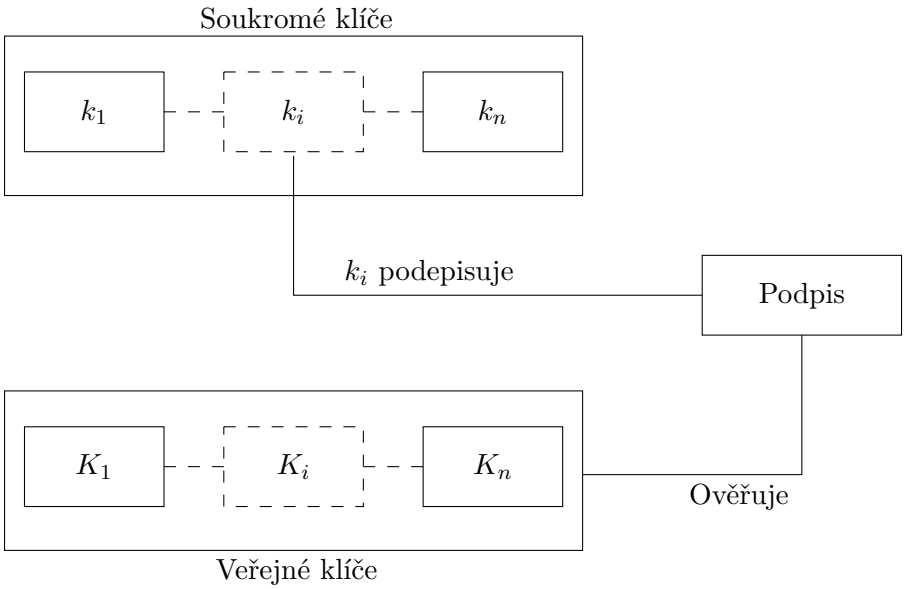
Pracuje nad eliptickou křivkou E s generátorem G, hashovací funkcí H. Součástí schématu je i množina n účastníků, které si oindexujme indexem i, $i \in [1, n]$. Skládá se ze 4 různých algoritmů [39]. Zde je uveden jejich hrubý popis:

GEN všichni si zvolí soukromý klíč k_i, vypočte veřejný klíč $K_i = k_i G$ a obraz veřejného klíče $I_k = k_i H(K_i)$; K_i a I_i se zveřejní

SIG podepisující s indexem s, $s \in [1, n]$, vezme zprávu m, množinu veřejných klíčů S', $S' = \{K_i\}_{i \neq s}$, svoji dvojici klíčů (k_s, K_s) a na výstup položí vypočtený podpis zprávy σ a množinu S, $S = S' \cup K_s$

VER vezme zprávu m, podpis σ, množinu S a na výstupu vrátí, jestli je podpis validní nebo ne

LNK vezme množinu ζ, $\zeta = \{I_i\}$, podpis σ a vrátí, jestli nějaká dvojice klíčů (k_i, K_i) již vygenerovala podpis



Obrázek 4.2: Diagram prstencového podpisu [39]

Protokol se slepým podpisem

Poslední protokol, stejně jako předchozí, umožňuje odpojení volby od reálné identity voliče, nicméně stále umožňuje připustit k volbě pouze ty oprávněné. V tomto protokolu [40] vystupují 3 role a sice

administrátor ověřuje voliče a slepě podepisuje jejich hlasy,

volič volí,

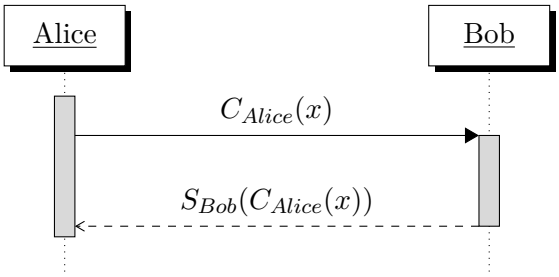
inspektor dohlíží na volby.

Slepý podpis [40] je další podpisové schéma, které umožňuje podepisovanému zaslat zprávu k podepsání nějaké autoritě bez toho, aby autorita tušila, co zpráva obsahuje. Ať Alice chce od Boba podepsat zprávu.

Bob exkluzivně vlastní podepisovací funkci S_{Bob}, ke které existuje veřejně známá inverze S_{Bob}^{-1}, platí tedy $S_{Bob}^{-1}(S_{Bob}(x)) = x$. Alice vlastní svoji dvojici funkcí, které jsou oboje tajné. Pro tyto funkce platí vztah 4.2 a x je podepisovaný blok dat, obvykle hash. Tyto funkce budou specifikovány dále v kapitole 5.

$$C_{Alice}^{-1}(S_{Bob}(C_{Alice}(x))) = S_{Bob}(x) \tag{4.2}$$

Aplikace funkcí na x nikterak nedává informaci o x. Podepisovací schéma je znázorněno na diagramu 4.3



Obrázek 4.3: Diagram slepého podpisu. Na přijatou zprávu od Boba je uplatněn vztah 4.2 a tedy Alice získá Bobův podpis bloku dat x, tedy $S_{Bob}(x)$

Volič se před samotnou volbou zaregistruje u administrátora a je zařazen do množiny voličů. Z kryptografického hlediska není potřeba seznam ukrývat[19]. Poté volič vytvoří hlas, což bude určitý binární řetězec, který nechá slepě podepsat a z nové blockchainové adresy odešle hlas v plaintextu spolu s podepsaným hlasem. Tak bude zaručena anonymita a zároveň platnost hlasu.

4.2.2 Návrh volebního systému

V předchozí kapitole byly popsány tři protokoly, které je možné použít pro zaručení kvality volebního systému. Open Vote Network je nejjednodušší jak z matematického, tak z implementačního hlediska a je nápadně podobný Diffe-Helmannově výměně klíčů. Jeho vlastnost však je, že musí odvolit každý registrovaný volič, což může být mimořádně vhodné pro menší povinné volby, například v nějaké komisi. O to více, že lze rozpoznat, který účastník nevolil. Určité řešení by bylo zavést registrační depozit a tedy finančně motivovat voliče, aby nezahodili svoji registraci, ale toto řešení je mimořádně těžkopádné.

Další je protokol, který užívá jednorázový prstencový podpis. Protokol je to robustní, nicméně je velmi citlivý na konkrétní implementaci, konkrétně na kvalitu hashovacích funkcí a kvalitu generátoru pseudonáhodných čísel.

Poslední představený protokol, který využívá slepý podpis, je implementačně relativně jednoduchý a zaručuje stejně kvalitní utajení jako předchozí protokly. Proto bude použit jako základ pro návrh nového systému. Ale ve své základní verzi neskrývá během voleb aktuální výsledky. Pro účely voleb do AS FIT bude třeba protokol mírně modifikovat.

[19] Právní hledisko bude rozvedeno v další kapitole.

4.2.3 Popis navrhovaného protokolu

V této podkapitole bude blíže představen a mírně modifikován protokol z podkapitoly 4.2.1. Pro lepší čitelnost definujme tři osoby a sice

Alice volič

Bob administrátor

Walder inspektor

Průběh bude rozdělen do 3 fází.

Registrace

Alice se chce zaregistrovat k volbám. Odešle tedy žádost spolu se svým veřejným klíčem PK_A k autorizační autoritě, což bude typicky Bob. Ten ověří její způsobilost k volbě a přidá její klíč na seznam oprávněných klíčů. Tento seznam bude po ukončení registrační fáze zveřejněn. Právní stránka tohoto kroku bude zanalyzována v následující kapitole.

Volba

Nejdříve musí Alice připravit svůj hlas, což bude v zásadě binární řetězec, který bude kódovat vlastní volbu.

$$hlas \in \overbrace{\underbrace{\{0,1\}^x}_{\text{volba}} \cdot \underbrace{\{0\}^y}_{\text{zero-string}} \cdot \underbrace{\{0,1\}^z}_{\text{sůl}}}^{\text{n}}$$

Platí evidentní vztah, že $n = x+y+z$. První část řetězce je samotné zakódování volby, které může být provedeno tak, že každý kandidát bude mít určený bit, který bude buď 1 nebo 0. Druhá část je povinný počet 0 pro oddělení a indikaci dobře sestaveného hlasu. A konečně poslední část slouží jako sůl a je to náhodně generovaný řetězec.

Následuje slepé podepsání hlasu nejdříve od Boba a pak od inspektorů, tedy od Waldera. V případě voleb v prostředí FIT by inspektoři mohli být dobrovolníci nebo volební komise. Alice tedy vlastní hlas, podpis od Boba a podpis od Waldera. Nyní Alice vygeneruje novou blockchainovou adresu. Pokud by nyní hlas odeslala z nové adresy, bude dosaženo anonymity a zároveň díky podpisům bude hlas ověřen, nicméně bylo by veřejně známo, kolik má který kandidát hlasů. Proto bude potřeba posledního kroku, kdy již v registrační fázi Bob vygeneruje dvojici volebního veřejného a volebního soukromého

klíče, kde soukromý klíč bude uchován. Alice pak zašifruje svůj hlas tímto veřejným klíčem a celou trojici[20] odešle.

Sčítání

Mějme tedy k dispozici množinu všech došlých hlasů. Bob zveřejní volební soukromý klíč a systém automaticky dešifruje každý přichozí hlas, stejně tak ověří platnost podpisů, což nakonec může udělat každý účastník.

4.2.4 Výběr platformy

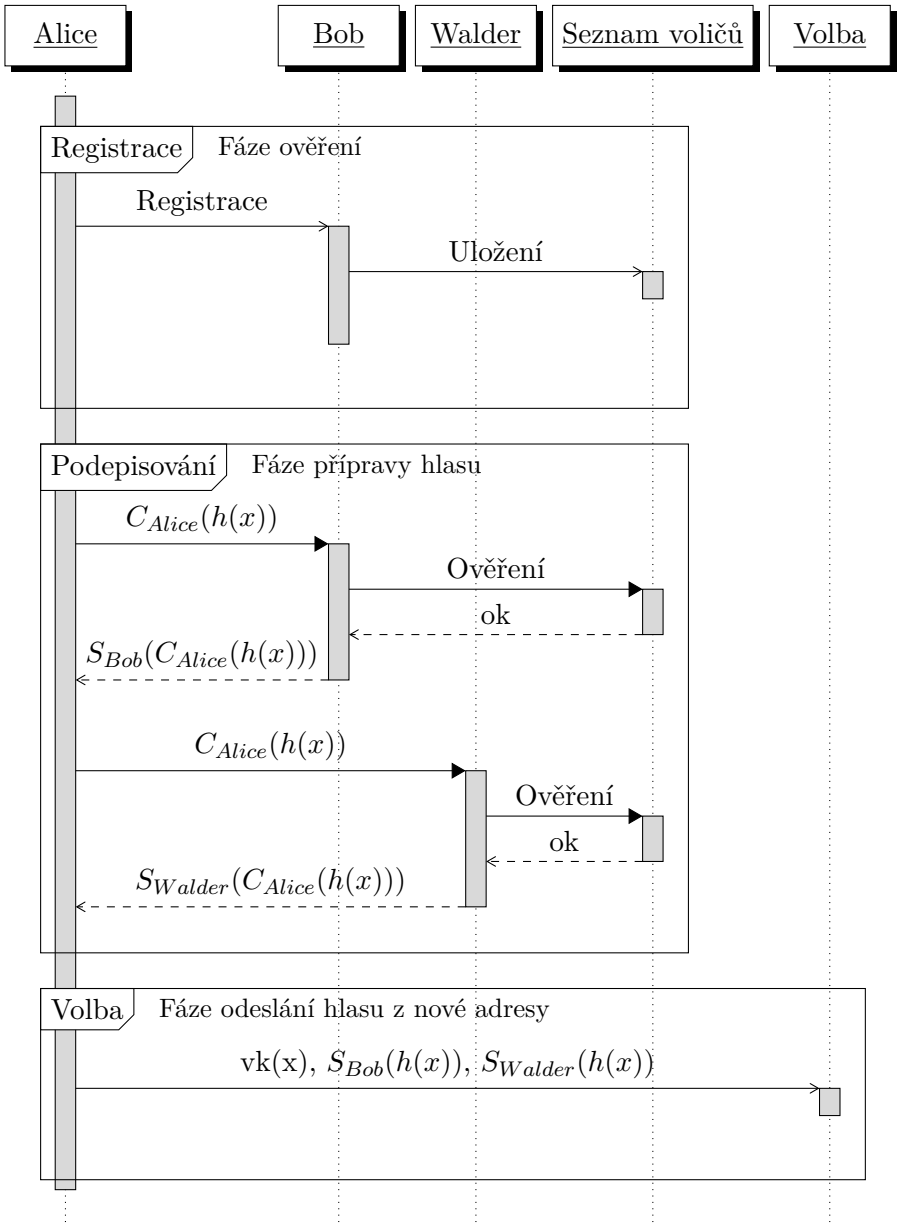
Přirozená vlastnost blockchainu, a to neměnnost uložených dat a transparentnost, našemu systému zaručí spravedlivý a transparentní průběh. Využití navrhovaného protokolu zase zaručí vlastnosti vhodné pro volby do AS FIT zmíněné v předchozích kapitolách. Takovýto systém lze bezpochyby implementovat do čistého blockchainu, kdy zasílané zprávy budou data v transakcích; avšak vyšší úroveň abstrakce poskytnou chytré kontrakty a to především uložitě a Turingovsky úplné provádění kódu. Tedy například ověřování a sčítání hlasů bude možné provádět přímo na blockchainu. Ne však slepé podepisování, protože při tom bude třeba používat funkce, které jsou tajemstvím podepisovaných.

4.2.5 Bezpečnostní analýza navrhovaného schématu

Otázka bezpečnosti takto navrženého schématu může být rozdělena do několika fází. Fáze registrace je transparentní a lze ji automatizovat. V případě prostředí FIT každý volič vlastní elektronickou identitu, takže fáze ověření identity a zařazení do množiny voličů může být plně automatizována. Tato registrační fáze je tedy plně v rukou volební autority, což je požadovaná vlastnost. Registrační fáze by měla být relativně dlouhá. V případě, že by oprávněný volič nebyl zařazen do seznamu oprávněných voličů, měl by mít možnost se odvolat. Tato fáze funguje tedy prakticky stejně jako konvenční volby. Systém by měl v této fázi veřejně oznámit, který veřejný klíč v množině oprávněných voličů patří kterému konkrétnímu člověku, čímž by bylo dosaženo požadované transparentnosti. Právní stránka tohoto kroku bude rozvedena v další kapitole.

Další fáze, tedy samotná volba, je již plně transparentní. Díky protokolu slepého podpisu administrátor i inspektoři podepisují zaslepený hlas bez toho, aby znali jeho obsah. Zároveň i stejný hlas bude mít rozdílný hash a to díky

[20] V tomto případě. Jestliže v systému existuje více inspektorů, klíčů bude přirozeně více.



Obrázek 4.4: Diagram prvních dvou fází, funkce vk je volební veřejný klíč a funkce h je hashovací funkce

náhodné části v hlasu. Administrátor a inspektoři zároveň kontrolují, jestli člověk hlasoval nebo ne. Při získání všech podpisů volič odešle hlas z nově vygenerované adresy. Díky tomu je zachována jeho anonymita a zároveň s hlasem odešle i podpisy, které budou ověřeny a tím bude zajištěno, že hlasující byl k hlasu oprávněn a zároveň, že nehlasoval vícetkrát. Zároveň by volič měl

používat například VPN nebo anonymizační síť TOR kvůli zamaskování IP adresy.

Nicméně i tak má takto navržené schéma určité slabiny. Ty se takřka výhradně týkají často příliš velké role autority a některé půjde vylepšit konkrétní implementací systémů třetí strany.

Velký problém je ověřování, kdy se musíme plně spolehnout na to, že autorita nezařadí do systému další klíče, popřípadě, že dokonce odmítne nějaký oprávněný klíč přidat. V druhém případě by volič měl využít právních mechanismů fakulty. V prvním případě je samozřejmě přirozené řešení zveřejnění seznamu všech oprávněných voličů s napárovanými veřejnými klíči.

Druhá slabina je na stejném principu a nachází se ve volební fázi. Volič podepisuje hlas volebním veřejným klíčem, administrátor může hlasy dešifrovat ještě před koncem. Řešením může být to, že před volbami se vygeneruje dvojice klíčů a soukromý klíč je uložen například k notáři nebo se důvěra svěří administrátorovi. Dešifrování hlasů před koncem je ekvivalentem otevření volební urny. V každém případě i v tomto posledním kroku je potřeba důvěřovat autoritě.

4.2.6 Výhody a nevýhody řešení

Autentizační část schématu se prakticky shoduje se současným řešením a nelze tedy hovořit o vylepšení vlastností. Zajímavá část je volební, kdy dochází k zajištění anonymity voliče a zároveň k plné transparentnosti, neboť každý účastník může kontrolovat, co se v blockchainové síti děje. V poslední fázi sčítání nelze považovat navrhovaný způsob za lepší než současný, neboť je opět potřeba důvěra v autoritu.

Mezi velkou nevýhodu patří určitá nepohodlnost pro uživatele, tedy voliče, který musí sám generovat klíče, podepisovat hlasy a podobně. Proto by měla instituce poskytovat nějaký otevřený nástroj, který tyto procesy zautomatizuje; nicméně hlasování by nemělo být závislé na použitých nástrojích. Toto bude více rozvedeno v implementační části, protože to s návrhem nesouvisí.

Mimoto navrhovaný systém netrpí žádnou zásadní nevýhodou oproti současnému řešení.

4.2.7 Možné modifikace

- Lze vynechat registrační fázi s tím, že bude administrátorovi a inspektorům poskytnut přístup do seznamu oprávněných voličů.

- Lze vygenerovat volební dvojici klíčů tak, že ihned po vygenerování bude volební soukromý klíč rozdělen a rozdistribuován, například s využitím Shamirova sdílení tajemství [41]. To je mimořádně problematické provést transparentně, ale může se to stát součástí předvolebního rituálu.

4.3 Legislativní aspekt navrženého řešení

Volby definované ve volební vyhlášce FIT [37] se všemi náležitostmi navrhované řešení takřka splňuje. Volba je tajná a při správném nastavení systému i přímá i dvouobvodová. Zároveň autentizace a přidání kandidátů na listinu je v rukou administrátora voleb, tedy zástupce autority, tedy fakulty. Potíž může být informace o sumárních výsledcích kandidátů; podle odstavce 4 článku 4 z [37] je informace o sumárních výsledcích jednotlivých kandidátů tajná. Nicméně pro kýžené zvýšení transparentnosti je potřeba tento bod nerespektovat a navrhovaný systém je možné nasadit až po změně tohoto předpisu. Stejně tak ze stejného odstavce nepřímo vyplývá, že anonymizace se lze provést až po skončení voleb; navrhovaný systém nemusí anonymizaci vůbec provádět.

V roce 2018 došlo k zásadní změně v oblasti právní ochrany osobních údajů vyhláškou Evropské unie, zvanou Obecné nařízení o ochraně osobních údajů, lépe známou jako GDPR. Ta mimo jiné může považovat veřejný klíč za osobní údaj [42], vedle jména, data narození nebo třeba i IP adresy. V zásadě je osobní údaj jakákoliv informace vedoucí přímo i nepřímo k identifikaci konkrétní osoby. V případě našeho systému by tedy veřejný klíč ve spojitosti s e-mailovou adresou byl jistě osobní údaj, nicméně veřejný klíč samotný a vygenerovaný anonymně pro jednoúčelové použití nikoliv [43]. Zveřejnění veřejného klíče ve spojitosti s identifikátorem (tedy e-mailovou adresou) v registrační fázi navrhovaného systému tedy podléhá souhlasu uživatele. Pro kvalitní transparentnost by uživatel měl souhlasit. Pokud souhlasit nebude, stejně by se mu nemělo bránit volbě a zveřejněn bude pouze veřejný klíč, který sám o sobě nevede k jednoznačné identifikaci, nicméně poté by měl existovat nějaký legislativní kontrolní mechanismus, nicméně to už záleží na konkrétní implementaci a případném rozhodnutí akademického senátu.

Kapitola 5

Proof of concept

Poslední kapitola se věnuje implementaci konceptu navrženého řešení. Bude využívat již existující blockchain Ethereum, jehož protokol bude popsán. Výhoda využití Etherea je v existenci EVM a jazyka Solidity, stejně tak v existenci mnoha nástrojů ulehčující vývoj.

5.1 Implementační návrh

Vycházíme ze systému navrženého v předchozí kapitole. Zásadní chytrý kontrakt je ten volební, nicméně proof of concept využije chytré kontrakty i jako uložiště pro zasílání hlasů. Není problém, že kdokoliv může vidět odeslaný hlas k podepsání, neboť tento hlas je zašifrován a připraven k slepým podpisům. Není problém, že všichni mohou teoreticky znát ty, kteří se účastní voleb, neboť do jisté míry je to odvoditelná informace. Pokud kdokoliv chce využít svého práva nevolit, ale zároveň o tom nikomu neříct, má možnost sestavit neplatný hlas, což systému nezpůsobí žádnou potíž.

5.2 Výběr technologií

K implementaci se využije veřejný blockchain Ethereum popsaný již v kapitole 3. Výhody využití této technologie vězí v existenci kvalitní dokumentace a v existenci mnoha nástrojů ulehčující práci. K zápisu chytrých kontraktů se využije jazyk Solidity s oficiálním kompilátorem do EVM bytekódu.

Pro provoz na fakultě by se nevyužívala veřejná síť Etherea. Mezi nevýhody potenciálního použití patří především ekonomická stránka věci; za vykonávání kódu je potřeba zaplatit. V případě našeho konceptu bude nasazena síť Ethereum a fakulta bude mít možnost natěžit dostatek tokenů, které bude

distribuovat jednotlivým účastníkům, kteří si jí případně mohou těžit také. V samotné implementaci by bylo vhodné tento mechanizmus v prostředí naší fakulty odstranit, popřípadě zavést jiný obranný mechanizmus proti *spamování* nebo DoS útokům, neboť potřeba platit za vykonávání kódu může být pro používání těžkopádné[21].

V proof of concept navrhovaného řešení se bude využívat implementace ethereovéhu uzlu `geth` [44], který poskytuje vše, co je potřeba:

- těžbu
- zasílání transakcí
- vytváření kontraktů
- prohlížení bloků v blockchainu

5.3 Implementace protokolu pro slepý podpis

Ke slepému podpisu se využije řešení s RSA podpisem [45], které vychází z návrhu v kapitole 4. Jedná se o modifikované klasické podpisové schéma RSA. Ať Alice chce podepsat zprávu m od Boba bez toho, aby prozradila obsah zprávy.

1. Bob vlastní RSA klíče a zveřejní veřejný klíč (e, n), kde e je veřejný exponent a n modul. Klíč (d, n) je jeho tajemství
2. Alice si zvolí náhodné číslo $r \ mod \ n$ tak, že $gcd(r, n) = 1$ (tedy existuje inverze r^{-1}, tedy $r \cdot r^{-1} \ mod \ n = 1$)
3. Alice vypočítá hash zprávy m, značíme $h(m)$, produktem je hash (na které lze nahlížet jako na 256bitové číslo v případě použití hashovací funkce SHA3)
4. Alice vypočítá zaslepenou zprávu b, $b = h(m) \cdot r^e \ mod \ n$ a odešle b
5. Bob standardně podepíše zprávu b, tedy $b_{sig} = b^d \ mod \ n$ a odešle. Z platnosti šifrovacího schématu RSA platí, že $b_{sig} = (h(m) \cdot r^e)^d \ mod \ n = h(m)^d \cdot r \ mod \ n$
6. Alice zná své zvolené r z bodu 2, tedy existence r^{-1} tak, že $r \cdot r^{-1} \ mod \ n = 1$ je zaručena, takže vypočítá $m_{sig} = b_{sig} \cdot r^{-1} \ mod \ n = h(m)^d \ mod \ n$

[21] I když bezcennou měnou, stále je potřeba ji vytěžit

7. $h(m)^d \ mod \ n$ je podpis Boba zprávy m

Pro tento první krok jsem vytvořil sadu nástrojů, které nabízí jednoduchou možnost pro tvorbu klíčů, podpisu, ověření a podobně. Výstupem je primitivní webová stránka, která slouží jako uživatelské rozhraní pro tyto nástroje.

Tento soubor krátkých funkcí slouží pro účely tohoto proof of concept; v reálně nasazeném systému by bylo vhodnější využít ověřenou knihovnu jako například `LibreSSL`. V případě pouhého konceptu je v této práci dána přednost jednoduchému a názornému řešení před kvalitní bezpečností. V reálné implementaci je třeba pravý opak. Důležité části jsou napsané ve výpisu kódu 5.3.

Nástroj k přípravě slepého podpisu využívá 2 zásadní knihovny. Je to knihovna `jsbn` pro manipulaci s velkými čísly, generování pseudonáhodných čísel a generování klíčů [46] a knihovna `Web3`, která je součástí projektu Ethereum pro hashování dle algoritmu Keccak-256 používaný v této síti.

5.4 Implementace kontraktů

V této podkapitole budou implementovány jednotlivé chytré kontrakty. Veškeré zdrojové kódy lze nalézt na přiloženém paměťovém médiu. Samotná architektura EVM přináší jisté problémy, jako například omezené úložiště. Tyto problémy se řeší technicky – například více úložišť, což nemá na funkčnost vliv. V implementovaném konceptu odhlédneme od těchto problémů, ale pro úplnost je nutno říct, že v ostré implementaci je třeba na tato omezení myslet.

5.4.1 Uložitě pro výměnu mezi voličem a podepisujícím

V této podkapitole bude implementována možnost zasílání zaslepených hlasů do sdílené sítě. Zasílání hlasů k podpisu a další výměny zpráv tedy budou probíhat transparentně na blockchainu. Úložiště bude napsáno v jazyce Solidity a zkompilováno do EVM byte kódu. Tabulka 5.2 ukazuje požadované rozhraní a popis funkčnosti jednotlivých komponentů. Samotný kód je k dispozici na přiloženém paměťovém médiu.

V konstruktoru kontraktu, který je zavolán právě při nasazení kontraktu do sítě lze dobře specifikovat, které volající adresy jsou administrátoři, inspektoři a v závislosti na konkrétním řešení i které adresy jsou oprávnění voliči, čili členové akademické obce. V proof of concept řešení bude oprávněný volič libovolná adresa. Přidání kontroly oprávněnosti je triviální úprava. Některé funkce lze použít v automatizační rouře – například podepisování může být plně zautomatizováno, nicméně to již přesahuje rozsah tohoto konceptu.

```
function blind(vote, e, n) {
        const h_m = new BigInteger (keccak_256(vote), 16)
        const arr = new Array(64)
        let r
        do {
                new SecureRandom().nextBytes(arr)
                r = new BigInteger(arr).mod(n)
        } while ( !r.gcd(n).equals(BigInteger.ONE) )
        return {
                hashed: h_m,
                r,
                blinded: h_m.multiply(r.modPow(e, n)).mod(n)
        }
}

function sign(b, d, n) {
        return b.modPow(d, n)
}

function unblindByMultiplyRandInv(b_sig, r, n) {
        return b_sig.multiply(r.modInverse(n)).mod(n)
}

function verifySignature(m_sig, e, n) {
        return m_sig.modPow(e,n)
}
```

Výpis kódu 4: Zápis algoritmu pro slepé podepsání v jazyce *JavaScript*; proměnné odpovídají konvencím z kapitoly 5.3

5.4.2 Volba

Tato podkapitola implementuje samotnou volbu. Vše potřebné pro tuto implementaci již bylo zmíněno v předchozích kapitolách. Kontrakt bude do jisté míry podobný předchozímu, nicméně samozřejmě slouží k jinému účelu. V tomto konceptu bude implementováno i šifrování hlasu veřejným volebním klíčem vydávaným volební komisí z důvodu již zmíněného; je požadováno, aby v průběhu volby nebyl znám stav hlasování. Koncept již nebude řešit uchovávání komplementárního soukromého klíče.

Samozřejmě neexistuje obrana proti tomu, aby někdo nahrál nezašifrovaný hlas, přestože kontrakt obsahuje jednoduchou kontrolu vstupů. Kontrakt stejně tak nepřijme hlas ve špatném formátu; to jest přirozená vlastnost kaž-

Tabulka 5.1: Veřejné rozhraní kontraktu `Storage`

Metoda	Popis
`add(b)`	Odešle zaslepený hlas k podpisu
`getBVote(addr)`	Pokud je volající oprávněn k podepisování hlasů, je mu vydán hlas z adresy `addr` k podpisu
`setSign(b, sigb)`	Vrátí do systému podepsaný hlas pokud volající má oprávnění podepisovat
`collect()`	Volajícímu je vydána struktura s jeho zaslepeným podpisem a příslušnými podpisy; volající je identifikován adresou; pokud daný podpis chybí, je přítomný byte `0x00`

dého strojového zpracování.

Je naprosto kruciální, aby hlasující s tímto kontraktem interagovali pod novou identitou v podobně nově vygenerované a jednorázové adresy. Jejich hlas by i nadále byl platný, ale tajnost jejich volby by byla kompromitována.

Tabulka 5.2: Veřejné rozhraní kontraktu `Ballot`

Metoda	Popis
`vote(v, aSig, wSig)`	Uloží zašifrovaný hlas a příslušné podpisy
`tally(privKey)`	Pokud je volající administrátor (ten, který zavolal konstruktor kontraktu), dojde k ukončení voleb a výsledky se uloží
`results(c)`	Vrátí počet hlasů pro volbu kódovanou bytem `c`; interně se používá struktura typu `mapping`

5.5 Testování

Pro otestování správnosti řešení je využito knihoven *Truffle* a testovacího blockchainu *Ganache* patřící do frameworku *Truffle Suite* [47] . Na přiloženém datovém médiu lze nalézt kód jednotkových testů společně s použitou konfigurací. Truffle poskytuje interface pro kompilaci a nasazení kontraktů, Ganache zase plnohodnotný uzel Etherea s přednastavenými účty založený na softwaru

```
function tally (bytes memory privKey) public {
        require(msg.sender == admin, "Tally can be called only by admin");
        uint len = votes.length;
        for (uint i = 0 ; i < len ; i++) {
                BallotVote memory v~= votes[i];
                decypher(v.encVote, privKey, v.sigAdmin, v.sigInspector);
        }
}
```

Výpis kódu 5: Metoda `tally(privKey)` je zásadní. Jako argument přijímá soukromý volební klíč, respektive exponent toho klíče, neboť modul je znám veřejně a jako strukturu pro ukládání využívá typ `mapping`

```
function decypher (
        bytes memory _base,
        bytes memory _exp,
        bytes memory sA,
        bytes memory sI
        ) private {
        bytes memory result = modexp(_base, _exp, ballotMod);
        bytes32 hash = keccak256(result);
        byte firstByte = result[0];
        bytes32 sA_kecc = Cut64bytesTo32Keccak(modexp(sA, pe, modKeyA));
        bytes32 sI_kecc = Cut64bytesTo32Keccak(modexp(sI, pe, modKeyI));
        if (hash == sA_kecc && hash == sI_kecc ){
                results[firstByte] += 1;
        }
}
```

Výpis kódu 6: Z předchozího výpisu je vidět využití metody `decypher`, jejíž argumenty jsou vlastní zašifrovaný hlas, dešifrovací volební expoment a podpisy; použité privátní metody nedělají nic překvapivého – metoda `modeexp` je modulární umocňování a metoda `Cut64bytesTo32Keccak` slouží ke změně typu `bytes` na `bytes32`

`geth`. Postup jak otestovat řešení je popsán v příloze C. Jsou přítomny celkem 2 sady testů pro oba kontrakty – pro `Storage` i pro `Ballot`. Otestovány jsou postupně všechny možné scénáře. Testy jsou popsány v souboru `README.md` na přiloženém paměťovém médiu.

Závěr

V této bakalářské práci byly rozebrány principy, na kterých fungují moderní blockchainové sítě a kryptoměny. Bylo probráno využití těchto technologií. Vysvětlen byl princip fungování chytrých kontraktů existující na této platformě.

V modelovém využití chytrých kontraktů společně s blockchainem byl zvolen problém voleb. Takové téma je samo o sobě mimořádně složité a citlivé. Současné řešení může být kvalitně zabezpečeno, nicméně stále se jedná o centralizovaný systém, kterému z podstaty je nutno důvěřovat. Systém navržený v této práci je decentralizovaný a důvěry vyžaduje řádově méně. Nejslabší články systému byly již vytyčeny v samotné práci, nicméně z hlediska tendence k vypouštění autorit je navržený systém jistě krokem vpřed. Byla provedena kryptografická analýza – systém prakticky závisí pouze na bezpečnosti asymetrické kryptografie a kryptografické hashovací funkce. Nevýhoda řešení je především v určité nepohodlnosti pro uživatele, který je navíc identifikován jako slabé místo systému, neboť jeho soukromí je čistě v jeho rukou.

Těžištěm této práce je čtvrtá kapitola, tedy vlastní návrh systému. Srovnal jsem několik protokolů pro bezpečné volby a vybral jsem takový, který má nejlepší průnik uživatelského pohodlí, snadnosti implementace do blockchainu a zabezpečení, ale neobešlo se bez mírné modifikace.

Funkčnost návrhu je dokázána v poslední kapitole, jejíž větší část je ve zdrojových kódech v příloze. Řešení je podpořeno automatizovanými testy, které lze nalézt tamtéž. Budoucnost projektu je spíš softwarově inženýrského rázu, neboť systém je již navržen. Pokud by mělo dojít k nasazení v příštích volbách, bylo by potřeba vytvořit pohodlné uživatelské rozhraní a rozumnou autentizaci a především je nutné vytvořit osvětovou činnost mezi členy akademické obce a řádně vysvětlit princip, na jakém nový typ voleb funguje.

Odměnou akademické obci bude fakt, že již nemusí věřit centrální relační databázi.

Literatura

[1] NARAYANAN, A.; BONNEAU, J.; FELTEN, E.; aj.: *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton, NJ, USA: Princeton University Press, 2016, ISBN 0691171696, 9780691171692.

[2] KLÍMA, V.: Tunely v hašovacích funkcích: kolize MD5 do minuty. 2006, [online] [cit. 15.10.2018]. Dostupné z: http://cryptography.hyperlink.cz/2006/tunely.pdf

[3] DWORKIN, M. J.: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. [online] [cit. 9.5.2019]. Dostupné z: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf

[4] WOOD, G.: Ethereum: A Secure Decentralised Generalised Transaction Ledger. 2019, [online] [cit. 27.3.2019]. Dostupné z: https://ethereum.github.io/yellowpaper/paper.pdf

[5] ANTONOPOULOS, A. M.: *Mastering bitcoin*. Sebastopol CA: O'Reilly, [2015]., ISBN 978-1-449-37404-4.

[6] LÓRENCZ, R.; KOKEŠ, J.: BI-BEZ: 6. přednáška: RSA, kryptografie s veřejným klíčem, DSA, El-Gamalův algoritmus. Fakulta informačních technologií ČVUT. [online] [cit. 15.10.2018]. Dostupné z: https://courses.fit.cvut.cz/BI-BEZ/media/bez-n6.pdf

[7] Cryptocurrency Market Capitalizations. [online] [cit. 27.3.2019]. Dostupné z: https://coinmarketcap.com/

[8] Secp256k1. 2010-, [online] [cit. 27.3.2019]. Dostupné z: https://en.bitcoin.it/wiki/Secp256k1

[9] KALVODA, T.; STAROSTA, S.; PETR, I.: *Matematika pro kryptologii*. Praha, 2019, [online] [cit. 12.5.2019]. Dostupné z: https://courses.fit.cvut.cz/MI-MKY/media/lectures/mi-mky-poznamky-v17.pdf

[10] LÓRENCZ, R.: BI-BEZ: 8. přednáška: Základy kryptografie eliptických křivek a kvantové kryptografie. Fakulta informačních technologií ČVUT. [online] [cit. 27.3.2019]. Dostupné z: https://courses.fit.cvut.cz/BI-BEZ/media/bez-n8.pdf

[11] BROWN, D.: SEC 2: Recommended Elliptic Curve Domain Parameters. 2010, [online] [cit. 27.3.2019]. Dostupné z: http://www.secg.org/sec2-v2.pdf

[12] JOHNSON, D.; MENEZES, A.: The Elliptic Curve Digital Signature Algorithm (ECDSA). *International journal of information security*, ročník 1, č. 1, 2001: s. 36–63.

[13] NAKAMOTO, S.: Bitcoin: A Peer-to-Peer Electronic Cash System. 2008, [online] [cit. 15.10.2018]. Dostupné z: http://bitcoin.org/bitcoin.pdf

[14] SCHOLLMEIER, R.: A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. In *Proceedings First International Conference on Peer-to-Peer Computing*, IEEE Comput. Soc, 2002, ISBN 0-7695-1503-7, doi:10.1109/P2P.2001.990434, [online] [cit. 15.10.2018]. Dostupné z: https://www.researchgate.net/publication/3940901_A_Definition_of_Peer-to-Peer_Networking_for_the_Classification_of_Peer-to-Peer_Architectures_and_Applications

[15] HABER, S.; STORNETTA, W. S.: How to Time-stamp a Digital Document. *Journal of Cryptology*, ročník 3, 1991: s. 99–111.

[16] MOUBARAK, J.; FILIOL, E.; CHAMOUN, M.: On blockchain security and relevant attacks. In *2018 IEEE Middle East and North Africa Communications Conference (MENACOMM)*, IEEE, 2018, ISBN 978-1-5386-1254-5, s. 1–6, doi:10.1109/MENACOMM.2018.8371010, [online] [cit. 5.4.2019]. Dostupné z: https://ieeexplore.ieee.org/document/8371010/

[17] Tor Project. [online] [cit. 13.5.2019]. Dostupné z: https://www.torproject.org/about/history/

[18] Confirmed Transactions Per Day. [online] [cit. 5.5.2019]. Dostupné z: https://www.blockchain.com/charts/n-transactions?timespan=all&daysAverageString=7

[19] Market Price (USD). [online] [cit. 5.5.2019]. Dostupné z: https://www.blockchain.com/charts/market-price?timespan=all&daysAverageString=7

[20] IBM Blockchain Platform. [online] [cit. 9.4.2019]. Dostupné z: https://www.ibm.com/downloads/cas/Q9DGBLV7

[21] CHAVEZ-DREYFUSS, G.: Sweden tests blockchain technology for land registry. *Reuters, June*, ročník 16, 2016. Dostupné z: https://cartorios.org/wp-content/uploads/2016/06/2016-06-26-sweden-tests-blockchain-technology-for-land-registry.pdf

[22] White Paper. [online] [cit. 9.4.2019]. Dostupné z: https://kodakone.com/fileadmin/white_paper/180424_kodakone_wp.pdf

[23] JAFARKARIMI, H.; SIM, A.; SAADATDOOST, R.; aj.: The Impact of ICT on Reinforcing Citizens' Role in Government Decision Making. *International Journal of Emerging Technology and Advanced Engineering*, ročník 4, č. 1, 2014: s. 642–646, [online] [cit. 13.5.2019]. Dostupné z: https://ijetae.com/files/Volume4Issue1/IJETAE_0114_109.pdf

[24] PILKINGTON, M.: 11 Blockchain technology: principles and applications. *Research handbook on digital transformations*, ročník 225, 2016.

[25] Danish Political Party May Be First to Use Block Chain For Internal Voting. Apr 2014, [online] [cit. 13.5.2019]. Dostupné z: https://www.newsbtc.com/2014/04/22/danish-political-party-may-first-use-block-chain-internal-voting/

[26] Follow my vote. 2019, [online] [cit. 13.5.2019]. Dostupné z: https://followmyvote.com/press/

[27] Follow my vote. 2019, [online] [cit. 13.5.2019]. Dostupné z: https://followmyvote.com/online-voting-technology/blockchain-technology/

[28] SZABO, N.: The Idea of Smart Contracts. 1998, [online] [cit. 10.4.2019]. Dostupné z: https://nakamotoinstitute.org/the-idea-of-smart-contracts/

[29] SZABO, N.: Secure Property Titles with Owner Authority. 1998, [online] [cit. 10.4.2019]. Dostupné z: https://nakamotoinstitute.org/secure-property-titles/

[30] SZABO, N.: A Formal Language for Analyzing Contracts. 2002, [online] [cit. 10.4.2019]. Dostupné z: https://nakamotoinstitute.org/contract-language/

[31] WOOD, G.; ANTONOPOULOS, A. M.: *Mastering Ethereum*. First edition vydání, 2018, ISBN 9781491971932, [online] [cit. 5.4.2019]. Dostupné z: https://github.com/ethereumbook/ethereumbook

[32] Solidity. 2019, [online] [cit. 24.4.2019]. Dostupné z: https://solidity.readthedocs.io/en/v0.4.13/

[33] Cardano. [online] [cit. 5.5.2019]. Dostupné z: https://www.cardano.org/en/home/

[34] EOS. [online] [cit. 5.5.2019]. Dostupné z: https://eos.io/

[35] Akademický senát FIT. [online] [cit. 28.3.2019]. Dostupné z: https://fit.cvut.cz/as

[36] Zákon o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách). 1998, § 25 odst. 1 písm. a). Dostupné z: http://www.msmt.cz/vyzkum-a-vyvoj-2/zakon-c-111-1998-sb-o-vysokych-skolach

[37] Volební řád akademického senátu Fakulty informačních technologií. 2017, [online] [cit. 28.3.2019]. Dostupné z: https://fit.cvut.cz/sites/default/files/AS_volebni_rad_2017-11-29.pdf

[38] HAO, F.; RYAN, P.; ZIELIŃSKI, P.: Anonymous voting by two-round public discussion. *IET Information Security*, ročník 4, č. 2, 2010, ISSN 17518709, doi:10.1049/iet-ifs.2008.0127, [online] [cit. 4.4.2019]. Dostupné z: https://digital-library.theiet.org/content/journals/10.1049/iet-ifs.2008.0127

[39] VAN SABERHAGEN, N.: CryptoNote v 2.0, Říjen 2013, [online] [cit. 5.4.2019]. Dostupné z: https://cryptonote.org/whitepaper.pdf

[40] LIU, Y.; WANG, Q.: An E-voting Protocol Based on Blockchain. *IACR Cryptology ePrint Archive*, ročník 2017, 2017: str. 1043.

[41] SHAMIR, A.: How to share a secret. *Communications of the ACM*, ročník 22, č. 11: s. 612–613, ISSN 00010782, doi:10.1145/359168.359176. Dostupné z: http://portal.acm.org/citation.cfm?doid=359168.359176

[42] FINCK, M.: Blockchains and Data Protection in the European Union. *SSRN Electronic Journal*, ISSN 1556-5068, doi:10.2139/ssrn.3080322, [online] [cit. 4.4.2019]. Dostupné z: https://www.ssrn.com/abstract=3080322

[43] Nařízení Evropského parlamentu a Rady (EU) 2016/679 ze dne 27. dubna 2016 o ochraně fyzických osob v souvislosti se zpracováním osobních údajů a o volném pohybu těchto údajů a o zrušení směrnice 95/46/ES (obecné nařízení o ochraně osobních údajů). *Úřední věstník Evropské unie*, ročník L119, Květen 2016: s. 1–88, [online] [cit. 4.4.2019]. Dostupné z: https://eur-lex.europa.eu/legal-content/CS/TXT/PDF/?uri=OJ:L:2016:119:FULL

[44] go-ethereum. [software][online] [cit. 13.5.2019]. Dostupné z: https://github.com/ethereum/go-ethereum/wiki

[45] GOLDWASSER, S.; BELLARE, M.: Lecture Notes on Cryptography. 2001, [online] [cit. 11.4.2019]. Dostupné z: http://cseweb.ucsd.edu/~mihir/papers/gb.pdf

[46] WU, T.: jsbn. [software] [online] [cit. 14.4.2019]. Dostupné z: http://www-cs-students.stanford.edu/~tjw/jsbn/

[47] Truffle Suite. 2019, [software][online] [cit. 13.5.2019]. Dostupné z: https://truffleframework.com/

Příloha A

Seznam použitých zkratek

AS Akademický senát.

DDoS Distributed denial-of-service.

DoS Denial-of-service.

ECDSA The Elliptic Curve Digital Signature Algorithm.

EVM Ethereum Virtual Machine.

FIT Fakulta informačních technologií.

GDPR General Data Protection Regulation.

IP Internet Protocol.

MD5 MD5 message-digest algorithm.

P2P Peer-to-peer.

SHA Secure Hash Algorithm.

TOR The Onion Router.

USA United States of America.

VPN Virtual private network.

ČVUT České vysoké učení technické.

Příloha **B**

Obsah přiloženého disku

- `readme.txt` ..... Popis obsahu
- `src`
  - `impl` ..... Zdrojový kód implementace
    - `tools` ..... Nástroj pro generování klíčů a slepé podepisování
    - `poc` ..... Proof of concept
  - `thesis` ..... LaTeX zdrojový kód zprávy
- `text`
  - `thesis.pdf` ..... Zpráva ve formátu pdf

Příloha C

Testování

Náplní této přílohy je postup, jak ověřit proof of concept řešení z poslední kapitoly pomocí přiložených jednotkových testů. Na paměťovém médiu lze nalézt soubor `README.md`, který je zkrácenou a více návodnou verzí této přílohy a navíc obsahuje bližší popis prováděných testů.

C.1 Prerekvizity

Jedinou prerekvizitou je nainstalovaná technologie `docker`, která umožňuje vytvořit oddělené prostředí od operačního systému. V daném prostředí se nadefinuje použitý operační systém, nainstalované balíčky a předdefinují se prováděné příkazy. Lze si to představit jako zjednodušenou virtualizaci. V operačních systémech používající balíčkový systém `apt` lze `docker` nainstalovat příkazem `apt install docker.io` a další informace o tomto softwaru je možné získat na `www.docker.io`.

C.2 Příprava

Splnění prerekvizity lze ověřit příkazem `docker --version`, jehož výstupem by měla být používaná verze. Ve složce `src/impl/poc` je přítomen soubor `Dockerfile`, který obsahuje definici pro oddělené prostředí – v našem případě nainstaluje testovací frameworky `Truffle` a `Ganache`. Obraz pro prostředí lze vytvořit příkazem `docker build --tag=<NAME> .`, kdy pracovní adresář je právě ve složce `src/impl/poc` a řetězec <NAME> je libovolné jméno obrazu. Po spuštění příkazu je možné, že na standardním výstupu budou vypsané

chyby[22], ale zásadní je, příkaz skončí úspěchem. Ve složce s kontrakty se také nachází soubor `Migrations.sol`, což je speciální kontrakt potřebný k fungování frameworku `Truffle` a není součástí práce.

C.3 Spuštění testů

Jakmile je připravený obraz, lze spustit samotné testy pomocí příkazu `docker run <NAME>`, kde řetězec <NAME> se musí shodovat s tím, který byl použit v předchozím kroku. Stanou se následující věci:

1. Spuštění uzlu Etherea s přednastavenými adresami a účty (`Ganache`)
2. Stažení správné verze kompilátoru `solc`
3. Zkompilování kontraktů a jejich nasazení na blockchain
4. Spuštění testů pro kontrakt `Ballot`
5. Spuštění testů pro kontrakt `Storage`

[22]Je to z důvodu, že toto konkrétní oddělené prostředí nemá předdefinované některé technologie, nicméně pro účely tohoto testu to nemá vliv