

УНИВЕРЗИТЕТ У БЕОГРАДУ  
ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ



# **РАЗВОЈ ВЕБ ПЛАТФОРМЕ ЗА ГЛАСАЊЕ БАЗИРАНЕ НА БЛОКЧЕЈН ТЕХНОЛОГИЈИ**

Дипломски рад

Ментор:

доц. др Дражен Драшковић

Кандидат:

Алекса Рајковић 2016/248

Београд, септембар 2020.

# САДРЖАЈ

<b>САДРЖАЈ .....</b>	<b>2</b>
<b>1. УВОД.....</b>	<b>4</b>
<b>2. БЛОКЧЕЈН .....</b>	<b>6</b>
2.1. ОСНОВНИ ПОЈМОВИ.....	6
2.1.1 Симетрично шифровање.....	6
2.1.2 Асиметрично шифровање .....	6
2.1.3 Хеи функција.....	7
2.1.4 Сертификат.....	7
2.2. ДЕФИНИЦИЈА БЛОКЧЕЈНА .....	8
2.3. НАСТАНАК БЛОКЧЕЈНА .....	9
2.4. СТРУКТУРА .....	9
2.5. МРЕЖА.....	10
2.6. ПОСЛЕДИЦЕ ДИЗАЈНА .....	11
2.7. КЛАСИФИКАЦИЈА .....	12
2.7.1 Јавни блокчејн.....	12
2.7.2 Приватни блокчејн.....	12
2.7.3 Хибридни блокчејн .....	12
2.7.4 Споредни блокчејн.....	12
2.8. НАПАДИ.....	12
2.8.1 Малвер рудар .....	12
2.8.2 Напад 51%.....	13
2.8.3 Лажне трансакције .....	13
2.8.4 Крађа приватног кључа.....	13
2.9. ПРЕДНОСТИ .....	13
2.9.1 Тачност ланца .....	13
2.9.2 Смањење трошкова.....	14
2.9.3 Децентрализација .....	14
2.9.4 Ефикасност трансакција.....	14
2.10. МАНЕ .....	14
2.10.1 Трошкови технологије и технике.....	15
2.10.2 Нелегални системи.....	15
2.10.3 Напади.....	15
2.11. ПРИМЕНА.....	15
2.11.1 Здравство.....	15
2.11.2 Информациони системи .....	16
2.11.3 Паметни уговори.....	16
2.11.4 Гласање .....	16
<b>3. ПОСТОЈЕЋА РЕШЕЊА .....</b>	<b>17</b>
3.1. ЕСТОНИЈСКИ МОДЕЛ.....	17
3.2. ПЛИМУТ МОДЕЛ.....	18
3.3. РЕЈКЈАВИК МОДЕЛ .....	19
<b>4. СПЕЦИФИКАЦИЈА СИСТЕМА.....</b>	<b>20</b>
4.1. ТИП ЛАНЦА .....	20
4.2. <i>PROOF OF AUTHORITY</i> АЛГОРИТАМ .....	21
4.3. УЛОГЕ .....	22

<b>5.</b>	<b>МОДЕЛ СИСТЕМА.....</b>	<b>23</b>
5.1.	СТРУКТУРА .....	23
5.2.	АРХИТЕКТУРА.....	26
5.3.	ПРОТОКОЛИ .....	27
5.3.1	<i>Протокол упознавања.....</i>	<i>27</i>
5.3.2	<i>Протоколи аутентификације.....</i>	<i>29</i>
5.3.3	<i>Протокол одржавања .....</i>	<i>30</i>
5.3.4	<i>Протокол елиминације.....</i>	<i>30</i>
5.3.5	<i>Протокол консензуса .....</i>	<i>31</i>
5.3.6	<i>Протокол ауторизације актера .....</i>	<i>31</i>
5.4.	ФУНКЦИОНАЛНОСТИ СИСТЕМА .....	32
5.4.1	<i>ClientAPI руте .....</i>	<i>32</i>
5.4.2	<i>ControllerAPI руте .....</i>	<i>34</i>
5.4.3	<i>NodeAPI руте.....</i>	<i>34</i>
5.5.	ПРИМЕР СТВАРАЊА ГЛАСАЊА.....	36
<b>6.</b>	<b>РЕАЛИЗАЦИЈА СИСТЕМА.....</b>	<b>37</b>
6.1.	ИЗБОР ТЕХНОЛОГИЈЕ .....	37
6.2.	ПОКРЕТАЊЕ СИСТЕМА.....	37
6.3.	ПРИКАЗ КОРИСНИЧКОГ ИНТЕРФЕЈСА.....	38
<b>7.</b>	<b>ЗАКЉУЧАК .....</b>	<b>42</b>
	<b>ЛИТЕРАТУРА.....</b>	<b>44</b>
	<b>СПИСАК СКРАЋЕНИЦА .....</b>	<b>45</b>
	<b>СПИСАК СЛИКА.....</b>	<b>46</b>

# 1. Увод

Сведоци смо времена у којем је присутан масовни раст технологија и платформи које исте имплементирају, блокчејн као једна од најновијих показала је широк спектар примене и ефикасности при употреби, паралелно подстичући индустрију на нове подухвате већом потражњом за сигурнијим и транспарентнијим процесима продукције, администрације и извештавања. Блокчејн је настао као криптовалута изазвајући велико интересовање и формирајући предмет поларизације између стручне и шире јавности у изненађујуће кратком року. Главни аспект који се уводи јесте потпуна децентрализација система и непостојање треће стране при извршавању трансакција уз потпуно задржавање аутентичности, интегритетета, поверљивости и неопорецивости истих. Једна од примена би самим тим логично била и реализација гласања на различитим нивоима формалности. Блокчејн трансакције би у идеалном случају представљале висок степен некомпромитоване демократије виртуелизоване над апстракцијама криптографије и сигурним интернет протоколима.

Теза за циљ има детаљну анализу примене блокчејн технологије у сврху формирања јединствене веб платформе као апликације за одржавања гласања различитог типа. Посматрањем теоријских основа криптографије и постулата интернет протокола разматраће се различите технике којима би се гласање интегрисало у једну интернет апликацију која је реализована. Сврха истраживања јесте поставити добар теоријски модел, такав да платформа пружа висок степен сигурности, као и отпор од реално могућих сајбер напада, и да онда на основу формиране спецификације изврши селекцију потребних и пре свега ефикасних технологија и алата за имплементацију исте. Задатак је и наћи могуће реализације структуре блокчејна као и метод за структурирање трансакција гласања у систему тако да су анонимност, неопорецивост и сигурност платформе инхерентне последице саме архитектуре. Апликација се развија као софтвер потпуног скупа технологија (енг. *full stack*) и тежи да омогући што једноставнији и интуитивнији кориснички интерфејс који нуди ефикасну употребу кориснику, али исто тако формира децентрализовану и скалабилну мрежу лако преносивих и за одржавање једноставних сервера коју би администратори одржавали.

На самом почетку овог документа разматрају се фундаменти криптографије, криптоанализе и њихове директне приемене уз додатно дефинисање потребних појмова и информационих структура. Додатно се анализирају други, већ постојећи, блокчејн системи за гласање имплементирани од стране других организација и установа и даје се коначан модел чијом се имплементацијом теза бави.

Саму срж рада чини детаљна реализација система, првенствено уз образложења изабраних технологија тј. технологије *Tech Stack* као и погодности које оне за сам пројекат доносе. Даје се подробна структура архитектуре система са свим интерфејсима и протоколима истог, пратећи модуле имплементационих јединица платформе као што су енкапсулације развијених ентитета, дефиниције њихових интерфејса, механизма перзистенције и међупроцесне комуникације развијене како моделом дељене променљиве тако и моделом размене порука. Све то пропраћено је разним UML (*Unified Modeling Language*) дијаграмима и шемама.

Коначно даје се преглед имплементираног уз анализу и могуће тачке проширења као и обсервацију осетљивих тачака система. Дате су слике на којима је приказан кориснички интерфејс свих приступних веб страна по улогама као и објашњења шта се формама на странама постиже и позива.

## 2. БЛОКЧЕЈН

Структура и имплементација самог блокчејна базирана је у основи на прецизним постулатима теорије криптографије. Анализирати један ланац значи анализирати и потребне структуре заштите података које исти и чине. У овом поглављу прво се дефинишу неопходни појмови теорије криптографије, а затим се прелази на дефиницију блокчејна.

### 2.1. Основни појмови

Разматрају се основне дефиниције структура заштита података које ће се користити у даљем излагању, али су исто тако кључне за разумевање истог.

#### 2.1.1 Симетрично шифровање

Представља класу алгоритама криптографије који користе идентични кључ за шифровање и дешифровање податка. У пракси ентитети који комуницирају користе дељени симетрични кључ како би одржали сигурну тј. поверљиву везу. Даље у раду при свакој итерацији анализе важиће следеће ознаке:

- $K_s$  - јесте ознака произвољног симетричног кључа
- $E(M, K_s)$  - јесте ознака енкрипције податка  $M$  симетричним кључем  $K_s$
- $D(S, K_s)$  - јесте ознака декрипције податка  $S$  симетричним кључем  $K_s$
- $D(E(M, K_s), K_s) = M$  - јесте последица механизма симетричног шифровања

Као један од најкоришћенијих имплементација симетричног шифровања наводи се *AES* (*Advanced Encryption Standard*) алгоритам.

#### 2.1.2 Асиметрично шифровање

Представља класу алгоритама који користе пар математички спрегнутих кључева како би шифровали и дешифровали податак. Шифрован података једним кључем се може дешифровати само његовим паром, али не и истим кључем као што је то био случај код симетричног шифровања. У пракси ентитети који комуницирају користе ова два кључа као приватни и јавни. Јавни је доступан свима и може се користити како би само власник упареног приватног кључа могао да дешифрује поруку. Шифровање јавним кључем дакле

омогућава поверљивост, док се аутентичност поруке реализује механизмима приватне енкрипције исте тј. потписивањем. Даље у раду при свакој итерацији анализе важиће следеће ознаке:

$(K_{pu}, K_{pr})$  – јесте ознака пара математички спрегнутих кључева који чине јавни и приватни кључ

$E(M, K_{pu})$  – јесте ознака енкрипције поруке  $M$  јавним кључем  $K_{pu}$

$D(M, K_{pu})$  – јесте ознака декрипције поруке  $M$  јавним кључем  $K_{pu}$

$E(M, K_{pr})$  – јесте ознака енкрипције поруке  $M$  приватним кључем  $K_{pr}$  која је уједно и потпис поруке  $M$  датим приватним кључем.

$D(M, K_{pr})$  – јесте ознака декрипције поруке  $M$  приватним кључем  $K_{pr}$

$D(E(M, K_{pu}), K_{pr}) = M$  – јесте последица механизма асиметричног шифровања

$D(E(M, K_{pr}), K_{pu}) = M$  – јесте последица механизма асиметричног шифровања

Као један од најкоришћенијих имплементација асиметричног шифровања наводи се RSA (*Rivest–Shamir–Adleman*) алгоритам.

### 2.1.3 Хеш функција

Хеш функција је сваки алгоритам који подацима произвољне дужине додељује податке фиксне дужине. Хеш вредност у криптографији је број генерисан из ниски текста. Хеш вредност је знатно мања од самог текста и генерисана је хеш алгоритмом на такав начин да је вероватноћа да неки други текст има исту хеш вредност занемарљива. Основи принцип који се постиже хеширањем јесте очување интегритета поруке. У пракси се поред саме поруке шаље и њен хеш који се на пријемној страни тестира на валидност. Даље у раду при свакој итерацији анализе важиће следеће ознаке:

$H(M)$  – јесте ознака хеш вредности поруке  $M$  применом алгоритма  $H$

Као један од најкоришћенијих имплементација криптографских хеширања наводи се SHA (*Secure Hash Algorithm*) алгоритам.

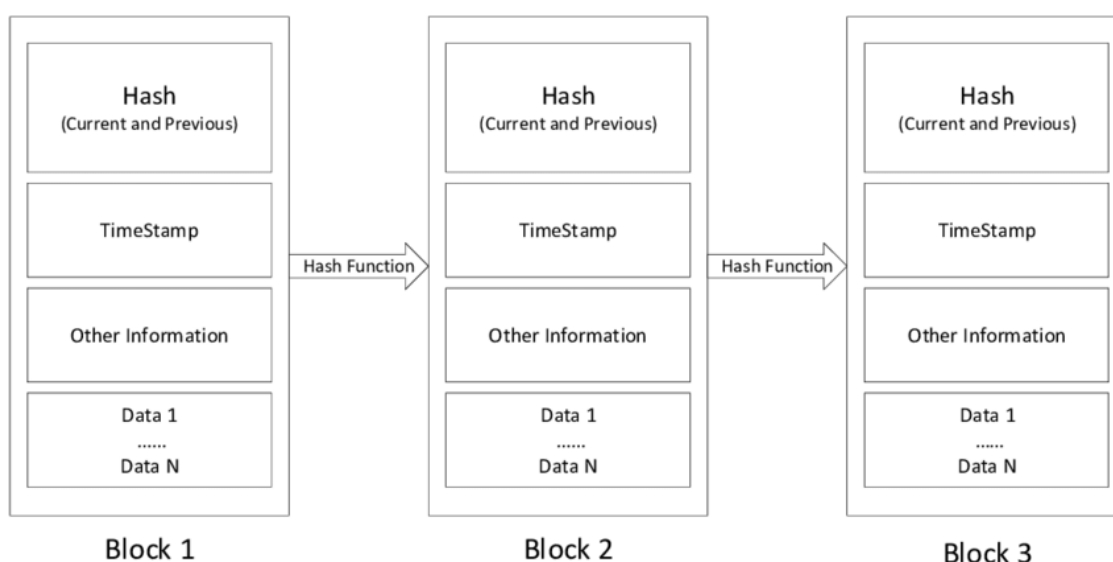
### 2.1.4 Сертификат

Електронски сертификат је електронски документ који издаје сертификационо тело. То је електронска потврда којом се потврђује веза између података за проверу електронског

потписа и идентитета потписника. Квалификовани електронски сертификат може да се схвати као дигитални идентификациони документ, јер садржи податке о кориснику сертификата као и податке о његовом издаваоцу, као што су информације о верзионисању, имену, временске печате валидности и најбитније јавни кључ и потпис самог сертификата. У пракси сертификати се користе за аутентификацију учесника и за размену сесијског симетричног кључа којим ће слати податке. Битно је напоменути да сертификати глобално формирају ланац поверења, тако да један сертификат може потписати као важећи неки други сертификат и тако до самог врха ланца где се налазе сертификати којима се по подешавању окружења верује.

## 2.2. Дефиниција блокчејна

Блокчејн је неограничено растућа листа података који се називају блоковима (колекције трансакција), угнежђеним криптографским механизмима тако да сваки блок садржи хеш вредност свог претходника у ланцу. Једна од кључних карактеристика јесте дистрибуираност саме структуре, сви учесници који желе да користе ланац уједно чувају и комплетну или редуковану верзију целе листе и заједничким договорима и протоколима одржавају сигурност мреже. Комуникација у мрежи је у потпуности P2P (*Peer-to-peer*) и не захтева постојање ауторитета од поверења за комуникацију чиме је сам систем у потпуности децентрализован.



Слика 2.2.1 Генеричка шема структуре блокчејна

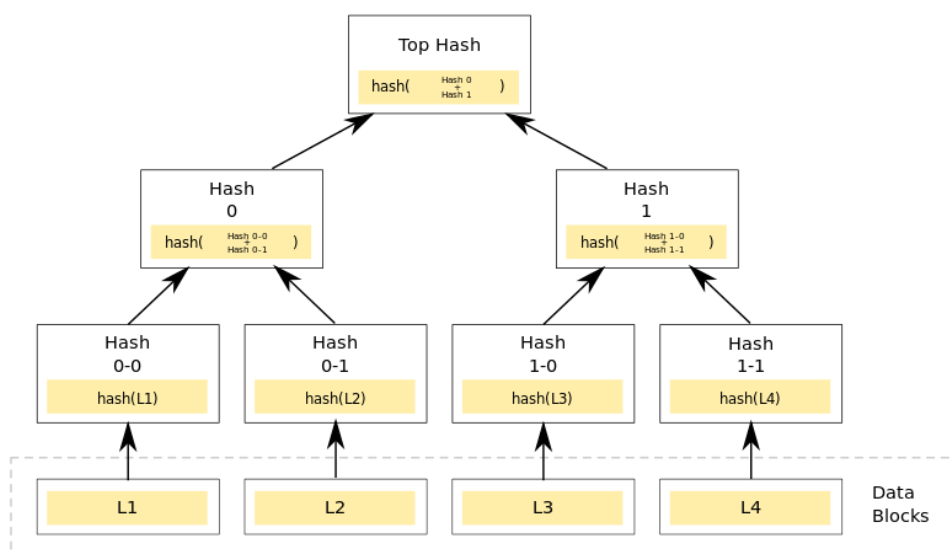


## 2.3. Настанак блокчејна

Псеудоним Сатоши Накамото (*Satoshi Nakamoto*) 2008. године објавио је рад и имплементацију првог блокчејна *Bitcoin*. Систем је био предвиђен да формира јединствену структуру за проток „до тада непостојеће, криптовалуте уједно елиминишући потребу за икаквим регулаторним телом у виду банке или државног тела. Учесници би алгоритмима консензуса одржавали сигурност, а надограђујући само мрежу (копањем нових блокова) били би аутоматски награђивани сумом криптовалуте.

## 2.4. Структура

Основна јединица структуре јесте сам блок. Блок као такав садржи колекцију својих трансакција формираних у Мерклово стабло. Сваки блок садржи свој хеш али и најважније хеш свог претходника у ланцу. Овим итеративним процесом успоставља се снажан интегритет ланца све до почетног *Genesis* блока.



Слика 2.4.1 Генерички приказ Мерклеовог стабла

Додатно сваки блок садржи и информације о временским жиговима када су створени. Трансакције се најчешће чувају у виду Мерклеовог стабла. Мерклеово стабло јесте стабло где сваки лист садржи хеш података складиштених у њему (у овом случају трансакције), док унутрашњи листови садрже хеш настао као хеш конкатенираних вредности хешева својих потомака у стаблу. На овај начин чува се и интегритет самих трансакција у оквиру једног блока, али се и омогућава складиштење мање верзије блокчејна. Неки учесници могу

памтити само вредности корена Мерклеовог стабла и тиме уштедети на локалном простору, али са друге стране губе одређене функционалности чворова који садрже пун ланац нпр. провера баланса једне адресе. У општем случају једна трансакција садржи адресе (јавне кључеве) учесника као и потписе формиране њиховим комплементим приватним кључевима као и податке које трансакција са собом носи (нпр. суму преноса криптовалуте)

## 2.5. Мрежа

Сваки учесник мреже који на локалном нивоу садржи копију целог ланца назива се чвором (енг. *node*). Чворови *P2P* комуникацијом размењују новокреиране блокове и складиште их у својим копијама. Не постоји централни ауторитет, тј. ниједном чвору се не верује да је његов ланац прави (каузално и структурално валидан), већ се алгоритмима консензуса сви чворови међусобно договарају око валидности ланца. Консензус се постиже различитим механизмима, али у суштини сваки алгоритам базиран је на међусобном емитовању информација о својим ланцима на основу којих се утврђује да ли је исти валидан, коначно ланци око којих се већина чворова сложи (потврди идентичност параметара са својим ланцем) бивају прихваћени као опште тачни. По општем правилу најтачнији је онај ланац који има највећу дубину (најдужа листа) блокова и притом јесте валидан.

Сваки чвор који креира блокове и верификује трансакције назива се рударем (енг. *miner*). Рудари су они чворови који пристигле трансакције верификују и формирају нови блок за смештање у ланац. Валидност блока се дефинитивно огледа у валидности хеша самог блока, али и у валидности додатог хеша његовог претходника, остали параметри валидности зависе од конкретног алгоритма консензуса. Као најрепрезентативнији пример дефинисаћемо *PoW (Proof of Work)* алгоритам: Копање једног блока овде првенствено налаже решавање процесорски захтеваног криптографског проблема. Сваки рудар добија задатак у виду проналажења инверзне вредности хеш функције за одређену карактеристику излаза, нпр. у општем случају треба се наћи  $M$  такво да је  $H(M) = X$  где је  $X$  променљива задатих карактеристика од стране мреже. Те карактеристике могу варирати и најчешће варирају тако да се сам процес решавања процесорски усложњава повећањем броја рудара нпр.  $X$  мора имати  $N$  нула. Повећавањем броја  $N$  расте и сложеност решавања проблема, чиме се постиже скалабилни раст копања, а тиме и раст самог блокчејна. Практично гледано што је проблем тежи (нпр. веће  $N$ ), више је потребно енергије (струје) да локални хардвер рудара нађе инверзан хеш. Рудар који први реши задати проблем и при томе валидно формира блок

исти емитује свим осталим чворовима мреже. Ако се већина сложи око валидности новоископаног блока рудар бива привилегован тиме да се његов блок уврсти у ланац сваког од чворова. Додатно у реализацији *Bitcoin* успешан рудар бива награђен сумом криптовалуте сразмерном тежини проблема који је успешно решио. Једноставно говорећи онај који је спреман да најбрже (највише) уложи у раст ланца бива награђен. Касније у раду биће анализе додатних алгоритама консензуса и умрежавања.

## 2.6. Последице дизајна

Оно што сама грађа блокчејна по дизајну омогућава јесте:

- Брзе трансакције – у банкарству процес преноса одређене вредности са једне адресе на другу траје релативно дуго због робусних и често дугих протокола верификације и аутентификације корисника уз посредовање трећег лица. У банкарским системима рачуни се одржавају као модели ентитета који имају своја стања у базама података система, приступи овим базама пролазе кроз неколицину провера сертификата и потврда идентитета што цео процес одужава. Када је у питању блокчејн једна трансакција само бива уграђена у одговарајући блок уз претходну проверу реализибилности претрагом по свим претходним трансакцијама које се односе на адресу пошаљиоца у циљу провере средстава. Немогуће или по протоколу невалидне трансакције бивају одбијене од стране свих доброћудних чворова.
- Непроменљиве трансакције – након што су додате у блок убрзо потом дати блокови бивају ископани чиме се перманентно утискују у ланац са практично немогућом шансом промене. Ако би неко желео да промени само једну трансакцију морао би то да учини прво налазећи инверзне вредности хешева за цео ланац тако да се трансакција нумерички уклапа, а онда то и реализовао над релативном већином (51%) учесника мреже.
- Поузданост – све трансакције са собом носе јавне адресе (јавне кључеве пошаљиоца и примаоца) са потписима њихових приватних кључева. Сигурност једне адресе лежи у сигурности чувања тајног кључа корисника.

## **2.7. Класификација**

Иако почетно замишљен као отворена и децентрализована мрежа, блокчејн је временом добио разне подтипове и селекције.

### **2.7.1 Јавни блокчејн**

Ланац без икаквих рестрикција приступа. Било ко са приступом интернету може постати чвор и рудар и учествовати у алгоритмима консензуса. Уобичајно овакве структуре имају уграђен систем награђивања за копање како би што боље искористили *PoW* алгоритам.

### **2.7.2 Приватни блокчејн**

Ланац са додатним забранама и рестрикцијама приступа. Чвор се постаје само по одређеном протокул, док је и сам приступ подацима углавном рестриктиван. Уобичајено је да приватне фирме ради својих интереса одржавања и сигурне администрације користе овакву структуру. Често се у литератури назива још и дистрибуираним регистром.

### **2.7.3 Хибридни блокчејн**

Комбинација претходне две структуре потпуно произвољног облика рестрикција. Овакав тип блокчејна је имплементиран у истраживању у оквиру ове тезе.

### **2.7.4 Споредни блокчејн**

Дигитална копија циљног блокчејна формирана на другачијим карактеристикама консензуса и технологији. Одржава се како би се јасно формирао независан клон главног ланца ради валидације и сигурније инфраструктуре.

## **2.8. Напади**

Иако доста сигуран блокчејн подлаже различитим типовима напада специфичних за дизајн система. Систем инхерентно по дизајну чини напад на централни ентитет немогућ јер таквог ни нема, но уочавају се другачији типови малверзација.

### **2.8.1 Малвер рудар**

Софтвер вирусног типа који на домаћину формира рудара таквог да све награде за успешно копање шаље на нападачев рачун.

### **2.8.2 *Напад 51%***

Ако би неки ентитет контролисао барем 51% укупне рударске снаге могао би да константим постизањем консензуса својом већином и кривотворењем трансакција или одбијањем валидних наруши интегритет ланца и у потпуности корумпира ланац. Срећом за велике мреже овај напад је доста тешко извести обзиром да потреба процесорска снага за напад расте порастом мреже.

### **2.8.3 *Лажне трансакције***

Теоријски је могуће, али изразито рачунарски тешко, прикрити малициозну трансакцију у блок, а потом га и успешно ископати. Да би реализовао овај напад нападач мора деформисати тренутни блок невалидном трансакцијом, тако да  $H(\text{Block}|\text{BadTransaction}) = H(\text{Block})$  и додатно при томе први реши хеш проблем.

### **2.8.4 *Крађа приватног кључа***

Овај напад је својеврсан пандан губитку кредитне картице. Учесник овиме дели идентитет са нападачем и могућно сва своја средства у ланцу.

## **2.9. Предности**

Нова технологија са собом носи велики број предности и погодности чије коришћење пружа. Махом све се односе на јаку структуру и тешко оборив интегритет платформе, али исто тако и на једноставност и неопорецивост трансакција система.

### **2.9.1 *Тачност ланца***

Велики број учесника одржава и формира ланац копањем, у случају великих система овај број достиже и милионе чворова. Имајући у обзир масовност мреже, која готово у потпуности аутоматизовано функционише, фактор људске грешке бива сведен на минималан. Са друге стране иако дође до рачунарске грешке у самом чвору, ова грешка ће испропагирати само кроз локални ланац посматраног чвора и евентуално у прави ланац (блокчејн већине) ако и само ако се идентична грешка појави на барем 51% свих чворова система.

### **2.9.2 Смањење трошкова**

Типично је за банкарске системе да се вид трансакција тј. извршавања трансакција наплаћује, или да се пак верификација документа наплаћује: нотари потписују и оверавају списе, матичари закључују брачну заједницу, итд. Са блокчејном ово једноставно није случај, не постоју централни ауторитет и самим тим су све трансакције и верификације над истим бесплатне.

### **2.9.3 Децентрализација**

Не постоји централна база података које чува логистику система и самим тим напад на једну јединицу постаје готово узалудан. Нападаци би у општем случају морали да инвалидирају барем пола чворова мреже док би код конвенционалних система нападали само један доман или у критичном случају само један сервер.

### **2.9.4 Ефикасност трансакција**

Трансакције јесу анонимне само до оне границе до које су и поверљиве, наиме аутоматизован процес потписивања и смештања трансакција са информацијама о адресама даје јасно раслојавање правог идентитета корисника (нпр. IP - *Internet Protocol* - адресе, корисничког имена платформе) од његовог кључа. Анонимност лежи у томе да адреса у виду јавног кључа указује на корисника само као учесника трансакције и ништа више. Лажно представљање је немогуће због спреге јавног и приватног кључа. Докле год корисник добро чува свој приватни кључ чува и апсолутни баланс својих трансакција, ослањајући се на математички модел који стоји иза теорије асиметричне екнрипције и механизма потписивања.

Додатно не постоји ни ограничење када се трансакције могу извршавати. Неке банке не раде викендом, неке државне управе не извршавају потписивања сваког дана у недељи, док је сервис који пружа блокчејн доступан сваког дана сваког месеца у години.

## **2.10. Мане**

Ни један система није савршен и сваки систем се може нападати са специјалних и за исти јединствених страна. Комплексна шема архитектуре и константа потреба за великим учешћем рачунарске снаге имају и своју цену.

### **2.10.1 Трошкови технологије и технике**

Иако све трансакције заиста јесу бесплатне, одржавање истих је генерално скупо. Валидација и копање једног блока може постати енормно енергетски скупа ако је степен симултаности мреже висок. *PoW* алгоритам може знатно повећати рачун за електричну енергију ако рудар користи велики број процесни јединица. Додатно и сама валидација блока од стране чворова који су блок примили, али не и ископали кошта и троши добру количину процесне снаге.

### **2.10.2 Нелегални системи**

Висок степен демократије и анонимности у једном блокчејну ни на који начин не зависи нити је ограничен законом неке државе. Не постоји уграђен начин да се одређене трансакције забране или инвалидирају, ако пак крше правни кодекс. Ово је доста искоришћено у трговини наркотицима и оружјем, како би се очувала анонимност учесника и ризик свео на минимум. Додатно могућ је вид прања новца трансформисањем зарађене реалне валуте са два рачуна исте особе у криптовалуту.

### **2.10.3 Напади**

Иако захтевају велику количину процесне моћи, напади су итекако могући. Ако нападач скупии довољно средстава (која се данас могу и изнајмљивати) апсолутно детерминистички и са очекиваним временом трајања може напасти неки блокчејн.

## **2.11. Примена**

Честа конотација до сада била је у светлу монетарног одржавања криптовалуте и опште употребе блокчејна ради извршавања преноса средстава, но ово није једина апликација ланца. Неминовно је то да би банкарска индустрија највише профитирала интеграцијом блокчејна у своју инфраструктуру, али су и остале гране индустрије успеле да искористе ланац.

### **2.11.1 Здравство**

Транспарентно и јасно складиштење картона пацијената, извештаја доктора и јавних набавки јесте сигуран начин да се побољша, аутоматизује и повећа ниво транспарентности једног здравственог система. Прегледи и рецепти могу одмах након издавања бити верификовани и постављени у ланац чиме се обарање и кривотворење чини узалудним. Од

оваквог система профитирали би сви корисници истог, али и институције и државе које систем регулишу.

### ***2.11.2 Информациони системи***

Честа је потреба модерних компанија да развију приватну инфраструктуру за чување и администрацију података. Јавни индекси потковани сервисирањем над блокчејном дају ефикасније и сигурније пословање фирме и смањује ризик од нелегалних радњи и искоришћавања, такође умањује се и степен људске грешке који потенцијално радници могу унети при верификацији и одржавању.

### ***2.11.3 Паметни уговори***

Теоријски гледано, могуће је пренети једну инфраструктуру нотарног система целе државе на сам ланац. Сви документи, све изјаве и уговори били би непроменљиво урезани у ланац, са веома малим трошком одржавања и још мањом могућности корупције, ако се систем добро аутоматизује. Један уговор био би смештен са свим информацијама о евентуалном плаћању и евентуалном извршењу. На пример ако би се корисник А паметним уговором обавезао да кориснику Б исплати одређену суму до одређеног датума у сврху издавања простора за живот, систем би могао да аутоматски поништи уговор ако сума до задатог датума не буде уплаћена.

### ***2.11.4 Гласање***

Блокчејн потенцијално може да отклони све проблеме са којима се данашњи системи гласања носе. Лажни гласови, намештање изборних резултата били би неизводљиви над овом структуром. Сви учесници би транспарентно и по дефинисаним протоколима гласали без икакве могућности да свој глас понове или пак инвалидирају, нити да трећа страна поништи њихов глас. Јасна пребројавања и баланси не остављају простор људском интересу да подмеће невалидне информације и гласове. У наставку тезе следи детаљан преглед и анализа ове примене уз давање аргумената за формирање спецификације и различите могуће начине имплементације система.

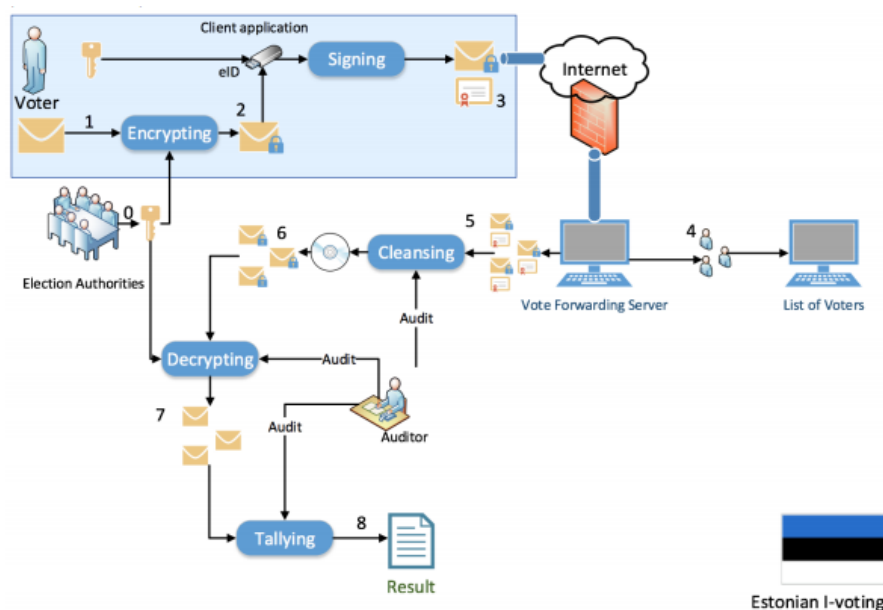


## 3. ПОСТОЈЕЋА РЕШЕЊА

Већ постојећи системи за гласање базирани на блокчејну у сржи користе исту идеју, а то је да се ланац искористи како би се обезбедила сигурност гласања и цео процес аутоматизовао. Апстракције варирају од пројекта до пројекта, али и од величине тела за који се исти реализује. Кратак опис посматраних система дат је у наставку.

### 3.1. Естонијски модел

Иако није базиран на дистрибуираном ланцу, пример директно указује на кључне ставке које инфраструктура мора обезбедити. Естонија је 2007. године била прва земља у свету која је увела гласање путем интернета. У 2015. години, 30,5% свих гласова је прикупљено електронским путем. Основа система лежи у националним идентификаторима који сви грађани Естоније добијају. Идентификатори садрже шифроване податке које власнику допуштају бројне сервисе, попут приступа банковним рачунима за плаћања, дигиталним потписом, приступ државним базама података и у овом случају електронском гласању. Овај идентификатор можемо сагледати као пар кључева који сваки потенцијалн гласач већ има.



Слика 3.1.1 Приказ естонијског модела

Процес гласања почиње пре свега онлајн регистрацијом за исто, корисник уноси креденцијале идентификатора преко читача картица или мобилног телефона и проверава статус гласања и да ли је за исто добио дозволу да гласа. Након потврде, гласач има четири дана да гласа и евентуално промени свој глас. Када се овај период заврши гласови се шаљу централној бази. Потом што се прикупе сви гласови, сервер који је прикупио податке прво, елиминише личне податке гласаче обезбеђујући анонимност, снима садржај на оптичке дискове и коначно брише целу базу. Прикупљени оптички дискови се физички шаљу на посебан сервер који сумира податаке и даје коначан извештај. Битно је напоменути да је сервер бројач у потпуности искључен са било које мреже и заштићен од стране државе. Систем је на крају гласања изанализиран као рањив обзиром да су машине гласача у потпуности биле остављене не заштићеним као и то да су се оптички дискови могли заразити непожељним злонамерним софтвером (енг. *malware*).

### 3.2. Плимут модел

Овај модел развили су професори Универзитета Плимут у Уједињеном Краљевству 2016. године. Процес гласања почиње регистрацијом, а како би систем обезбедио аутентификацију корисника, формирана су два модела приступа један поштанским путем, и други веб приступом. Регистрација заједно са свим креденцијалима и додатним дескрипторима бива прослеђена на ТИП1 блокчејн као вид уговора који касније бива ископан од стране владиних рудара. Рудари поред копања такође и лично верификују трансакцију (регистрацију) одобравајући или одбијајући грађанину право на гласање. Ископана верификација бива урезана у ланац, а формира се и додатна трансакција за ланац ТИП2 у којој се адреси посматраног гласача даје један токен/криптовалута/глас на употребу. Гласање у пракси само конзумира овај токен.

Сама мрежа подељена је на три апстрактна типа реона: национални, регионални и локални. Локални реон садржи гласачке станице чворова које комуницирају само са станицама у свом регионалном реону. Регионални реон предстаља подмрежу свих станица за гласање неке административне целине државе и своје интерне чворове. Национални реон предстаља целу мрежу заједно са националним чворовима. Систему су потребни и независни сервери који исти и надгледају преко националног чвора верификујући све трансакције и копајући ланац. Трансакције пропагирају од гласачке станице, па све до националног чвора коначним копањем. Блокови се додатно чувају шифрованим симетричним кључем од стране

сервера који га је ископао, те иако нападач на неки начин успе да пробије један регионални сервер неће моћи да прочита све податке већ само оне које је посматрани сервер ископао.

### **3.3. Рејкјавик модел**

Модел развијен од групе аутора на Рејкјавик Универзитету на Исланду представља јединствен приступ решавању овог проблема. Формиране су улоге у систему такве да омогућавају протокол гласања, администратори надгледају и управљају процесом гласања, гласачи гласају, покрајинске чворове који прикупљају гласове и копају ланац као и бут (енг. *Boot*) чворове који покрајинским омогућавају реализацију протокола упознавања и придруживања мрежи.

На почетку гласања сваки покрајински чвор добија свој паметни уговор који бива уграђен у ланац, овиме се датом терминалу допушта да врши обраду трансакција и приступа мрежи. Администратор дели ове уговоре и са њима сваком чвору шаље и листу гласача у виду блока. За сваког гласача бива изгенерисан пар кључева, у овом случају његов виртуелни новчаник директно повезан са паметним уговором покрајинског чвора. При регистрацији и добијању кључева, гласач задаје глас који у виду трансакције која пропагира кроз ланац, али само са додатно валидним кључем паметног уговора који је додељен његовом покрајинском чвору.

## 4. СПЕЦИФИКАЦИЈА СИСТЕМА

Пре него што се сам систем моделује битно је саставити детаљну спецификацију и поставити јасне циљеве реализације платформе. У овом поглављу претежно се наводе претпоставке везане за формирање случајева употребе, као и конкретног модела којим се посматрани случајеви могу реализовати. Као главне карактеристике спецификације узете су за разматрање оне које се намећу и у неелектронским гласањима, а представљају основ једног демократски слободног гласања:

- Систем мора обезбедити потпуну анонимност гласача. Тежи се ка томе да је по дизајну немогуће ући у траг гласачу и његовом гласу.
- Глас мора бити у потпуности аутентичан и сигуран.
- Висок степен заштите мреже са малом вероватноћом пробоја.
- Пребројавање гласова се не препушта једном ентитету већ је процес у потпуности децентрализован.
- Глас може дати само онај корисник који је на исто и позван, тј. мора постојати механизам верификације гласача за посматрано гласање.
- Немогућност поновног изласка једног гласача на исто гласање за које је позван.

### 4.1. Тип ланца

Ако би за реализацију усвојили структуру јавног блокчејна, свако ко потенцијално жели могао би да учествује у мрежи као чвор и рудар, ово би се за посматрани систем могло имплементирати тако да или свако ко жели да гласа мора и да чествује у одржавању. Јавна инфраструктура првенствено своју сигурност одржава у тежини криптографског проблема израчунавања инверзног хеша те би се од сваког корисника одузимала процесна моћ ради учествовања у платформи. Додатно овакав начин би експлоатисао енергију корисника заузврат не дајући им ништа осим саме сигурности, олакшица би можда била у томе да систем награђује копања. Ако би награда била токен за гласање цела платформа би се знатно успорила због наметнутог правила да свако ко жели да гласа мора прво да ископа један блок, додатно би се и сами гласачи/рудари утркивали за право на глас што и практично нема

смисла. Друга опција била би да је наградни токен право на формирање једног гласања, но ово поново само смањује моћ мреже да одржава и валидира гласања. Закључује се да јавна структура није оптимални модел за овакав проблем, првенствено из нефер црпљења корисничких ресурса за прикупљање информација, али и због потенцијалног загушења мреже које је честа појава код јавних блокчејнова. Те је битно искористити хибридно решење. Хибридизација се огледа у формирању протокола за добијање софтвера чвора и сертификацију ентитета да може као добронамеран члан мреже да чува и обезбеђује ланац. Протокол за добијање софтвера би био следећи:

- 1) Ентитет подноси захтев за сертификацију власницима платформе.
- 2) Након валидације ентитета он добија софтвер чвора, кључ за приступ мрежи као и сертификат којим представља домен мреже ланца.
- 3) Ентитет покреће софтвер и активно учествује у одржавању блокчејна.

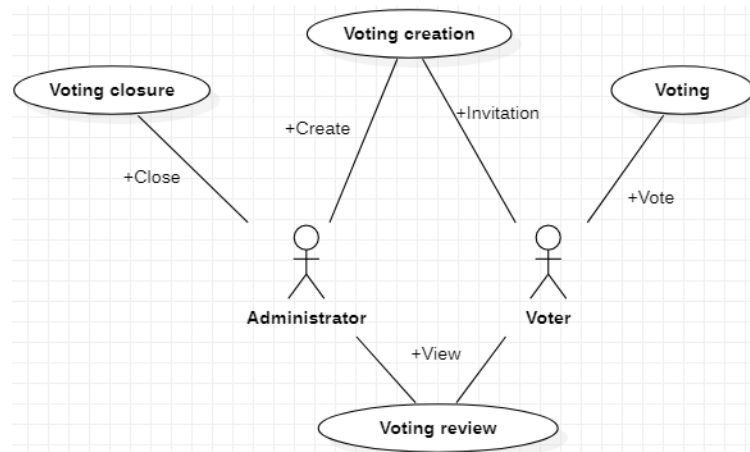
Овај поступак за проширење мора бити спровођен по јасним и конкретним проверама како би се вероватноћа малверзације свела на минималну. Сви чворови би у најидеалнијем случају представљали домен институције или фирме која платформу одржава без додатног допуштања ентитетима ван домена да учествују у мрежи, но сама инфраструктура ово не мора експлицитно да забрани.

## **4.2. *Proof of Authority* алгоритам**

Од чворова се не очекује да решавају криптографски проблем, већ се степен сигурности ланца и његове веродостојности директно пребацује на част и одговорност сертификованих рудара. PoW алгоритам изискује релативно велику количину струје за већ постојне вредности само у циљу добијања награде и успоравања процеса копања у једном ланцу, обзриом да ове карактеристике нису пожељне за овај тип система PoA алгоритам даје боље решење. Сваки сертификовани чвор приликом придруживања мрежи добија и пар RSA асиметричних кључева којима ће потписивати сваки блок који ископа. Сви остали чворови могу лако провером верификоати бродкастовани блок и уверити се у добронамерност рудара. Недефинисано понашање тј. подметање невалидних трансакција и корумпираних блокова резултује избацивањем рудара из мреже и одбијањем блока. Тим поступком би ентитет изгубио своју сертификацију и могућност да даље учествује у мрежи. Процес је лако аутоматизовати тако да се на одређено време прозивају различити чворови мреже да ископају тренути блок, чиме се додатно и тестира њихова добронамерност.

### 4.3. Улоге

Дефинисаћемо две улоге, једна је улога администратора система који гласања отвара, позива гласаче и коначно затвара гласања. Гласач је улога која добија позив за гласање и на исто евентуално излази (гласа).



Слика 4.2.1 Дијаграм употребе система

Сам начин стварања улоге јесте слободан за тип гласача, од њега се при регистрацији траже креденцијали и адреса електронске поште. Ради сигурније аутентификације самих гласача покретач система би могао да у исти допушта само регистрације са мејловима из свог службеног домена како би нпр. само стварни чланови институције/фирме могли да постану активни учесници. Додатно, добро би било оставити администраторску улогу заштићеном од стварања преко веб форми већ да се скуп администратора формира пре него што се сама платформа покрене. Овиме се систем штити од потенцијално лажних администратора и евентуалног стварања невалидних администраторских ентитета.

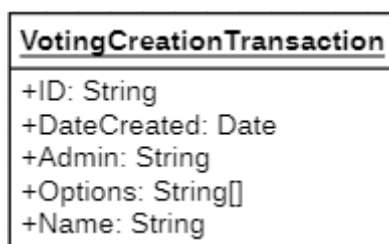
## 5. МОДЕЛ СИСТЕМА

У овом поглављу у жижу се ставља реализација и конструкција самог система. Детаљно ће бити описана структура, архитектура, протоколи и сви сервиси који платформа треба да омогући ради испуњавања спецификације.

### 5.1. Структура

Сама функција блокчејна јесте да се инфраструктура система интегрише са сигурносим механизмима ланца. Блокчејн би требао да омогући транспарентно и сигурно смештање сих потребних информација о гласањима. Дефинисаћемо четири типа трансакција за дати ланац:

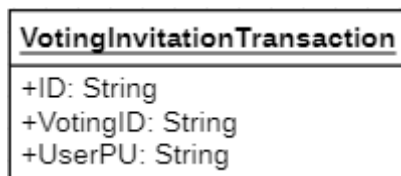
**Трансакција стварања гласања** – трансакција се формира након обраде администраторског захтева за стварање гласања уз све потребне податке. Приликом обраде захтева сви позвани гласачи добијају пар асиметричних кључева као позив за дато гласање. Ова трансакција бива смештена у блок који ће непосредно бити ископан ради веће сигурности.



Слика 5.1 Објектни дијаграм трансакције стварања гласања

- *ID* – Идентификатор трансакције који је уједно и идентификатор новокреираног гласања
- *DateCreated* – Датум отварања гласања
- *Admin* – Корисничко име администратора који је отворио гласање
- *Options* – Низ опција за дато гласање
- *Name* – Назив датог гласања

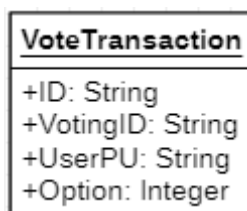
**Трансакција позива на гласање** – апстрахује везу између једног јавног кључа (гласача) и гласања за које је позив упућен. Ово је исти кључ који се формира у обради трансакције стварања гласања. Све трансакције овог типа везане за позиве једног гласања бивају смештени у блок који се непосредно ископава ради што боље сигурности.



Слика 5.1.1 Објектни дијаграм трансакције стварања гласања

- *ID* – Идентификатор трансакције
- *VotingID* – Идентификатор посматраног гласања на које се упућује позив
- *UserPU* – Јавни кључ генерисаног пара кључева позваног гласача у PEM (*Privacy-Enhanced Mail*) формату.

**Трансакција гласа** – представља извршавање једног гласа. Гласач доставља сервису свој јавни кључ, идентификатор гласања као и дигитални потпис формиран упареним приватним кључем. Трансакција након смештања у блок бива такође непосредно ископана.

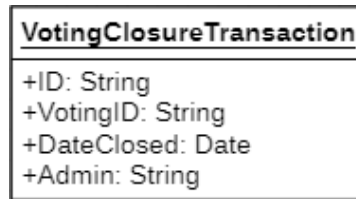


Слика 5.1.2 Објектни дијаграм трансакције гласа

- *ID* – Идентификатор трансакције
- *VotingID* – Идентификатор посматраног гласања на којем се гласа
- *UserPU* – Јавни кључ генерисаног пара кључева позваног гласача у PEM (*Privacy-Enhanced Mail*) формату.
- *Option* – Редни број опције за коју се гласа из низа опција које су задате за посматрано гласање.

**Трансакција затварања гласања** – својеврстан комплемент трансакцији стварања гласања. Администратор затвара гласање и тиме онемогућава даље трансакције гласова. Трансакција бива директно ископана у блоку након обраде захтева.





Слика 5.1.3 Објектни дијаграм трансакције гласа

- *ID* – Идентификатор трансакције
- *VotingID* – Идентификатор посматраног гласања које се затвара
- *DateClosed* – Датум затварања гласања
- *Admin* – Корисничко име администратора који је гласање затворио

Последице формирања структуре трансакција су следеће:

- 1) Анонимност – нигде се у систему не чува директна веза између идентитета гласача и евентуалног гласа. Кључеви за позив бивају генерисани унутар радне меморије и сигурно послати на сигуран начин датом кориснику. Ланац само памти јавни кључ као адресу која има право на глас.
- 2) Немогућност понављања – немогуће је искористи једну позивницу за више гласова. Једном искоришћен јавни кључ бива перманентно урезан у ланац и све остале употребе истог бивају одбациване као невалидне.
- 3) Децентрализован регистар – не постоји централно тело које гласове прикупља, сви гласови чувају се дистрибуирано на ланцу те обраћање било ком чвору на упит о стању гласања даје идентични одговор.
- 4) Интегритет система - сваки тип трансакције бива непосредно ископан у своме блоку након пристизања захтева. Овиме се смањује могућност инвалидације пристиглих трансакција или евентуалних модификација.
- 5) Аутентичност – само прави власник приватног кључа упареног за јавним кључем из позива на гласање може искористити позив због механизма дигиталног потписивања.

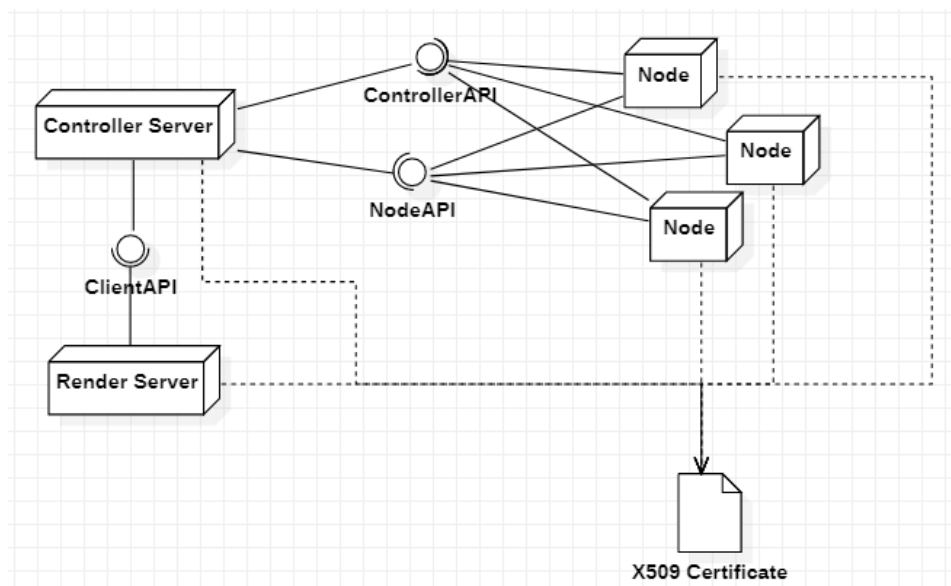
Овиме се по дизајну омогућавају основне тачке спецификације. Потребно је још дефинисати архитектуру платформе и осигурати комуникације у систему али и ка систему. Додатно како би се јасно раздвојиле информације о самим гласовима и о стварању и затварању гласова структура ланца је подељена на два подланца. У један *voteChain* смештају

се трансакције типа трансакције гласа, док се у *invChain* смештају све остале. Ова модификација омогућава лакше одржавање платформе.

## 5.2. Архитектура

Под описом архитектуре сматрају се начини повезаности и имплементација комуникационих линкова које ентитети мреже остварују. Било би пожељно раздвојити интерфејс услуге коју неки учесник нуди од самог учесника, тј. омогућити да се мрежи приступа као сервису преко интернета независно од саме технологије на којој су сервери имплементирани и постављени. Оптималан модел за форматирање оваквих серверских софтера јесте *REST* (*Representational state transfer*).

*REST* модел може да примењује и *HTTP* (*Hypertext Transfer Protocol*) у циљу размене информација текстуалног типа у нотацији *JSON* (*Javascript Object Notation*). Овај тип сервера подразумева отварање портова сервера ка спољашности преко имплементираних рута које засебно означавају једну услугу самог софтвера и тиме формирају апликативни програмски интерфејс (енг. *Application programming interface*, скр. *API*) сервера. Управо преко формираних *API* комуницирају сви учесници система. Актери из спољашњости преко веб клијента приступају рутама контролног чвора преко *ClientAPI* док се међусобна комуникација контролера и чворова реализује *ControllerAPI* и *NodeAPI* рутама.



Слика 5.2.1 Архитектура система

*Render Sever* представља сервер генератор страница и скрипти које се извршавају као клијентски код и има улогу да омогући корисницима јасан и прилагодљив графички интерфејс.

*Controller Server* представља сервер као приступну тачку систему, њему се обраћа сервер генератор као би уградио позиве рута у скрипте и задао захтев за неком услугом, контролер све захтеве преусмерава балансирањем на *Node* сервере који имају улогу чвора ланца мреже и теројски их може бити произвољно много. Такође преко контролера чворови приступају мрежи упознавајући се и успостављајући међусобно поверење.

Битно је напоменути да је поверљивост комуникације остварена путем облика *HTTP* протокола такозваног *HTTPS* (*Hypertext Transfer Protocol Secure*) протокола. У суштини *HTTPS* захтева постојање дигиталног сертификата коме иницијатори комуникације верују. Јавним кључем сертификата се успоставља комуникација ка серверу тако што клијент јавном енкрипцијом серверу шаље сесијски симетрични кључ којим ће даља комуникација бити остваривана. Нужно је да сви сервери мреже приликом покретања имају овај сертификат као и приватни кључ истог. На овај начин се штити поверљивост свих комуникационих веза унутар система али и ка систему. Свака линија на шеми заштићена је *HTTPS* протоколом, али такође и клијентски веб претраживач мора веровати посматраном сертификату како би комуникација уопште била могућа. Сертификат у суштину представља доменски сертификат ове мреже, цела мрежа би у идеалном случају предстаљала један домен под истим сертификатом којем се актери обраћају за услуге.

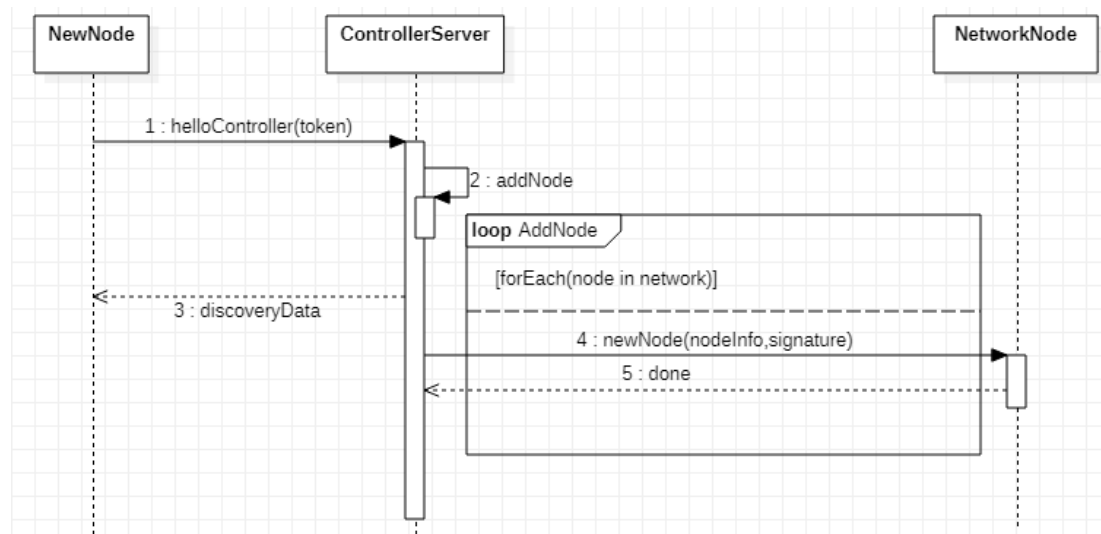
### 5.3. Протоколи

Конструкција начина комуникације по јасно дефинисаним корацима је битна како би систем могао да функционише на одржив и сигуран начин. Протоколи се или односе на саме протоколе унутар мреже чворова или између чвора и контролера или између клијента и контролера.

#### 5.3.1 Протокол упознавања

Након што је ентитет сертифициван за пуноправног чвора учесника мреже мора проћи кроз протокол упознавања у сврху добијања листе свих тренутних чворова у мрежи али и ажурирања листи осталих чворова. Приликом сертификације ентитет добија URL (*Uniform*

*Resource Locator*) адресу контролног сервера, сертификат са приватним кључем домена као и симетрични кључ као лозинку приступа.



Слика 5.3.1 Дијаграм секвенце протокола упознавања

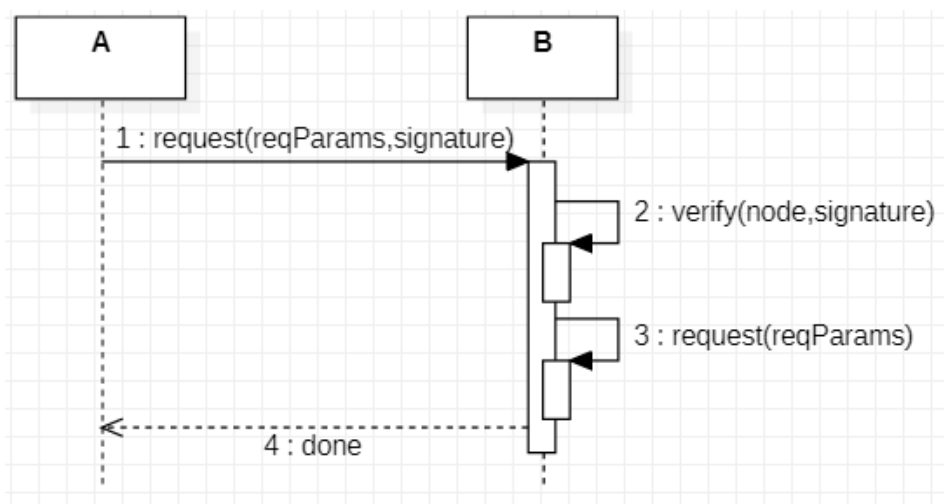
- 1) *helloController(token)* – ентитет шаље сертификацијом добијену приступну поруку шифровану симетричним кључем такође добијеним при сертификацији. Сервер добија овај податак у виду  $E(secret, K_s)$  где је *secret* предефинисана приступна порука. Ако контролни сервер верификује да је  $D(token, K_s) = secret$  наставља протокол. На дијаграму дат је валидан ток приступања. Овим токен штити приступ мрежи, тако да мрежи могу приступити само сертификовани корисници. Дати токен би било пожељну мењати од стране администратора платформе тако да се евентуални пробој може избећи
- 2) *addNode* – контролер анализира и памти IP адресу са кога је пристигао захтев као URL адресу на којој се отворен нови сервер чвора. Креира пар асиметричних RSA кључева које ће новоприспирани чвор користити за аутентификацију себе и блокова које ископа и коначно додељује идентификатор чвору.
- 3) *discoveryData* – новом чвору шаљу се креирани јавни и приватни кључ, идентификатор као и листа свих тренутно присутних чворова у мрежи са садржајем њихових адреса, идентификатора и јавних кључевима.
- 4) *newNode(nodeInfo, signature)* – сваком чвору који је био у мрежи пре прикључивања новог се шаљу адреса, идентификатор и јавни кључ новоприспиралог како би проширили свој локални регистар чворова у виду *nodeInfo* објекта. Потписом

*signature* се контролер аутентификује чвору, ово ће бити детаљно образложено у следећем протоколу.

5) *done* – потврда обраде

### 5.3.2 Протоколи аутентификације

Свако обраћање контролера чвору, свако обраћање чвора контролеру или чвору ради извршавања неког захтева пролази процес аутентификације механизмом дигиталног потписивања одређеним приватним кључевима носиоца тј. иницијатора комуникације. Захтев је било која порука која се може остварити између два учесника било да је то наредба додавања новог чвора неком чвору од стране контролера, преусмеравање захтева клијента за бродкастовањем трансакције или пак копање ланца.



Слика 5.3.2 Дијаграм секвенце генеричког захтева ентитета А ка ентитету Б

Дата је анализа секвенце свих могућих секвенци по улогама (А,Б)

(Контролер,Чвор) – контролер шаље предефинисану контролерску поруку потписану својим приватним кључем из сертификата тј. шаље се  $E(M, Kpr)$  где је  $M$  генеричка порука а  $Kpr$  приватни кључ сертификата. Чвор верификује потпис и обрађује захтев.

(Чвор, Контролер) – чвор шаље свој идентификатор као  $E(ID, Kpr)$  где је  $ID$  идентификатор додељен чвору при прикључивању у мрежу а  $Kpr$  његов приватни кључ такође додељен при прикључивању. Контролер прво гледа адресу са које је захтев стигао, асоцира адресу преко регистра чворорова на потенцијални идентификатор који представља

адресу и ову добијену вредност користи како би верификовао чвор и коначно наставио са обрадом захтева.

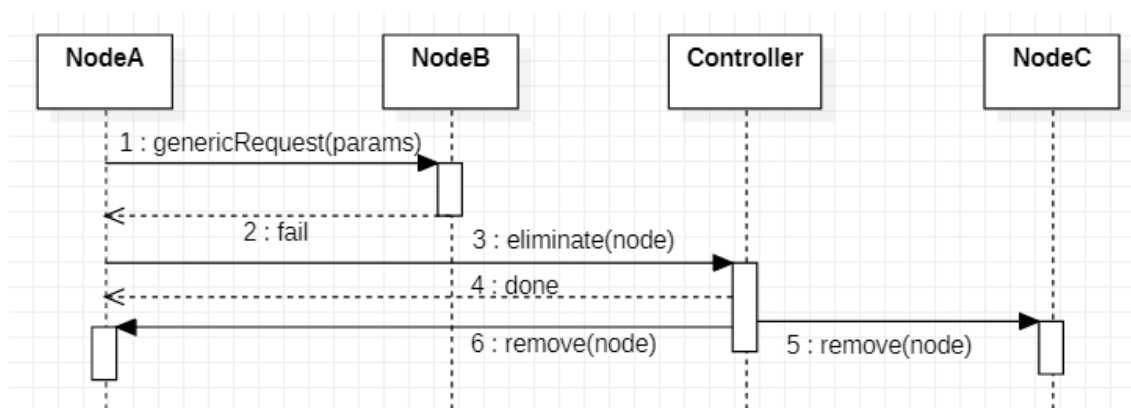
(Чвор, Чвор) – процес тече идентично као и у претходној комбинацији обзиром да сви чворови и контролер имају идентични регистар чворова.

### 5.3.3 Протокол одржавања

Контролер циклично прозива све чворове мреже са одређеном учестаношћу у циљу провере њихове живости. Једноставним одговором на захтев чвор потврђује да је активан и спреман на обраду.

### 5.3.4 Протокол елиминације

Јасно дефинисани протоколи имају гране грешке, тј. онај део контроле кода који изазива неочекивано понашање као што је грешка при обради захтева, неаутентификован приступ или одбијање поступања по предвиђеном протоколу. Сви ови случајеви јесу довољан разлог да се чвору више не верује и да се исти избаци из мреже. Чвор може пријавити други чвор контролеру на елиминацију или га и сам контролер може елиминисати.



Слика 5.3.3 Дијаграм секвенце генеричког захтева чвора А ка чвору Б уз додатну елиминацију

- 1) *genericRequest(params)* – чвор А шаље захтев чвору Б нпр. као захтев за скледиштење новокреиране трансакције.
- 2) *fail* - у чвору Б долази до грешке обраде захтева тј. непоступања по протоколу
- 3) *eliminate(node)* – контролеру се шаље идентификатор чвора кога треба елиминисати и он га избацује из регистра чворова мреже и самим тим из мреже
- 4) *done* – потврда обраде

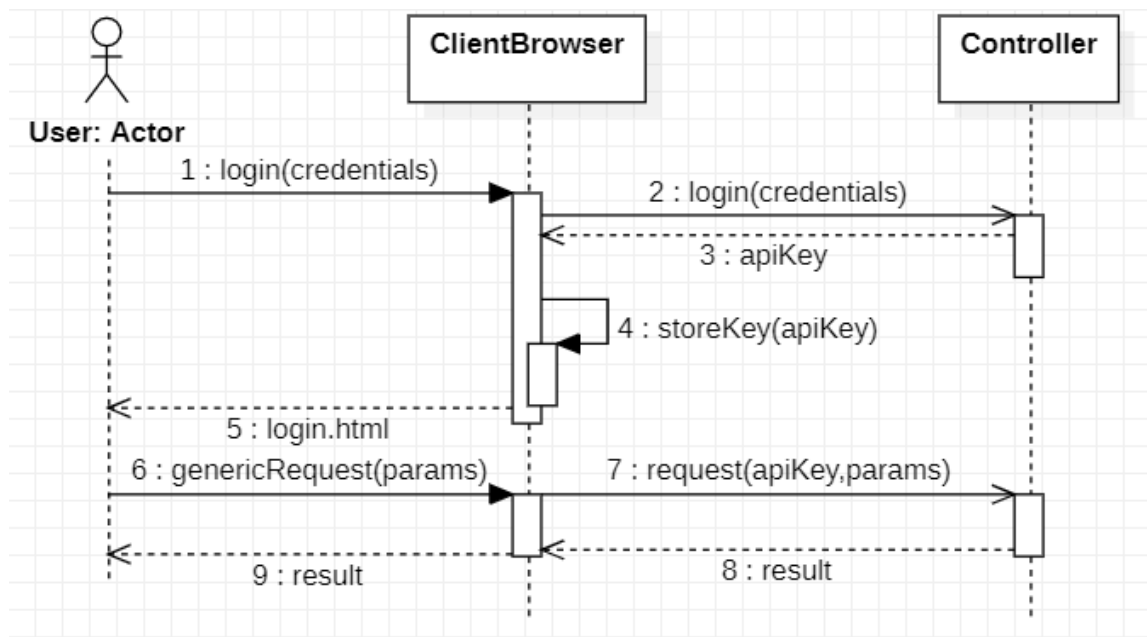
- 5) *remove(node)* -контролер шаље свим осталим и даље валидним чворовима наредбу за брисањем корумпираног чвора Б

### 5.3.5 Протокол консензуса

Дефинише начин успостављања договора у мрежи који је ланац валидан тј. шта представља тренутно исправан блокчејн. Проток се одвија над једним чвором тако што прво прозове све остале чворове ради добијања информација о њихоим ланцима. Те информације користи за валидацију самог ланца, проверавајући све хешеве и потписе блокова. Ако се његова верзија разликује од најдужег валидног лана он свој и ажурира. Ова функционалност се користи при покретању чвора одмах након пролаза кроз протокол упознавања како би добио актуелну верзију блокчејна.

### 5.3.6 Протокол ауторизације актера

Клијентски API се штити од приступа неауторизованих захтева механизмом издавања кључа приступа систему.



Слика 5.3.4 Дијаграм секвенце добијања кључа интерфејса и приступања систему

- 1) *login(credentials)* – корисник кроз интернет претраживач задаје корисничко име и лозинку
- 2) *login(credentials)* – креденцијали се предају контролном серверу

- 3) *apiKey* – сервер враћа кључ за приступ интерфејсу система у виду дигиталног потписа корисничког имена приватним кључем сервера формираним за ову функционалност
- 4) *storeKey(apiKey)* – браузер складишти кључ за даљу употребу
- 5) *login.html* – кориснику се приказује почетна страна система
- 6) *genericRequest(params)* – генерички захтев који корисник може послати систему преко веб клијента
- 7) *request(apiKey,params)* – интернет претраживач аутоматски пакује и претходно добијен кључ интерфејса за приступ као и прослеђене параметре
- 8) *result* – контролер верификује кључ интерфејса и шаље одговор

## 5.4. Функционалности система

До сада су посматрани конструкти који омогућавају сигурност саме мреже као и успостављање комуникационих веза. У овом потпоглављу акценат се ставља на све сервисе које инфраструктура пружа подељене по серверима и њиховим рутама. Битно је напоменути да контролер врши расподелу захтева техникама балансирања над доступним чворовима мреже класичним RR (*Round Robin*) алгоритмом. Сваки захтев од клијента који прими преко *ClientAPI* који садржи обраду над ланцем контролер преусмерава на следећег по реду из RR реда за обраду. RR се модуларно помера по регистру свих чворова тако да сваки следећи захтев обради други чвор по реду приспећа у мрежу, овај начин расподеле посла узет је јер сви чворови обављају идентичне захтеве сличних дужина трајања. Формат руте је дат као `< тип метода > /< путања > ( < аргументи > )`

### 5.4.1 *ClientAPI* руте

- 1) POST /registration (username,password,name,surname,email)

Представља захтев за креирање гласачког налога са прослеђеним аргументима. Корисничка имена и електронске адресе су јединствени на нивоу логистичке базе коју одржава контролер.

- 2) GET /login (username,password)

Представља улазну тачку у платформу и као резултат враћа клијентов кључ за приступ интерфејсу у случају успеха пријаве на систем.

- 3) POST /add\_voting (name,options[],usernames[],apiKey)



Представља обраду захтева стварања гласања. Захтев прво пролази кроз верификацију кључа интерфејса и проверу да ли је дати кључ везан за администраторску улогу. Након тога следе комуникације са ланцем. У invChain смешта се трансакција о креирању гласања и она бива непосредно ископана са својим блоком. Након тога се такође у invChain смешта N трансакција позива на гласање где је N број позваних корисника са информацијама о идентификатору гласања и генерисаним јавним кључем, клијенту се на приложеноу електронску адресу шаљу кључеви на сигуран начин. Ове трансакције након смештања у блок бивају ископане. Коначно у voteChain бива урезано N трансакција са вредностима идентификатора гласања као и претходно генерисаним јавним кључем за сваког корисника. Кориснику се враћа извештај о успешности операције.

4) GET /voting\_options (votingID,apiKey)

Представља наредбу којом клијент након верификације кључа интерфејса добија опције за гласање везано за прослеђени идентификатор гласања.

5) POST /add\_vote (votingID,PUKey,signature,option,apiKey)

Прво се проверава кључ интерфејса. Позвани гласач задаје идентификатор гласања, сигурно достављен јавни кључ, опцију потписану у достављеним упареним приватним кључем као и саму опцију. Дигиталним потписивањем се осигурава да је глас извршен заиста од стране гласача који је позван. Након верификације јавног кључа провером постојања трансакције позива која везује прослеђени јавни кључ и идентификатор гласања проверава се и валидност потписа као и саме опције. Такође се проверава да ли је дати кључ можда већ искоришћен. Ако све провере прођу и ако је гласање и даље актуелно у voteChain бива непосредно урезана трансакција гласа са свим потребним информацијама ископавањем њеног блока.

6) POST /close\_voting (votingID,apiKey)

Поново се проверава кључ интерфејса и да ли је дати кључ везан за улогу администратора. Након тога ако постоји још отворено гласање везано за прослеђени идентификатор оно се затвара додавањем трансакције затварања гласања у invChain коначим копањем блока у који је иста смештена.

7) GET /voting\_results (votingID,apiKey)

Након провере кључа интерфејса кориснику се, ако постоје, шаљу резултати гласања у виду низа бројача за сваку опцију гласања.

#### 8) GET /usernames

Рута за администраторску улогу која олакшава креирање гласања и формирање листе званица. Кориснику се враћа листа свих корисничих имена система.

### 5.4.2 *ControllerAPI пуме*

#### 1) POST /node\_registration (secret)

Први корак у протоколу упознавања јесте приступ посматраној рути. Новокреирани чвор шаље шифровану предефинисану тајну како би се успешно регистровао у мрежи и доказао да јесте сертифициован идентитет. Након провере добија регистар свих чворова мреже, док се осталим чворовима шаљу његове информације (адреса, идентификатор и јавни кључ)

#### 2) POST/eliminate (nodeID,authToken)

Рута за обраду елиминације неког чвора из мреже. Контролер прима идентификатор чвора који треба избацити ако аутентификација преко прослеђеног токена подносиоца захтева прође успешно.

### 5.4.3 *NodeAPI пуме*

#### 1) POST /add\_node (nodeID,nodeURL,nodePU,authToken)

Чвор мреже за време обраде овог захтева након верификације контролера преко токена додаје идентификатор, адресу и јавни кључ новопридруженог чвора мреже.

#### 2) POST /add\_transaction (chainType,data,authToken)

Обрада захтева ако верификација чвора преко токена прође резултује додавањем нове трансакције у ред трансакција још неископаног блока.

#### 3) POST /broadcast\_transaction (chainType,data,authToken)

Овај захтев чвор прима од контролера и након неопходне аутентификације шаље захтев свим осталим чворовима као и себи гађајући /add\_transaction руту само са различитим токеном за аутентификацију.

#### 4) POST /mine (chainType,authToken)

Захтев се шаље следећем у реду чвору од стране контролера како би извршио копање тренутног блока. Након што формира и потпише блок, чвор га бродкастује свим осталим чворовима мреже гађајући /add\_block руту.

5) POST /transact\_and\_mine (chainType,data,authToken)

Овај рута се такође захтева од стране контролера и обухвата уједињено понашање додавања трансакције и непосредног ископавања блока у који је иста смештена. Ова функционалност се користи када се жели да једна трансакција буде урезана што је пре могуће.

6) POST /add\_block (chainType,block,authToken)

Чвор прима блок од чвора који је блок ископао. Верификује се и сам чвор али и блок, проверавају му се хеш и индекс као и валидност потписа. Ако све провере прођу блок се додаје у локалну верзију ланца.

7) POST /eliminate (nodeID,authToken)

Руту погађа контролер и на тај начин задаје идентификатор чвора кога треба избацити из мреже због невалидног понашања.

8) GET /chain\_length(chainType,authToken)

Рута коју користе међусобно чворови како би извршавали алгоритам консензуса у циљу придобијања информација о ланцу, у овом случају дужине ланца.

9) GET /chain (chainType,authToken)

Слично као код претходне руте и у исту сврху ова рута враћа ланац чвору ради постизања консензуса.

10) GET /voting\_options (votingID, authToken)

Одговор на редиректовану руту из ClientAPI-ја само се додатно аутентификује редирекција контролера преко токена.

11) GET /key\_valid (votingID,userPU, authToken)

Провера да ли дати јавни кључ јесте у вези са идентификатором гласања и да ли је евентуално већ искоришћен.

12) GET /voting\_results (votingID,authToken)

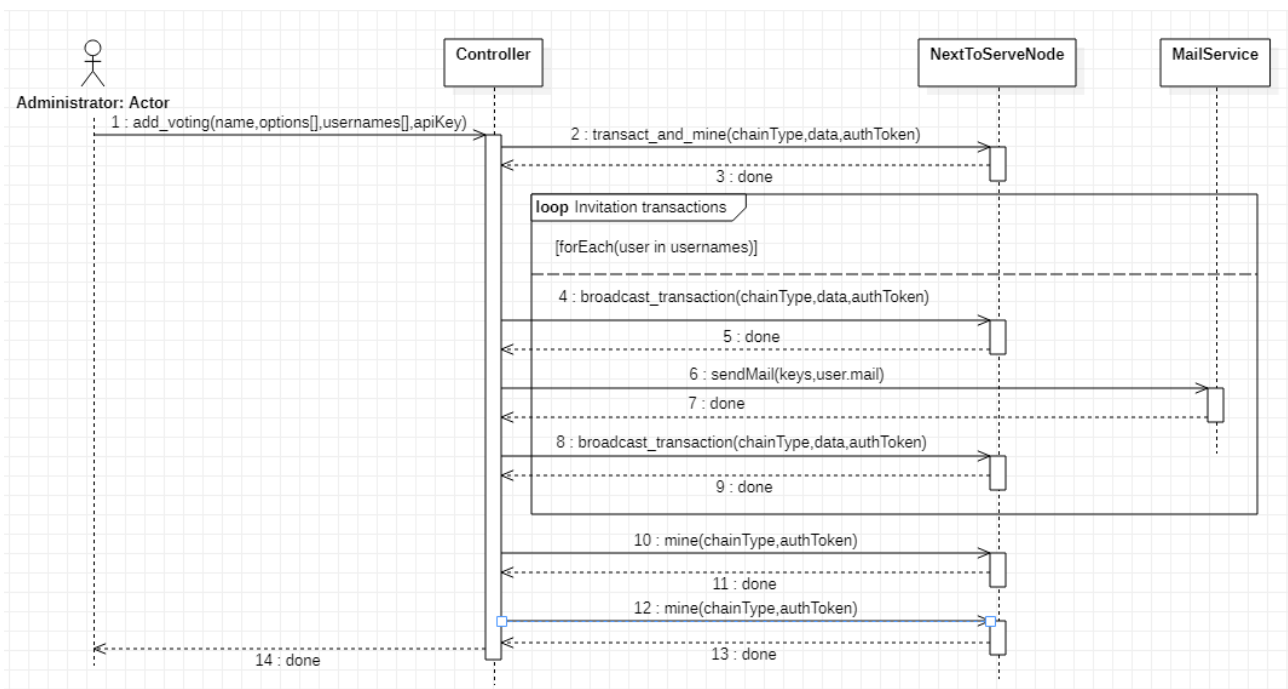
Одговор на редиректовану руту из ClientAPI-ја само се додатно аутентификује редирекција контролера преко токена.

### 13) GET /heartbeat (authToken)

Једноставна рута која само уверава контролера да дати чвор и даље функционише.

## 5.5. Пример стварања гласања

Битно је напоменути да је за сигурну доставу пара кључева гласачу за дато гласање изабрана електронска пошта. Приликом креирања гласања контролни сервер у .zip архиву пакује фајлове са идентификатором гласања, јавним и приватним кључем у PEM формату и целу архиву шифрује лозинком посматраног корисника. Под *NextToServeNode* сматра се ентитет чвор који је добијен као следећи из RR реда за обраду захтева. При свакој поруци *NextToServeNode*-у контактира се конкретно другачији чвор који ће захтев обработити али због једноставности UML дијаграма ово се у дијаграму изоставља.



Слика 5.5.1 Дијаграм секвенце стварања једног гласања

## 6. РЕАЛИЗАЦИЈА СИСТЕМА

Следи преглед технологија које су искоришћене како би се систем развио као и механизми приступа модулима и формирању сервера. Прво се врши преглед конкретних технологија уз давање образложења за избор истих.

### 6.1. Избор технологије

MEAN (*MongoDB*, *Express.js*, *Angular*, *Node.js*) нуди једоставан и целовит приступ изради веб система. Технологија је базирана на програмском језику *JavaScript*, те се клијентски и серверски део система развијају на истој платформи и интерпретирају на истој машини *Node.js*. Овиме се олакшава одржавања кода као и међусобна интеграција. *MongoDB* јесте *NoSQL* база која податке чува у формату сличном JSON-у, дозвољавајући висок степен адаптивности самим објектима инфраструктуре јер се не намећу скоро па никаква типизациона ограничења. *Express.js* је модуларни радни оквир (енг. *framework*) пакет намењен за *Node.js* замишљен као једноставно оруђе за креирање сигурних веб сервера и једноставно форматирање њихових рута. *Angular* представља радни оквир за развијање клијентског кода за обраду и одржавање *HTML* страница, као и комуникацију са серверским делом. *Node.js* је развијен као *JavaScript* окружење у реалном времену за интерпретирање кода високог нивоа асптракције на серверском делу. Базиран је на асинхронном моделу који укључује ред догађаја и повратних позива на дате догађаје. Овакав модел директно подржава међупроцесну комуникацију имплементирану над HTTP/HTTPS протоколима док је за потребну синхронизацију моделом дељене променљиве за потребе овог пројекта развијен једноставан код налик на семафору који омогућава ексклузивност приступа блоковима кода.

За израду самопотписаног сертификата коришћен је *OpenSSL* софтвер.

### 6.2. Покретање система

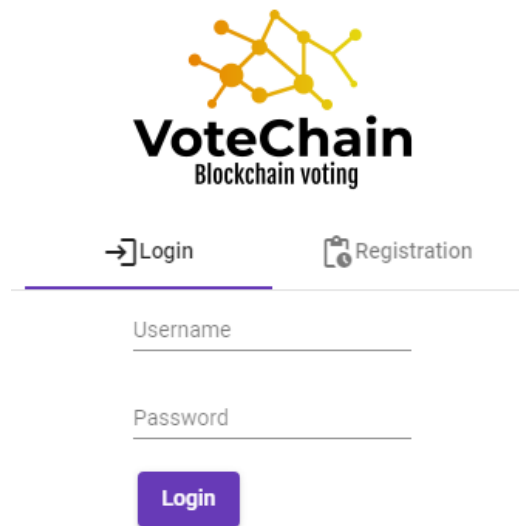
NPM (*Node Package Manager*) јесте помоћни алат *Node.js* задужен за олакшано интерпретирање команди, али и за одржавање и верзионисање библиотека. У систем су директно уграђене команде:

- `npm run startController` – покреће серверски програмски код сервер контролера
- `npm run startNodes` – покреће пет сервера чвора на серверској страни, којима је интерно прослеђена адреса контролера
- `npm run startNodes` – покреће клијентски сервер који формира странице и директно комуницира са веб прегледачем клијента

Наравно у реалном случају примене система нужно би било прво купити валидан сертификат, уградити га у окружење на којем су сервери постављени али такође и непосредно у окружење уградити предефинисане параметре за сигурну комуникацију као што су кључеви и лозинке. Систем је подешен да стално брише целу базу приликом покретања ради тестирања, али се ово понашање може и мора променити при реалној употреби како би нпр. у случају нестанка струје цео ланац на свим чворовима остао непромењен након поновног покретања.


### 6.3. Приказ корисничког интерфејса

Дат је приказ страница које улоге користе за приступ систему. Свака слика дефинише једну од могућих функционалности и начина на које се исте примењују.



Слика 6.3.1 Почетна страна платформе

Задају се корисничко име и лозинка за приступ систему, корисничко име, лозинка, име, презиме и адреса е-поште за отварање гласачког налога.



Create new voting

Close voting

View voting results

---

Name

Izbor predstavnika logistike

New option...

Petar G. ✕

Marko S. ✕

Nemanja D. ✕

---

New username...

nensi ✕


mare ✕

v

vlada

Слика 6.3.2 Администраторска страна за стварање гласања

Задају се име гласања, листа опција преко чипсета, као и листа корисничких имена који су позвани са задатом опцијом аутоматског допуњавања корисничких имена ради бољег корисничког искуства.



Create new voting

Close voting

View voting results

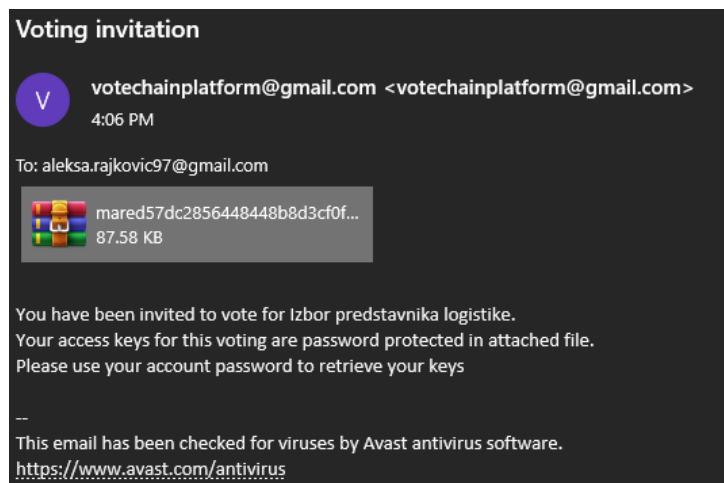
---

Voting ID

Close

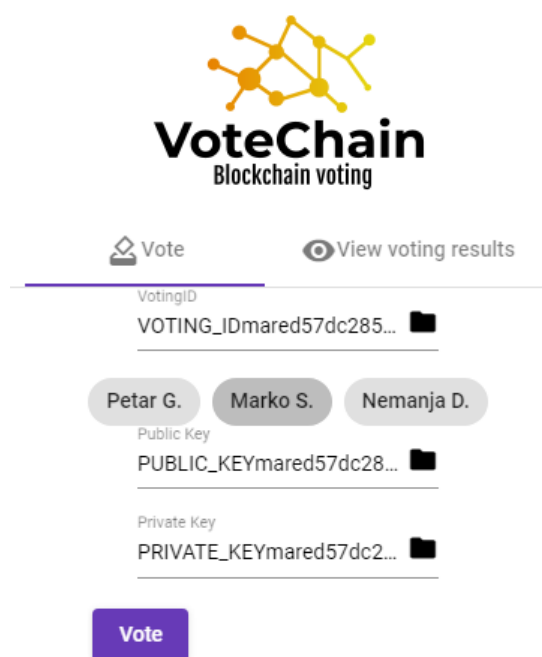
Слика 6.3.3 Администраторска страна за затварање гласања

Задаје се само идентификатор гласања и посматрано гласање се евентуално затвара.



Слика 6.3.4 Приказ мејла који пристиже кориснику као позив на гласање

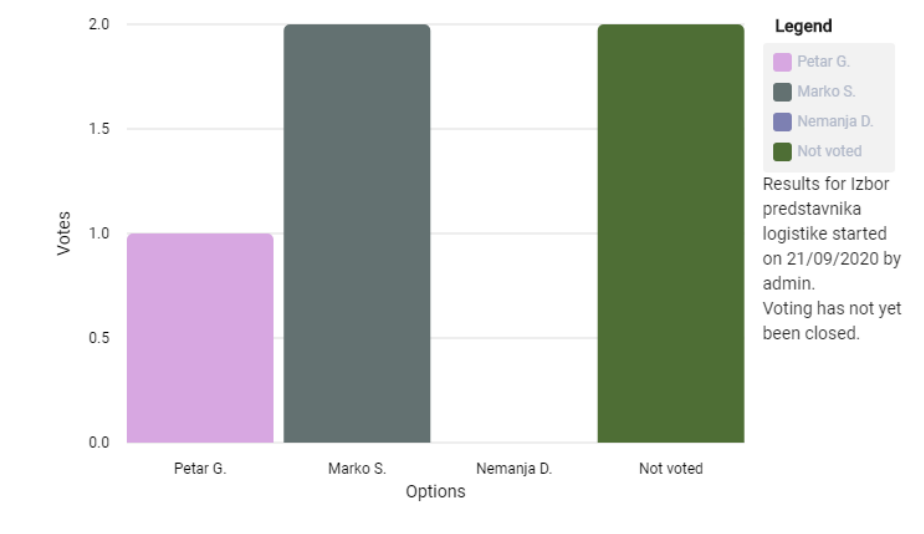
Послата архива шифрована је *AES* алгоритмом, а за лозинку је искоришћена шифра корисника коме се шаље. Архива садржи три текстуална фајла један са идентификатором гласања, један са јавним кључем и један са приватним кључем за дато гласање.



Слика 6.3.5 Приказ форме за гласање

Гласач прво распакује архиву са фајловима уносећи своју лозинку. Након тога уноси све потребне фајлове и бира опцију за гласање.





**Слика 6.3.5 Приказ извештавања о гласању**

Корисник преко форме уноси идентификатор жељеног гласања и добија генерисан график тренутних резултата са описом стања.

## 7. ЗАКЉУЧАК

У овом дипломском раду детаљно је анализирана технологија блокчејн, као и његова примена за дату реализацију. Закључује се да је технологија довољна да обезбеди основне сигурносне механизме, али такође да избегне пропусте који се јављају код конвенционалних гласања. Постигнуто је јасно раслојавање интерфејса система уз јасно дефинисан кориснички интерфејс који омогућава приступ свим наведеним функционалностима. Инфраструктура у сржи јесте јединствена и садржи неке аспекте од разних пројеката који су се овом темом већ бавили. Платформа је замишљена као таква да је покреће одређена фирма или институција у циљу прикупљања потребних информација за своје интерне договоре и планове. У потпуности се одржавање система, обезбеђивање кључева као и формирање предефинисаних шифри оставља покретачу платформе. Покретач такође мора добро обезбедити приступне тачке, првествено дефинишући јасан и конкретан поступак за сертификацију једног чвора. Додатно мора се дефинисати и начин стварања корисничких улога са електронским адресама које би у идеалном случају биле адресе домена покретача. Покретач мора да обезбеди сигурност налога електронске поште своје устанве како не би долазило до евентуалних пробоја и малверзација крађама приватних кључева који су генерисани при позиву на гласање. Идеално би било да покретач сертификује сваки рачунар који се налази у фирми и који је већ придружен локалној инфраструктури. Овиме се сигурност пробоја смањује јер се сви релевантни чворови држе под истом мрежом.

Систем дефинитивно има рањиве тачке, генерално један од напада би могао бити DDoS (*Distributed Denial of service*). Као одговор могло би се у сам софтвер платформе уградити ограничење брзине слања захтева и на серверској и на клијентској страни. Наиме, позадински сервери би само примали захтеве који пристижу до одређеног нивоа брзине док би се фронт подесио тако да захтеве и шаље у дефинисаним дозвољеним опсезима. Сви остали захтеви који стижу већом брзином од ограничене се одбијају јер могу бити злонамерни и угушити систем. Ова солуција само пластично успорава систем те би се заштита од DDoS најбоље имплементирала на нижим слојевима апстракције мрежних протокола мреже покретача.

Добра количина сигурности пребачена је на сам HTTPS и сертификат који представља домен платформе. Компромитовање приватног кључа система у потпуности би отворило целу мрежу чинећи сваки комуникациони канал неповерљивим и опасним за коришћење. Сходно томе велики напори морају бити уложени у очување и одржавање сертификата.

Дати модел се може узети као база за даља проширења и унапређења, као што су нови протоколи и функционалности система, такође модел је могуће уз додатне модификације искористити и за друге примене у којима су потребна пребројавања високог степена сигурности.

Блокчејн је технологија која у свету бива све прихваћенија и примењиванија, доста држава је процесу транзиције монетарних јединица са класичних система на дистрибуиране али се и у приватном сектору све више фирми одлучује за примену ланца. Аутор се нада да је обрадом ове тезе допринео барем малим делом глобалном истраживању блокчејна и његових примена.

## ЛИТЕРАТУРА

- [1] William Stallings, *Cryptography and Network Security Fourth Edition*, Prentice Hall, 2005.
- [2] Tiana Laurence, *Blockchain For Dummies*, John Wiley & Sons (Wiley), 2019.
- [3] Elad Elrom, *The Blockchain Developer: A Practical Guide for Designing, Implementing, Publishing, Testing, and Securing Distributed Blockchain-based Projects*, Apress, 2019.
- [4] Nathan Reiff, *Blockchain explained*, Available: <https://www.investopedia.com/terms/b/blockchain.asp> (09.20.2020.)
- [5] *Proof-of-Authority consensus*, Available: <https://apla.readthedocs.io/en/latest/concepts/consensus.html> (09.20.2020.)
- [6] Andrew Barnes, Christopher Brake and Thomas Perry, *Digital Voting with the use of Blockchain Technology*, Available: <https://www.economist.com/sites/default/files/plymouth.pdf> (09.20.2020.)
- [7] Friðrik Þ. Hjálmarsson, Gunnlaugur K. Hreiðarsson, *Blockchain-Based E-Voting System*, Available <https://skemman.is/bitstream/1946/31161/1/Research-Paper-BBEVS.pdf> (09.20.2020.)
- [8] Satoshi Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*, Available <https://bitcoin.org/bitcoin.pdf> (09.20.2020.)
- [9] *Node.js Documentation*, Available <https://nodejs.org/en/docs/> (09.20.2020.)

## СПИСАК СКРАЋЕНИЦА

AES - *Advanced Encryption Standard*

API – *Application Programming Interface*

DDoS - *Distributed Denial of service*

HTTP - *Hypertext Transfer Protocol*

HTTPS - *Hypertext Transfer Protocol Secure*

IP – *Internet Protocol*

JSON - *Javascript Object Notation*

MEAN - *MongoDB, Express.js, Angular, Node.js*

NPM - *Node Package Manager*

P2P – *Peer to peer*

PEM - *Privacy-Enhanced Mail*

PoA – *Proof of Authority*

PoW - *Proof of Work*

REST - *Representational State Transfer*

RR – *Round Robin*

RSA – *Rivest Shamir Adleman*

SHA - *Secure Hash Algorithm*

UML - *Unified Modeling Language*

URL - *Uniform Resource Locator*

## СПИСАК СЛИКА

Слика 2.2.1 Генеричка шема структуре блокчејна .....	8
Слика 2.4.1 Генерички приказ Мерклеовог стабла.....	9
Слика 3.1.1 Приказ естонијског модела.....	17
Слика 4.2.1 Дијаграм употребе система .....	22
Слика 5.1 Објектни дијаграм трансакције стварања гласања.....	23
Слика 5.1.1 Објектни дијаграм трансакције стварања гласања.....	24
Слика 5.1.2 Објектни дијаграм трансакције гласа .....	24
Слика 5.1.3 Објектни дијаграм трансакције гласа .....	25
Слика 5.2.1 Архитектура система.....	26
Слика 5.3.1 Дијаграм секвенце протокола упознавања.....	28
Слика 5.3.2 Дијаграм секвенце генеричког захтева ентитета А ка ентитету Б.....	29
Слика 5.3.3 Дијаграм секвенце генеричког захтева чвора А ка чвору Б уз додатну елиминацију.....	30
Слика 5.3.4 Дијаграм секвенце добијања кључа интерфејса и приступања систему.....	31
Слика 5.5.1 Дијаграм секвенце стварања једног гласања .....	36
Слика 6.3.1 Почетна страна платформе .....	38
Слика 6.3.2 Администраторска страна за стварање гласања .....	39
Слика 6.3.3 Администраторска страна за затварање гласања .....	39
Слика 6.3.4 Приказ мејла који пристиже кориснику као позив на гласање.....	40
Слика 6.3.5 Приказ форме за гласање .....	40
Слика 6.3.5 Приказ извештавања о гласању.....	41