Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет навчально-науковий центр заочної форми навчання
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Модель децентралізованої системи автентифікації

(тема)

Виконав:

студент II курсу, групи КСМзм-21-1

Шокота Т. А.
(прізвище, ініціали)

Спеціальність

123 «Комп'ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма

Комп'ютерні системи та мережі
(повна назва освітньої програми)

Керівник: доц. Ільїна І.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет навчально-науковий центр заочної форми навчання

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп'ютерна інженерія»
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп'ютерні системи та мережі
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
"____" _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Шокоті Тетяні Анатоліївні
(прізвище, ім'я, по батькові)

1. Тема роботи Модель децентралізованої системи автентифікації

затверджена наказом по університету від " 24 " жовтня 2022 р. № 178Стз

2. Термін подання студентом роботи до екзаменаційної комісії 13 грудня 2022 р.

3. Вхідні дані до роботи Вимоги до системи автентифікації

4. Перелік питань, що потрібно опрацювати у роботі

Аналіз предметної області;

Аналіз проблеми та огляд існуючих рішень

Розробка програмних модулів

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 15 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз проблеми та огляд існуючих рішень	25.10.22–01.11.22	
2	Вибір технології розробки та	02.11.22–17.11.22	
	інструментальних засобів		
3	Розробка алгоритмічного забезпечення	22.11.22–28.11.22	
4	Розробка програмних модулів	29.11.22–02.12.22	
5	Відлагодження програмних модулів	03.12.22–06.12.22	
6	Оформлення матеріалів кваліфікаційної роботи	07.12.22–08.12.22	
7	Подання кваліфікаційної роботи керівникові та		
	її попередній захист	09.12.22–11.12.22	
8	Подання кваліфікаційної роботи на рецензу-	12.12.22–13.12.22	
	вання		

Дата видачі завдання 24 жовтня 2022 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Ільїна І.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 66 с., 10 рис., 1 дод., 22 джерел.

ДЕЦЕНТРАЛІЗАЦІЯ, ІНФОРМАЦІЙНА СИСТЕМА, АВТЕНТИФІКАЦІЯ, ІНФОРМАЦІЙНА БЕЗПЕКА

Метою кваліфікаційної роботи є розробка для мережі Інтернет єдиної та універсальної системи ідентифікації та автентифікації.

Для досягнення поставленої мети вирішуються наступні задачі:

- аналіз існуючих протоколів аутентифікації, їх переваг та недоліків;
- підбір відповідного протоколу аутентифікації та наявної інфраструктури (якщо така у нього є): сховище особистостей, сховище даних, вузли мережі, додаток-клієнт;
- розробка бібліотек розробника та розповсюдження їх по каналах Open Source;
- розробка тестового сайту, який використовує систему автентифікації;
- аналіз рішення на зазначені вище критерії.

ABSTRACT

Master’s thesis: 66 pages, 10 figures, 1 appendices, 22 sources.

DECENTRALIZATION, INFORMATION SYSTEM, AUTHENTICATION, INFORMATION SECURITY

The major goal of this thesis is to develop a single and universal identification and authentication system for the Internet.

To achieve the goal, the following tasks are solved:

- analysis of existing authentication protocols, their advantages and disadvantages;
- selection of the appropriate authentication protocol and existing infrastructure (if it has one): identity storage, data storage, network nodes, client application;
- development of developer libraries and their distribution through Open Source channels;
- development of a test site that uses an authentication system;
- analysis of the decision based on the above-mentioned criteria.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД – бази даних

ІТ – інформаційна система

ІБ – інформаційна безпека

ІС – інформаційна система

OpenID – открытый стандарт децентрализованной системы аутентификации, предоставляющей пользователю возможность создать единую учётную запись для аутентификации на множестве не связанных друг с другом интернет-ресурсов, используя услуги третьих лиц

OAuth 2.0 – протокол авторизації, що дозволяє видати одному сервісу (додатку) права на доступ до ресурсів користувача на іншому сервісі

ВСТУП

Одним із наріжних каменів інформаційної безпеки є питання аутентифікації, тобто перевірки справжності однієї із сторін інформаційного обміну. Традиційно автентифікація викликає безліч складнощів як у звичайних користувачів, так і у компаній, які вимагають підтвердження у користувачів. Неповний список цих проблем включає в себе втрати паролів, крадіжки акаунтів, витоку даних користувача. З аутентифікацією також нерозривно пов'язана ширша проблема ідентифікації, тобто наявності в людини законної особи – набору персональних даних про людину як суб'єкт права, її права та обов'язки. Згідно з некомерційною організацією ID2020, що проводить щорічні конференції в штаб-квартирі ООН, одне п'яте населення Землі живе без надійного способу ідентифікувати себе, таким чином випадаючи з правового поля та стаючи вразливими для залучення до кримінальної діяльності [1].

Для більшості людей на даний момент найголовніший спосіб засвідчити свою особу – це паспорти, що видаються державними органами. При цьому одна і та сама людина може бути громадянином кількох країн і мати кілька документів, що ідентифікують, а отже — кілька осіб.

Виникає ще більше труднощів, коли йдеться про електронну ідентифікацію. Багато постачальників послуг в Інтернеті вимагають реєстрації, зберігання даних на своїх серверах. Кожен такий обліковий запис робить внесок у роздробленість особистостей. Виникає обґрунтоване бажання розробити єдиний механізм ідентифікації користувача та аутентифікації. І такі механізми вже є; головна їхня проблема в тому, що серед них немає однозначно глобального та універсального, а головне прозорого. Такі механізми, як правило, повністю покладаються на будь-яку комерційну організацію, якій ми довіряємо свої дані та електронні особи. У цій роботі будуть розглянуті вже існуючі рішення для єдиної аутентифікації

користувачів та запропонуємо альтернативний варіант на базі технології blockchain, покликаної усунути залежність від однієї компанії, що засвідчує, і децентралізувати механізм видачі законних осіб у мережі Інтернет.

Метою даної роботи є розробка для мережі Інтернет єдиної та універсальної системи ідентифікації та автентифікації, яка відповідає критеріям, сформульованим нижче. Під системою ідентифікації розумітимемо:

- сховище особистостей, кожна з яких асоціюється з певною реальною людиною;

- спосіб, яким людина може довести володіння цією особою (автентифікація);

Під системою аутентифікації розумітимемо набір наступних структур:

- сховище даних, де містяться зашифровані персональні дані користувачів та їх публічні ключі;

- вузли мережі, що обслуговують сховище даних та відповідають за логіку процесу автентифікації;

- додаток-клієнт, здатний приймати та підтверджувати запити на автентифікацію, пов'язані з конкретною особистістю;

- протокол, яким взаємодіють перелічені вище частини системи.

- бібліотеки, за допомогою яких розробник легко зможе вбудувати аутентифікацію протоколу на свій сайт.

Водночас представлена тут система ідентифікації та аутентифікації має відповідати таким критеріям:

- універсальність: маючи особистість у цій системі, людина повинна мати можливість автентифікуватися на будь-якому підключеному до системи веб-сайті;

- безпека: користувачі контролюють безпеку доступу до своїх імен і не залежать від політик безпеки третіх сторін;

- децентралізованість: жодна фізична чи юридична особа не може одноосібно контролювати систему та процес аутентифікації користувачів.

Володіння ім'ям підтверджується чи відкидається з урахуванням колективного рішення учасників мережі. Жоден вузол не є точкою відмови

Тому завдання роботи входить:

- аналіз існуючих протоколів аутентифікації, їх переваг та недоліків;
- підбір відповідного протоколу аутентифікації та наявної інфраструктури (якщо така у нього є): сховище особистостей, сховище даних, вузли мережі, додаток-клієнт;
- розробка бібліотек розробника та розповсюдження їх по каналах Open Source;
- розробка тестового сайту, який використовує систему автентифікації;
- аналіз рішення на зазначені вище критерії.

1 ПІДХОДИ ДО АУТЕНТИФІКАЦІЇ

1.1 Зберігання паролей у незашифрованому вигляді на боці постачальника послуг

Переваги:

- простота реалізації.
- можливість відновлення старого втраченого пароля.

Недоліки:

- нестійкість до злому сховища.Система аутентифікації з таким методом зберігання паролів є незахищеною. При зломі постачальника послуг усі паролі стають доступними хакеру, і він може у будь-який час автентифікуватись під виглядом іншого користувача. Дані користувача знаходяться під великою загрозою;
- неуніверсальність. Користувач змушений щоразу заводити новий обліковий запис для кожного постачальника послуг з таким методом автентифікації. Оскільки заводити однакові пари логін-пароль вкрай небезпечно (а часто просто неможливо), це створює велику плутанину.

1.2 Зберігання паролів у зашифрованому вигляді на стороні постачальника послуг

Такі системи схожі на попередні, за винятком того, що постачальник послуг зберігає на своїх серверах паролі не у відкритому вигляді, а у вигляді секретного ключа, який генерується один раз на основі пароля. Для цього використовуються спеціальні алгоритми, наприклад PBKDF2 [2].

Щоразу, коли користувач вводить свій пароль, система знову обчислює по ньому ключ, і якщо він збігається з раніше збереженим для цього користувача, то користувач успішно проходить автентифікацію.

Переваги:

- стійкість до злому сховища. Навіть у разі злому постачальника послуг хакер отримує не самі паролі, а результати застосування алгоритмів формування ключа до паролів. Такі алгоритми спеціально розроблені так, щоб ускладнити отримання пароля

Недоліки:

- неуніверсальність. Користувачі все ще вимушені заводитиокремі облікові записи для кожного постачальника послуг.

1.3 OAuth, OpenID

Два відкриті стандарти, що дозволяють делегувати аутентифікацію деякій третій стороні — identity provider, постачальнику особистостей. Як правило, як identity provider виступає величезна корпорація, сервісів якої більшість користувачів вже мають обліковий запис (наприклад, Google, Facebook).

Переваги:

- універсальність. Маючи обліковий запис у identity provider, можна заходити під своїм ім'ям на будь-які сайти, що підтримують OpenID або OAuth. Такий обліковий запис служить універсальною мережевою особистістю і дозволяє використовувати одні й самі способи аутентифікації на безлічі майданчиків;

- відкритість. OpenID та OAuth – повністю відкриті стандарти, не захищені патентами та копірайтом.

Недоліки:

- довіра до третього боку. Користуючись OpenID або OAuth, користувачі змушені довіряти аутентифікацію деякої третьої сторони. З цього випливає;

- нестійкість до зламів. Користувач не може контролювати зберігання своїх даних на серверах третьої сторони і теоретично вони можуть зберігатися там у відкритому вигляді;

- відсутність анонімності. Identity provider бачить і логує дії клієнта: коли він аутентифікувався, куди, з якої IP-адреси, як часто;

- єдина точка відмови. Якщо сервер автентифікації третьої сторони виявиться на якийсь час недоступним, або третя сторона зовсім припинить існування, користувач втратить можливість аутентифікуватися під своїм єдиним ім'ям.

Будь-якої миті третя сторона може скористатися ім'ям користувача, оскільки саме вона ним і володіє.

У будь-який момент третя сторона може конфіскувати чи заблокувати особу користувача.

Цього недоліку можна уникнути, якщо запустити власний identity server (у разі OpenID). Однак це потребує певної технічної підготовки та обчислювальних ресурсів.

Не всі сайти можуть підтримувати кращий для користувача identity provider (у випадку OAuth).

1.4 Mozilla Persona

Єдина система аутентифікації, розроблена у Mozilla під впливом ідей VerifiedEmailProtocol. Аутентифікація будується на основі сертифіката, який зберігається у пам'яті браузера користувача. Кожен сертифікат містить верифікований email користувача та криптографічно доводить, що користувач дійсно володіє цією адресою. Кожні 24 години сертифікат необхідно оновлювати, надіславши пароль на сервер Mozilla. Однак оновлений сертифікат наступних 24 годин служить засобом автентифікації на всіх сайтах, які підтримують Mozilla Persona, і не вимагає введення пароля.

Переваги:

- універсальність;
- відкритість;
- відносна анонімність.

З огляду на особливості протоколу, identity provider (у разі Mozilla) бачить лише факт оновлення сертифіката. Identity provider не може зрозуміти, де і коли цей сертифікат використовується для аутентифікації.

Недоліки:

- довіра до третього боку. У випадку з Mozilla Persona, з цього випливає менше недоліків, ніж у OpenID та OAuth. Протокол розроблено так, щоб забезпечувати базову анонімність та стійкість до зламів;
- відсутність підтримки з боку розробника.Mozilla припинила розробку проекту і віддала його під опіку спільноті.

1.5 Актуальність стеганографії

Blockchain – розподілена база даних, яку вперше почали використовувати для криптовалют, таких як Bitcoin, творець якої Сатоші Накамото вперше і ввів цей термін. Blockchain - база даних, в криптовалютах вона зберігає послідовність всіх фінансових транзакцій. Копія цієї бази даних живе одночасно на всіх вузлах системи (хоча для продуктивності деякі вузли можуть мати урізані права та не зберігати базу у повному вигляді). Додавання до блокчейну нової транзакції можливе лише з урахуванням консенсусу, тобто спільної верифікації кількома вузлами системи. [7] Згодом технологія blockchain знайшла застосування в багатьох сферах, де була потрібна надійність, захищеність і відсутність єдиної сторони, якій потрібно довіряти. Останнім часом Blockchain почав використовуватись і в області ідентифікації та аутентифікації:

Рішення цих двох компаній, Blockstack та uPort, є досить перспективними розробками, проте також не позбавлені недоліків.

1.5.1 Blockstack

Blockstack спочатку є системою безсерверних веб-додатків, яка дозволяє створювати односторінкові сайти та безкоштовно викладати їх у блокчейн замість традиційного хостингу. У процесі було розроблено концепцію Blockchain ID, з допомогою якої можна створити у блокчейне публічний профіль і його аутентифікувати себе [4].

Система Blockstack використовує вже існуючу інфраструктуру Bitcoin. Кожна транзакція Bitcoin може містити опціональне поле OP_RETURN, де зберігається довільна інформація. Завдяки такій можливості Blockstack будує на основі блокчейна Bitcoin віртуальний блокчейн (virtualchain), який складається тільки з транзакцій з непустими полями OP_RETURN, що містять повідомлення потрібного формату. Такі транзакції передають можуть передавати інформацію про такі операції:

- реєстрація імені.Вхід цієї операції ім'я прив'язується добиткоин-адресу, що зареєстрував його;
- оновлення профілю.У профілі можуть бути абсолютно довільні поля;
- надсилання імені іншій адресі.

Таким чином, віртуальний блокчейн Blockstack грає роль, схожу роль DNS.

Переваги:

- універсальність.Зареєструвавши ім'я в системі Blockstack, користувач може автентифікуватись під цим ім'ям на будь-яких сайтах, на яких працює протокол автентифікації Blockstack;
- відкритість.Розробка системи ведеться за принципами Open Source;
- анонімність. Інформація про те, коли і куди аутентифікується користувач, доступна лише користувачеві та сайту, куди він аутентифікується;
- відсутність довіри до третього боку.Система Blockstack працює на базі розвиненого блокчейну Bitcoin.

Це означає що:

Систему практично неможливо зламати: у такій системі взагалі немає такого поняття, як злом чи витік даних. Інформація про зареєстровані імена і так публічна та доступна всім. Вкрасти можна тільки саме ім'я, але завдяки криптографічним основам блокчейна це можливе лише в тому випадку, якщо зловмисник отримає до рук приватний ключ користувача. Оскільки приватний ключ зберігається тільки на стороні користувача, і ключ зашифрований за допомогою його майстер-паролю, ця ситуація рівноцінна компрометації одночасно сховища користувача та його майстер-паролю, а не системи загалом.

У системі немає єдиної точки відмови.Система може вийти з ладу тільки якщо вийдуть з ладу всі вузли блокчейну Bitcoin.

Ніхто не може скористатися ім'ям користувача, крім нього самого.

Ніхто не може конфіскувати чи заблокувати особу користувача. Усі дії в блокчейні приймаються з урахуванням консенсусу кількох вузлів. Конфіскація імені користувача можлива лише за умови схвалення значної кількості вузлів мережі, що рівноцінно компрометації кількох тисяч серверів по всьому світу.

Недоліки:

- новизна. Протокол все ще перебуває на стадії розробки. До хоч скільки робочого стану основні частини системи аутентифікації привели тільки до кінця 2016 року. Система може працювати нестабільно, з помилками, відсутня важлива функціональність;

- незручність користувача. Хоча розробники прагнуть виправити цю проблему, зараз робота з протоколом досить незручна. Користувачеві необхідно завантажити, запустити та налаштувати локально додаток-клієнт, який на момент написання роботи в готовому вигляді існує тільки для операційної системи OS X; користувачі Windows та Linux змушені завантажувати та налаштовувати частини системи вручну. Користувачеві також необхідно замовити собі хоча б одне ім'я; в силу особливостей

протоколу система вимагає від користувача невелику плату за кожну операцію з віртуальним блокчейном, у тому числі за реєстрацію імені. Це означає, що користувачеві треба купити біткоїни та передати їх на свою адресу. Немає повноцінних бібліотек для вбудовування автентифікації на веб-сайті.

1.5.2 uPort

uPort, на відміну від Blockstack, є спеціалізованою системою, заточеною саме під ідентифікацію та автентифікацію. Творці обіцяють можливість створювати окремі особи для різних аспектів життя людини (банківська діяльність, акаунти у соціальних мережах, професійна діяльність тощо). Особу можна буде використовувати як єдиний обліковий запис для аутентифікації, підписувати нею заяви, виставляти напоказ свій публічний профіль.

uPort все ще знаходиться в розробці, автори зараз випустили White Paper і запрошують всіх бажаючих взяти участь у тесті альфа-версії. Система побудована з урахуванням розумних контрактів на платформі Ethereum. Це дає творцям багаті можливості використання вже існуючого коду, вбудованого в платформу.

Система складається з [5]:

- мобільного клієнтського додатка, що зберігає приватний ключкористувача;
- бібліотек, за допомогою яких розробники можуть підключитиаутентифікацію до свого сайту;
- розумні контракти Ethereum, що містять логіку програми.

Хоча uPort виглядає досить перспективною розробкою, зараз вона знаходиться в самій початковій стадії. Головним недоліком є – закритий код мобільних програм uPort, що не дає повністю зрозуміти механізм роботи, і стадія альфа-тестування, і як наслідок — нестабільна робота системи.

Переваги:

- універсальність;
- анонімність;
- відсутність довіри до третього боку;
- зручність користувача.З усіх систем аутентифікації на основі blockchain uPort має найзручніший мобільний додаток-клієнт.

Недоліки:

- новизна.Система знаходиться в стадії напівзакритої альфи: взяти участь у тесті може будь-хто, хто бажає, але якщо той зареєструється в програмі тестування. Мобільний додаток працює нестабільно та активно доопрацьовується;
- частково закритий код.Репозиторії клієнтських програм для iOS та Android закриті, хоча розробники обіцяють незабаром опублікувати вихідний код.

1.5.3 EMCSSL

Рішення, що ґрунтується на криптовалюті EmerCoin. Ця криптовалюта продовжує ідеї NameCoin – криптовалюти, де будується система альтернативних кореневих DNS-серверів. EmerCoin має розподілене сховище загального призначення із відкритим інтерфейсом: EmerCoin NVS. Це дозволяє зберігати у блокчейні інформацію будь-якого роду.

Одним із застосувань такого блокчейну є випуск SSL-сертифікатів. На них будується автентифікація користувачів за допомогою EmerCoin. На відміну від Blockstack і uPort, які розробляють власні протоколи аутентифікації клієнтів, написані на Javascript і запущені в браузері, EmerCoin використовують існуючу технологію аутентифікації за допомогою клієнтських SSL-сертифікатів.

Різниця між звичайними клієнтськими сертифікатами та сертифікатами EMCSSL у тому, що як центр сертифікації виступає не деяка третя довірена

сторона, а сам блокчейн. Головний секрет SSL – приватний ключ центру сертифікації – в EMCSSL є відкритим, і валідація сертифіката здійснюється не через підпис від CA, а через хеш у блокчейні.

Така система не розкриває секрет користувача в процесі аутентифікації сервера та використовує децентралізоване зберігання облікових записів. Як і у Blockstack і uPort, EMCSSL має можливість прикріпити до імені довільну інформацію у вигляді профілю [7].

Переваги:

- універсальність;
- відкритість;
- анонімність;
- використання існуючого протоколу (автентифікація через SSL-сертифікат клієнта). Це полегшує життя розробникам сайтів, на яких буде використовуватися така модель аутентифікації;
- відсутність довіри до третього боку.

Недоліки:

- напрямок на специфічну цільову аудиторію технічних фахівців. Прямо зараз система не має достатньо зручного способу для кінцевих користувачів зареєструвати ім'я та використовувати його як засіб аутентифікації. Користувачі змушені пройти процедуру генерації сертифіката і завантаження його в браузер [8], і хоча розробники надають докладну інструкцію для здійснення цих дій, вона поки що спрямована більше на людей, які хоч скільки-небудь знаються на криптографії;
- новизна. Єдиний сервіс, який покликаний допомогти кінцевому користувачеві зареєструватися та використовувати своє ім'я як засіб аутентифікації, Remme знаходиться у стадії закритої бети [9].

2 АРХІТЕКТУРА СИСТЕМИ

2.1 Огляд

Система, представлена у межах цієї роботи, виконує такі цілі:

- забезпечує безпечні та розподілені методи зберігання, реєстрації та передачі імен користувачів;

- забезпечує локальне зберігання приватного ключа користувача, за допомогою якого може довести своє володіння ім'ям;

- надає можливість користувачеві підписувати своїм приватним ключем запити на автентифікацію і таким чином заходити під своїм ім'ям на сайти, які підтримують цей протокол;

- містить бібліотеки, за допомогою яких розробники можуть легко налаштувати автентифікацію користувачів на веб-сайті;

Система складається з наступних компонентів:

- інфраструктура Blockstack:Virtualchain, Blockstack Core, Blockstack Portal, Onename;

- бібліотека Python, за допомогою якої можна створювати іпідписувати Authentication Request та Authentication Response об'єкти;

- програма для фреймворку Django, що реалізує аутентифікацію з використанням протоколу Blockstack на стороні сервера.

В рамках цієї роботи були реалізовані друга та третя частини системи (Python-бібліотека та Django-додаток). Перший компонент, тобто інфраструктура Blockstack, є відкритою розробкою Blockstack Inc., і в рамках цієї роботи вона була вивчена та використана як частина системи. Також було написано тестовий сайт, який використовує систему аутентифікації Blockstack.

2.2 Інфраструктура Blockstack

Blockstack – це цілий комплекс програмного забезпечення, що знаходиться в розробці, кінцевою метою якої є створення нового децентралізованого інтернету. Це стартап із Нью-Йорка, який співпрацює з Microsoft і отримав інвестиції у розмірі 4 мільйонів доларів в інвестиційному раунді Y Combinator [10].

Метою компанії є побудова у блокчейні окремої системи імен, схожої на DNS. Це дозволяє перевести такий важливий і фундаментальний рівень всесвітньої павутини, як система імен, децентралізовану область. Якщо перевести всі імена в блокчейн, то усунуть поодинокі точки відмови та сторони, яким ми змушені довіряти, такі як кореневі DNS-вузли та Identity Providers. Імена Blockstack можна використовувати як для адресації веб-сайтів, так і для автентифікації користувачів. Фактично, ім'я в блокчейні може бути як ім'ям сайту, так і ім'ям користувача, або тим, і іншим.

Програми Blockstack – програми без сервера, односторінкові веб-сайти, що складаються зі статичних файлів (HTML, CSS, Javascript). Ці статичні файли розміщуються в особистому хмарному сховищі розробника (Google Drive, Dropbox тощо) [11].

Інфраструктура Blockstack розбивається на кілька шарів [12]. Ключовою технологією інфраструктури Blockstack є блокчейн

Bitcoin, де будується система альтернативних кореневих DNS-серверів. Blockchain Layer інфраструктури Blockstack – це в чистому вигляді блокчейн Bitcoin; завдяки сильній криптографічній основі, децентралізованій природі та механізмам консенсусу він забезпечує розподіленість даних, безпеку та той факт, що у користувача не можна відібрати ім'я без компрометації його приватного ключа.

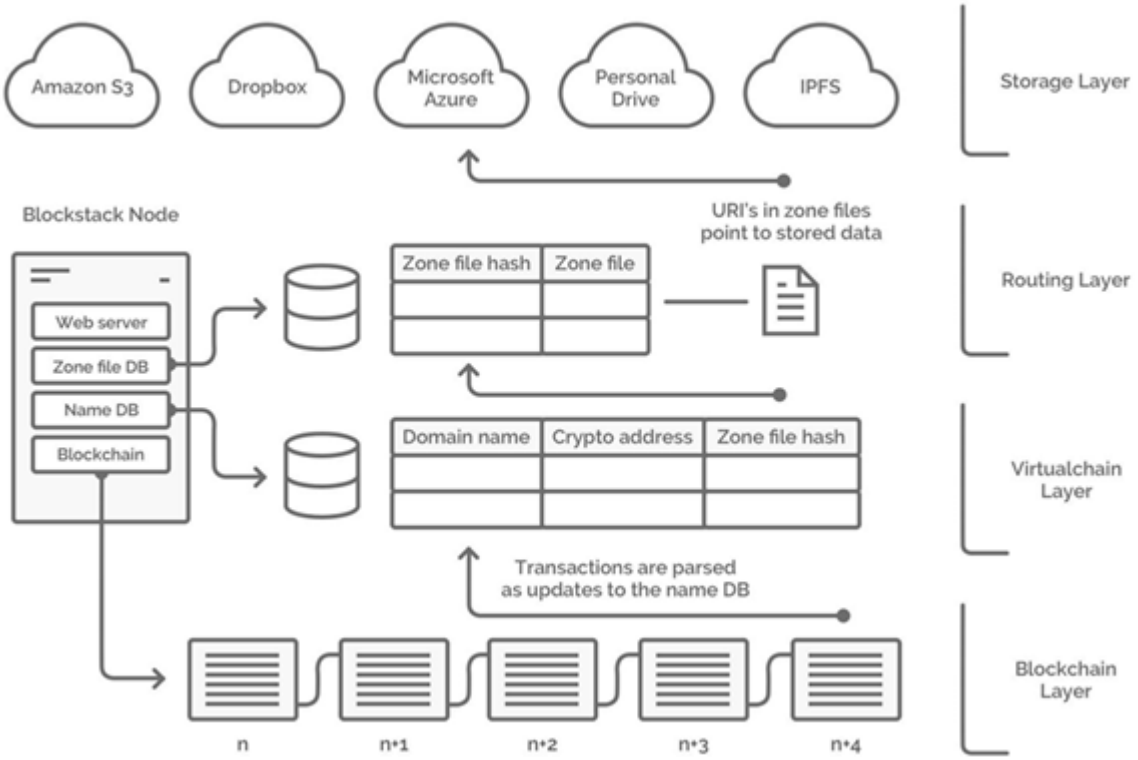

Рисунок 2.1 – Огляд Blockstack

Virtualchain Layer – надбудова над блокчейном, яка дозволяє зберігати в блокчейні інформацію, пов'язану з протоколом Blockstack: імена, їхніх власників, операції над іменами, інформацію про імена.

До кожного імені додається Zone File у такому самому форматі, в якому він використовується в DNS. Він містить посилання на інформацію про ім'я, наприклад профіль (якщо це ім'я користувача) або статичні файли [13-14]. Zone Files складають Routing Layer.

На всіх раніше згаданих трьох рівнях працює Blockstack Node. Це,по-перше, звичайний майнер Bitcoin, тобто Bitcoin Full Node, що індексує блокчейн. По-друге, такий сервер запускає у себе BlockstackCore: основну бібліотеку для роботи з Blockstack. бібліотеказапускається як сервер, що індексує Virtualchain і розпізнає Zone File. Посилання в Zone File ведуть на деяке віддалене сховище, якеможе утримувати як легковагові текстові описи профілів, так і важкі статичні файли. Цей останній шар називається Storage Layer.

Нарешті, Blockstack Portal – додаток користувача, який відповідає за доступ користувачів до Blockstack-сайтів і автентифікацію.

На даний момент ще не готова та частина системи, яка дозволяє користувачеві на ім'я сайту отримати до нього доступ, тобто завантажити статичні файли по дорозі в Zone File. Проте вже зараз можна випробувати ту частину системи, яка відповідає за індивідуальні акаунти та їхню автентифікацію на будь-яких сайтах, які не тільки використовують Blockstack. Ця можливість і буде використана в даній роботі для створення системи автентифікації щодо звичайних серверних сайтів [15].

Для простоти надалі процес аутентифікації через ім'я в системі Blockstack буде називатися просто «протоколом Blockstack», хоча самі розробники не використовують такий термін, оскільки назва "Blockstack" відноситься до цілого комплексу ПЗ, пов'язаного не тільки з аутентифікацією.

Розробники Blockstack запровадили нове поняття віртуального блокчейну. Це шар програмних компонентів поверх звичайного блокчейну, який вводить нову функціональність без змін блокчейна, що лежить в основі. Всі нові операції вводяться на рівні віртуального блокчейну і кодуються в мета-дані транзакцій звичайного блокчейну. І хоча звичайні вузли блокчейна теж бачать цю інформацію, логічне значення вона має лише вузлів віртуального блокчейна [16].

Прикладами операцій є попереднє замовлення імені, реєстрація імені, зміна власника, відкликання імені, продовження реєстрації імені та оновлення Zone File.

Як конкретний блокчейн, на якому працює система імен Blockstack, був обраний блокчейн Bitcoin як найрозвиненіший блокчейн мережі. Це пов'язано з тим, що маленькі блокчейни (з невеликою кількістю майнерів) сильніше схильні до ризику атаки: зловмисникові необхідні менші обчислювальні потужності в порівнянні з великим блокчейном. Крім того, маленькі блокчейни (такі як Namecoin, який так само використовується як

система альтернативних DNS-серверів) уразливі перед атакою 51%, коли один майнер або група майнерів у змові контролюють більше 51% обчислювальної потужності блокчейну, що веде до захоплення контролю над блокчейном однією стороною [17].

Інформація про операції Blockstack вбудовується через поле OP_RETURN транзакцій Bitcoin. Згідно з сайтом OP_RETURN Stats, який збирає інформацію про кастомні протоколи на основі блокчейну Bitcoin, протокол Blockstack є найбільшим за обсягом транзакцій серед нефінансових додатків [18].

Це основний CLI-додаток, написаний на Python, який здійснює значну частину роботи з протоколом, що здійснюється на сервері.

Blockstack Core включає:

- Blockstack API. Надає REST API для отримання інформації про інфраструктуру Blockstack: інформація про імена, гаманці, профілі, вузли і т.п. Ця програма потрібна для користувача, тому що API використовується в Blockstack Portal. Blockstack Core, запущений у режимі API, не перетворює локальну машину користувача на повний вузол Blockstack і таким чином не завантажує локально весь блокчейн; натомість він покладається на повний вузол, доступний за адресою node.blockstack.org;

- Blockstack Client. Консольна утиліта, за допомогою якої користувач може керувати своїм гаманцем та прив'язаним до нього іменами. До консольного клієнта прив'язані два Bitcoin-гаманці: owner та payment. Owner-гаманець має імена, саме його адреса вказується при передачі імені облікового запису користувача. На payment-адресу нараховуються біткоїни, щоб за допомогою них можна було платити невелику технічну плату кожну транзакцію (реєстрація імені, оновлення Zone File, передача імені);

- Blockstack Registrar. Бібліотека для Identity Provider'ів, за допомогою якого можна легко реєструвати нові імена, заповнювати профілі користувачів та надсилати ці імена на їхні особисті адреси;

- Blockstack Core. Повний вузол Blockstack, що індексує транзакції блокчейну Bitcoin і віртуального блокчейну.

У цій роботі в першу чергу нас цікавитимуть API та Client, оскільки вони необхідні для роботи Blockstack Portal.

Blockstack Portal – програма-сервер, що запускається локально на комп'ютері користувача [20]. За допомогою нього користувач може:

- завести собі новий owner-гаманець;
- зайти зі старого owner-гаманця за допомогою backup phrase;
- придбати собі ім'я;
- заповнити профіль цього імені;
- приймати запити на автентифікацію, логінуватись на веб-сайтах.

Рисунок 2.2 – Домашня сторінка Blockstack Portal.

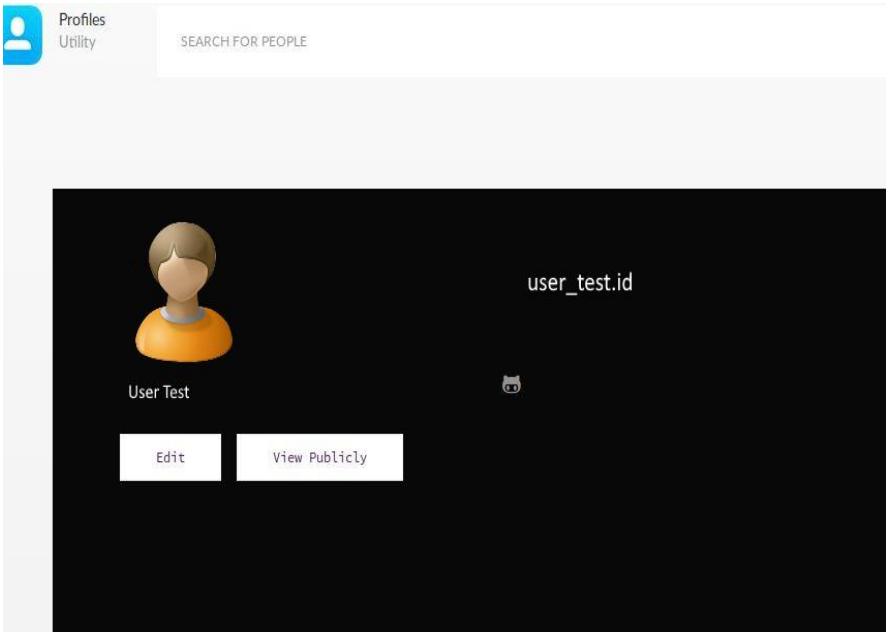

Рисунок 2.3 – Сторінка профілю користувача у Blockstack Portal.

Identity Provider від Blockstack, розташований за адресою https://onename.com

За допомогою цього сайту можна зареєструвати собі ім'я, редагувати профіль, верифікувати своє ім'я у Facebook, Twitter та Github, а також перенести створене ім'я на локальний owner-гаманець. Будь-яка плата за транзакції сплачується самим сайтом.

Через особливості роботи з блокчейном, ім'я може досить довго перебувати в процесі реєстрації (до декількох днів): воно відображатиметься на сайті Onename, але фактично не буде зафіксовано в блокчейні. Це пов'язано з тим, що будь-які транзакції в блокчейні спершу мають бути об'єднані в блок та пройти процедуру підтвердження іншими майнерами.

2.3 Python-бібліотека

На відміну від раніше описаних технологій Python-бібліотека була розроблена в рамках цієї роботи.

Бібліотека дозволяє:

- згенерувати запит на автентифікацію з боку сервера та відповідь на цей запит згідно із заданим JSON-форматом;
- підписати запит на ключ сервера;
- підписати відповідь на запит ключем користувача;
- перетворити запит та відповідь на запит у форму JWT і назад на JSON;
- верифікувати запит на автентифікацію;
- верифікувати відповідь на запит автентифікації.

2.4 Django-додаток

Django-додаток також було розроблено в рамках цієї роботи. Це зручний спосіб вбудувати на веб-сайт автентифікацію користувачів через

Blockstack. Використовуючи можливості раніше написаної Python-бібліотеки, програма дозволяє:

- за допомогою першого URL налаштувати процес аутентифікації: при переході по цьому URL сервер за допомогою Python-бібліотеки генерує запит на автентифікацію і перенаправляє користувача до Blockstack Portal;
- за допомогою другого URL (callback URL) програма приймає відповідь на запит аутентифікації, верифікує його і в разі успіху аутентифікує користувача;
- на ім'я користувача отримати його Blockstack-профіль.

3 РЕАЛІЗАЦІЯ ПРОТОКОЛУ АВТЕНТИФІКАЦІЇ BLOCKSTACK

3.1 Огляд

Реалізація поставлених завдань включає:

- аналіз відкритого протоколу автентифікації Blockstack, його інфраструктури, виявлення переваг та недоліків, використання його для реалізації бібліотек розробника blockchainauth та django-blockstack;
- розробку на основі протоколу Blockstack Python-бібліотеки blockchainauth;
- розробку Django-додатку django-blockstack, що використовує blockchainauth;
- розробка тестового веб-сайту, що використовує django-blockstack для автентифікації користувачів.

3.2 Терміни

Ім'я – рядок для ідентифікації в мережі Blockstack. Може використовуватися як для ідентифікації користувачів, так і доменів, але в рамках цієї роботи розглядається лише ідентифікація користувачів.

Користувач – одна із сторін процесу аутентифікації, деяка особа, у володінні якої є ім'я. Користувач характеризується парою закритого та відкритого ключа;

Аутентифікація – у разі процес, у якому користувач доводить своє володіння ім'ям.

Закритий ключ – HEX-рядок, головний криптографічний секрет, що асоціюється з користувачем. У випадку Blockstack закритий ключ є також закритим ключем Bitcoin, тобто по ньому здійснюється доступ до Bitcoin-

гаманця. Закритим ключем користувач підписує всі операції, пов'язані з його ім'ям, у тому числі автентифікаційні операції.

Відкритий ключ – HEX-рядок, публічна частина ідентифікації користувача на рівні блокчейну. За допомогою нього програма може верифікувати відповідь на автентифікацію, тобто перевірити, чи дійсно відповідь підписана закритим ключем користувача.

Адреса – адреса Bitcoin, пов'язана з відкритим ключем користувача. Усі зареєстровані імена належать якійсь адресі.

Додаток – одна із сторін аутентифікації, деякий сервіс, найчастіше веб-сайт, для повноцінної роботи з яким користувачеві необхідно автентифікуватись. Тут розглядаються ті програми, які підтримують протокол Blockstack.

Запит на автентифікацію – рядок, що генерується на стороні програми, в якому закодована інформація про програму та деяка мета-інформація, необхідна користувачеві для аутентифікації. Для більшої безпеки запит може підписуватися закритим ключем програми.

Відповідь на запит автентифікації – рядок, що генерується на стороні користувача, в якому закодована інформація про користувача та деяка мета-інформація, необхідна додатку для того, щоб перевірити справжність відповіді та автентифікувати користувача. Відповідь обов'язково підписується закритим ключем користувача.

Owner-гаманець – Bitcoin-гаманець, якому належать імена. Саме на адресу цього гаманця слід надсилати запити на передачу імені від адреси на адресу.

Payment-гаманець – Bitcoin-гаманець, з якого необхідно платити невелику сервісну плату за будь-яку операцію з іменами, що передбачає зміну блокчейна та нову транзакцію (реєстрація, попереднє замовлення, передачі імені, оновлення Zone-файлу). Саме на адресу цього гаманця слід відправляти біткоїни. Користувач завжди володіє як мінімум однією парою owner-гаманця та payment-гаманця.

3.3 Процес аутентифікації

Клієнтом для користувача є BlockstackPortal, для роботиякого також потрібний Blockstack Core.

Інструкції зі встановлення розташовані в публічному репозиторії на Github:https://github.com/blockstack/blockstack-portal.

Щоб запустити у себе Blockstack Portal, користувачеві необхідно:

- встановити Blockstack Core;
- налаштувати Blockstack Core, створивши пару owner-і payment-гаманців. Через особливості програмного забезпечення Blockstack на даний момент owner-гаманець з Blockstack Core та owner-гаманець з Blockstack;
- portal несумісні, при настроюванні BlockstackPortal користувачевідоведеться згенерувати новий owner-гаманець;
- запустити Blockstack Core у режимі API, підключившись до повного вузла Blockstack;
- завантажити Blockstack Portal;
- запустити CORS-проксі;
- запустити Blockstack Portal;
- завести в Blockstack Portal новий обліковий запис (тобто згенерувати owner-гаманець) або ввести backup phrase з 24 слів, відновивши доступ до старого owner-гаманця.

Увімкнути доступ до протоколу автентифікації у налаштуваннях Blockstack Portal.

Перш, ніж автентифікуватися в будь-якій програмі, користувач має придбати собі ім'я. Зробити це можна одним із наступних способів:

- завести біткоїни на свій payment-гаманець через інтерфейс Blockstack Portal замовити собі ім'я;
- придбати ім'я іншим способом (через сайт onename.com або через консольний клієнт Blockstack Core) та передати ім'я на owner-гаманець у Blockstack Portal.

На даний момент процес встановлення клієнта та придбання імені у системі Blockstack досить незручний для кінцевого користувача. У майбутньому розробники планують значно доопрацювати його.

Після набуття імені процес аутентифікації проходить для користувача тривіальним чином:

- з запущеним клієнтом він заходить на сайт програми і намагається автентифікуватись (рисунок 3.1);
- додаток генерує запит на автентифікацію та перенаправляє користувача до Blockstack Portal (рисунок 3.2);

Рисунок 3.1 – Домашня сторінка тестового веб-сайту “Hello, Blockstack”, користувач ще не автентифіковано.

Рисунок 3.2 – Інтерфейс Blockstack Portal при вхідному запиті на автентифікацію

- користувач кліком підтверджує запит, Blockstack Portal генерує відповідь на запит автентифікації та перенаправляє користувача назад до програми. Програма автентифікує користувача (рисунок 3.3).

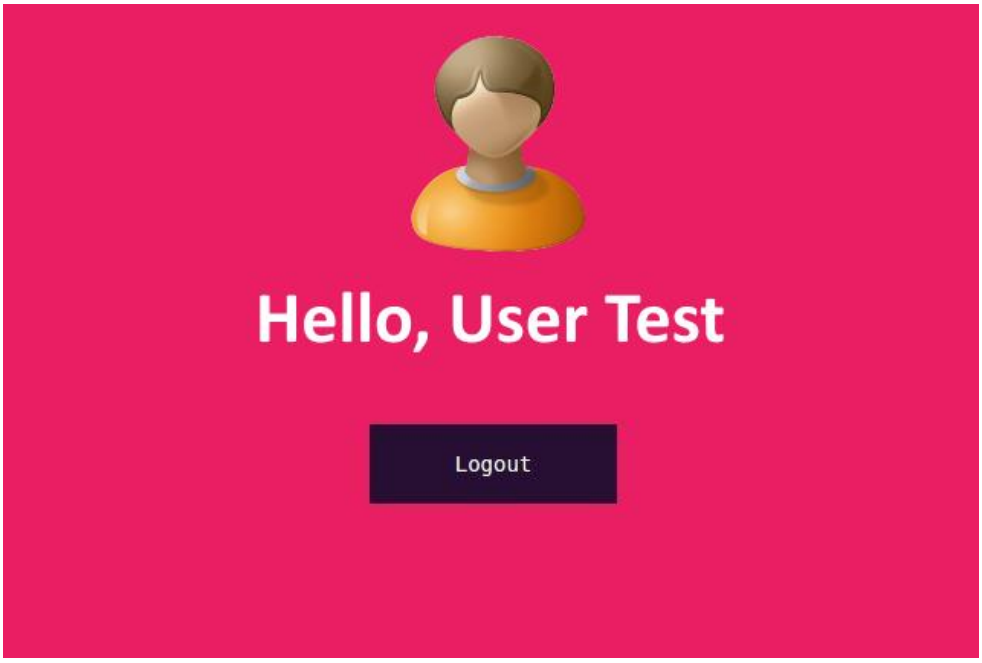

Рисунок 3.3 – Домашня сторінка тестового веб-сайту “Hello, Blockstack”, користувач автентифіковано

Щоб вбудувати на свій сайт аутентифікацію через Blockstack, розробнику необхідно підключити спеціальну бібліотеку. Як тільки користувач заходить на сторінку аутентифікації та намагається залогінитися, відбувається таке:

- бібліотека на основі поточного часу та даних про саму програму генерує запит на автентифікацію;
- запит на аутентифікацію підписується закритим ключем програми та кодується у вигляді одного рядка (JSON Web Token);
- програма перенаправляє користувача в Blockstack Portal за спеціальним посиланням, що містить раніше згенерований рядок;
- Blockstack Portal декодує рядок, верифікує запит і отримує дані про програму.

Як тільки користувач підтверджує аутентифікацію, Blockstack Portal перенаправляє його на адресу, вказану самим додатком у запиті на аутентифікацію, вказавши як GET-запиту закодовану відповідь на запит аутентифікації.

Програма верифікує відповідь на запит і у разі успіху аутентифікує користувача та отримує дані його профілю.

3.4. blockchainauth

Оскільки Blockstack розроблявся як мережа для безсерверних додатків, основна бібліотека для генерації запитів на аутентифікацію та відповідей на запит аутентифікації – бібліотека Javascript, код якої виконується на стороні клієнта:https://github.com/blockstack/blockstack.js.

Однак у застосуванні до традиційних серверних програм клієнтська бібліотека не підходить з міркувань безпеки. Додавши на сторінку свій скрипт, зловмисник може досягти неправильної роботи програми, підмінивши всю логіку автентифікації. Таким чином, можна, наприклад, автентифікуватися під чужим ім'ям і отримати доступ до чужих даних. Щоб такого не трапилося, була розроблена бібліотека мовою Python, що виконується на рівні сервера, що відповідає за автентифікацію користувачів, blockchainauth: https://github.com/spankratov/blockstack-auth-python

Після закінчення розробки зміни, зроблені в рамках даної роботи, були прийняті розробниками Blockstack до репозиторій проекту: https://github.com/blockstack/blockstack-auth-python

Запити на аутентифікацію в рамках бібліотеки представлені у вигляді об'єктів AuthRequest. AuthRequest зберігає таку інформацію:

- закритий ключ програми (private_key);
- доменна назва програми (domain_name);

- посилання, за яким можна отримати файл з інформацією про програму (manifest_uri). Значення за замовчуванням: доменне ім'я + "/manifest.json". Посилання має повертатися JSON у наступному форматі:

Лістинг 3.1 – Приклад JSON

```
{
"name": "Hello, Blockstack", "start_url": "localhost:5000",
"description": "A simple demo of Blockstack Auth", "icons": [{
"src":
"https://helloblockstack.com/icon-192x192.png", "sizes":
"192x192",
"type": "image/png"
}]
}
```

- посилання, на яке клієнтська програма надсилає відповіді на запити автентифікації (redirect_uri). Значення за промовчанням – доменне ім'я;

- список дозволів (scopes) – частина протоколу, що тимчасово не використовується. Значення за замовчуванням – порожній масив;

- Unix-час, через який закінчується термін дії цього запиту (expires_at). Значення за замовчуванням: за годину.

Запит, що генерується на основі цих даних, має наступний JSON-формат:

Лістинг 3.2 – Приклад JSON, значення за замовчуванням

```
{
"header": {"type": "JWT", "alg": "ES256K"}, "payload": {
"domain_name": "http://localhost:5000", "exp": 1493412486,
"iat": 1493408886,
"iss":
"did:btc-addr:1NZNxhoxobqwsNvTb16pdeiqvFvce3Yg8U", "jti":
"75719c8a-3679-45b7-9551-21b6dfc28444",
"manifest_uri": "http://localhost:5000/manifest.json",
"public_keys":
["027d28f9951ce46538951e3697c62588a87f1f1f295de4a14fdd4
c780fc52cfe69"],
"redirect_uri": "http://localhost:5000", "scopes": []
```

```
},
"signature":
"HBwhcgPj7hrKg_IOGyaMJ9L-U_kwE5EweK8H54E2fuNONeWElfJg-h
10LJvbwrvf_3TcgzQRbqdxGSmro8Ey6A"
}
```

Формат запиту вибрано спеціально для того, щоб закодувати його у вигляді JSON Web Token [21]:

а) перша частина запиту, header містить інформацію про те, що це JWT, а також алгоритм підпису (ES256K – синонім для SECP256k1, основна еліптична крива, що використовується в Bitcoin);

б) друга частина, payload, містить всю основну інформацію про запит;

1) domain_name – доменне ім'я;

2) exp – Unix-час, коли закінчується термін дії запиту;

3) iat – Unix-час, коли було створено запит;

4) iss – рядок у форматі або “did:btc-addr:<bitcoin address>”, або “did:ecdsa-pub:<public key>”, який означає спосіб адресації до додатку в блокчейні (адреса або відкритий ключ);

5) jti – унікальний ідентифікатор запиту (UUID);

6) manifest_uri–посилання на manifest.json програми;

7) public_keys – список відкритих ключів програми (зараз підтримується лише один відкритий ключ);

8) redirect_uri – посилання, за яким програма приймає відповіді на запит автентифікації;

9) scopes – список дозволів, які на даний момент не використовуються;

в) третя частина, signature, є криптографічним підписом даних, що відправляються. Підпис робиться за алгоритмом ECDSA з хеш-функцією SHA256.

Відповідно до формату JWT, всі ці три частини кодуються за допомогою Base64 і записуються послідовно з точкою-розділювачем. Приклад використання бібліотеки для генерації токена:

Лістинг 3.3 – Приклад signature

```
>>> з blockchainauth import AuthRequest
>>> private_key =
"a5c61c6ca7b3e7e55edee68566aeab22e4da26baa285c7bd10e8d2
218aa3b22901"
>>> auth_request = AuthRequest(private_key, "localhost:5000")
>>> auth_request_token = auth_request.token()
>>> print auth_request_token
eyJhbGciOiJFUzI1NiIsInR5cCI6IkpXVCJ9.eyJzY29wZXMiOltdLC
JyZWRpcmVjdF91cmkiOiJsb2NhbGhvc3Q6NTAwMCIsImp0aSI6IjI3M
WYxNGI5LTc3YmEtNDZiMy1iNmRjLTI2YzZjNjAyYWY5YiIsImV4cCI6
IjE0OTM1NTI0MDQuMDYiLCJwdWJsaWNfa2V5cyI6WyIwMjdkMjhmOTk
1MWNlNDY1Mzg5NTFlMzY5N2M2MjU4OGE4N2YxZjFmMjk1ZGU0YTE0Zm
RkNGM3ODBmYzUyY2ZlNjkiXSwiaXNzIjoiZGlkOmJ0Yy1hZGRyOjFOW
k54aG94b2Jxd3NOdlRiMTZwZGVpcXZGdmNlM1lnOFUiLCJpYXQiOiIx
NDkzNTQ4ODA3Ljk2IiwibWFuaWZlc3RfdXJpIjoibG9jYWxob3N0OjU
wMDAvbWFuaWZlc3QuanNvbiIsImRvbWFpbl9uYW1lIjoibG9jYWxob3
N0OjUwMDAifQ.EotiYm1yGOJt3qqUWKzB0FQqkk8onxEB_rBgIllIUZ
5l8gujREPPX1osWy8Wm6q8p0q81k41w1K8YyAAIgWucg
```

За допомогою бібліотеки запити можна верифікувати. Запит проходить верифікацію, якщо він містить правильний підпис, згенерований на основі даних з payload відкритим ключем зі списку public_keys, адреса поля iss – похідний відкритого ключа з public_keys, час iat вже минув, час exp ще не настав.

Запити на аутентифікацію в рамках бібліотеки представлені у вигляді об'єктів AuthResponse. AuthResponse зберігає таку інформацію:

- закритий ключ програми (private_key);
- профіль користувача у форматі JSON (profile);
- ім'я, яким користувач аутентифікується (username);
- Unix-час, через який закінчується терміндії цього запиту(expires_at). Значення за промовчанням: через місяць.

Відповідь на запит, що генерується на основі цих даних, має наступний JSON-формат, багато в чому схожий з форматом AuthRequest:

Лістинг 3.4 – AuthRequest

```
{
"header": {"type": "JWT", "alg": "ES256K"}, "payload": {
"exp": 1496141298.31,
"iat": 1493549308.76,
"iss":
"did:btc-addr:1NZNxhoxobqwsNvTb16pdeiqvFvce3Yg8U", "jti":
"b483adf1-1c3d-435c-be5e-bb88000bb2b4", "profile": {...},
"public_keys":
["027d28f9951ce46538951e3697c62588a87f1f1f295de4a14fdd4
c780fc52cfe69"],
"username": "stanislav_pankratov.id"
},
"signature":
"CoOAxoYEOAYd0ajaa31AWxFzvBhSxzdEUa-tm2LDlkICj_sNIgF-jH JY-
6WPGfk4OmLLZ-8Uos5HdWes1PviRQ"
}
```

Робота з об'єктами AuthResponse також схожа з об'єктами AuthRequest:

Лістинг 3.5 – AuthResponse

```
>>> з blockchainauth import AuthResponse
>>> STANISLAV_PANKRATOV_PROFILE = {...}
>>> private_key =
"a5c61c6ca7b3e7e55edee68566aeab22e4da26baa285c7bd10e8d2
218aa3b22901"
>>> username = "stanislav_pankratov.id"
>>> з blockchainauth import AuthResponse
>>> auth_response = AuthResponse(private_key,
STANISLAV_PANKRATOV_PROFILE, username)
>>> auth_response_token = auth_response.token()
>>> print auth_response_token
eyJhbGciOiJFUzI1NiIsInR5cCI6IkpXVCJ9.eyJw...
```

За допомогою бібліотеки відповіді на запити можна верифікувати. Відповідь проходить верифікацію, якщо виконуються ті самі умови, що й для самих запитів, а також якщо відповідь проходить додаткову верифікацію: адреса з поля iss повинна співпадати з публічною адресою власника цього імені. Для перевірки власника імені використовується локально запущений Blockstack Core API, або публічний, запущений авторами Blockstack.

3.5 django-blockstack

Щоб розробнику не доводилося вручну пробляти на своєму веб-сервері всі описані кроки, в рамках цієї роботи було розроблено програму Django для легкої інтеграції: https://github.com/spankratov/django-blockstack.

Django – найпопулярніший веб-фреймворк загального призначення на Python і п'ятий за популярністю веб-фреймворк загалом [22]. Він має вбудовану систему автентифікації та модель користувача, що відображається у базі даних. Django вимагає мінімального налаштування, щоб встановити стандартну систему зберігання паролів у зашифрованому вигляді на стороні сервера. Але Django підтримує багато інших систем аутентифікації, включаючи OAuth та OpenID. Для підключення додаткової функціональності використовуються програми Django — бібліотеки з інтеграцією з Django.

django-blockstack залежить від наступних пакетів: blockchainauth (тобто розробленої в рамках цієї роботи Python-бібліотеки), самого django та blockstack-profiles (бібліотека з відкритим вихідним кодом для роботи з Blockstack-профілями).

Одна з головних складових програми - BlockstackAuthBackend. У термінах Django це Authentication Backend. Щоразу, коли користувач намагається автентифікуватися та надає деяку інформацію про себе (наприклад, логін та пароль), Django викликає метод authenticate у всіх перерахованих в налаштуваннях Authentication Backend об'єктів. Користувач аутентифікується, якщо хоч один із них повертає об'єкт User.

Лістинг 3.5 – Користувач аутентифікується

```
class BlockstackAuthBackend(object): def authenticate(self,
request,
auth_response_token = None):
...
```

Оскільки в протоколі Blockstack користувач аутентифікується за токеном, згенерованим його клієнтом, метод authenticate об'єкта BlockstackAuthBackend приймає аргумент auth_response_token замість традиційних username та password.

За допомогою раніше створеної бібліотеки blockchainauth токен верифікується, у разі успіху користувач шукається в базі або створюється по Blockstack-імені як username. На ім'я шукається профіль, з якого в базу записується інформація про ім'я користувача.

Спеціально для пошуку профілю функція fetch_profile. Спочатку вона намагається отримати профіль із локально запущеного Blockstack Core API. Якщо сервер API не запущено або профіль з інших причин не вдалося отримати, функція намагається отримати його з віддаленого Blockstack Core API.

django-blockstack додає три посилання на сайт. Одна з них відповідає за генерацію об'єкта AuthRequest та перенаправлення користувача до його клієнта. Як redirect_uri для AuthRequest вказується друге посилання, яке приймає токен AuthResponse та аутентифікує користувача. За третім посиланням Blockstack Portal може отримати файл manifest.json сайту.

За логіку, яка стоїть за цими посиланнями, відповідають функції views у термінах Django:

Лістинг 3.6 – Views

```
# django-blockstack/views.py
def blockstack_request(request):
...


def blockstack_response(request):
...


def manifest(request):
...
```

І, нарешті, функції переглядів асоціюються з конкретними URL:

Лістинг 3.7 – Views

```
# django-blockstack/urls.py urlpatterns = [
url(r'^blockstack/request', views.blockstack_request,
name='request'),
url(r'^blockstack/response', views.blockstack_response,
name='response'), url(r'^blockstack/manifest.json',
views.manifest,
name='manifest')
]
```

Як і будь-які пакети Python, django-blockstack встановлюється однією командою через pip.

Подальші кроки описані у репозиторії на Github. Щоб інтегрувати аутентифікацію Blockstack на своєму сайті, розробнику потрібно:

Додати “django_blockstack” до списку встановлених програм:

Лістинг 3.8 – Встановлення django_blockstack

```
INSTALLED_APPS = [
...
'django_blockstack',
]
```

Додати “django_blockstack.backends.BlockstackAuthBackend” вналаштування AUTHENTICATION_BACKENDS:

Лістинг 3.9 – Встановлення django_blockstack

```
AUTHENTICATION_BACKENDS = [
'django_blockstack.backends.BlockstackAuthBackend',
...
]
```

Конфігурувати програму через налаштування BLOCKSTACK_AUTH. Всі дані settings використовуються для ініціалізації запиту на автентифікацію, це поле має обов'язково містити значення domain_name. Поле manifest містить обов'язкову інформацію про сайт. blockstack_api – опціональне налаштування, використовується, якщо на сервері запущено Blockstack Core у режимі API на нестандартному порті.

Лістинг 3.10 – BLOCKSTACK_AUTH

```
BLOCKSTACK_AUTH = {
'settings': {
'private_key':
'a5c61c6ca7b3e7e55edee68566aeab22e4da26baa285c7bd10e8d2
218aa3b22901',
'domain_name': 'http://localhost:8000'
},
'manifest': {
'name': 'Blockstack Auth Test',

'start_url': 'http://localhost:8000', 'description': 'A simle
demo of Blockstack
Auth',
'icons': [{
'src': 'http://localhost:8000/static/icon.png',
'sizes': '420x420', 'type': 'image/png'
}]
},
'blockstack_api': 'http://localhost:6270'
}]
}
```

Підключити URL Blockstack у головному файлі проекту urls.py:

При спробі користувача аутентифікуватись через Blockstack сайт просто перенаправляє його на посилання /blockstack/request.

Коли сайт повинен отримати доступ до профілю користувача, він викликає метод fetch_profile

3.6 Розробка сайту

Для тестування протоколу був написаний простой сайт, Котрий відображає основну інформацію з профілю користувача Blockstack: ім'я, аватар, фонове зображення та посилання на підтверджені акаунти в Github, Facebook та Twitter. Сайт написаний на Django та використовує програму django-blockstack:https://github.com/spankratov/django-blockstack-test

При першому відвідуванні сайт пропонує користувачеві автентифікуватись (рисунок 3.4).

При натисканні на кнопку користувача перенаправляють на Blockstack Portal, де відображається основна інформація про програму (ім'я, іконка) (рисунок 3.5).

Рисунок 3.4 – Домашня сторінка тестового сайту. Користувач не автентифікований

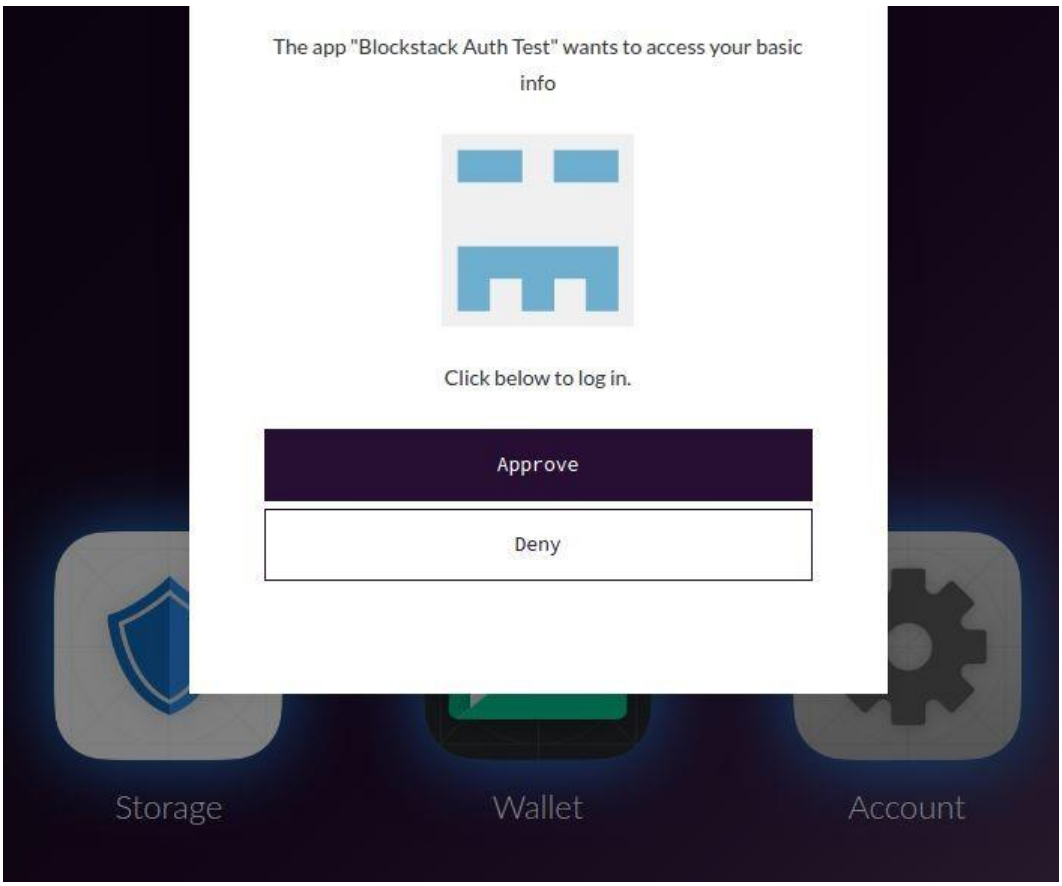

Рисунок 3.5 – Інтерфейс Blockstack Portal, запит на автентифікацію тестового сайту

Як тільки користувач натискає кнопку Approve, його перенаправляють назад на сайт, де він аутентифікується. Видно основна інформація з його профілю (рисунок 3.6).

4 АНАЛІЗ РІШЕННЯ

У цьому розділі буде детально розглянуто основні проблеми, які вирішує протокол Blockstack. Буде описано переваги для користувачів щодо зручності зберігання паролів. Будуть вивчені вектори атаки на централізовані системи зберігання облікових записів користувачів і чому подібні атаки неможливі при використанні протоколу Blockstack. Також необхідно порушити питання порушень з боку централізованого провайдера імен. Наприкінці будуть наведені недоліки, які існують у протоколі

4.1 Універсальність

Однією з незаперечних переваг протоколу є єдиний вхід від однієї й тієї імені на безліч сервісів. У світі з величезною кількістю веб-сервісів це позбавляє необхідності створювати безліч різних облікових записів для цих сервісів, щоразу наново заповнювати всю необхідну інформацію, зберігати паролі окремо для кожного. Ім'я у системі Blockstack може використовуватись на всіх сайтах, які підключили цей протокол, що можна легко зробити за допомогою бібліотек, розроблених у рамках цієї роботи. Таким чином, вкладом цієї роботи є спрощення застосування протоколу Blockstack як протоколу єдиного входу.

Однак технології єдиного входу не є новими. Акаунти в таких сервісах, як Google або Facebook, використовуються для аутентифікації на безлічі веб-сайтів, і вони так само зручні для користувачів тим, що дозволяють не створювати окремі облікові записи для кожного сайту.

У наступних розділах порівняння протоколу Blockstack проводитиметься лише з сервісами, що надають механізми Single Sign On.

4.2 Безпека

Найбільшою проблемою, пов'язаною із централізованими сховищами облікових записів, є безпека. Оскільки облікові записи користувачів зберігаються не в самих користувачів, вони не можуть контролювати заходи безпеки, які вживаються компаніями-постачальниками імен. Сервери компаній можуть бути зламані, і через них зловмисники можуть отримати доступ до облікових записів користувачів.

Зламування облікового запису на одному з сервісів не завжди означає компрометацію всіх облікових записів користувача, але на практиці користувачі часто використовують один і той же пароль для декількох сервісів. Якщо хакери дізнаються пароль від облікового запису на якомусь малозначимому сервісі, а той же пароль використовується, наприклад, для електронної пошти, то зловмисник отримає доступ відразу і до пошти, і до всіх прив'язаних до неї облікових записів.

Особливо небезпечні масові зломи постачальників імен, що надають різними протоколи Single Sign On (OAuth, OpenID), або зломи провайдерів електронної пошти, оскільки до електронної пошти часто-густо прив'язуються інші облікові записи. Це створює принципово дуже небезпечну ситуацію, коли великі постачальники імен (Google, Facebook, Twitter, Microsoft, Yahoo, Yandex, VK) є гігантською точкою відмови всієї мережі: злом одного такого провайдера викликає витік колосальної кількості облікових записів, що в свою чергу спричиняє компрометацію облікових записів на інших сервісах, які використовують аутентифікацію через зламаного провайдера.

Щоб оцінити масштаби такого злому, необхідно оцінити частку облікових записів всесвітньої павутини, у тому чи іншому вигляді прив'язаних до постачальників осіб. І хоча повноцінної статистики з цих питань немає, можна все одно простежити загальні тенденції щодо непрямих даних.

Розглянемо звіт 2016 року, проведений компанією-розробником рішень автентифікації LoginRadius [28]. Згідно з аналізом 160000 веб-сайтів з функцією Social Login (вхід через соціальні мережі), 93% користувачів цих сайтів надавали перевагу входу через соціальні мережі звичайній реєстрації електронною поштою (рисунок 4.1).

При цьому переважна більшість користувачів, які вибрали Social Login, є користувачами двох великих постачальників імен: Facebook та Google (рисунок 4.3).

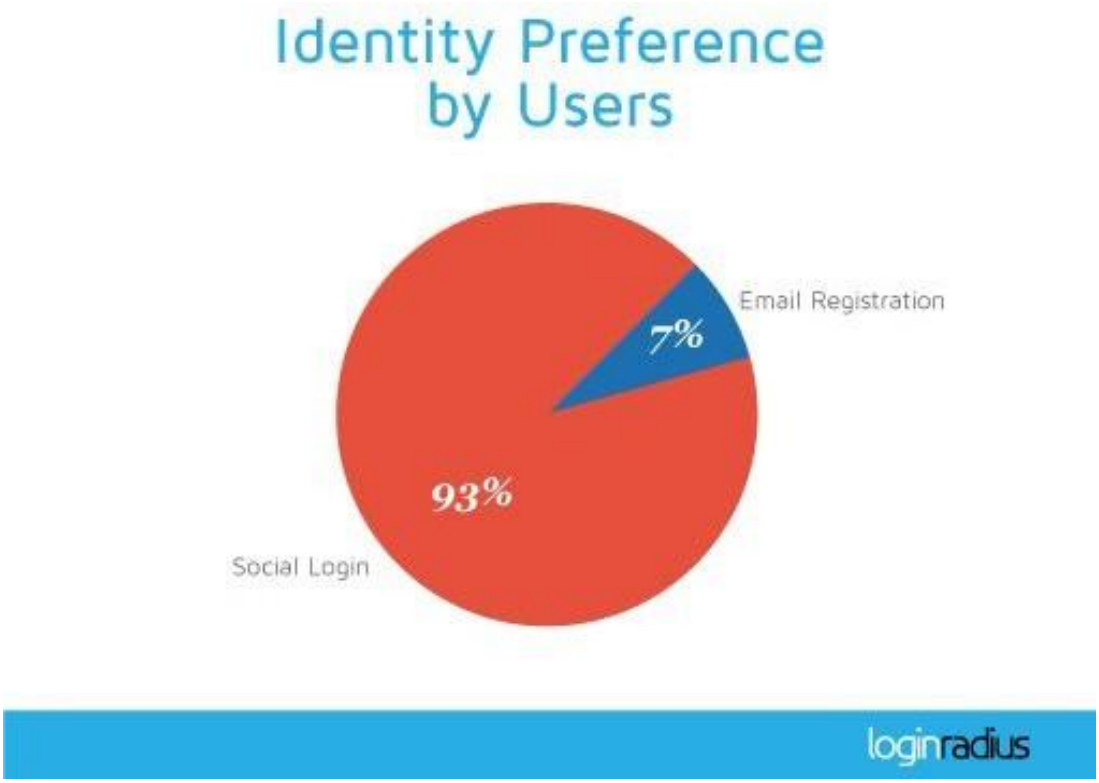

Рисунок 4.1 – Переваги користувачів у методах автентифікації. Вхід через соціальні мережі проти стандартної реєстрації через пошту

На рисунку 4.2 показано наступну форму – після вибору варіанту приховування даних для одного файлу.

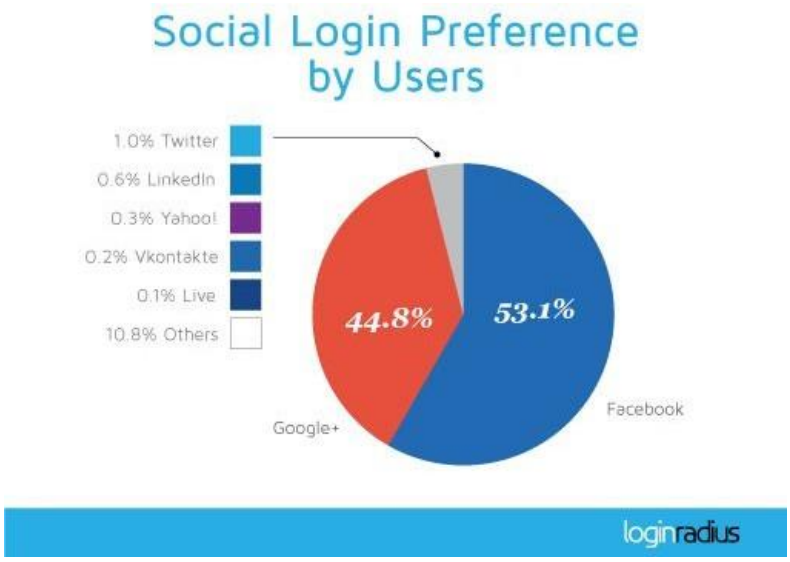

Рисунок 4.2 – Розподіл користувачів із сервісів Social Login

З цього можна дійти невтішного висновку, що це користувачі, скористалися можливістю аутентифікуватися через соціальні мережі, перебувають під загрозою крадіжки їх акаунта у разі злому їх постачальника особи, причому майже 98% користувачів залежить від двох сервісів, які є великими точками відмови. Усі користувачі цих двох сервісів (майже 2 мільярди активних користувачів Facebook [29] та більше мільярда активних користувачів Google [30]) можуть скористатися послугами єдиного входу та потрапити у залежність від цих постачальників імен.

Власники безлічі сайтів дозволяють своїм користувачам автентифікуватися через облікові записи великих постачальників імен. Так, через Facebook можна автентифікуватись приблизно на 14,5 мільйонах сайтів [31].Невсевеб-сайти, однак, надають можливість автентифікуватись через постачальників імен. Тим не менш, якщо веб-сайт не надає цієї можливості, то практично завжди з метою захисту від роботыв і від забудькуватості користувача требує при реєстрації електронну пошту, через яку користувач згодом може відновити доступ до облікового запису. Злом постачальника електронної пошти може спричинити крадіжку всіх облікових записів, прив'язаних до пошти користувачів.

Як і у випадку з соціальними мережами, більшість облікових записів прив'язані до кількох великих постачальників послуг. Згідно з звітом MailChimp, великого сервісу розсилки електронної пошти, що досліджував 81,69 мільярда листів, надісланих з їх серверів, більшість користувачів польуються поштою від п'яти великих компаній: Google, Hotmail, Yahoo, AOL і Comcast [32].

Хоча і не маючи якихось точних оцінок через відсутність потрібної статистики, можна зробити висновок, що з точки зору безпеки користувачі-власники облікових записів та імен знаходяться в занадто великій залежності від кількох великих постачальників послуг і піддаються ризику компрометації своїх облікових записів при компрометації цих сервісів.

Можна було б сказати, що великі компанії наймають досить кваліфікований персонал і захищають дані своїх користувачів настільки надійно, наскільки це можливо. Однак від помилок не застрахований ніхто, і, як показує історія, перед зломами вразливі навіть гігантські корпорації. Декілька найбільш значущих прикладів:

Злом серверів Sony у 2011 році. Паролі зберігалися в нехешованому вигляді, а доступ до них був отриманий звичайною SQL-ін'єкцією [23].

Шість із половиною мільйонів паролів користувачів LinkedIn були вкрадені у 2012 році. Паролі були захешовані, але без криптографічної солі. Це дозволило хакерам легко отримати паролі з вкрадених хешів [24].

Близько мільярда облікових записів користувачів Yahoo в 2013 році були скомпрометовані через підроблені cookie-файли [25].

Фішингова атака на електронні скриньки Google у 2017 році, було зламано близько мільйона облікових записів [26]. Облікові записи були зламані через вразливість у протоколі OAuth, про яку дослідники попереджали ще в 2011 році [27].

Небезпека може критися і не на боці самої компанії, а десь серед технологій, які вона використовує. Появу таких уразливостей складно контролювати і неможливо передбачити. Так, одну з найвідоміших

уразливостей останніх років, Heartbleed, знайшли в OpenSSL – програмному забезпеченні, що лежить в основі сучасного Інтернету. Виявилися вразливими сервери таких компаній, як Amazon, GitHub, Pinterest,

Reddit, Tumblr, Wikimedia, Yahoo. Вразливість була присутня в коді OpenSSL протягом двох років [33].

На прикладі Heartbleed можна побачити, як легко зловмиснику, знаючи конкретну вразливість, отримати доступ до облікових записів користувачів. Для аналізу веб-сайту досить вільно розповсюджуваних програм [34]. Ось як виглядає аналіз вмісту пам'яті сервера за допомогою програми Metasploit Framework:

Лістинг 4.1 – Metasploit Framework

```
root@kali :~# msfconsole
msf > use auxiliary/scanner/ssl/openssl_heartbleed msf
auxiliary(openssl_heartbleed) > set RHOSTS 192.168.154.137
RHOSTS => 192.168.154.133
msf auxiliary(openssl_heartbleed) > run
[*] 192.168.154.137:443 - Sending Client Hello... [*]
192.168.154.137:443 - Sending Heartbeat...
[*] 192.168.154.137:443 - Heartbeat response, checking if there
is data leaked...
[+] 192.168.154.137:443 - Heartbeat response with leak
Gecko)
Chrome/34.0.1847.116 Safari/537.36Accept-Encoding:
gzip,deflate,sdchAccept-Language: pt-BR,pt;q=0.8,en-
US;q=0.6,en;q=0.4Cookie: nessus-session=false ] 5gOZ pt-BR, pt;
q = 0.8, en-
US;q=0.6,en;q=0.4Cookie: nessus-session=falseql|Z4EKT [*]
Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution
```

За оцінками дослідників, зробивши всього 3 таких запити на хвилину, зловмисник може отримати 30% усієї пам'яті, яку зараз читає сервер [35]. Якщо зловмисник шукає у пам'яті пари "логін-пароль", то досить скоро він наткнеться на дані такого вигляду, що прийшли відкористувача веб-програми у вигляді POST-запиту на автентифікацію:

Лістинг 4.3 – POST-запит на автентифікацію

```
 [*] 192.168.154.137:443 - Printable info leaked:
… login_username=user123&password=jh345sfhds32
…
```

Використовуючи тестовий фреймворк RUBiS, що імітує сайт інтернет-аукціону, дослідники змогли за всього 100 активних користувачів за 10 хвилин отримати близько 7000 рядків з парами "логін-пароль", експлуатуючи вразливість Heartbleed [36].

Звичайно, розроблена в рамках цієї роботи система не вирішує глобальне питання вразливостей у програмному забезпеченні. Навіть при використанні протоколу Blockstack зловмисники можуть зробити наступні атаки:

Атака на комп'ютер користувача. Хоча приватні ключі і зашифровані локально майстер-паролем користувача, залежно від отриманих прав доступу зловмисник все одно може вкрасти обліковий запис користувача (наприклад, виудавши у користувача пароль кейлоггером або представившись на сайті під ім'ям користувача через існуючу сесію в Blockstack Portal).

Атака на сервер програми, що використовує протокол аутентифікації Blockstack. Зловмисник може отримати доступ до всієї користувальницької інформації, що стосується цієї програми (наприклад, до особистої листування в соціальній мережі).

Однак при використанні протоколу Blockstack принципово неможливий той сценарій розвитку подій, коли злом одного сервісу спричиняє крадіжку акаунтів на іншому. Злом однієї сторони аутентифікаційного процесу виливається або в компрометацію одного облікового запису (у разі злому локального комп'ютера користувача), або в компрометацію багатьох облікових записів, пов'язаних тільки з цією стороною (у разі злому сервера програми). Постачальник імен у разі

замінюється блокчейном, і злом сховища імен рівноцінний компрометації гігантського блокчейна Bitcoin і контролю понад 51% вузлів мережі.

У рамках цієї роботи як протокол аутентифікації свідомо був обраний протокол Blockstack через свої сильні характеристики безпеки. При активному використанні цього протоколу в мережі може зникнути проблема великих точок відмови у вигляді постачальників.

4.4 Децентралізованість

При класичному сценарії зберігання імен користувачів, коли зберігання здійснюється на одній стороні (компанії-постачальнику імен), фактично ім'я контролюється не користувачем, а самою компанією. Компанія-постачальник на свій розсуд може:

- повністю видалити ім'я користувача;
- змінити дані;
- переглядати його дані та передавати третім сторонам. Приклад: PRISM;
- використовувати дані користувачів у комерційних цілях. Приклад:таргетована реклама.

Тому і не можна говорити, що користувач повною мірою має ім'я. Ім'ям володіє компанія, яка дає його користувачеві у безстрокову оренду на своїх умовах.

Централізоване зберігання імен користувачів може бути небезпечним ще й тим, що компанія-постачальник може просто припинити існування. В найближчому майбутньому це малоймовірний результат для найбільших компаній (Google, Facebook), але є і безліч дрібніших і менш стабільних постачальників імен, від яких залежать користувачі.

Протокол Blockstack радикально вирішує проблему, використовуючи блокчейн – децентралізовану базу даних. Жоден вузол мережі не може повністю контролювати систему, нові блоки додаються до блокчейну на

основі консенсусу безлічі вузлів. Перераховані вище проблеми в блокчейні можуть виникнути тільки при порушенні роботи більше 51% вузлів блокчейну Bitcoin. У травні 2017 року кількість повних вузлів мережі Bitcoin – близько 7000 [37]. Це означає, що для компрометації назв імен зловмисникам довелося б контролювати більше 3500 повних вузлів мережі, що представляється практично неможливим.

4.5 Недоліки та можливості їх подолання

Як і будь-яка технологія, протокол Blockstack та розроблена навколо нього система має свої недоліки.

Одним із головних, непереборних недоліків є невисока швидкість роботи системи загалом. Це випливає з важливих особливостей протоколу. Оскільки всі імена зберігаються в блокчейні Bitcoin, всі операції так чи інакше пов'язані зі зміною цього сховища (реєстрація імені, оновлення zone-файлу, передача імені) проходять верифікацію та приймаються чи відкидаються з урахуванням консенсусу кількох вузлів мережі. Така транзакція може проходити від кількох хвилин до кількох годин.

Через свою новизну протокол перебуває в ще дуже сирому стані і не зовсім зручний для звичайних користувачів:

За кожну операцію, пов'язану з володінням ім'ям і зміною блокчейна, користувачеві доводиться платити невелику суму в біткоінах. Це відбувається з того, що кожна така операція – це транзакція Bitcoin, яку необхідно підписати закритим ключем користувача. А всі транзакції Bitcoin мають на увазі передачі хоча б невеликої суми.

Щоб скористатися протоколом, користувач змушений ставити додаткове програмне забезпечення (Blockstack Core, Blockstack Portal), яке знаходиться на ранній стадії розробки і часто не готове для зручної установки. Так, готовий інсталяційний пакет існує тільки для операційної системи OS X. На даний момент користувачі Windows та Linux змушені

вручну збирати та запускати всі частини програмного забезпечення, хоча розробники активно працюють над виправленням цієї проблеми.

Поки що існує дуже мала кількість сайтів, де підключено протокол і де користувач може спробувати автентифікацію через Blockstack.

У силу того, що протокол знаходиться на ранній стадії розвитку, він поки не набув широкого застосування. Проте він дуже важливий хоча б як доказ того, що така важлива частина життя, як ідентифікація людини, може керуватися без контролю третіми сторонами. Крім того, протокол має всі шанси поширитися по всьому світу завдяки підтримки Microsoft, із якими співпрацюють розробники Blockstack, щоб створити децентралізовану систему ідентифікації [3].

Головним внеском цієї роботи є часткове подолання незручності, що з роботою з протоколом. Так, розробники традиційних серверних додатків не мали можливості легко вбудувати автентифікацію користувачів на сервері. В інфраструктурі Blockstack є бібліотека Javascript, blockstack.js, яка може генерувати запити на автентифікацію та відповіді на них, але ця бібліотека призначена в першу чергу для безсерверних додатків. Такий підхід передбачає проходження автентифікації на стороні клієнта, що може призвести до зловживань (наприклад, підміни коду, спробі видати себе за інших людей). Завдяки розробленим бібліотекам всі веб-сайти, що використовують Python та Django, можуть вбудувати автентифікацію користувачів через Blockstack, що сприяє більш широкому прийняттю протоколу.

Розробники Blockstack працюють над подоланням перерахованих недоліків, і є всі підстави вважати, що незабаром протокол автентифікації буде зручнішим для широкого застосування.

ВИСНОВКИ

У цій роботі було проаналізовано підходи до аутентифікації, що застосовуються у світі. Були виявлені критичні недоліки, такі як централізованість та залежність процесу аутентифікації від третіх сторін. Аналіз великих витоків інформації та зловживань із боку постачальників імен показав серйозність цих недоліків.

Як відправна точка було обрано перспективний протокол автентифікації децентралізованої системи імен Blockstack. За рахунок зберігання імен у блокчейні він сам по собі вирішував наведені вище недоліки, проте через свою новизну був занадто незручний як для звичайних користувачів, так і для розробників веб-додатків, які могли б вбудувати автентифікацію через Blockstack у свою програму.

Вклад цієї роботи – спрощення встановлення системи автентифікації Blockstack для розробників веб-застосунків.

Python-бібліотека blockchainauth призначена для зручної генерації запитів на аутентифікацію та відповідей на запит аутентифікації. Ця бібліотека може використовуватися як сама по собі, для ручного керування процесом аутентифікації з боку розробника, так і у складі Django-додатку django-blockstack, що дозволяє за допомогою однієї команди установки пакета та кількох рядків конфігураційних файлів вбудувати аутентифікацію через Blockstack на веб-сайт.

Розробка тестового веб-сайту показала, що використання цих бібліотек зручно і не змушує розробника вкотре замислюватися про налаштування автентифікації.

У рамках цієї роботи були отримані такі результати:

Проаналізовано існуючі протоколи аутентифікації, їх переваги та недоліки. Було виділено клас найперспективніших протоколів з урахуванням технології блокчейн.

Було обрано для вивчення один конкретний протокол із системи Blockstack, який використовує раніше розроблену інфраструктуру: сховище особистостей у блокчейні, вузли мережі, додаток-клієнт. Було вивчено його можливості, переваги та недоліки, докладно розглянуто його застосування в автентифікації користувачів.

Було розроблено дві бібліотеки мовою Python: blockchainauth і django-blockstack. Обидві бібліотеки є придатними для легкої інтеграції системи аутентифікації Blockstack на веб-сайти.

Був розроблений тестовий сайт, який повною мірою використовує бібліотеки blockchainauth і django-blockstack.

Система була докладно вивчена, були розглянуті проблеми існуючих систем аутентифікації та як вона їх долає. Було вивчено недоліки самого протоколу Blockstack і як вони частково вирішуються за допомогою розроблених бібліотек.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Офіційний сайт ID2020. - URL: http://id2020.org/

2. . Kaliski. PKCS #5: Password-Based Cryptography Specification Version 2.0 // RFC 2898. – 2000. – С. 9-11

3. Microsoft Building Open Blockchain-Based Identity System With Blockstack, ConsenSys, Bitcoin Magazine. – URL: https://bitcoinmagazine.com /articles/microsoft-building-open-blockchain-based-id entity-system-with-block stack-consensys-1464968713

4. Introducing the Blockstack Identity System, Blockstack blog. — URL: https://blockstack.org/blog/introducing-the-blockstack-identity-system

5. Dr. Christian Lundkvist, Rouven Heck, Joel Torstensson, Zac Mitton, Michael Sena. UPORT: A PLATFORM FOR SELF-SOVEREIGN IDENTITY. – URL: https://whitepaper.uport.me/uPort_whitepaper_DRAFT20170221.pdf – С. 2

6. Andreas M. Antonopoulos. Mastering Bitcoin: Unlocking digital currencies. – "O'Reilly Media, Inc." – 2014.

7. Занурення у технологію блокчейн: Децентралізована безпарольна система безпеки. Хабрахабр, блог компанії Microsoft. – URL: https://habrahabr.ru/company/microsoft/blog/316864/

8. EMCSSL – Система идентификации пользователей WWW на основе подсистемы NVS криптовалюты EmerCoin и децентрализованных клиентских SSL-сертификатов. Хабрахабр. – URL: https://habrahabr.ru/post/257605/

9. Офіційний сайт Remme. – URL: http://remme.io/

10. Blockstack | AgenlList. – URL: https://angel.co/blockstack

11. Blockstack – Intro, офіційний сайт Blockstack. – URL: https://blockstack.org/intro

12. Muneeb Ali, Jude Nelson, Ryan Shea, Michael J. Freedman. Bootstrapping Trust in Distributed Systems with Blockchains // ;login: – VOL. 41, NO. 3 – 2016. – С. 56

13. Blockstack, Onename, and future applications – Identity – Blockstack Forum. URL – https://forum.blockstack.org/t/blockstack-onename-and-future-applications/529

14. Muneeb Ali, Jude Nelson, Ryan Shea, Michael J. Freedman. Blockstack: A Global Naming and Storage System Secured by Blockchains // 2016 USENIX Annual Technical Conference. – 2016. – С. 188

15. Diploma project about Blockstack authentication protocol, Blockstack forum. – URL: https://forum.blockstack.org/t/diploma-project-about-blockstack-authentication-pr otocol/850

16. Library for scanning blockchains and running Blockstack state engines, Github. – URL: https://github.com/blockstack/virtualchain

17. Muneeb Ali, Jude Nelson, Ryan Shea, Michael J. Freedman. Blockstack: A Global Naming and Storage System Secured by Blockchains // 2016 USENIX Annual Technical Conference. – 2016. – С. 184—186

18. OP_RETURN Stats. – URL: http://opreturn.org

19. The reference implementation of Blockstack, Github. – URL: https://github.com/blockstack/blockstack-core

20. The Blockstack Browser Portal, Github. – URL: https://github.com/blockstack/blockstack-portal

21. JSON Web Tokens, jwt.io – URL: https://jwt.io

22. Шокота Т. А. (2022). Аналіз систем автентифікації. Збірник тез доповідей підготовлено за матеріалами Міжнародної наукової інтернет-конференції (випуск 69) 4-5 липня 2022 р. на сайті www. konferenciaonline. org. ua, 131.