НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
Навчально-науковий фізико-технічний інститут
Кафедра математичних методів захисту інформації

«На правах рукопису»
УДК 043.2

«До захисту допущено»
В.о. завідувача кафедри
_____ Сергій ЯКОВЛЄВ
«__» _____ 2023 р.

Дипломна робота

на здобуття ступеня бакалавра
за освітньо-професійною програмою
«Математичні методи криптографічного захисту інформації»

зі спеціальності: 113 Прикладна математика
на тему: **«Побудова атак на протокол реєстрації супутнього пристрою в WhatsApp»**

Виконала:
студентка IV курсу, групи ФІ-94
Немкович Ольга Михайлівна _____

Керівник:
к.ф.-м.н., ст. викладач
Фесенко Андрій В'ячеславович _____

Рецензент:
ст. викладач, доктор філософії
Яйлимова Ганна Олексіївна _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студентка _____

Київ — 2023

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
Навчально-науковий фізико-технічний інститут
Кафедра математичних методів захисту інформації

Рівень вищої освіти — перший (бакалаврський)
Спеціальність — 113 Прикладна математика,
ОПП «Математичні методи криптографічного захисту інформації»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Сергій ЯКОВЛЄВ
«__» _____ 2023 р.

ЗАВДАННЯ
на дипломну роботу

Студентка: Немкович Ольга Михайлівна

1. Тема роботи: «*Побудова атак на протокол реєстрації супутнього пристрою в WhatsApp*», науковий керівник дипломної роботи: к.ф.-м.н., ст. викладач Фесенко Андрій В'ячеславович,

затверджені наказом по університету №__ від «__» _____ 2023 р.

2. Термін подання студентом роботи: «__» _____ 2023 р.

3. Об'єкт дослідження: інформаційні процеси в системах захищеного обміну повідомленнями

4. Предмет дослідження: стійкість протоколу реєстрації супутніх пристроїв в системі захищеного обміну повідомленнями WhatsApp

5. Перелік завдань: дослідження стійкості протоколів реєстрації супутніх пристроїв у системах Signal та WhatsApp; порівняльний аналіз; побудова атак; розгляд протоколу реєстрації в системах Session і Viber

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: презентація доповіді

7. Орієнтовний перелік публікацій: опубліковано на всеукраїнській науково-практичній конференції

8. Дата видачі завдання: 10 вересня 2022 р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання	Примітка
1	Узгодження теми роботи із науковим керівником	01-15 вересня 2022 р.	Виконано
2	Огляд опублікованих джерел за тематикою дослідження	Вересень-жовтень 2022 р.	Виконано
3	Дослідження протоколу реєстрації супутніх пристроїв в системі миттєвих повідомлень Signal	Листопад - грудень 2022 р.	Виконано
4	Аналіз протоколу реєстрації в месенджері WhatsApp	Січень-лютий 2023 р.	Виконано
5	Порівняльний аналіз реєстрації супутніх пристроїв в системах Signal та WhatsApp	Березень 2023 р.	Виконано
6	Побудова атак на протокол реєстрації супутнього пристрою в WhatsApp.	Квітень 2023 р.	Виконано
7	Розглянути протокол реєстрації в системах обміну повідомленнями Session та Viber. Аналіз змін в протоколі реєстрації в системі WhatsApp.	Травень 2023 р.	Виконано
8	Оформлення диплому	Червень 2023 р.	Виконано

Студентка _____ Ольга Немкович

Керівник _____ Андрій Фесенко

РЕФЕРАТ

Кваліфікаційна робота містить: 63 стор., 9 рисунків, 6 таблиці, 34 джерела.

Метою є дослідження стійкості протоколу реєстрації супутніх пристроїв в системі захищеного обміну повідомленнями WhatsApp.

Об'єктом дослідження є інформаційні процеси в системах захищеного обміну повідомленнями.

Предметом дослідження є стійкість протоколу реєстрації супутніх пристроїв в системі захищеного обміну повідомленнями WhatsApp.

В результаті проведеного дослідження проаналізовано особливості протоколів реєстрації супутніх пристроїв в різних системах захищеного обміну повідомленнями. Проведено порівняльний аналіз протоколів реєстрації в системах WhatsApp і Signal, виявлено відмінності та недоліки в протоколі реєстрації системи WhatsApp, які впливають на безпеку та конфіденційність користувачів.

Побудовано вперше атаки на новітній протокол реєстрації супутніх пристроїв в системі WhatsApp, які дозволяють зловмиснику отримати доступ до всіх даних користувача, відновити секретний особистий ключ або зареєструвати зловмисний супутній пристрій.

На основі проведеного детального дослідження протоколу реєстрації супутніх пристроїв в системі WhatsApp та інших системах захищеного обміну повідомленнями було сформовано загальні рекомендації до побудови протоколів такого типу.

СИСТЕМИ ЗАХИЩЕНОГО ОБМІНУ ПОВІДОМЛЕННЯМИ, СИСТЕМА WHATSAPP, ПРОТОКОЛИ РЕЄСТРАЦІЇ ДОДАТКОВИХ ПРИСТРОЇВ

ABSTRACT

The qualification work contains: 63 p., 9 drawings, 6 tables, 34 sources

The research objective is the security analysis of the companion device registration protocol in the WhatsApp private messaging system.

The object of the research is the information processes in private messaging systems.

The subject of the research is the security of the companion device registration protocol in the WhatsApp private messaging system.

As a result of the research, the properties of the companion device registration protocols in various systems of private messaging were analyzed. A comparative analysis of the registration protocols in the WhatsApp and Signal systems was carried out, revealed differences and flaws in the WhatsApp registration protocol that affect user security and privacy.

Attacks on the recent companion device registration protocol in the WhatsApp system have been constructed for the first time, which allow an attacker to gain access to all user data, recover a secret private key, or register a malicious companion device.

General recommendations for the construction of protocols were formulated which are based on a detailed analysis of the companion device registration protocols in the WhatsApp system and other systems of private messaging,

ЗМІСТ

ВСТУП

Актуальність дослідження. WhatsApp є одним з найпопулярніших месенджерів у світі, яким користуються мільйони людей. Його актуальність продовжує зростати щодня. Обсяги особистої інформації користувачів з кожним днем зростають. Все більше людей проводять конфіденційні розмови в месенджерах, надіючись, що ця інформація залишиться приватною і недоступною для сторонніх осіб. Захищені месенджери допомагають запобігти кіберзлочинам, таким як шпигунство, крадіжка особистої інформації, фішинг або викрадення облікових даних

На відміну від інших складових частин, саме протоколи реєстрації є найменш стандартизованими. З огляду на використання багатьох пристроїв та зручності перемикатися між ними, реєстрація декількох пристроїв стала важливою функцією для користувачів. Актуальним питанням залишається безпека реєстрації та користування кількома супутніми пристроями.

Метою дослідження є дослідження стійкості протоколу реєстрації супутніх пристроїв в системі захищеного обміну повідомленнями WhatsApp. Для досягнення мети необхідно вирішити такі завдання:

1) провести огляд опублікованих джерел за тематикою дослідження;

2) проаналізувати протокол реєстрації супутніх пристроїв в системі захищеного обміну повідомленнями Signal та наявні атаки на нього;

3) дослідити особливості побудови новітнього протоколу реєстрації супутніх пристроїв в системі WhatsApp;

4) провести порівняльний аналіз особливостей реєстрації супутніх пристроїв в системах Signal та WhatsApp;

5) побудувати нові атаки на протокол реєстрації супутнього пристрою в системі WhatsApp;

6) розглянути протоколи реєстрації супутніх пристроїв в системах

Session і Viber та побудувати загальні рекомендації до протоколу реєстрації супутніх пристроїв.

Об'єктом дослідження є інформаційні процеси в системах захищеного обміну повідомленнями.

Предметом дослідження є стійкість протоколу реєстрації супутніх пристроїв в системі захищеного обміну повідомленнями WhatsApp.

При розв'язанні поставлених завдань використовувались такі *методи дослідження*: методи лінійної та абстрактної алгебри, симетричної та асиметричної криптографії.

Наукова новизна отриманих результатів полягає в побудові одного з перших досліджень новітнього протоколу реєстрації супутніх пристроїв в системі WhatsApp; побудові перших атак на протокол реєстрації супутніх пристроїв системи WhatsApp, результатом застосування яких є реєстрація зловмисного пристрою або отримання ключових даних.

Практичне значення результатів полягає в тому, що доведена нестійкість протоколів реєстрації супутніх пристроїв призвела до побудови атак, які практично реалізуються і призводять до втрати конфіденційних даних користувача, що може вплинути на вибір системи захищеного обміну повідомленнями користувача.

Апробація результатів та публікації. Основні результати роботи представлено на XXI Всеукраїнській науково-практичній конференції студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики» (11-12 травня 2023 р., м. Київ, Україна).

1 ПОПУЛЯРНІ ПРОТОКОЛИ МЕСЕНДЖЕРІВ. АНАЛІЗ СТІЙКОСТІ СИСТЕМИ SIGNAL

Месенджери відіграють важливу роль у сучасному житті. Вимогами до них є захищеність, анонімність та гарантія конфіденційного спілкування. Спочатку розглянемо одну з найкращих сучасних систем обміну повідомленнями сьогодення — Signal. Розберемо коротко його складові: розширений потрійний протокол Діффі-Хеллмана (англ. X3DH), протокол подвійного храповика (англ. Double Ratchet Algorithm) та протокол Sesame. Розглянемо наявні недоліки системи Signal, а також наявні атаки на її підпротоколи. Проаналізуємо можливості покращення властивостей систем захищеного обміну повідомленнями з підвищенням анонімності, гарантій безпеки та захисту від компрометації.

1.1 Актуальність потреби створення системи Signal

Конфіденційність даних завжди була важливим аспектом у комунікаціях. Масові спроби стеження змусили користувачів задуматись про приватність спілкування. У відповідь вчені та розробники запропонували методи, які можуть забезпечити безпеку для кінцевих користувачів, навіть якщо вони не повністю довіряють постачальникам послуг. Хоча під час появи більшість додатків для обміну повідомленнями зосереджувалися на функціях, а не на безпеці, за останні роки це змінилося: відтоді багато адаптували наскрізне шифрування як стандартну функцію.

Ранні системи обміну миттєвими повідомленнями не забезпечували потрібної безпеки. Хоча деякі системи шифрували трафік між користувачем і постачальником послуг, проте постачальник послуг зберігав можливість читати відкритий текст повідомлень користувачів.

Протокол Off the Record (OTR) [1], [2] був одним із перших протоколів безпеки для обміну миттєвими повідомленнями: діючи як плагін до різноманітних програм обміну миттєвими повідомленнями, користувачі могли автентифікувати один одного за допомогою відкритих ключів або загальної секретної фрази-паролю та отримати наскрізну конфіденційність і цілісність. Одна нова особливість OTR полягала в оновленні ключів: разом із кожним повідомленням користувачі встановлювали свіжий ефемерний спільний ключ Діффі — Хеллмана (DH).

Неможливість перейти від наступного стану до попереднього та розшифрувати минулі повідомлення стала відомою як алгоритм храповика. OTR отримав відносно обмежене застосування, але його техніку храповика можна побачити в сучасних протоколах безпеки.

Храповий механізм (ratchet) — це просто пристрій, який рухається лише вперед, крок за кроком. Він складається з круглої шестерні або лінійної зубчастої рейки зі сходинками та поворотної підпружиненої собачки. Зубці однакові, але зазвичай асиметричні.

Механізм зображено на рис. 1.1. Коли шестерня рухається вперед,

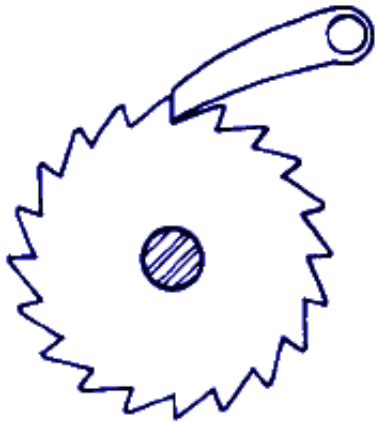

Рисунок 1.1 – Храповий механізм

то собачка легко ковзає вгору по плавно нахиленим краям зубів за допомогою пружини, що змушує її впадати у заглиблення між східцями,

коли він проходить через кінчик кожного зубця. Однак, коли шестерня рухається в протилежному напрямку, собачка зачепиться за крутий край першого зуба, який зустрінеться, тим самим зафіксувавши його на сходинці та запобігаючи будь-якому подальшому руху в цьому напрямку.

Основною відмінністю храпових протоколів є постійна зміна спільного ключа. Фактично першим храповим протоколом є протокол OTR, запропонований Борисовим та ін. в роботі [1] 2004 року. Хоча треба відзначити, що сам термін храпового (ratchet) протоколу або механізму з'явився вже пізніше.

Вперше в протоколі OTR був представлений DH храповий протокол. Його спрощений опис, оновлення та процедуру обміну ключами між A та B можна узагальнити таким чином:

$$A \to B : g^{x_1}, B \to A : g^{y_1}, A \to B : g^{x_2},$$
$$AES - CTR(M_1, k_{11}),$$
$$B \to A : G^{y_2},$$
$$AES - CTR(M_2, k_{21}),$$
$$A \to B : g^{x_3},$$
$$AES - CTR(M_3, k_{22}),$$

де $k_{ij} = trunk128(SHA - 1(g^{x_i y_j}))$, а значення $x_1, x_2, \ldots$ і $y_1, y_2, \ldots$ є випадковими значеннями, як і в протоколі узгодження ключів Діффі-Хеллмана.

Як видно, процес оновлення ключа складається з трьох кроків: передача повідомлення про ключ іншій стороні, підтвердження отримання інформації про новий ключ від іншої сторони та безпосереднє застосування цього ключа для шифрування. Тому протокол OTR іноді називають триетапним храповим протоколом (Triple ratchet).

Протокол SCIMP [3], розроблений компанією Silent Circle у 2015 році для їх програми миттєвого обміну повідомленнями Silent Text, є іншим варіантом постійного оновлення ключів в рамках храпового протоколу.

На рис. 1.2 представлена процедура генерування початкового ключа

протоколу SCIMP. Якщо учасники вже обмінювалися повідомленнями, то попереднє спільне таємне значення cs_i є непорожнім. В іншому випадку використовується випадкове значення, яке не впливає на роботу протоколу.

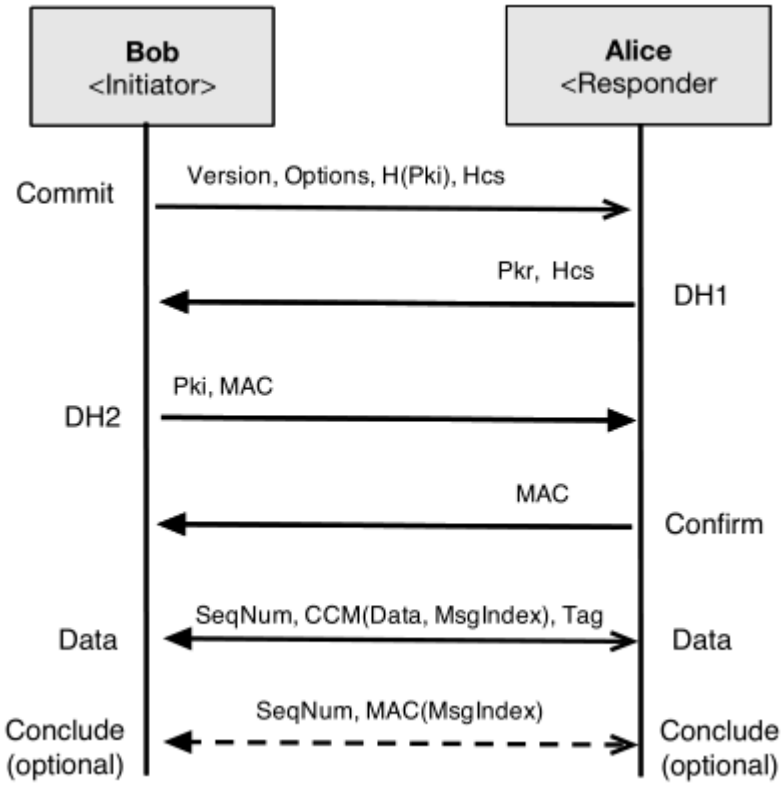

Рисунок 1.2 – Генерування початкового ключа протоколу SCIMP

Відправник генерує нову пару відкритого та особистого ключів та обчислює значення $Hcsi = MAC(cs_i||H(Pk_i)||"initiator")$ (використовується перших 64 біти), яке відправляється отримувачу.

Отримувач отримує перший пакет і також генерує нову пару відкритого та особистого ключів. Формує значення та відправляє його в повідомленні $DH1$, де $Hcsr = MAC(cs_i||H(Pkr)||"Responder")$ (використовується перших 64 біти), а $DH1 = (Pkr||Hcsi)$.

Відправник отримує пакет $DH1$ від отримувача. Формує пакет $DH2 = (Pk_i||HMAC(cs_i))$ і відправляє його отримувачу. Відкритий ключ відправника отримувач отримує тільки на цьому кроці.

Отримувач отримує пакет $DH2 = (Pk_i||HMAC(cs_i))$ від відправника. Перевіряє, що отримане раніше значення $H(Pk_i)$ збігається із застосуванням геш-функції до отриманого значення Pk_i, та оновлює спільний секрет, відправляючи його MAC значення відправнику для підтвердження. Протокол видає повідомлення про помилку, якщо два учасники намагаються ініціювати узгодження ключа одночасно. В такому випадку далі дії вирішуються не на рівні протоколу. Наприклад, повідомлення можуть порівнюватися за деяким числовим параметром, таким як геш-значення від обох повідомлень.

Варіанти наборів криптомітивів, які можуть використовуватися в протоколі SCIMP наведено в таблиці 1.1.

Таблиця 1.1 – Набори криптопримітивів протоколу SCIMP

Набір	Геш-функція	MAC код	Шифр	Ключі
1	SHA-256	HMAC-SHA-256	AES-128	ECC-384
2	SHA-512/256	HMAC-SHA-512	AES 256	ECC-384
3	SKEIN-512/256	SKEIN-MAC-512	AES-256	ECC-384

Протокол SCIMP має особливість в процедурі оновлення ключів, яка не потребує підтвердження від іншої сторони. Вона описується лише застосуванням функції оновлення ключа для кожного нового повідомлення. Коректна робота цієї частини протоколу гарантується правильним розшифруванням і вимогою перегенерації ключів при необхідності. Цей підхід дозволяє мінімізувати часові простої при оновленні ключа, навіть якщо одна зі сторін довгий час відсутня у мережі.

1.2 Підпротоколи системи Signal та їх опис

Під час активності у створенні протоколів наскрізного шифрування для обміну миттєвими повідомленнями [4], [5], найвидатнішою розробкою в цьому просторі стала система обміну повідомленнями Signal. Її підпротоколи містить кілька незвичайних властивостей безпеки (таких як «майбутня секретність» або «безпека після компрометації»), що забезпечується новою технікою під назвою подвійний храповик [6] .

Система Signal складається з трьох підпротоколів, відомих як розширений потрійний протокол Діффі – Хеллмана, протокол подвійного храповика [7] та протокол Sesame. Перший підпротокол можна розглядати як тип протоколу обміну ключами, який дозволяє двом сторонам обмінюватися безпечним початковим ключем сеансу. Другий підпротокол виконується після протоколу X3DH і дозволяє двом сторонам виконувати безпечну зворотну доставку повідомлень.

Підтримка кількох пристроїв забезпечується підпротоколом Sesame у системі Signal. Sesame додає новий рівень складності, який часто не відображається в поточному криптографічному аналізі [8].

На відміну від інших підпротоколів системи Signal, специфікація Sesame менш точна та залишає багато свободи для фактичної реалізації протоколу. Таким чином, незрозуміло, чи забезпечує Signal безпеку після зламу в загальному випадку, коли користувачі мають кілька пристроїв.

Розглянемо більш детально ці протоколи та їх складові, наведемо їх схеми та криптографічні записи.

Протокол узгодження ключів X3DH

Протокол X3DH використовує комбінацію асиметричної криптографії та обчислювальних засобів для узгодження спільного секретного ключа між двома сторонами. Основна ідея полягає в тому, що сторони обмінюються публічними ключами, а потім використовують їх для створення спільного секретного ключа, який може бути використаний

для подальшого шифрування комунікації.

Таблиця 1.2 – Параметри X3DH

Назва	Визначення
Еліптична крива	X25519 або X448
Геш-функція	256 або 512-бітна геш-функція (наприклад SHA-256 or SHA-512)
Інформація	рядок ASCII, що ідентифікує програму

Наприклад, програма може вибрати криву як X25519, хеш як SHA-512, а інформацію як "MyProtocol".

Додаток має додатково визначити функцію кодування Encode(PK) для кодування відкритого ключа X25519 або X448 PK у послідовність байтів. Рекомендоване кодування складається з деякої однобайтової константи для представлення типу кривої, за якою слідує кодування u - координати з прямим кінцем, як зазначено в [9].

Криптографічний запис

• DH(PK1, PK2) — це послідовність байтів, яка є спільним секретним виходом функції Діффі - Хеллмана на еліптичній кривій, що включає пари ключів, представлені відкритими ключами PK1 і PK2. Функція еліптичної кривої Діффі - Хеллмана буде функцією X25519 або X448 з таблиці 3.1 , залежно від параметра кривої.

• Sig(PK, M) представляє послідовність байтів, яка є підписом XEdDSA на послідовності байтів M і перевіряється відкритим ключем PK, яка була створена шляхом підписання M відповідним закритим ключем PK. Функції підписання та перевірки для XEdDSA визначено в [10].

• KDF(KM) представляє 32 байти вихідних даних алгоритму HKDF [11] із входами:

◦Матеріал вхідного ключа HKDF $= F||KM$, де KM — вхідна

послідовність байтів, що містить матеріал секретного ключа, а F — послідовність байтів, що містить 32 байти 0xFF, якщо крива — X25519, і 57 байтів 0xFF кривої — X448. F використовується для криптографічного поділу домену з XEdDSA [10].

∘Сіль HKDF = послідовність байтів із заповненням нулями, довжина якої дорівнює довжині виведення хешу.

∘HKDF інформація = інформаційний параметр із таблиці 3.1.

Налаштування сеансу в протоколі системи Signal включає три сторони : Алісу та Боба, а також центральний сервер S. Це пов'язано з тим, що Signal прагне забезпечити безпечний обмін повідомленнями в асинхронних налаштуваннях, тобто можна ініціювати чати та зашифровані повідомлення; обмінюватися повідомленнями, навіть якщо не всі партнери по спілкуванню онлайн.

Ключі

Всі відкриті ключі мають відповідний секретний ключ, але для спрощення опису ми зосередимося на відкритих ключах.

Таблиця 1.3 – Відкриті ключі

Назва	Визначення
IK_A	Ключ ідентифікації Аліси
EK_B	Ефемерний ключ Аліси
IK_B	Ідентифікаційний ключ Боба
SPK_B	Ключ підпису Боба
OPK_B	Одноразовий ключ Боба

Відкриті ключі, які бачимо в таблиці 1.3, використовуються в рамках запуску протоколу X3DH, повинні або всі бути у формі X25519, або всі вони повинні бути у формі X448, залежно від параметра кривої. Під час кожного запуску протоколу Аліса генерує нову пару ефемерних ключів із відкритим ключем EK_A. Після успішного виконання протоколу

Аліса та Боб поділяться 32-байтовим секретним ключем SK. Цей ключ можна використовувати в деяких протоколах захищеного зв'язку після X3DH відповідно до міркувань безпеки.

X3DH має три фази:

1)Боб публікує свій ідентифікаційний ключ і попередні ключі на сервері.

2) Аліса отримує «набір попередніх ключів» із сервера та використовує його для надсилання початкового повідомлення Бобу.

3) Боб отримує та обробляє початкове повідомлення Аліси.

Щоб виконати узгодження ключа X3DH з Бобом, Аліса зв'язується з сервером і отримує «набір попередніх ключів», що містить такі значення:

•Ключ ідентифікації Боба IK_B

•Попередньо підписаний ключ Боба SPK_B

•Підпис попереднього ключа Боба $Sig(IK_B, Encode(SPK_B))$

•(Необов'язково) одноразовий ключ OPK_B Боба

Сервер повинен надати один з одноразових попередніх ключів Боба, якщо він існує, а потім видалити його. Якщо всі одноразові попередні ключі Боба на сервері видалено, пакет не міститиме одноразового попереднього ключа.

Аліса перевіряє підпис попереднього ключа та перериває протокол, якщо перевірка не вдається. Потім Аліса генерує пару ефемерних ключів із відкритим ключем EK_A.

Якщо пакет не містить одноразового попереднього ключа, вона обчислює:

$$DH1 = DH(IK_A, SPK_B)$$
$$DH2 = DH(EK_A, IK_B)$$
$$DH3 = DH(EK_A, SPK_B)$$
$$SK = KDF(DH1||DH2||DH3)$$

Якщо комплект містить одноразовий попередній ключ, розрахунок змінюється, щоб включити додатковий DH:

$$DH4 = DH(EK_A, OPK_B)$$

$$SK = KDF(DH1 \,||\, DH2 \,||\, DH3 \,||\, DH4)$$

На рис. 1.3 показано обчислення DH між ключами. Зверніть увагу, що DH1 і DH2 забезпечують взаємну автентифікацію, тоді як DH3 і DH4 забезпечують пряму секретність.

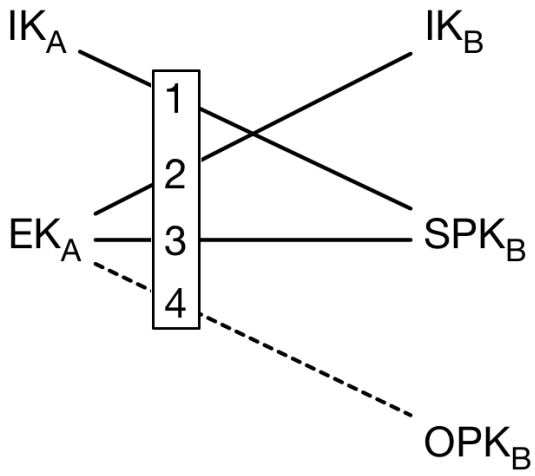

Рисунок 1.3 – Обчислення DH між ключами

Початковий зашифрований текст зазвичай є першим повідомленням у протоколі зв'язку після X3DH. Іншими словами, цей зашифрований текст зазвичай виконує дві ролі: виступає першим повідомленням у певному протоколі після X3DH і як частина початкового повідомлення X3DH Аліси.

Подвійний алгоритм храповика

Подвійний алгоритм храповика (англ. DR - Double Ratchet Algorithm або Axolotl Ratchet) — це широко використовуваний протокол для обміну повідомленнями між двома сторонами, який гарантує надійні властивості безпеки, такі як пряма секретність і безпека після зламу. Складається з двох частин: симетричного та асиметричного храповика, останнього також називають храповиком Діффі – Хеллмана (DH).

Для отримання більш детальної інформації пропоную ознайомитись з документацією [12].

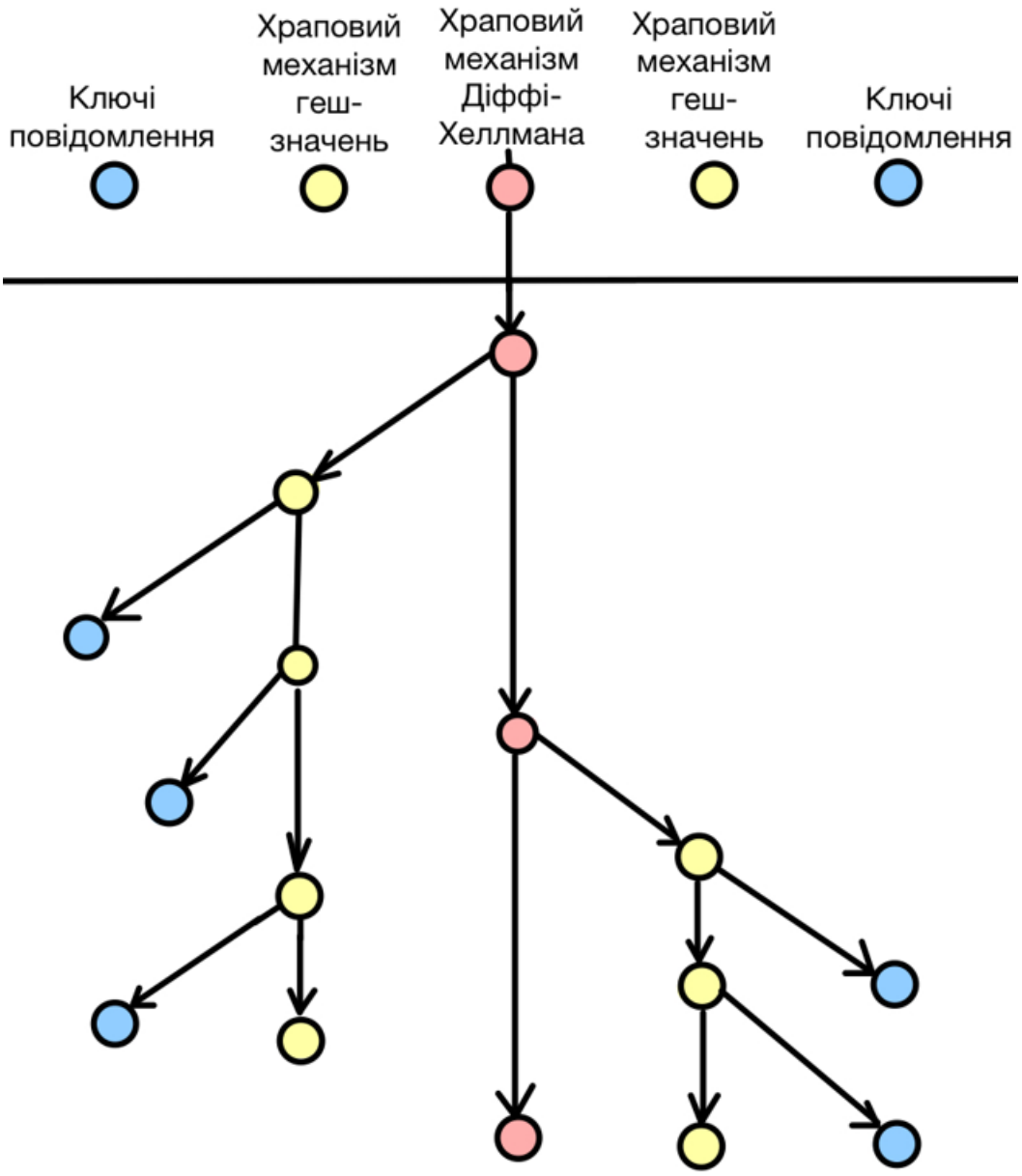

Рисунок 1.4 – Храпові механізми протоколу Double Ratchet

Як передумова DR очікує, що сторони автентифікують одна одну та встановлять спільний ключ перед запуском протоколу, наприклад, Signal використовує розширений протокол потрійного Діффі – Хеллмана (X3DH). Подвійний храповий механізм складається з ієрархії трьох типів ключів: кореневий ключ, який є спільним ключем між сторонами на початку протоколу, ланцюгові ключі, отримані з кореневого ключа, і ключі повідомлень, отримані з ланцюжкових ключів. Коли сторони міняються ролями, скажімо, від відправника до одержувача, вони виконують асиметричний крок, а коли вони зберігають ту саму роль, вони виконують симетричний крок.

Протокол подвійного храпового механізму є вдалим поєднанням храпових протоколів SCIMP та OTR, якому вдалося поєднати декілька храпових механізмів, див. рис. 1.4. Його переваги у порівнянні з SCIMP є достатньо очевидними та призвели до більшого розповсюдження протоколу із заміною SCIMP.

Протокол OTR, який ще називають “three step ratchet” – для кожного тимчасового ключа необхідно три обміни – створення такого ключа, підтвердження іншою стороною та використання. Тобто на відміну від подвійного храпового протоколу відправник не може використовувати оновлення ключів, не отримавши підтвердження від отримувача з його новим тимчасовим ключем.

Далі розберемо набори значень кожної сторони (Аліси та Боба), які зазначено в таблиці 3.2, та їхню взаємодію. Буде наведена схема ініціалізації, відправлення повідомлення та отримання повідомлення.

Ініціалізація:

Аліса (відправник, знає Eph^{Bob}) обчислює значення:

$$HK_r, NHK_r, CK_r, NHK_s,$$
$$RK = HKDF(master\_secret),$$
$$Eph_r = Eph^{Bob},\ N_s^A = N_r^A = 0,$$
$$PN^A = 0\ ,\ r\_flag^A = 1$$

Боб (отримувач) обчислює значення:

$$HK_s,\ NHK_r,\ NHK_s,\ CK_s,$$
$$RK = HKDF\ (master\_secret),$$
$$N_S^B = N_r^B = 0,\ PN^B = 0,$$
$$r\_flag^B = 0,\ Eph_s = Eph^{Bob}$$

$master\_secret$ — спільне таємне значення з ECDH.

Таблиця 1.4 – Набори значень кожної сторони

Назва	Визначення
RK	поточне значення Root Key — 32 байти
CK_s	власне поточне значення Chain Key — 32 байти
CK_r	поточне значення Chain Key іншої сторони — 32 байти
MK_s	власне поточне значення Message Key — 80 байтів
MK_r	поточне значення Message Key іншої сторони — 80 байтів
N_s, N_r	кількість відправлених та отриманих повідомлень відповідно
PN_s	кількість повідомлень відправлених з минулим тимчасовим ключем
r_flag	флаг, який встановлюється, якщо з наступним повідомленням буде створено новий тимчасовий ключ
HK_s, NHK_s, HK_r, NHK_r	значення ключів шифрування заголовків поточне та наступне, власне та іншої сторони відповідно
Eph_s, Eph_r	тимчасові значення сторін для ECDH

Відправлення повідомлення:

1) Якщо $r\_flag = 1$, то необхідно оновити ключі -— згенерувати нове

значення Eph_s , оновити значення ключа заголовка $HK_s = NHK_s$, де $eph_secret = ECDH(Eph_r, Eph_s)$ (з новим значенням Eph_s).
2) $PN_s = N_s$, $N_s = 0$, $r\_flag = 0$.
3) $MK = HMAC - SHA256(CK,0x01)$ — обчислення ключа шифрування повідомлення
4) $cip\_txt = Enc(HK_s,N_s||PN_S||Eph_s)||Enc(MK,plain\_txt)$ — шифрування повідомлення та ключів (створення заголовка).
5) $N_s = N_s + 1$ — оновлення лічильника відправлених повідомлень.
6) $CK = HMAC - SHA256(CK,0x02)$ – оновлення значення Chain Key.

За такого підходу, знаючи значення MK, обчислювано важко знайти поточне або наступне значення K. Після генерування стороною нового значення Eph_s воно буде відсилатися кожен раз поки не надійде відповідь від іншої сторони з оновленими ключовими даними.

Отримання повідомлення:

Якщо $HK_r <> none$ і можна правильно розшифрувати заголовок за допомогою HK_r, то не було оновлення значень ключів іншої сторони. Отже, обчислити ключ $MK_r = HMAC - SHA256(CK_r,0x01)$ для розшифрування.

В іншому випадку відбулося оновлення ключів і необхідно використати $NHK_r (HK_r = NHK_r)$ для розшифрування заголовка та отримати значення N_r^{new}, PN_r^{new}, Eph_r^{new}.
$eph\_secret = ECDH(Eph_r^{new}, Eph_s)$ — обчислення нового тимчасового спільного секретного значення.
$CK_r, NHK_r, RK = HKDF(RK,eph\_secret)$ — оновлення значень ключів іншої сторони та Root Key.

Обчислити ключ $MK_r = HMAC - SHA256(CK_r,0x01)$ для розшифрування повідомлення. Встановити $r\_flag = 1$ — тепер наша черга оновлювати ключові дані.

Враховуючи можливість некоректних даних, необхідно використовувати більше умов розгалуження та тимчасові змінні. Якщо деякі повідомлення були втрачені, то значення лічильника дозволяє все

одно отримати вірні значення CK та MK.

Краще також зберігати всі значення Eph_r і Eph_s для перевірки всіх ключів та невпорядкованого надходження повідомлень. Можна не використовувати окремі ключі для заголовків, а породжувати їх зі значень CK.

Опис протоколу Sesame

Протокол дає змогу користувачеві зв'язати кілька пристроїв зі своїм обліковим записом і забезпечує синхронізацію надісланих і отриманих повідомлень із пристроями їхніх партнерів.

Sesame був розроблений для керування сеансами Double Ratchet, створеними за допомогою угоди ключа X3DH [12]. Однак Sesame — це загальний алгоритм, який може працювати з довільним алгоритмом шифрування повідомлень на основі сеансу, який відповідає певним умовам.

Він описує два сценарії. Перший — для кожного користувача, коли один ідентифікаційний ключ використовується на всіх пристроях. Другий сценарій — для кожного пристрою, де кожен пристрій має власний ідентифікаційний ключ. Єдина різниця полягає в місці, де зберігаються ідентифікаційні ключі — або в записах користувача, або в записах пристрою. Отже, обидва варіанти обробляються подібним чином.

Крім того, Sesame використовує сервер поштової скриньки для зберігання повідомлень, надісланих на пристрої, доки вони не зможуть їх отримати, що забезпечує асинхронний зв'язок.

Попри те, що Sesame описує, як синхронізуються повідомлення на всіх пристроях у сценарії з кількома пристроями, він не охоплює реєстрацію нових пристроїв: ці деталі повністю залишаються на розсуд реалізації, виключаючи їх із міркувань щодо безпеки Sesame.

1.3 Стійкість підпротоколів системи Signal

У цій роботі [13] розпочатий формальний аналіз безпечного обміну повідомленнями, враховуючи рівень обробки сеансів, і застосовується підхід до протоколу Sesame, керування сеансами Signal.

В роботі [14] представлено перший повний опис складного криптографічного протоколу TextSecure, проведено аналіз безпеки трьох його основних компонентів (обмін ключами, отримання ключів і автентифіковане шифрування) і обговорено основні претензії щодо його безпеки. Наступник TextSecure месенджер Signal продовжує використовувати базовий протокол для обміну текстовими повідомленнями.

Більшість месенджерів, які відновлюють безпеку для широкого класу супротивників навіть зі скомпрометованими секретами, засновані на підпротоколах системи Signal. Крім того, було показано, що він справді гарантує сильнішу ідею посткомпромісної безпеки у випадку використання одного пристрою на користувача [7].

За останні два роки уявлення про безпеку вийшли за межі того, що досягається цією безпечною системою обміну повідомленнями. Добірка з них подана в цій роботі [8].

Автори роботи [15] надали повноцінний аналіз безпеки наскрізного протоколу обміну повідомленнями в системі Signal у рамках UC. Він потребує аналізу безпеки в рамках, що допускає модульний аналіз і компоновані гарантії безпеки. Перші кроки в цьому напрямку були зроблені роботою Йоста (англ. Jost), Маурера (англ. Maurer) та Мулярчика (англ. Mularczyk) [16], яка визначає абстрактну службу храповика в структурі конструктивної криптографії [17], а також паралельною роботою Бінштока (англ. Bienstock) та ін. [18], який формулює ідеальну функціональність підпротоколів Signal в рамках UC. Однак жодна з цих робіт не дає модульної декомпозиції Signal на його

основні компоненти (як описано в [19]).

1.4 Недоліки та спроби вдосконалення системи Signal

Захищені програми обміну повідомленнями спостерігали експоненціальне зростання використання протягом останнього десятиліття. Надзвичайно важливо проаналізувати їхню безпеку та пом'якшити наявні недоліки. Роботи щодо вдосконалення системи Signal дуже актуальні, більшість з них була написана за останні 2 роки, тобто 2021—2022 роки.

2019:

Розглядаючи систему Signal, описуючи деякі атаки на оригінальний дизайн, автори роботи [20] пропонують SAID: Signal Authenticated і IDentity-based. З назви видно, що новий протокол базується на налаштуваннях на основі ідентифікації, що дозволяє обійтися без централізованого сервера Signal. Використовуючи довгострокові секрети на основі ідентифікації, вони отримують постійну та явну автентифікацію, щоб SAID досягав вищих гарантій безпеки, ніж Signal.

2021:

Поточна реалізація протоколу реєстрації пристрою дозволяє зловмиснику зареєструвати власний шкідливий пристрій, що дає йому необмежений доступ до всіх майбутніх комунікацій жертви та навіть дозволяє повністю видати себе за іншу особу. Це показує, що нинішнє втілення системи Signal не гарантує безпеку після зламу. Саме тому було обговорено декілька контрзаходів [21], як простих, спрямованих на підвищення виявленості атаки, так і ширшого підходу, спрямованого на розв'язання основної проблеми, а саме слабкого потоку реєстрації пристроїв.

У роботі Штадлера (англ.Stadler) [22] описано гібридну версію системи Signal. Розробка протоколу є результатом спільного дослідницького проєкту між захищеною електронною поштою Tutanota

та Університетом Лейбніца в Ганновері. Його буде впроваджено для досягнення постквантової безпеки для Tutanota та для додавання передової та майбутньої секретності до наскрізних зашифрованих електронних листів.

Безперервні угоди групового ключа (CGKA) [23] — це клас протоколів, які можуть забезпечити сильні гарантії безпеки для захисту протоколів групового обміну повідомленнями, таких як Messaging Layer Security (MLS) та протоколу в системі Signal. Захист від компрометації пристрою забезпечується повідомленнями про фіксацію: кожен учасник групи може регулярно оновлювати свій ключ, завантажуючи повідомлення про фіксацію, яке потім завантажується та обробляється всіма іншими учасниками.

В роботі [24] починають з демонстрації обмежень розгорнутого механізму в системі Signal, спостерігаючи, що він призводить до відносно слабких властивостей анонімності, і показують нову атаку зловживання некоректними повідомленнями, яка дозволяє надіслати одержувачу спам із повідомленнями, які неможливо відстежити, через що у пристрою одержувача розряджатиметься акумулятор. Тому було розроблено новий протокол під назвою Orca, який дозволяє одержувачам реєструвати на платформі список блокувань, що зберігає конфіденційність.

По-перше, протокол в системі Signal не дає сторонам можливість виявити, чи отримували їхні колеги підроблені повідомлення від їх імені під час корупції. Такі ситуації можуть виникнути, коли будь-яка сторона була пошкоджена в минулому, а потім відновилася. Це є відомою слабкістю Signal [25].

По-друге, як зазначено в документації системи Signal [19], коли одна зі сторін скомпрометована, зловмисник може «поділити» сеанс обміну повідомленнями. Тобто супротивник може створити атаку «людина посередині», коли обидві сторони вважають, що вони розмовляють одна з одною під час спільного сеансу, але насправді вони обидві розмовляють із супротивником. Крім того, це може залишатися такою нескінченно довго,

навіть якщо жодна сторона більше не скомпрометована.

Також у розділах роботи [15] розкривають деякі недоліки підпротоколів системи Signal, які раніше не обговорювалися.

2022:

У роботі [26] пропонується безпечна заміна протоколу обміну ключами в системі Signal X3DH на основі SIDH і надають докази безпеки в моделі Signal-adapted-CK, демонструючи, що їхній протокол задовольняє всі властивості безпеки оригінального Signal X3DH. Названо цей новий протокол SI-X3DH.

Доступ до контактів дає змогу зручно користуватись з людьми, які є в адресній книзі. У цій роботі [27] досліджено три популярних месенджери (WhatsApp, Signal і Telegram) та продемонстровано, що існують серйозні проблеми з конфіденційністю в розгорнутих на цей час в методах виявлення контактів. Кількісно оцінено, що навіть зловмисники з обмеженими ресурсами здатні збирати дані мільярдів користувачів. Вони пропонують відповідні засоби пом'якшення, такі як кілька технічних заходів для постачальників послуг, щоб запобігти таким атакам у майбутньому. Широкомасштабні атаки сканування все ще можливі та ці дані можуть бути використані для різних цілей. У системі Signal визнали, що проблему нумераційних атак не можна повністю запобігти, але все ж скорегували їх обмеження частоти та запровадили додаткові засоби захисту від сканування. WhatsApp розгорнув покращений захист для синхронізації контактів.

Побудувати протокол з однаковими гарантіями безпеки, який можна ефективно масштабувати для великих груп, є складною проблемою. Основним слабким місцем є часта ротація ключів, яку користувачі повинні виконувати, щоб досягти безпеки після компрометації. У цій роботі [28] пропонують CoCoA. Це нова схема, яка дозволяє виконувати одночасні оновлення, які не є ні шкідливими, ні дорогими. Тобто вони не додають витрат на майбутні операції, але потребують лише зв'язку для кожного користувача.

Ця робота [29] мотивована спостереженням, що попередні анклавні реалізації алгоритмів системи Signal є неоптимальними як асимптотично, так і конкретно. Отримана система під назвою EnigMap досягає 5,5-кратного прискорення порівняно з реалізацією лінійного сканування Signal і 21-кратного прискорення порівняно з попередньою реалізацією кращого непомітного алгоритму за реалістичного розміру бази даних 256 мільйонів і розміру пакета 1000. Прискорення є асимптотичним за своєю природою та буде ще більше, оскільки база користувачів Signal зростатиме.

У роботі [30] посилюють властивість безпеки після зламу, забезпечуючи швидке відновлення. Протокол в системі Signal потребує до двох повних ланцюжків повідомлень перед відновленням, коли ж їх протокол дозволяє відновлення після еквівалентного ланцюга лише з одного повідомлення. Також вони надають додатковий захист від атак із захопленням сесії. Будують її на основі вже наявної магістралі Signal, не послаблюючи інших її припущень щодо безпеки, і залишаючись сумісними з функцією обробки повідомлень. Результати впровадження показують, що, всупереч тому, що MARSHAL повільніше, ніж Signal (як і очікувалося), винятковий приріст у швидкості відновлення досягається напрочуд низькою ціною, при цьому окремі етапи (включно з отриманням ключа, шифруванням і дешифруванням) займають менш як 6 мс.

Автори роботи [31] зосередились на методі, здатному приховати інформацію про членство з точки зору не членів групи в протоколі безпечного групового обміну повідомленнями (SGM), який називають «конфіденційністю членства». Хоча Чейз та ін. (ACM CCS 2020) розглядали те саме поняття, їхня пропозиція є розширенням Signal, так званого «парного сигналу», де групове повідомлення багаторазово надсилається через окремі канали Signal. Таким чином, їхній протокол не є масштабованим. У цій роботі розширили протокол SGM (ACM CCS 2018), який вони називають протоколом асинхронні дерева храповика (ART), щоб додати конфіденційність членства.

Висновки до розділу 1

В цьому розділі обговорено актуальність та потребу захищених месенджерів та наскрізного шифрування. Досліджено схеми протоколів OTR та Scimp, процедури оновлення ключів та особливості цих протоколів захищеного обміну повідомленнями для мобільних систем миттєвого обміну повідомленнями. Розглянуто складові сучасного месенджеру Signal та показано, чому алгоритм подвійного храповика є більш вдалим, ніж його попередники. Побудували загальні моделі цих протоколів.

Розглянуто стійкість підпротоколів системи Signal. Вони забезпечують конфіденційність, цілісність, автентифікацію, узгодженість учасників, перевірку місця призначення, пряму секретність, посткомпромісну безпеку (так звану майбутню секретність), збереження причинності, незв'язність повідомлень, відмову від повідомлень, відмову від участі та асинхронність. Вони не забезпечують збереження анонімності та вимагає серверів для ретрансляції повідомлень і зберігання матеріалу відкритого ключа.

Підпротоколи в системі Signal також є не стійкими до декількох відомих атак, наприклад "атака людина посередині"та "атака зловживанням некоректних повідомлень". Зроблена робота показує, що необхідно забезпечити стійкість до всіх відомих атак та підвищення їх виявлення. Показано, що система має безліч варіантів покращень та усунення недоліків. В роботах запропоновані варіанти швидшої роботи без втрати безпеки, оптимізація алгоритму та покращення гарантій безпеки.

2 ПРОТОКОЛИ РЕЄСТРАЦІЇ СУПУТНІХ ПРИСТРОЇВ В СИСТЕМАХ ОБМІНУ ПОВІДОМЛЕННЯМИ

В наш час ми проводимо своє життя, користуючись багатьма такими пристроями як смартфони, планшети чи ноутбуки, і ми очікуємо, що буде легко та ефективно перемикатися між ними без втрати часу чи безпеки. Однак більшість програм розроблено для використання на одному пристрої. Щоб дозволити користувачам надсилати та отримувати повідомлення в системі захищеного обміну з кількох пристроїв, був представлений протокол Sesame.

Проаналізуємо останні оновлення підтримки декількох пристроїв в системі WhatsApp у їхньому блозі. Також опишемо протоколи реєстрації супутніх пристроїв в захищених системах обміну повідомленнями WhatsApp та Signal, які набирають популярності в користувачів.

2.1 Припущення протоколу Sesame

Протокол Sesame створений для керування сесіями в системах захищеного обміну повідомленнями в асинхронному режимі з використанням кількох пристроїв. В першу чергу він орієнтований на використання протоколу Double Ratchet та протоколу узгодження ключа X3DH. Однак Sesame — це загальний протокол, який може працювати з довільним алгоритмом шифрування повідомлень на основі сесії, який відповідає певним умовам.

Sesame базується на наступних припущеннях:

– Сервер

Існує сервер, де зберігаються поточні записи всіх користувачів і пристроїв. Сервер тимчасово зберігає повідомлення, які пристрої надсилають один одному, доки повідомлення не будуть отримані.

– Користувачі

У будь-який момент часу існує набір користувачів. Вони можуть бути додані або видалені в будь-який час. Кожен користувач має UserID (наприклад, ім'я користувача або номер телефону). Після видалення користувача його UserID може отримати новий користувач.

– Пристрої

У будь-який момент часу кожен користувач має хоча б один пристрій. Їх можна додавати або видаляти пристрої в будь-який час. Кожен пристрій має DeviceID, унікальний для UserID. Пристрої можуть запитувати у сервера інформацію про інших користувачів та пристрої. Вони можуть підтримувати стан, але в будь-який момент часу цей стан може бути повністю або частково видалено (наприклад, через збій апаратного забезпечення чи дії користувача) або повернуто до попереднього стану (наприклад, шляхом відновлення резервної копії). Пристрої мають годинник, який може вимірювати час, що минув, але не синхронізований.

– Поштові скриньки

Сервер зберігає поштову скриньку для кожного пристрою. Поштова скринька містить набір повідомлень, надісланих на пристрій. Повідомлення видаляються з поштової скриньки після отримання. Пристрої можуть надсилати повідомлення на поштові скриньки інших пристроїв. Сервер зберігає UserID і DeviceID пристрою, що надсилає їх, разом із повідомленням.

Пристрої можуть отримувати повідомлення з власної поштової скриньки. Пристрій одержувача отримує повідомлення та UserID відправника та DeviceID із сервера. Повідомлення, надіслані до поштової скриньки, є ненадійними за нормальної роботи - вони можуть бути пошкоджені, видалені, перевпорядковані, затримані або дубльовані до того, як надійдуть до поштової скриньки.

У нормальній роботі повідомлення, яке не надійшло до поштової скриньки протягом певного проміжку часу (MAXLATENCY), було втрачено. Повідомлення, надіслані до поштової скриньки, можуть бути

предметом дій зловмисника – зловмисник (включаючи сервер) може пошкодити, видалити, змінити порядок, дублювати або підробити повідомлення, перш ніж вони надійдуть до поштової скриньки.

– Сеанси

Повідомлення можна шифрувати та розшифровувати за допомогою сеансу. Розшифрувати повідомлення може не вдатися (наприклад, якщо зашифрований текст було змінено або підроблено, тому він не проходить перевірку автентифікації). Зашифроване повідомлення можна розшифрувати лише за допомогою відповідного сеансу.

Існує певний ідентифікатор сеансу, який є унікальним. Сеанс може містити інші дані після шифрування або дешифрування повідомлення (наприклад, ключі можуть бути видалені після їх використання для збереження секретності).

– Створення сеансу для відправників

Кожен пристрій має пару особистих ключів, що складається з відкритого та закритого ключів. Пристрій може створити новий початковий сеанс у будь-який час.

Для створення початкового сеансу потрібно вказати особистий відкритий ключ для передбачуваного пристрою одержувача, який отримає відповідний сеанс. Створити новий сеанс може не вдатися (наприклад, пристрій-відправник може отримати та криптографічно автентифікувати параметри, пов'язані з пристроєм-одержувачем).

– Створення сеансу для одержувачів

Усі повідомлення, зашифровані початковим сеансом, є початковими повідомленнями. Вони всі містять відкритий особистий ключ пристрою-відправника в незашифрованому заголовку.

Після отримання початкового повідомлення призначений пристрій одержувача може створити відповідний сеанс, який використовується для дешифрування початкового повідомлення. Створення відповідного сеансу може виявитися невдалим (наприклад, якщо криптографічна автентифікація початкового повідомлення не вдається).

Після першого розшифровування повідомлення початковий сеанс стає звичайним сеансом.

2.2 Реєстрація зловмисного пристрою

Один з ключових аспектів реєстрації зловмисних пристроїв полягає в тому, щоб такий пристрій був помічений як законний і отримав дозвіл на доступ до системи. Зловмисники використовують різні методи, такі як викрадення аутентифікаційних даних, підробка ідентифікаторів або використання вразливостей у протоколах комунікації, щоб обійти захист і успішно зареєструвати новий зловмисний пристрій.

Наслідками реєстрації зловмисного пристрою можуть бути порушення конфіденційності, крадіжку особистих даних, фінансові втрати або навіть небезпеку для фізичної безпеки. Реєстрація зловмисного пристрою становить серйозну загрозу для індивідуальної та корпоративної безпеки. Розуміння методів атак та розробка ефективних заходів для виявлення та запобігання реєстрації зловмисних пристроїв є критично важливим для захисту від цієї загрози.

Якщо зловмиснику вдається тимчасово скомпрометувати основний пристрій жертви таким чином, що розкриває певні приватні значення, а саме ключ ідентифікації жертви IK і пароль $API\ pw_A$, то він може імітувати реєстрацію пристрою та додати шкідливий пристрій. Як показано на рис. 2.1, основний пристрій взаємодіє з сервером під час реєстрації пристрою лише при запиті коду підтвердження та надсилання зашифрованих особистих даних. Для цього потрібні лише облікові дані API та приватний ключ ідентифікації. Якщо зловмисник отримав доступ до цих значень, то він може повністю емулювати протокол і налаштувати новий пристрій без жодної взаємодії з боку жертви чи її програми.

В роботі [21] було досліджено, що поточна реалізація протоколу реєстрації пристрою в системі Signal дозволяє зловмиснику зареєструвати власний шкідливий пристрій, що дає йому необмежений доступ до всіх

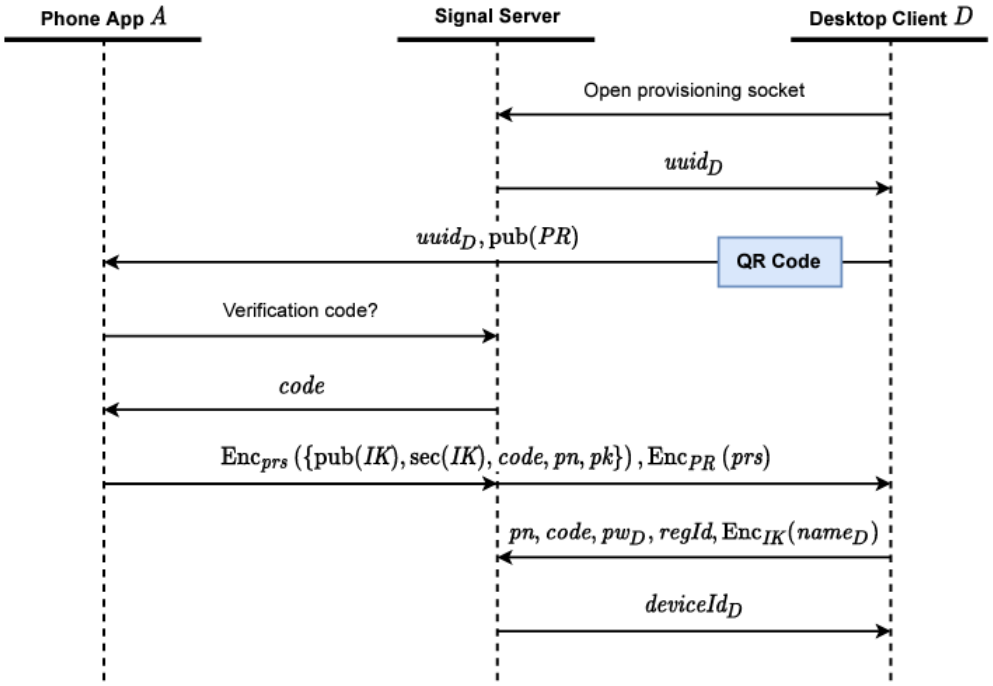

Рисунок 2.1 – Реєстрація нового пристрою D на своєму основному пристрої A

майбутніх комунікацій жертви та навіть дозволяє повністю видати себе за іншу особу. Це прямо показує, що поточна реалізація не гарантує безпеку після зламу.

Також вони створили програму на C#, яка отримує приватний особистий ключ та облікові дані, а потім запускає описані виклики. Показано, що якщо зловмисник має певний рівень контролю над (ненадійним) сервером, він може легко маніпулювати списком пристроїв, щоб виключити свій підроблений пристрій, роблячи атаку майже непомітною.

Альтернативним та більш радикальним підходом була б заміна самого протоколу Sesame, адже технології не стоять на місці. Ефективний протокол реєстрації дозволив би зручно та безпечно обробляти велику кількість нових пристроїв, що зареєстровуються в системі. Це особливо важливо в сучасному світі, де кількість підключених пристроїв постійно зростає. Протокол реєстрації повинен бути сумісним з різними пристроями та платформами. Це дозволяє забезпечити широку підтримку

та інтеграцію з різноманітними пристроями, що підключаються до системи.

Запропоновані зміни в протоколі Sesame

Автори статті [32] пропонують рішення, яке відрізняється від рішень Sesame тим, що в їхній конструкції користувач не знає про пристрої свого кореспондента. Повідомлення, надіслане Бобом, шифрується лише один раз для Аліси, а не для кожного пристрою Аліси. Це повідомлення також шифрується лише один раз для всіх інших пристроїв Боба, а не один раз на пристрій. Автентифікація нового пристрою також відрізняється.

Протокол Sesame пропонує два варіанти: перший вимагає, щоб усі пристрої Аліси мали спільний IDkey. Коли додається новий пристрій, він отримує цей IDkey. Боб розпізнає новий пристрій як пристрій Аліси, оскільки він має той самий IDkey. Це робить IDkey дуже конфіденційними даними. У другому варіанті пристрої не мають спільного ключа. Коли Аліса додає пристрій, Боб повинен фізично автентифікувати цей пристрій, щоб переконатися, що він чесний і належить Алісі. У їхньому рішенні вони лише вимагають, щоб новий пристрій Аліси був автентифікований іншим її пристроєм.

Було представлено високорівневий вигляд їхнього рішення, де Аліса надсилає повідомлення з будь-яких пристроїв, а Боб отримує їх на всіх своїх пристроях.

На рис. 2.2 пояснено, що зараз кожен окремий пристрій користувача — це як окремий абонент зі своїм ключем. Ліворуч на ньому представлено наявний протокол Sesame для кількох пристроїв. Кожен масив відповідає попарному сигнальному каналу. Аналізуючи схему, виникають питання, як відкликати такі ключі, наприклад, коли зловмисник отримав 2 з 3 можливих ключів.

Праворуч їхній протокол динамічного обміну ключами з декількома пристроями. Між Алісою та Бобом потрібен лише один канал сигналу.

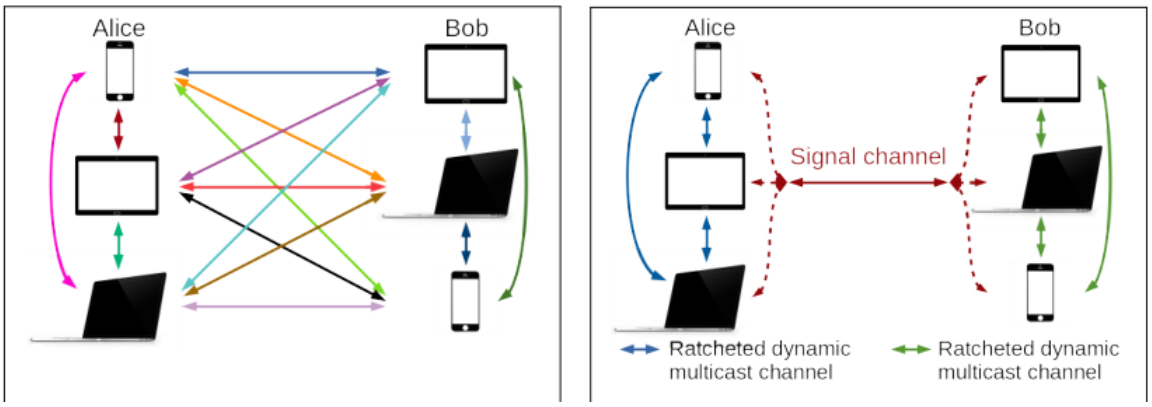

Рисунок 2.2 – Схема сигнальних каналів пристроїв протоколу Sesame

2.3 Протоколи реєстрації супутніх пристроїв в системах обміну повідомленнями WhatsApp та Signal

WhatsApp — найпопулярніша система захищеного обміну миттєвими повідомленнями у світі. Щодня в ній надсилається понад 100 мільярдів повідомлень, які містять особисті та ділові повідомлення. WhatsApp забезпечує зв'язок із наскрізним шифруванням, де жодна сторона, крім відправника та одержувача, не має можливості читати (або змінювати) повідомлення.

В липні 2021 року WhatsApp оголосив про запуск обмеженого загальнодоступного бета-тесту оновленої функції для кількох пристроїв. Таким чином він дав змогу підтримувати не лише один пристрій на номер, а декілька на різних пристроях. Кожен пристрій-супутник підключатиметься до основного пристрою WhatsApp незалежно, зберігаючи той самий рівень конфіденційності та безпеки завдяки наскрізному шифруванню, якого очікують люди, які використовують WhatsApp.

До введення кількох пристроїв усі користувачі WhatsApp ідентифікувалися за допомогою єдиного ідентифікаційного ключа, з

якого отримували всі зашифровані комунікаційні ключі. Завдяки мультипристроям кожен пристрій тепер має власний ідентифікаційний ключ. Особи, які користувались кількома пристроями, мали змогу бачити всі пристрої-супутники, пов'язані з їхнім обліковим записом, а також час їх останнього використання. З основного пристрою вони зможуть дистанційно вилучити додаткові пристрої, якщо потрібно.

WhatsApp синхронізував історію повідомлень, а також інші дані про стан програми (наприклад, імена контактів, чи заархівовано чат, чи позначено повідомлення зірочкою) на всіх пристроях. Всі ці дані синхронізуються та наскрізно шифруються між вашими пристроями.

У своєму останньому оновленні 25 квітня 2023 року в блозі WhatsApp представив процедуру для підключення свого телефону як одного із щонайбільше чотирьох додаткових супутніх пристроїв (оновлення на цю версію відбувається впродовж декількох тижнів). Кожен телефон, що підключений до WhatsApp, забезпечує наскрізне шифрування особистих повідомлень, медіафайлів і дзвінків, незалежно від інших підключених пристроїв. У разі, якщо ваш основний пристрій неактивний протягом тривалого періоду, вони автоматично вийдуть з вашого облікового запису на всіх інших додаткових пристроях.

Безумовно, це зручно, досить функціонально та дозволяє зберігати дані для швидкого доступу до них. Проте існує ряд недоліків в синхронізації, безпеці та надмірних сповіщеннях. Основною ж проблемою постає можливість відновити доступ до основного пристрою (Master Device) при його втраті. Головний пристрій надає змогу переглянути всі підключені пристрої та вийти з підключених до нього при потребі. У додаткових пристроях цієї функції не передбачається. Отже, при втраті головного пристрою займе певний час для відновлення основного або ж зовсім не вийде його відновити.

Але особливу увагу треба звернути саме на протокол реєстрації нових супутніх пристроїв у системі WhatsApp, оскільки він є новим і має певні відмінності в порівнянні з аналогічним протоколом в системі Signal.

Протокол реєстрації супутніх пристроїв в системі WhatsApp

Основний пристрій (primary) — пристрій, який використовується для реєстрації облікового запису WhatsApp із номером телефону. Кожен обліковий запис WhatsApp пов'язано з одним основним пристроєм. Цей основний пристрій можна використовувати для підключення додаткових супутніх пристроїв до облікового запису.

Пристрій-супутник (companion) —- пристрій, який пов'язано з наявним обліковим записом WhatsApp основним пристроєм облікового запису.

Під час реєстрації клієнт WhatsApp передає на сервер свій загальнодоступний ідентифікаційний ключ id_pk^p, публічний підписаний попередній ключ $SignPreKey$ (пара середньострокових ключів Curve25519, створена під час встановлення, підписана особистим ключем і періодично змінювана) (зі своїм підписом) і пакет загальнодоступних One-Time Pre Keys (черга пар ключів Curve25519 для одноразового використання, створена під час встановлення та поповнена за потреби). Сервер WhatsApp зберігає ці відкриті ключі, пов'язані з ідентифікатором користувача.

Всі використані ключі містять 32 байти. У протоколі реєстрації супутнього пристрою беруть участь троє учасників: головний пристрій, супутній пристрій та сервер.

Алгоритм 2.1. Протокол реєстрації супутнії пристроїв в системі WhatsApp

Крок 1. Супутній пристрій випадковим чином створює пару ключів (id_sk^c, id_pk^c) та симетричний ключ для шифрування lsk^c.

Крок 2. Супутній пристрій формує QR-код, в який передає публічний ключ id_pk^c та ключ шифрування lsk^c.

Крок 3. Супутній пристрій надсилає по захищеному каналі $QR(lsk, id)$ на головний пристрій.

Крок 4. Головний пристрій зберігає у свою базу даних id_pk^c та дістає свій власний id_sk^p.

Крок 5. Головний пристрій генерує блок метаданих $meta^c$ та оновлені дані списку пристроїв $list^p$, що містять новий супутній пристрій.

Крок 6. Головний пристрій формує підпис Curve25519 s_a з використанням власного секретного ключа id_sk^p. Цей підпис обчислений на основі префікса 0x0600, містить метадані $meta^c$ та публічний ключ компаньйона id_pk^c, який отримав раніше. Він робить це для того, щоб підтвердити, що головний пристрій знає і отримав ключ та сам створив метадані.

Крок 7. Головний пристрій формує підпис Curve25519 s_{dev} з використанням власного секретного ключа id_sk^p. Цей підпис обчислений на основі префікса 0x0602 та містить оновлений список пристроїв $list^p$.

Крок 8. Головний пристрій формує значення d, яке складається з метаданих $meta^c$, власного секретного ключа id_sk^p та підпис s_a.

Крок 9. Головний пристрій обчислює $h = HMAC(lsk^c,d)$.

Крок 10. Головний пристрій передає зашифровані дані $list^p, s_{dev}, d$ та h на сервер.

Крок 11. Сервер частину даних про оновлені пристрої $list^p$ та підпис s_{dev} зберігає собі в базу даних.

Крок 12. Сервер передає частину даних (d,h) на супутній пристрій.

Крок 13. Відбувається верифікація HMAC на супутньому пристрої. Також він отримав метадані $meta^c$, загальний особистий секретний ключ id_sk^p та підпис s_a.

Крок 14. Супутній пристрій зберігає метадані $meta^c$ собі в пам'ять.

Крок 15. Супутній пристрій формує підпис Curve25519 s_d^c з використанням власного публічного ключа id_pk^c. Він обчислений на основі префікса 0x0601 та містить метадані $meta^c$, власний публічний ключ id_pk^c та приватний особистий ключ основного пристрою id_sk^p.

Крок 16. Супутній пристрій передає дані $meta^c$, підпис s_a, підпис s_d^c, свій особистий публічний ключ is_pk^c і пакет публічних одноразових

попередніх ключів $keys$ на сервер.

Крок 17. Сервер зберігає всі дані у свою базу даних.

Алгоритм 2.2. Протокол реєстрації супутніх пристроїв в системі Signal

Крок 1. Супутній пристрій звертається до сервера, щоб зареєструватись як новий пристрій.

Крок 2. Сервер випадковим чином обирає йому певне значення $uuid^c$ для його ідентифікації та передає це значення супутньому пристрою.

Крок 3. Супутній пристрій випадковим чином створює пару ключів (id_sk^c, id_pk^c).

Крок 4. Супутній пристрій формує QR-код, в який передає публічний ключ id_pk^c та значення $uuid^c$.

Крок 5. Супутній пристрій надсилає по захищеному каналі $QR(uuid^c, id\_pk^c)$ на головний пристрій.

Крок 6. Головний пристрій зберігає у свою базу даних публічний особистий ключ супутнього пристрою id_pk^c та значення $uuid^c$ та дістає свій власний секретний ключ id_sk^p.

Крок 7. Головний пристрій надсилає запит коду перевірки на сервер.

Крок 8. Сервер формує та надсилає одноразовий код перевірки $code$ на головний пристрій.

Крок 9. Головний пристрій випадковим чином обирає симетричний ключ для шифрування lsk^p.

Крок 10. Головний пристрій обчислює значення c_1, яке зашифроване симетричним ключем для шифрування lsk^p та містить власний публічний ключ id_pk^p, секретний особистий ключ id_sk^p, одноразовий код перевірки $code$, номер телефону pn та пароль профілю pk.

Крок 11. Головний пристрій обчислює значення c_2, яке асиметрично зашифроване за допомогою публічного ключа супутнього пристрою id_pk^c та містить симетричний ключ для шифрування lsk^p.

Крок 12. Головний пристрій передає значення c_1 та c_2 на сервер.

Крок 13. Сервер передає значення c_1 та c_2 на супутній пристрій.

Крок 14. Супутній пристрій розшифровує за допомогою секретного особистого ключа id_sk^c значення c_2 та отримує симетричний ключ для шифрування lsk^p.

Крок 15. Супутній пристрій розшифровує за допомогою симетричного ключа для шифрування lsk^p значення c_1 та отримує значення публічного ключа основного пристрою id_pk^p, секретного ключа основного пристрою id_sk^p, одноразовий код перевірки $code$, номер телефону pn та пароль профілю pk.

Висновки до розділу 2

Описано відомий протокол Sesame, який був створений для використання декількох супутніх пристроїв; припущення, на яких він базується. Наведено наявні варіанти реєстрації зловмисних пристроїв та їх наслідки, такі як крадіжка особистих даних чи розкриття секретних значень жертви. Авторами однієї з робіт було створено програму на #, за допомогою якої можна виконати практично непомітну атаку.

Розглянуто нещодавні оновлення в системі WhatsApp, які дали змогу підключати щонайбільше чотири додаткові супутні пристрої. Наведено протоколи реєстрації супутніх пристроїв в системах захищеного обміну повідомленнями Signal та WhatsApp, які було відновлено з офіційної технічної документації. Описано вимоги до ключів, кількість учасників в кожному протоколі реєстрації супутніх пристроїв; послідовність кроків протоколу, які потрібно виконати, щоб під'єднати додатковий девайс.

3 ПОБУДОВА АТАК НА НОВІТНІЙ ПРОТОКОЛ РЕЄСТРАЦІЇ В СИСТЕМІ WHATSAPP

Протоколи реєстрації супутнього пристрою грають критичну роль у забезпеченні безпеки цих систем та захисту від атак. Цей розділ присвячений порівняльному аналізу протоколу реєстрації супутнього пристрою в Signal і WhatsApp. Досліджено основні аспекти цих протоколів, їхні переваги та недоліки з точки зору безпеки та приватності, щоб визначити, який з них краще відповідає вимогам безпеки та приватності в контексті реєстрації супутнього пристрою.

Крім того, розглянуто різні типи атак, які можуть бути спрямовані на протокол WhatsApp, а також описано деякі конкретні атаки, що використовуються проти цього протоколу.

3.1 Порівняльний аналіз протоколів реєстрації супутнього пристрою в системах Signal та WhatsApp

В розділі 2 розібрано детально по кроках протокол реєстрації для систем WhatsApp (алгоритм 2.1) та Signal (алгоритм 2.2). Зараз для кращого сприйняття та побудови порівняльного аналізу спробуємо побудувати схематичне зображення цих алгоритмів, щоб дійсно могли порівняти, чим вони відрізняються.

На рис. 3.1 наведено схему реєстрації супутнього пристрою у системі WhatsApp. На рис. 3.2 зображено схему реєстрації супутнього пристрою в системі Signal.

Проведений аналіз новітнього протоколу реєстрації супутнього пристрою у захищеній системі обміну повідомленнями WhatsApp, а також порівняльний аналіз із відповідним протоколом у системі Signal, дозволив виявити такі переваги та недоліки цих протоколів:

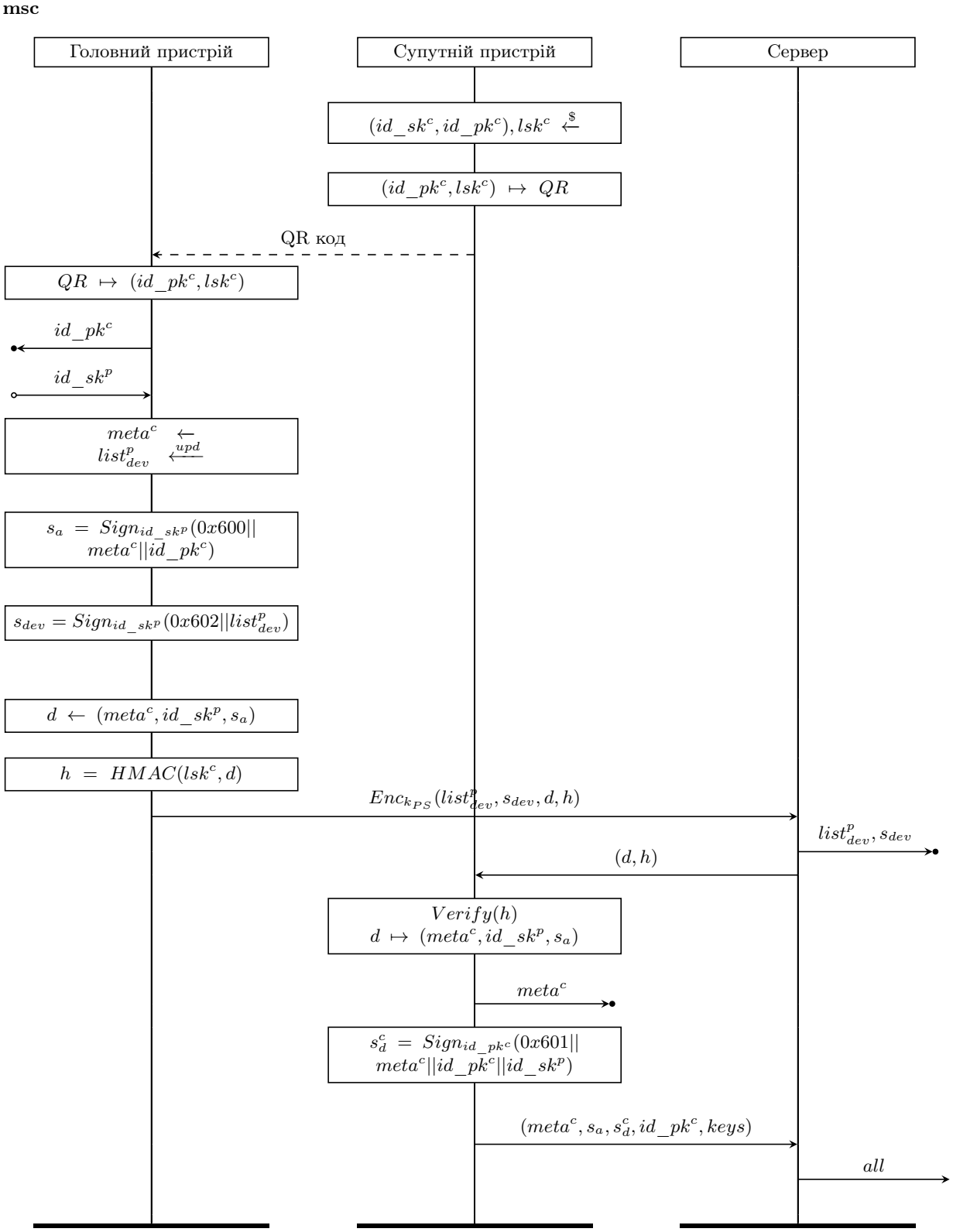

Рисунок 3.1 – Реєстрація супутнього пристрою (WhatsApp)

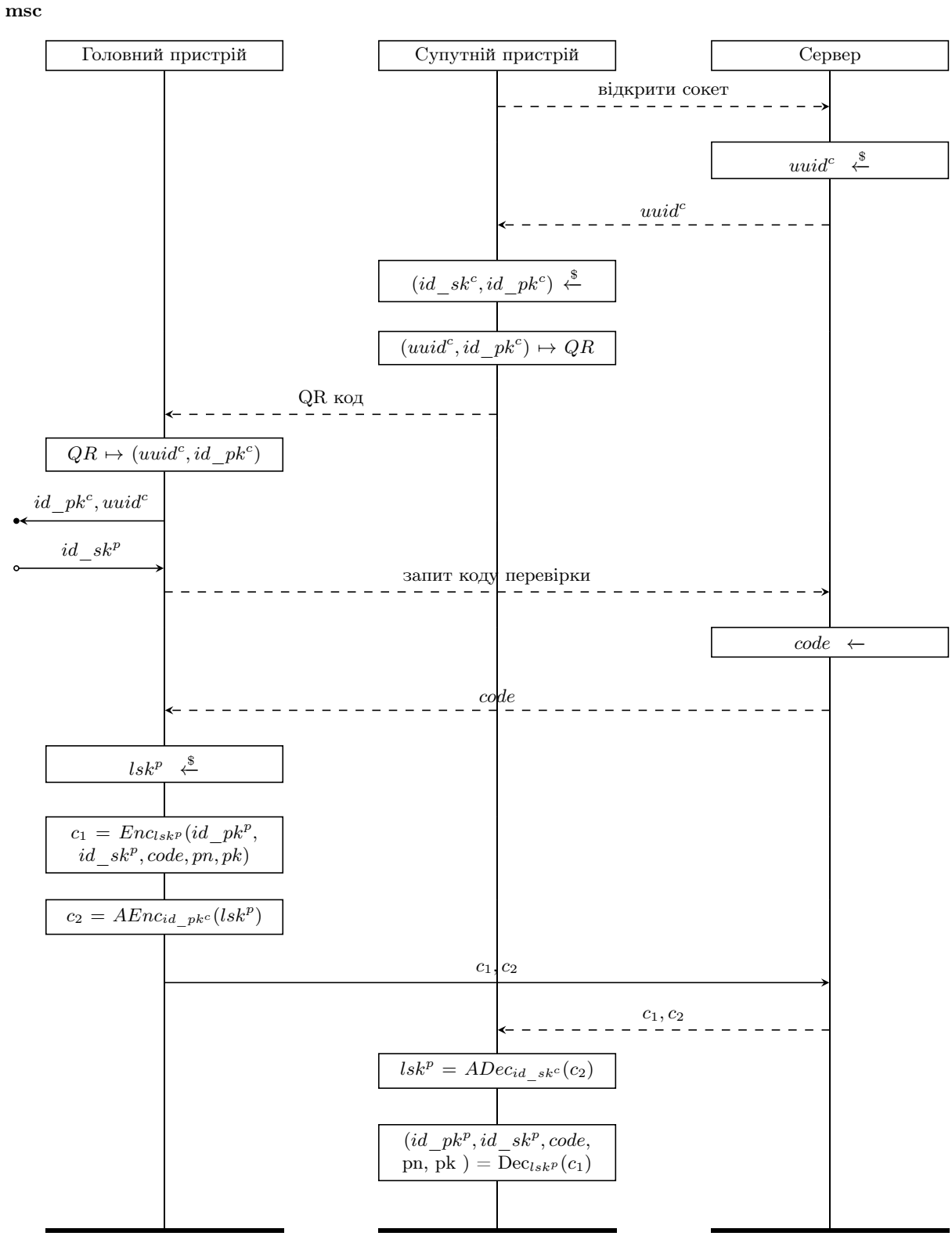

Рисунок 3.2 – Реєстрація супутнього пристрою (Signal)

– І Signal, і WhatsApp використовують сценарій протоколу Sesame, орієнтований на користувача, коли всі пристрої використовують його єдиний особистий ключ, що створює додаткові загрози для довготривалого ключа: Його можна отримати при вдалій спробі реєстрації зловмисного супутнього пристрою або при заволодінні зловмисником одного з супутніх пристроїв в деякий момент часу. Це також вимагає більш ретельного контролю за можливою компрометацією цього ключа і відповідним оновленням на всіх пристроях.

– Абсолютно зайвим є обчислення та передача цифрового підпису $s_a = Sign_{id\_sk^p}(0x600||meta^c||id\_pk^c)$ у системі WhatsApp. Разом з цим значенням передається ключ, яким робиться підпис, та значення $meta^c$. Отже, цей підпис не виконує жодної криптографічної функції та може бути обчисленим супутнім пристроєм після отримання всіх даних для подальшої передачі серверу. Виглядає дуже суттєвим недоліком протоколу.

– Атака реєстрації зловмисного супутнього пристрою з роботи [21] на Signal також є застосовною до WhatsApp, і навіть стає набагато простішою. Зловмиснику достатньо отримати тимчасовий доступ тільки до особистого приватного ключа на головному пристрої id_sk^p, щоб зареєструвати новий (зловмисний) супутній пристрій. В Signal для цього необхідно було додатково знати ім'я користувача та відповідний пароль для вебверсії. Signal є складнішим для проведення таких атак, оскільки є додаткові обміни повідомленнями з сервером як головного пристрою, так і супутнього пристрою. У WhatsApp тільки знання поточного особистого приватного ключа на головному пристрої id_sk^p дає змогу реєструвати нові супутні пристрої на сервері.

– Більш того, атака реєстрації зловмисного супутнього пристрою стає реальною практичною загрозою, оскільки за відсутності додаткових взаємодій з сервером і додаткових перевірок, достатньо підготувати зловмисний QR код заздалегідь і отримати доступ до розблокованої програми WhatsApp на декілька секунд, яких вистачить для сканування

цього QR коду. Весь подальший процес реєстрації зловмисного супутнього пристрою відбувається без участі користувача і не потрібним є навіть знання особистого приватного ключа.

– Створення ключа lsk^c для аутентифікації даних реєстрації на стороні нового супутнього пристрою у WhatsApp дає додаткову можливість перевірити знання цього ключа для аутентифікації супутнього пристрою, оскільки вся інша інформація в QR коді є відкритою.

– На відміну від Signal, у системі обміну повідомленнями WhatsApp ключ lsk^c, який використовуються для аутентифікації даних під час реєстрації супутнього пристрою, передається у QR коді у відкритому вигляді, що дозволяє зловмиснику дізнатися цей ключ, наприклад, спробувавши також зчитати QR код з більшої відстані. Знання цього ключа дозволить під'єднати супутній пристрій до зловмисного головного пристрою.

– У WhatsApp сервер дізнається про додавання нового пристрою вже наприкінці протоколу і має просто додати необхідну інформацію про цей пристрій в записі користувача та переслати йому дані. Це дозволяє використовувати атаку повторення за певних обставин: якщо відбулося декілька змін в списку пристроїв без зміни ключа шифрування між головним пристроєм та сервером, то можна повторити попередні повідомлення, щоб змінити список пристроїв (за протоколом не передбачено, що сервер щось відправляє на головний пристрій).

– Використання цифрових підписів s_a і s_{dev} у WhatsApp допомагає супутньому пристрою впевнитись, що саме головний пристрій передав дані, оскільки для створення підписів необхідним є знання особистого ключа.

– Використання саме коду аутентифікації HMAC в WhatsApp для обчислення значення h для забезпечення автентичності повідомлення та цілісності, а не цифрового підпису, скоріше за все, пов'язане з меншою вразливістю симетричних криптопримітивів до атак з використанням квантового комп'ютера. З цією метою краще було б використати один з

режимів автентифікованого шифрування, які саме для цього і призначені.

– Супутній пристрій у WhatsApp формує підпис Curve25519 s_d^c з використанням власного публічного ключа id_pk^c. Він обчислений на основі префікса 0x0601 та містить метадані $meta^c$, власний публічний ключ id_pk^c та приватний особистий ключ основного пристрою id_sk^p. Підпис відкритим ключем id_pk^c є дивним рішенням.

Провівши аналіз, можна продемонструвати коротко переваги в таблиці 3.1.

Таблиця 3.1 – Переваги протоколів реєстрації в системах WhatsApp та Signal

WhatsApp	Signal
	Переваги
Створення ключа lsk^c для аутентифікації даних реєстрації на стороні нового супутнього пристрою;	Ключ lsk^c, який використовуються для аутентифікації даних, створюється на головному пристрої та не передається у відкритому вигляді;
Використання цифрових підписів s_a і s_{dev} допомагає супутньому пристрою впевнитись, що саме головний пристрій передав дані;	Додаткові перевірки супутнього пристрою через сервер та використання запиту коду перевірки роблять Signal складнішим для проведення вже відомих атак;

Переваги є в обох протоколів реєстрації супутніх пристроїв систем WhatsApp та Signal. Більше значення вони мають в протоколі в системі Signal, адже надалі саме його переваги призведуть до недоліків в протоколі реєстрації в системі WhatsApp. Тепер можемо переглянути їхні доведені

недоліки в таблиці 3.2.

Таблиця 3.2 – Недоліки протоколів реєстрації в системах WhatsApp та Signal

WhatsApp	Signal
Недоліки	
Сценарій протоколу Sesame, орієнтований на користувача, коли всі пристрої використовують його єдиний особистий ключ;	
Зайве обчислення цифрового підпису s_a	– – –
Спрощення вже відомих атак через відсутність додаткових взаємодій з сервером, атака реєстрації зловмисного супутнього пристрою стає реальною практичною загрозою;	– – –
Ключ lsk_c передається у відкритому вигляді в QR-коді;	– – –
Використання коду аутентифікації HMAC, а не цифрового підпису;	– – –
Супутній пристрій формує підпис Curve25519 s_d^c з використанням власного публічного ключа id_pk^c;	– – –

Як видно з таблиць, недоліків у протоколі реєстрації супутніх пристроїв в системі WhatsApp більше, ніж переваг. Також їх суттєво

більше, ніж в протоколі реєстрації у системі Signal. Користувачі йдуть на великий ризик використовуючи цей месенджер на кількох пристроях.

3.2 Побудова атак на протокол реєстрації супутніх пристроїв в системі WhatsApp

В результаті проведеного детального дослідження протоколів реєстрації в системі WhatsApp було виявлено вразливості, які описано вище в попередньому підрозділі. З використанням цих вразливостей вперше на цей протокол мною побудовано атаки, де розглянуто передумови, кроки та результати кожної атаки.

Атаки на протокол реєстрації в системі WhatsApp можуть мати різноманітні цілі, такі як порушення приватності користувачів, отримання несанкціонованого доступу до їхніх повідомлень або навіть викрадення їхньої особистої інформації. Важливо зрозуміти, що такі атаки створюють серйозні загрози для безпеки та конфіденційності даних учасників спілкування.

Алгоритм 3.1. (Атака додавання зловмисного пристрою)

Передумови: потрібен розблокований основний пристрій користувача з додатком WhatsApp (якщо є окремо пароль на додаток, то він повинен бути відкритим).

Крок 1. Підготувати заранння зловмисний QR-код того пристрою, який стане в майбутньому супутнім пристроєм жертви.

Крок 2. Просканувати залишеним розблокованим пристроєм зловмисний QR-код. Реєстрація зловмисного супутнього пристрою відбувається без участі користувача і не є потрібним навіть знання особистого приватного ключа.

Результат атаки: зловмисник стає повноцінним супутнім пристроєм, який може читати вже надіслані та отримані повідомлення та володіти доступом до того часу, поки людина не помітить, що до його

основного пристрою було додано зловмисний додатковий пристрій.

Алгоритм 3.2. (Атака відновлення ключа)

Передумови: людина хоче зісканувати за допомогою основного пристрою свій QR-код для підключення додаткового.

Крок 1. Ми припускаємо, що додатковий пристрій хороший і з ним все добре. Проводимо спробу перехопити ключ між цією передачею.

Крок 2. Зловмисник перехопив $QR(id\_pk^c, lsk^c)$ (наприклад, сфотографував з великої відстані), який сканує людина для того, щоб додати супутній пристрій.

Крок 3. lsk^c та id_pk^c знаходиться у QR-коді у відкритому вигляді, отже нам не потрібно їх розшифровувати, а лише витягнути звідти.

Крок 4. Маючи ці ключі ми можемо перехопити його пакет до сервера, розшифрувати дані, отримати тимчасовий доступ до особистого приватного ключа id_sk^p.

Результат атаки: знання приватного особистого ключа id_sk^p.

Алгоритм 3.3. (Атака додавання нових зловмисних пристроїв на справжньому супутньому пристрої)

Зауваження: WhatsApp дозволяє додавати супутні пристрої лише з основного пристрою.

Передумови: отримання зловмисником фізичного доступу до супутнього пристрою, на якому вже налаштований обліковий запис WhatsApp.

Крок 1. Зловмисник отримує доступ до захищених файлів або пам'яті пристрою, де зберігається ключ ідентифікації (identity key) для облікового запису користувача.

Крок 2. Він копіює або перехоплює ключ ідентифікації з супутнього пристрою.

Крок 3. Зловмисник використовує отриманий ключ ідентифікації для реєстрації нових пристроїв у якості додаткових супутніх пристроїв облікового запису WhatsApp користувача.

Крок 4. Сервер приймає запити на реєстрацію нових пристроїв, оскільки ключ ідентифікації вважається валідним.

Крок 5. Зловмисник успішно додає нові пристрої до облікового запису WhatsApp без відома або підтвердження користувача.

Результат атаки: зловмисник може створити підроблений пристрій або використати інший супутній пристрій, який не належить користувачу, і зареєструвати його в якості додаткового пристрою до облікового запису WhatsApp. Зокрема, він отримує несанкціонований доступ до повідомлень, контактів та інших конфіденційних даних, які зберігаються в обліковому запису користувача на нових пристроях.

Алгоритм 3.4. (MITM-атака для перехоплення ідентифікаційних даних)

Передумови: зловмисник повинен мати можливість перехоплювати мережевий трафік між супутнім пристроєм і сервером WhatsApp, створити підроблений супутній пристрій.

Крок 1. Зловмисник перехоплює мережевий трафік між супутнім пристроєм та сервером WhatsApp, наприклад, використовуючи метод атаки "Man-in-the-Middle"(MITM).

Крок 2. Зловмисник здійснює атаку на аутентифікацію, перехоплюючи та зберігаючи токен доступу (access token) або інші ідентифікаційні дані, які використовуються для взаємодії між супутнім пристроєм та сервером WhatsApp.

Крок 3. Зловмисник створює підроблений супутній пристрій, використовуючи перехоплені ідентифікаційні дані та аутентифікаційний токен.

Крок 4. Зловмисник реєструє підроблений супутній пристрій на сервері WhatsApp, використовуючи підроблені ідентифікаційні дані.

Крок 5. Сервер WhatsApp, сприймаючи підроблені дані як законні, реєструє підроблений супутній пристрій і надає йому доступ до облікового запису.

Крок 6. Зловмисник отримує повний доступ до повідомлень, даних

та функцій облікового запису на підробленому супутньому пристрої. *Результат атаки*: отримання зловмисником повного контролю над обліковим записом користувача на підробленому супутньому пристрої. Зловмисник може отримувати доступ до всіх повідомлень, даних і функцій облікового запису, які зазвичай доступні на справжньому супутньому пристрої.

Складність атак

Алгоритмічна складність є тривіальною. Всі створені мною та наведені вище атаки є ефективними та такими, що практично реалізуються.

3.3 Протоколи реєстрації супутніх пристроїв в системах обміну повідомленнями Session та Viber

Безліч месенджерів пропонують підтримку декількох пристроїв, проте мало з них описують протокол реєстрації супутніх пристроїв. Її реалізація залишається на розсуд розробника. Інколи кроки цього протоколу можна відновити за офіційною документацією або ж кодом. Частіше за все вона залишається прихованою, що не дає змогу користувачам впевнитись, що використання є безпечним та не призведе до втрати даних.

Один із можливих підходів для забезпечення безпеки та надійності реєстрації супутніх пристроїв полягає у відкритій документації, яка описує всі протоколи та методи. Це допомогло б виявити потенційні слабкі місця в реалізації протоколу та виправити їх до випуску оновлення.

Проаналізуємо інші протоколи реєстрації супутніх пристроїв для того, щоб показати, що їх існує дуже багато і в цьому є проблема. Кожен розробник намагається створити свій протокол, проте не всім вдається побудувати стійкий протокол. Розглянемо більш детально два популярні протоколи реєстрації супутніх пристроїв в месенджерах Session та Viber.

Система захищеного обміну повідомленнями Session

Session [33] — це система миттєвого обміну повідомленнями на основі відкритих ключів з відкритим вихідним кодом. Перша версія вийшла 4 лютого 2020 року. Месенджер використовує наскрізне шифрування та набір децентралізованих серверів на основі блокчейну із мінімальним доступом до метаданих користувача. Використовується розширений протокол узгодження ключів Діффі—Хеллмана (X3DH) та алгоритм подвійного храповика для отримання ключів повідомлення.

Багато систем вимагають підтвердження телефонного номера або електронної адреси для реєстрації. Однак такі вимоги створюють серйозні проблеми з конфіденційністю та безпекою для користувачів. Session не використовує їх як основу своєї системи облікових записів. Натомість ідентифікація користувача встановлюється через генерацію пари ключів X25519. Ці ключі не потрібно пов'язувати з будь-яким іншим ідентифікатором і можна генерувати нові пари ключів на пристрої за секунди. Користувачам пропонується записати свій довгостроковий закритий ключ, щоб відновити свій обліковий запис, якщо його пристрій буде втрачено або знищено.

Вони розробили власний протокол на основі системи реєстрації супутніх пристроїв в Signal, але спрощений і відповідно до їхніх потреб. У березні 2021 року у своєму блозі оголосили про його запуск для користувачів.

Розглянемо користувача A з двома пристроями (A1; A2), який хоче почати спілкуватися з користувачем B, який також має два пристрої (B1;B2). Для цього необхідно, щоб була проведена синхронізація ключів між двома пристроями.

Алгоритм 3.5. Протокол реєстрації супутніх пристроїв в системі Session

Крок 1. Користувач A отримує публічний відкритий ключ основного пристрою B, тобто id_pk^{B1}.

Крок 2. Використовуючи цей відкритий ключ, один із пристроїв A

(наприклад A1) надсилає запит про дружбу до групи основного пристрою B, що містить пакет попередніх ключів і список пов'язаних пристроїв (A2).

Крок 3. Коли B отримує запит користувача A, то будь-який із пристроїв може прийняти його і розпочати сеанс з A1.

Крок 4. Пристрій, який приймає запит, наприклад B2, потім використовує попередньо встановлений парний канал, щоб сповістити інший пристрій B1 про вжиті ним дії. Також він дає вказівку B1 почати сеанс з усіма пристроями A.

Крок 5. B2 також встановлює інші сеанси з пристроями A. Нарешті, B2 надає A відкритий ключ усіх своїх власних зв'язаних пристроїв (B1), щоб A знав, що їх потрібно зв'язати.

На рис. 3.3 зображено пересилання інформації з використанням кількох пристроїв. Після цього процесу A та B можуть почати спілкуватися.

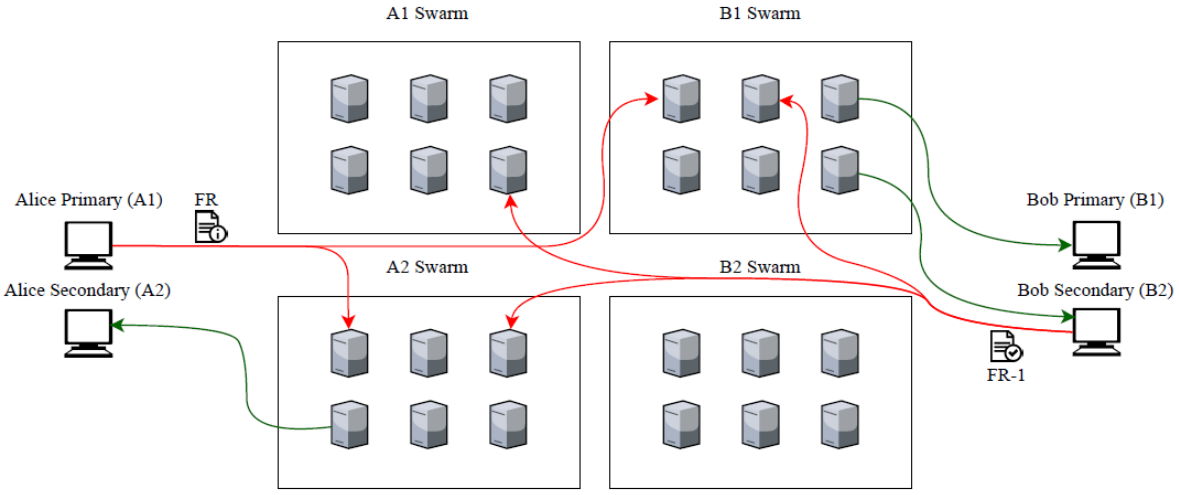

Рисунок 3.3 – Запит A до B для підключення кількох пристроїв

В подальшому користуванні A одночасно надсилатиме повідомлення іншому користувачу на пристрої B1, B2 та на власний супутній пристрій A2.

Система захищеного обміну повідомленнями Viber

Viber [34] — це система миттєвого обміну повідомленнями, яка використовує наскрізне шифрування. При реєстрації підв'язується до номера мобільного телефону.

Автентифікація здійснюється шляхом спільного використання приватного ідентифікаційного ключа id_sk^c між усіма пристроями одного облікового запису. Ідентифікаційний ключ id_sk^c генерується лише основним пристроєм і передається до супутніх пристроїв під час реєстрації безпечним способом.

Алгоритм 3.6. Протокол реєстрації супутніх пристроїв в системі Viber

Крок 1. Супутній пристрій генерує випадковим чином короткочасну 256-бітну пару ключів Curve-25519.

Крок 2. Супутній пристрій випадковим чином генерує QR-код, що містить $UDID$ пристрою (загальнодоступний унікальний ідентифікатор пристрою, згенерований Viber) і публічний ключ з пари тимчасових ключів.

Крок 3. Користувач використовує свій основний пристрій для сканування QR-коду.

Крок 4. Основний пристрій також генерує 256-бітну пару ключів Curve-25519 і виконує обчислення DH за допомогою відкритого ключа з QR. Результат хешується за допомогою SHA256 для створення спільного секрету. Хеш скорочується до 160 біт.

Крок 5. Основний пристрій зашифровує власний приватний ідентифікаційний ключ id_sk^p цим секретом і надсилає його та публічну частину ефемерного ключа через сервери Viber на супутній пристрій, ідентифікований за його $UDID$, який було отримано з QR-коду.

Крок 6. Зашифрований текст підписується за допомогою HMACSHA256.

Крок 7. Супутній пристрій отримує повідомлення, виконує той

самий DH і хеш, щоб отримати ключ, і використовує його для розшифрування основного приватного ідентифікаційного ключа id_sk^p.

Особистий ключ є частиною ланцюга DH, який створює спільні ключі для сеансів один на один. Таким чином, без правильного ідентифікатора супутній пристрій не може брати участь у жодних безпечних розмовах, частиною яких є основний пристрій.

3.4 Загальні рекомендації для побудови надійного протоколу реєстрації супутніх пристроїв

Всі наведені системи використовують подвійного храповика, який є стандартизованим та стійким. Проте досі не існує єдиного надійного протоколу реєстрації супутніх пристроїв. Поки він не готовий, можна сформувати перелік вимог, з огляду проведеного дослідження, побудованих атак та додатково розглянутих систем:

– побудувати єдиний протокол, щоб всі узгодили та погодились з ним, адже це простіше, аніж будувати кожен свій. Як показало моє дослідження, новостворені протоколи не завжди є кращими, аніж їх попередники;

– не передавати довготривалий ключ в QR-коді, тому що це може бути легко перехоплено зловмисником;

– при розробці та оновленнях перевіряти протокол на наявність функціональних помилок;

– використовувати наявні стандартні криптопримітиви з їх основною метою застосування;

– підключати сервер для перевірки супутніх пристроїв, щоб вони не були нав'язані пізніше без підтвердження з боку сервера;

– не виконувати зайві обчислення, які не виконують жодних криптографічних функцій;

– використовувати відносно захищений канал комунікації,

наприклад, шифрування через SSL/TLS, для захисту від перехоплення та прослуховування передаваних даних;

– містити заходи для запобігання повторного використання реєстраційних даних. Це може бути використання одноразових токенів або випадкових значень для кожної реєстрації нового супутнього пристрою;

– повинен мати можливість моніторингу подій, пов'язаних з реєстрацією супутніх пристроїв. Це допоможе виявити будь-які підозрілі або аномальні активності.

Висновки до розділу 3

Досліджено можливі ризики, пов'язані з використанням протоколу реєстрації в системі WhatsApp. Наведено приклади вразливостей, які можуть бути використані зловмисниками для отримання несанкціонованого доступу до облікових записів користувачів. Діаграми протоколів відновлено за оновленою технічною документацією.

Крім того, був проведений порівняльний аналіз протоколів реєстрації супутніх пристроїв в системах Signal і WhatsApp. Виявлено певні суттєві недоліки, в більшості випадків для системи WhatsApp. На основі детального аналізу протоколу реєстрації супутніх пристроїв в системі WhatsApp та наявності вразливостей було побудовано цілий перелік атак на цей протокол: атака додавання зловмисного пристрою; атака відновлення ключа; атака додавання нових зловмисних пристроїв на справжньому супутньому пристрої; MITM-атака для перехоплення ідентифікаційних даних, яка дає повний контроль над обліковим записом користувача.

Розглянуто, що існують інші протоколи реєстрації супутніх пристроїв, наприклад в системах Session та Viber. Це призвело до того, що було сформовано загальні рекомендації для побудови протоколу реєстрації супутніх пристроїв.

ВИСНОВКИ

Проведений аналіз опублікованих джерел за тематикою систем захищеного обміну повідомленнями та їх складових показав, що вони мають безліч варіантів покращень та усунення недоліків. Досліджена стійкість протоколу реєстрації в системі Signal доводить, що навіть у такому найбільш захищеному протоколі є слабкі місця. Проаналізовано всі наявні особливості побудови систем захищеного обміну повідомленнями, проведено стійкості системи Signal на анонімність та безпеку. Більшість складових частин систем захищеного обміну повідомленнями є в більшості встановленими та використовують однакові протоколи, наприклад X3DH, подвійного храповика.

Основну увагу було приділено протоколу реєстрації супутніх пристроїв в системах захищеного обміну повідомленнями та розглянуто, які особливості побудови вони мають, які є наявні вразливості та побудовані атаки на нього.

1. В роботі проведено одне з перших досліджень новітнього протоколу реєстрації супутніх пристроїв в системі WhatsApp, який повноцінно ввели в користування лише 25 квітня 2023 року. За описом алгоритму зроблено схему та досліджено особливості його побудови та функціонування.

2. Вперше проведено порівняльний аналіз особливостей протоколів реєстрації в системах Signal та WhatsApp, виявлено відмінності та недоліки в протоколі реєстрації системи WhatsApp, які впливають на безпеку та конфіденційність користувачів. В результаті було вперше сформовано список переваг та недоліків кожного протоколу, а також знайдено цілий перелік недоліків в протоколі реєстрації в системі WhatsApp.

3. Використовуючи знайдені особливості та вразливості протоколу реєстрації супутніх пристроїв в системі WhatsApp побудовано мною вперше низку атак на цей протокол реєстрації: атака додавання

зловмисного пристрою; атака відновлення ключа; атака додавання нових зловмисних пристроїв на справжньому супутньому пристрої; MITM-атака для перехоплення ідентифікаційних даних, яка дає повний контроль над обліковим записом користувача. Обчислювальна складність є тривіальною, але всі атаки є такими, що практично реалізуються. Більшість розглянутих передумов до атак не є складними для виконання. Вони можуть бути втілені без додаткових засобів, особливо в сучасних умовах, адже телефон може бути легко залишений без нагляду на невеликий термін. Результати атак є практичними та дають змогу отримати доступ абсолютно до всіх даних користувача, адже пристрої синхронізуються. Також вони дають змогу дізнатись приватний особистий ключ id_sk^p або ж стати зловмисним супутнім пристроєм та пасивно спостерігати за новими діями користувача.

4. Додатково було проаналізовано протоколи реєстрації супутніх пристроїв в системах Session та Viber. З урахуванням наявності великої кількості різних протоколів в різних системах на основі власного аналізу було сформовано рекомендації щодо встановлення єдиного протоколу реєстрації супутніх пристроїв в системах захищеного обміну повідомленнями та його властивості.

Ці результати мають важливе значення для користувачів, оскільки підкреслюють необхідність покращення безпеки та захисту протоколу реєстрації в системі WhatsApp. Розуміння слабких місць у протоколі дозволяє розробникам зосередитися на вдосконаленні механізмів реєстрації та забезпеченні конфіденційності облікових записів.

Актуальною темою подальших досліджень виглядає побудова єдиного протоколу реєстрації супутніх пристроїв, який буде не вразливим до відомих атак.

ПЕРЕЛІК ПОСИЛАНЬ

1. *Borisov N.*, *Goldberg I.*, *Brewer E.* Off-the-record communication, or, why not to use PGP // Proceedings of the 2004 ACM workshop on Privacy in the electronic society. — ACM, 10.2004. — DOI: 10.1145/1029179.1029200.

2. ASICS: authenticated key exchange security incorporating certification systems / C. Boyd [та ін.] // International Journal of Information Security. — 2016. — Січ. — Т. 16, № 2. — С. 151—171. — DOI: 10.1007/s10207-015-0312-y.

3. *P.Z.V M.* SilentCircle.SCIMP|SilentCircle //. — 2015. — URL: https://web.archive.org/web/20150718145410/https://silentcircle.com/scimp-protocol.

4. Electronic Frontier Foundation. Secure Messaging Scorecard. //. — 2016. — URL: https://www.eff.org/node/82654.

5. SoK: Secure Messaging / N. Unger [та ін.] // 2015 IEEE Symposium on Security and Privacy. — IEEE, 05.2015. — DOI: 10.1109/sp.2015.22.

6. *Marlinspike M.* Advanced cryptographic ratcheting //. — 2013. — URL: https://whispersystems.org/blog/advanced-ratcheting/.

7. A Formal Security Analysis of the Signal Messaging Protocol / K. Cohn-Gordon [та ін.] // Journal of Cryptology. — 2020. — Вер. — Т. 33, № 4. — С. 1914—1983. — DOI: 10.1007/s00145-020-09360-1.

8. *Alwen J.*, *Coretti S.*, *Dodis Y.* The Double Ratchet: Security Notions, Proofs, and Modularization for the Signal Protocol // Advances in Cryptology – EUROCRYPT 2019. — Springer International Publishing, 2019. — С. 129—158. — DOI: 10.1007/978-3-030-17653-2_5.

9. *Langley A.*, *Hamburg M.*, *Turner S.* Elliptic Curves for Security : тех. звіт. — 01.2016. — DOI: 10.17487/rfc7748.

10. *Perrin T.* "The XEdDSA and VXEdDSA Signature Schemes" //. — 2016. — URL: https://whispersystems.org/docs/specifications/xeddsa/.

11. *Krawczyk H.*, *Eronen P.* HMAC-based Extract-and-Expand Key Derivation Function (HKDF) : тех. звіт. — 05.2010. — DOI: 10.17487/rfc5869.

12. *Perrin T.*, *Marlinspike M.* "The Double Ratchet algorithm" //. — 2016. — URL: https://signal.org/docs/%20specifications/doubleratchet/.

13. *Cremers C.*, *Jacomme C.*, *Naska A.* Formal Analysis of Session-Handling in Secure Messaging: Lifting Security from Sessions to Conversations. — 2022. — URL: https://eprint.iacr.org/2022/1710. Cryptology ePrint Archive, Paper 2022/1710.

14. How Secure is TextSecure? / T. Frosch [та ін.] // 2016 IEEE European Symposium on Security and Privacy (EuroS&ampP). — IEEE, 03.2016. — DOI: 10.1109/eurosp.2016.41.

15. Universally Composable End-to-End Secure Messaging / R. Canetti [та ін.] // Advances in Cryptology – CRYPTO 2022. — Springer Nature Switzerland, 2022. — С. 3—33. — DOI: 10.1007/978-3-031-15979-4_1.

16. *Jost D.*, *Maurer U.*, *Mularczyk M.* A Unified and Composable Take on Ratcheting // Theory of Cryptography. — Springer International Publishing, 2019. — С. 180—210. — DOI: 10.1007/978-3-030-36033-7_7.

17. *Maurer U.* Constructive Cryptography – A New Paradigm for Security Definitions and Proofs // Theory of Security and Applications. — Springer Berlin Heidelberg, 2012. — С. 33—56. — DOI: 10.1007/978-3-642-27375-9_3.

18. A More Complete Analysis of the Signal Double Ratchet Algorithm / A. Bienstock [та ін.] // Advances in Cryptology – CRYPTO 2022. — Springer Nature Switzerland, 2022. — С. 784—813. — DOI: 10.1007/978-3-031-15802-5_27.

19. *Perrin T.*, *Marlinspike M.* Open Whisper Systems: Technical information: Specifications and software libraries for developers //. — 2016. — URL: https://signal.org/docs/.

20. SAID: Reshaping Signal into an Identity-Based Asynchronous Messaging Protocol with Authenticated Ratcheting / O. Blazy [та ін.] // 2019 IEEE European Symposium on Security and Privacy (EuroS&ampP). — IEEE, 06.2019. — DOI: 10.1109/eurosp.2019.00030.

21. Help, My Signal has Bad Device! / J. Wichelmann [та ін.] // Detection of Intrusions and Malware, and Vulnerability Assessment. — Springer International Publishing, 2021. — С. 88—105. — DOI: 10.1007/978-3-030-80825-9_5.

22. Hybrid Signal protocol for post-quantum email encryption / S. Stadler [та ін.]. — 2021. — URL: https://eprint.iacr.org/2021/875. Cryptology ePrint Archive, Paper 2021/875.

23. A Concrete Treatment of Efficient Continuous Group Key Agreement via Multi-Recipient PKEs / K. Hashimoto [та ін.] // Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. — ACM, 11.2021. — DOI: 10.1145/3460120.3484817.

24. Orca: Blocklisting in Sender-Anonymous Messaging / N. Tyagi [та ін.]. — 2021. — URL: https://eprint.iacr.org/2021/1380 ; https://eprint.iacr.org/2021/1380. Cryptology ePrint Archive, Paper 2021/1380.

25. *Caforio A.*, *Durak F. B.*, *Vaudenay S.* Beyond Security and Efficiency: On-Demand Ratcheting with Security Awareness // Public-Key Cryptography – PKC 2021. — Springer International Publishing, 2021. — С. 649—677. — DOI: 10.1007/978-3-030-75248-4_23.

26. *Dobson S.*, *Galbraith S. D.* Post-Quantum Signal Key Agreement from SIDH // Post-Quantum Cryptography. — Springer International Publishing, 2022. — С. 422—450. — DOI: 10.1007/978-3-031-17234-2_20.

27. Contact Discovery in Mobile Messengers: Low-cost Attacks, Quantitative Analyses, and Efficient Mitigations / C. Hagen [та ін.]. — 2022. — URL: https://eprint.iacr.org/2022/875. Cryptology ePrint Archive, Paper 2022/875.

28. CoCoA: Concurrent Continuous Group Key Agreement / J. Alwen [та ін.] // Advances in Cryptology – EUROCRYPT 2022. — Springer International Publishing, 2022. — С. 815—844. — DOI: 10.1007/978-3-031-07085-3_28.

29. *Tinoco A.*, *Gao S.*, *Shi E.* Enigmap : External-Memory Oblivious Map for Secure Enclaves. — 2022. — URL: https://eprint.iacr.org/2022/1083. Cryptology ePrint Archive, Paper 2022/1083.

30. MARSHAL: Messaging with Asynchronous Ratchets and Signatures for faster HeALing / O. Blazy [та ін.]. — 2022. — DOI: 10.1145/3477314.3507044. — URL: https://eprint.iacr.org/2022/486. Cryptology ePrint Archive, Paper 2022/486.

31. Membership Privacy for Asynchronous Group Messaging / K. Emura [та ін.]. — 2022. — URL: https://eprint.iacr.org/2022/046. Cryptology ePrint Archive, Paper 2022/046.

32. Multi-Device for Signal / S. Campion [та ін.]. — 2019. — URL: https://eprint.iacr.org/2019/1363. Cryptology ePrint Archive, Paper 2019/1363.

33. *Jefferys K.*, *Shishmarev M.*, *Harman S.* Session: A Model for End-To-End Encrypted Conversations With Minimal Metadata Leakage //. — 2020. — URL: https://arxiv.org/pdf/2002.04609.pdf.

34. Viber Encryption Overview //. — 2018. — URL: https://www.viber.com/app/uploads/viber-encryption-overview.pdf.