НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

Проектування і розробка криптографічних систем

Лабораторна робота №3

Дослідження криптографічних протоколів систем WebMoney, PayPal

Виконали:
Сковрон Роман ФІ-42мн
Волинець Сергій ФІ-42мн
Молдован Дмитро ФІ-42мн

Київ – 2025

1 Мета роботи

Метою роботи є дослідження особливостей реалізації криптографічних механізмів у сучасних платіжних системах та оцінка їхнього впливу на конфіденційність, цілісність і автентичність платіжних даних. У рамках дослідження передбачається проаналізувати використані криптографічні примітиви та протоколи, а також обґрунтувати їх придатність і рівень захищеності в типових сценаріях електронних платежів.

2 Постановка задачі

Дослідити особливості реалізації криптографічних протоколів, а також особливості роботи з електронними гаманцями систем WebMoney та PayPal. Розробити детальний опис проведеного дослідження особливостей реалізації криптографічних механізмів протоколів систем WebMoney та PayPal. Дослідити містити загальні теоретичні відомості побудови платіжних систем та їх основні характеристики (специфікація SET), зокрема систем мікроплатежів, таких як Payword та Micromint, та протоколи електронних грошей. Для кожної наведеної системи або протоколу необхідно обґрунтувати його захищеність та вибір криптографічних примітивів.

3 Огляд захищених платіжних систем та специфікації SET

Електронні платіжні системи повинні забезпечувати конфіденційність, цілісність та автентичність транзакцій. На початкових етапах розвитку онлайн-платежів галузеві консорціуми розробили стандарти на кшталт Secure Electronic Transaction (SET) для захисту операцій з кредитними картками в Інтернеті. SET не був самостійною платіжною системою, а являв собою набір криптографічних протоколів і форматів, накладених поверх уже існуючих карткових мереж. Він спирався на цифрові сертифікати X.509 для всіх учасників (держатель картки, торговець, банк) і використовував різноманітні криптографічні механізми (шифрування та електронні підписи) для виконання вимог безпеки. Ключовими властивостями були взаємна автентифікація клієнта та продавця, конфіденційність платіжних даних і забезпечення цілісності повідомлень. Для забезпечення конфіденційності SET застосовував симетричне шифрування (спочатку алгоритм DES). Для цілісності та автентифікації використовувалися електронні підписи RSA (у поєднанні з геш-функцією SHA-1), а також інколи HMAC на основі SHA-1.

Однією з помітних інновацій у SET був механізм подвійного підпису. Цей криптографічний прийом пов'язує інформацію про замовлення клієнта (Order Information, OI) та платіжну інформацію (Payment Information, PI) в єдиний підписаний блок, не розкриваючи при цьому реквізити картки продавця. Ідея полягає в тому, що клієнт окремо обчислює геші від OI та PI, потім конкатенує ці два геші, ще раз гешує результат і підписує отримане значення своїм закритим ключем. Утворений подвійний підпис надсилається як продавцю, так і банку. Продавець бачить лише OI (без номера картки), але може перевірити, що подвійний підпис узгоджується з гешем OI, який він обчислює самостійно; аналогічно банк перевіряє підпис відносно геша PI. Таким чином забезпечується незмінність транзакції та логічний зв'язок конкретного платежу з конкретним замовленням без розкриття чутливих даних стороннім суб'єктам. SET забезпечував високий рівень безпеки та довіри: він задовольняв вимоги конфіденційності, цілісності та взаємної автентифікації. Водночас протокол був складним і вимагав спеціалізованого клієнтського програмного забезпечення й розвиненої інфраструктури цифрових сертифікатів, тому не отримав широкого поширення на

ринку. Нині в реальних платіжних системах частіше застосовують простіші рішення, такі як протокол 3-D Secure.

4 Система PayPal: криптографічні механізми та захист електронного гаманця

PayPal є однією з найбільших у світі систем електронних гаманців, яка дає змогу користувачам надсилати й отримувати платежі онлайн. Безпека PayPal спирається на галузеві стандарти криптографічних протоколів для захисту даних як під час передавання, так і під час зберігання. Кожна взаємодія з платформою PayPal захищена потужним транспортним шифруванням. На практиці PayPal використовує захищені HTTPS-з'єднання з протоколом TLS (Transport Layer Security) високого рівня для всіх комунікацій. Усі транзакції є зашифрованими за принципом «від кінця до кінця»: дані шифруються на пристрої користувача та розшифровуються лише на серверах PayPal, що унеможливлює їх пасивне перехоплення. Зокрема, PayPal застосовує SSL/TLS зі 128-бітним або сильнішим шифруванням (нині зазвичай — 256-бітне), тому перехоплена інформація виглядає для зловмисника як випадковий набір символів. Такий рівень шифрування вважається практично незламним для атакуючих і забезпечує конфіденційність деталей транзакцій. Крім того, PayPal здійснює перевірку цілісності браузера, щоб гарантувати, що користувачі під'єднуються через безпечні, актуальні конфігурації TLS. Ці заходи захищають від атак «людина посередині» та інших мережевих загроз, оскільки PayPal блокує вхід з браузерів, які не відповідають його вимогам безпеки.

Окрім транспортного захисту, PayPal застосовує розвинені механізми автентифікації та обробки даних. Паролі користувачів передаються виключно зашифрованими каналами й зберігаються в незворотній формі (з використанням надійного хешування та випадкової «солі»), що відповідає найкращим практикам захисту облікових даних. Платформа підтримує двофакторну автентифікацію (2FA) через PayPal Security Key — одноразові PIN-коди, які надсилаються на пристрій користувача і виступають додатковим криптографічним шаром захисту, щоб до гаманця мав доступ лише авторизований власник. Останніми роками PayPal також додав підтримку *passkeys* (FIDO2/WebAuthn), що дає змогу використовувати криптографію з відкритим ключем для входу в обліковий запис: приватний ключ зберігається на пристрої користувача, а PayPal перевіряє коректність підпису криптографічного виклику.

Під час здійснення платежу архітектура системи PayPal побудована так, що інша сторона операції ніколи не бачить конфіденційних фінансових реквізитів користувача, таких як номер кредитної картки чи банківського рахунку. PayPal виступає довіреним посередником, який приховує платіжну інформацію від іншої сторони. Хоча це не окремий криптографічний протокол, такий дизайн істотно знижує ризики: навіть якщо сайт продавця буде зламано, зловмисник не отримає дані картки, оскільки транзакція проходить через сервери PayPal у токенізованому вигляді. Усередині системи платіжна інформація користувачів зберігається відповідно до стандартів PCI DSS в зашифрованому вигляді.

Крім того, служби сповіщень PayPal також використовують криптографічну перевірку. Класичний механізм Instant Payment Notification (IPN) вимагав, щоб продавець підтверджував отримане сповіщення, надсилаючи його назад до PayPal по HTTPS (покладаючись на TLS для забезпечення автентичності). У сучасних вебхуках PayPal включає до HTTP-заголовків криптографічний підпис і надає публічний сертифікат, за допомогою якого можна перевірити, що сповіщення справді надійшло від PayPal. Платформа підписує вміст вебхуків (наприклад, з використанням алгоритмів на кшталт RSA-SHA256), а розробники

можуть перевіряти ці підписи, застосовуючи публічний ключ PayPal, тим самим гарантуючи автентичність і цілісність повідомлень.

Підсумовуючи, модель безпеки PayPal побудована на стандартних криптографічних «будівельних блоках»: шифрування TLS під час передавання даних, безпечне зберігання облікових даних (хешування та шифрування), можливості автентифікації на основі криптографії з відкритим ключем (API-сертифікати, FIDO-passkeys) і підписування критично важливих даних (коди платіжних кнопок, вебхуки) для запобігання їх підробці. Завдяки такому вибору PayPal забезпечує конфіденційність даних користувача, цілісність транзакцій і достовірність тверджень про ідентичність, водночас залишаючись сумісним з існуючою фінансовою інфраструктурою. Застосування перевірених криптографічних примітивів (RSA, AES, SHA-256 тощо) й протоколів дає змогу збалансувати високий рівень безпеки з зручністю, якої очікують користувачі глобальної платіжної системи. Важливо, що на відміну від деяких інших систем, PayPal не вимагає від кінцевих користувачів самостійного керування криптографічними ключами — уся складність прихована всередині платформи. Це суттєво спрощує користування системою та є одним з чинників її широкого розповсюдження.

5 Система WebMoney: криптографічні протоколи та захист електронного гаманця

WebMoney Transfer є системою електронного гаманця та платіжною системою, яка застосовує інший підхід до безпеки, роблячи акцент на криптографічних ключах і облікових даних, що перебувають під контролем самого користувача. Заснована у 1998 році, WebMoney спочатку була спроєктована як система для захищених онлайн-транзакцій, у межах якої кожному користувачеві надається унікальний ідентифікатор та можливість самостійно керувати криптографічною автентифікацією свого облікового запису. Користувач WebMoney отримує WMID (WebMoney Identifier) — 12-значний номер, який слугує основним ідентифікатором його облікового запису. Кошти зберігаються у *purses* (гаманцях), деномінованих у різних валютах або активах (наприклад, WMZ для доларів США). Відмінною рисою WebMoney є спосіб захисту доступу до рахунку та авторизації транзакцій: кожна операція може бути санкціонована *цифровим підписом* користувача з використанням ключів, що зберігаються в користувача, а не лише на стороні сервера.

WebMoney підтримує кілька способів автентифікації, усі з яких ґрунтуються на криптографічних механізмах. Для базового доступу користувач може входити за допомогою WMID (або прив'язаних e-mail/телефону) та пароля, як правило, у поєднанні з одноразовим кодом, надісланим на телефон, що реалізує двофакторну перевірку. Важливішу роль відіграють розширені клієнти WebMoney, які використовують криптографію з відкритим ключем: користувачам видаються персональні цифрові сертифікати та файли секретних ключів, які застосовуються для автентифікації та підпису транзакцій. У класичному програмному забезпеченні WebMoney Keeper (WinPro/Classic) особу користувача та його повноваження щодо розпорядження коштами підтверджує володіння приватним ключем (збереженим у файлі, захищеному паролем, обраним користувачем) та відповідним сертифікатом. Коли ініціюється транзакція, програмний клієнт WebMoney формує цифровий підпис над її параметрами за допомогою приватного ключа користувача — цей підпис виступає криптографічним еквівалентом особистого «підтвердження» операції. Сервери системи перевіряють цей підпис за допомогою відкритого ключа користувача (що міститься в його сертифікаті), тим самим переконуючись, що запит справді походить від власника рахунку та не був змінений під час передавання. Такий підхід означає, що навіть оператори системи не проведуть транзакцію, якщо вона не підписана ключем користувача, що забез-

печує невідмовність. Фактично *довіра переноситься на криптографічний ключ*: володіння приватним ключем означає контроль над коштами. WebMoney офіційно наголошує на необхідності безпечного зберігання резервних копій файлів ключів; втрата приватного ключа без резерву може означати безповоротну втрату доступу до гаманця, оскільки сам сервіс не може просто «скинути» цей ключ.

Для підтримки такої інфраструктури відкритих ключів WebMoney випускає персональні сертифікати у межах власної служби. Під час реєстрації у веб-клієнті WebPro (браузерна версія Keeper) пара ключів генерується безпосередньо на пристрої користувача, а отриманий сертифікат прив'язується до відповідного WMID. Кореневий центр сертифікації WebMoney підписує ці сертифікати користувачів, завдяки чому система (і навіть сторонні сервіси) може довіряти їм для автентифікації. Фактично WebMoney створює повноцінну PKI (інфраструктуру відкритих ключів), у якій центральний сервер виступає центр ом сертифікації для користувацьких відкритих ключів. Настільні клієнти, такі як Keeper WinPro, можуть також інтегруватися з зовнішніми засобами зберігання ключів — наприклад, апаратними токенами (eToken PRO), що додатково підвищує рівень захисту. Таким чином, WebMoney переносить контроль над криптографічними операціями до користувача, на відміну від PayPal, де облікові дані зберігаються переважно централізовано. Така децентралізована модель облікових даних часто пояснюється як одна з причин високого рівня безпеки WebMoney: оскільки приватні ключі користувачів не зберігаються на центральних серверах, масштабні витоки облікових даних є менш імовірними. Дійсно, WebMoney не повідомляла про серйозні системні злами, які б компрометували ядро криптографії; типові інциденти безпеки пов'язані радше з фішинговими атаками або шкідливим ПЗ на стороні користувача (коли зловмисник змушує користувача розкрити свої ключі чи паролі), а не з криптографічними вразливостями.

Окрім механізмів автентифікації, WebMoney забезпечує безпеку транзакцій і на інших рівнях. Усі з'єднання з серверами WebMoney здійснюються за допомогою зашифрованих каналів (HTTPS/SSL). До того ж транзакції у WebMoney є атомарними й захищеними від атак «людина посередині» та збоїв з'єднання: згідно з офіційною документацією, операція або виконується повністю, або взагалі не застосовується, і не існує проміжного стану, в якому кошти могли б «зникнути». Це забезпечується архітектурою системи, у якій перекази відображаються в централізованому, але надійному реєстрі: гроші завжди знаходяться або на гаманці відправника, або на гаманці одержувача, і ніколи не «зависають» між ними. Для чутливих операцій WebMoney також використовує коди підтвердження або одноразові паролі (через SMS або мобільний додаток E-Num), що додає ще один рівень «людської» перевірки поверх цифрового підпису.

Для взаємодії з продавцями WebMoney надає криптографічно захищені сповіщення, подібні до IPN у PayPal. Коли покупець здійснює платіж продавцеві через мерчант-інтерфейс WebMoney, система надсилає продавцю зворотний виклик із деталями платежу та «контрольним підписом» (LMI_HASH), який обчислюється або за допомогою HMAC-SHA256, або у вигляді повноцінного цифрового підпису (з використанням служби підписування WebMoney). Продавець може перевірити цей контрольний підпис, повторно обчисливши HMAC із використанням спільного секрету, або перевіривши цифровий підпис з використанням публічного ключа WebMoney, тим самим переконавшись у цілісності та автентичності отриманих даних. Якщо використовується метод SHA-256 HMAC, до гешу включається секрет, відомий лише продавцю та системі; якщо ж застосовується режим SIGN, WebMoney формує справжній цифровий підпис, використовуючи свій приватний ключ (компонент WMSignerX). В обох випадках продавець криптографічно впевнений, що сповіщення справді надійшло від WebMoney і що дані платежу не були змінені під час передавання. За своєю суттю це є аналогом криптографічної перевірки вебхуків у PayPal.

Отже, підхід WebMoney до безпеки характеризується зосередженням на криптографії на стороні користувача: персональні ключі та сертифікати аутентифікують користувачів і санкціонують транзакції. Використовувані криптографічні примітиви включають RSA (для цифрових підписів і сертифікат-орієнтованої автентифікації) та геш-функції сімейства SHA (для гешування даних і в складі HMAC при передачі повідомлень). Завдяки цьому WebMoney досягає високого рівня автентифікації (лише володар приватного ключа може ініціювати переказ), цілісності (цифрові підписи на транзакціях, HMAC на повідомленнях) і невідмовності. Вибір криптографічних примітивів у WebMoney зумовлений прагненням до більшої незалежності від сторонніх довірених компонентів (користувач не повинен довіряти базі даних паролів, натомість він довіряє власному ключу) і бажанням досягти високого рівня гарантій в умовах потенційно недовірених клієнтів та мереж. Ціна цього підходу — необхідність відповідального ставлення користувача до зберігання й резервування ключів, однак система надає відповідні інструменти (резервне копіювання, міграція ключів тощо). Практика показує, що така архітектура є досить стійкою протягом багатьох років: WebMoney загалом уникає масштабного шахрайства, доки користувачі належним чином захищають свої облікові дані.

6 Системи мікроплатежів: PayWord та MicroMint

Системи мікроплатежів — це спеціалізовані цифрові платіжні протоколи, призначені для надзвичайно малих платежів (наприклад, частки цента або кілька центів), у яких використання «важкої» криптографії на основі відкритого ключа для кожної транзакції було б занадто неефективним. У середині 1990-х років Р. Рівест і А. Шамір запропонували дві схеми мікроплатежів — *PayWord* та *MicroMint*, які демонструють різні криптографічні підходи до цієї проблеми. Головною метою обох схем є мінімізація дорогих операцій з відкритим ключем і перехід до легких криптографічних примітивів (наприклад, односторонніх геш-функцій), щоб зменшити накладні витрати на один платіж. Безпека в цих схемах має відносний характер: вони прагнуть зробити шахрайство або криптографічно неможливим, або економічно невигідним з огляду на дуже малу вартість кожного окремого платежу.

6.1 Схема PayWord: геш-ланцюги для мікроплатежів

PayWord — це кредитна схема мікроплатежів, оптимізована для послідовностей малих платежів одному продавцеві. Вона використовує вдалу комбінацію одноразового застосування криптографії з відкритим ключем і ланцюгів геш-значень для амортизації витрат. Типовий сценарій роботи протоколу виглядає так:

1. **Відкриття рахунку.** Користувач (платник) спершу відкриває рахунок у брокера (фінансового посередника) й отримує від нього *цифровий сертифікат*, який підтверджує особу користувача та його платоспроможність. Цей сертифікат містить відкритий ключ користувача і, можливо, інформацію про кредитний ліміт чи посилання на рахунок; він підписаний закритим ключем брокера і відіграє роль криптографічної «банківської гарантії». Цей етап виконується одноразово.
2. **Генерація геш-ланцюга.** Коли користувач хоче здійснити багато мікроплатежів одному продавцеві, він офлайн обчислює *ланцюг payword*. Ланцюг payword — це послідовність значень, утворених шляхом ітераційного застосування криптографічної геш-функції. Наприклад, користувач випадково обирає фінальне значення w_n, після чого обчислює

$$w_{n-1} = h(w_n), \quad w_{n-2} = h(w_{n-1}), \ldots, \quad w_0 = h(w_1),$$

де $h(\cdot)$ — одностороння геш-функція (Рівест і Шамір у свій час пропонували MD5). Ланцюг будується так, що, маючи будь-який елемент w_i, можна загешувати його й отримати наступне значення w_{i-1}, але неможливо ефективно перейти в зворотному напрямку (через односторонність h). Значення w_0 використовується як *комітмент* до всього ланцюга, а кожне наступне w_i (так званий payword) відповідає одній одиниці вартості. Довжина ланцюга n обирається так, щоб покрити очікувану кількість мікроплатежів.

3. **Комітмент продавцю (одноразовий підпис).** Далі користувач звертається до продавця й передає йому два об'єкти: (a) сертифікат, виданий брокером (щоб продавець знав відкритий ключ користувача та був упевнений, що брокер гарантує платежі), і (b) корінь геш-ланцюга w_0, підписаний закритим ключем користувача. Це звичайний цифровий підпис (наприклад, RSA), але він створюється *один раз* на початку платіжної сесії. Підписуючи w_0, користувач зобов'язується використовувати цей ланцюг для платежів і «обіцяє», що розкриватиме наступні прегеш-значення як оплату. Підписане повідомлення може включати w_0, довжину ланцюга, дату, ідентифікатор продавця тощо, щоб запобігти повторному використанню ланцюга. Продавець перевіряє підпис користувача за відкритим ключем із сертифіката, а також перевіряє підпис брокера на самому сертифікаті, щоб упевнитися в його чинності. На цьому етапі встановлюється довіра: відомо, що користувач має рахунок і може здійснити до n мікроплатежів.
4. **Мікроплатежі через розкриття геш-значень.** Для кожного мікроплатежу (наприклад, за кожен доступ до статті, кліку чи інший невеликий ресурс) користувач розкриває наступне геш-значення в ланцюгу. Протокол починається з «кінця» ланцюга і рухається вперед: перший платіж здійснюється шляхом розкриття w_1. Продавець обчислює $h(w_1)$ і перевіряє, що $h(w_1) = w_0$ (значення w_0 для нього вже відоме з підписаного комітменту). Це доводить, що w_1 є коректним і послідовним елементом ланцюга, тому продавець приймає його як оплату. Для наступного платежу користувач розкриває w_2; продавець перевіряє рівність $h(w_2) = w_1$, і так далі. Для i-го платежу користувач надсилає w_i, а продавець перевіряє, що $h(w_i) = w_{i-1}$, який уже був підтверджений або отриманий раніше. Такі геш-обчислення дуже швидкі (набагато дешевші, ніж операції з відкритим ключем), тож навіть тисячі мікроплатежів можуть бути оброблені з мінімальними накладними витратами. Кожен розкритий payword унікальний і не може бути повторно використаний або сфальсифікований продавцем, оскільки він не може обчислити жодне w_{i+1} без участі користувача (це вимагало б знаходження прегеша для h).
5. **Погашення (redemption).** Продавець накопичує отримані payword-и (значення до гешування) як доказ здійснених платежів. Пізніше він звертається до брокера для погашення — отримання «справжніх» грошей. Наприклад, якщо користувач розкрив значення до w_k, це означає, що було здійснено k мікроплатежів. Продавець передає брокеру підписаний користувачем комітмент (із w_0) та останнє отримане значення w_k. Брокер перевіряє цифровий підпис, а потім k разів послідовно гешує w_k, щоб переконатися, що $h^k(w_k) = w_0$ (тобто при k-кратному застосуванні геш-функції отримується корінь ланцюга). Це підтверджує, що продавець дійсно отримав послідовний ланцюг довжини k. Після цього брокер перераховує продавцю суму, що відповідає k мікроплатежам. Одночасно брокер контролює кредитні обмеження користувача, оскільки в сертифікаті могли бути закладені умови щодо максимальної суми або строку дії. Зазвичай ланцюг прив'язується до конкретного продавця (для кожного продавця користувач створює окремий ланцюг), що зменшує ризики й запобігає спробам одного продавця пред'явити payword-и, призначені для іншого.

Безпека PayWord базується на односторонності геш-ланцюга та цифрових підписах, що використовуються для початкового комітменту. Зловмисний користувач не може застосувати той самий ланцюг до двох різних продавців, оскільки комітмент, як правило, прив'язаний до конкретного продавця або не може бути повторно використаний без виявлення. Користувач також не може «перескочити» деякі платежі: він не здатен коректно розкрити w_5, не розкривши послідовно $w_1, \dots, w_4$, оскільки продавець одразу виявить невідповідність. З боку продавця ризик полягає лише в тому, що користувач може раптово перестати надсилати нові payword-и (наприклад, розірвати з'єднання); у цьому разі продавець просто не отримує подальших платежів, але може погасити вже накопичені. Шахрайство з боку продавця стримується тим, що брокер не виплатить йому кошти за payword-и, які не гешуються до задекларованого w_0. Цифровий підпис брокера на сертифікаті користувача унеможливлює участь фіктивних клієнтів. Найбільш ресурсомісткі операції з відкритим ключем виконуються лише один раз на сесію, що дає значну економію обчислювальних ресурсів. Як зазначали Рівест і Шамір, геш-операції на порядки швидші від RSA-операцій, тому PayWord може підтримувати дуже велику кількість мікроплатежів, спираючись по суті на одну перевірку підпису та велику кількість швидких гешів. Вибір примітивів (MD5 або SHA-1 як h, RSA для підписів) був доречним для кінця 1990-х; у сучасній реалізації їх можна замінити на SHA-256 та ECDSA. Важливо, що з огляду на обмежену вартість одного ланцюга схема може допустити невеликий залишковий ризик за умови, що вона «утримує чесних користувачів чесними». Загалом PayWord вважається безпечною за умови, що геш-функція є односторонньою (стійкою до прегеш-атак), а цифрові підписи — невразливі до підробки.

6.2 Схема MicroMint: «монети» на основі геш-колізій

MicroMint — це принципово інша схема, також запропонована Рівестом і Шаміром, яка намагається повністю *усунути* використання криптографії з відкритим ключем у процесі мікроплатежів. Це різновид системи електронних монет, що базується на ідеї геш-колізій. Філософія MicroMint полягає в тому, щоб прийняти дещо нижчий рівень криптографічного захисту (допускаючи невелику ймовірність шахрайства) заради надзвичайно високої ефективності. У цій схемі «монета» — це по суті набір бітів, який складно згенерувати у великій кількості без істотних обчислень, але дуже легко перевірити. Спрощено протокол працює так:

1. **Випуск монет через геш-колізії.** Монета MicroMint представляється як специфічна *k-кратна колізія* геш-функції. Наприклад, обирається геш-функція h з n-бітним виходом. k-колізія означає знаходження k різних вхідних значень $x_1, x_2, \dots, x_k$ таких, що
$$h(x_1) = h(x_2) = \cdots = h(x_k) = Y.$$
Значення Y (разом з набором x_i або їх компактним поданням) і є «монетою». Знайти такі колізії випадковим перебором дуже складно: ймовірність того, що k випадкових значень дадуть однаковий геш, мізерно мала, отже доводиться обчислювати величезну кількість гешів цілеспрямовано. «Монетний двір» (брокер) вкладає істотні обчислювальні ресурси для генерування великої кількості таких k-колізій і таким чином *карбує* партію монет. Важливо, що MicroMint сконструйовано так, що масове виробництво монет стає вигіднішим: після того, як виконано великий обсяг геш-обчислень і знайдено перші кілька колізій, знаходження наступних колізій стає легшим (або принаймні дешевшим у середньому). Іншими словами, «генерувати багато монет значно дешевше в перерахунку на одну монету, ніж генерувати небагато монет», тобто спостерігається ефект економії на масштабі. Ця властивість має комбінаторний характер:

після достатньої кількості гешів імовірність появи нових колізій зростає, і набір монет після деякого «розігріву» може додавати нові монети з меншими додатковими витратами. Параметр k (наприклад, $k = 5$ чи $k = 10$) обирається так, щоб карбування монет було відносно легким для брокера (із потужними обчислювальними ресурсами), але надто дорогим для пересічного користувача чи зловмисника, який захотів би виготовляти монети у великому обсязі.

2. **Розповсюдження монет.** Брокер продає згенеровані монети користувачам (обмінюючи їх на звичайні гроші). Кожна монета має ідентифікатор (значення гешу Y або серійний номер), а також, як правило, обмежений строк дії чи приналежність до певної партії. Брокер веде облік, які монети були видані яким користувачам, але при цьому самі монети не потребують цифрового підпису на кожній одиниці: «вартість» монети закладена у властивості геш-колізії. Завдяки цьому монети дуже компактні й швидко перевіряються (для валідації достатньо лише гешування, без перевірки підписів).
3. **Платіж і перевірка.** Щоб здійснити мікроплатіж, користувач просто «пред'являє» монету (дані колізії геш-функції) продавцеві. Продавець дуже швидко перевіряє справжність монети: він пересвідчується, що дійсно існують k різних вхідних значень, які дають однаковий геш Y, і що Y належить допустимому діапазону ідентифікаторів монет поточної партії. Оскільки перевірка полягає лише в гешуванні кількох значень і порівнянні результатів, вона здійснюється майже миттєво. У режимі, де система дозволяє офлайн-прийом, продавцеві не обов'язково негайно під'єднуватися до брокера для кожного платежу — достатньо періодично оновлювати список анульованих монет або вже під час погашення.
4. **Погашення та запобігання повторному використанню (double-spending).** З часом продавець накопичує монети й пред'являє їх брокеру для обміну на реальні гроші. Ключовою проблемою безпеки є запобігання або виявлення подвійної витрати, коли той самий номінал монети кілька разів використовується різними продавцями. У MicroMint монети не прив'язуються до конкретних осіб чи транзакцій, тож користувач теоретично може скопіювати ту саму монету. Захист переважно процедурний: брокер у процесі погашення бачить, якщо один і той самий ідентифікатор монети надходить від різних продавців. Оскільки брокер відстежує, кому було продано яку партію монет, повторне пред'явлення одного й того самого ID означає, що користувач подвійно витратив монету. У такому випадку брокер може заблокувати цього користувача (або притягнути до відповідальності, якщо його особу можна встановити). Очікується, що через малу вартість кожної монети стимул до шахрайства невисокий, особливо якщо за спробу обману користувач втрачає весь свій рахунок. Крім того, користувачеві дуже важко «карбувати» власні монети: навіть одинична дійсна монета вимагає розв'язання задачі про k-кратну колізію, яка спеціально обрана достатньо складною, якщо не мати ресурсів брокера. Рівест і Шамір зазначали, що дрібні фальсифікації не становлять серйозної проблеми: вартість обчислень для підробки хоча б однієї монети має перевищувати її номінал, якщо тільки зловмисник не інвестує у ресурси рівня монетного набору. Масове фальшування потребувало б порівнянних з брокером обчислювальних потужностей, а поява великої кількості підозрілих монет була б швидко виявлена при погашенні. Таким чином, метою безпеки в MicroMint є радше економічне стримування, а не абсолютна криптографічна неможливість підробки: схема припускає, що атакувальники поводяться економічно раціонально.

У центрі MicroMint лежить одностороння геш-функція та складність побудови багатократних колізій. На відміну від PayWord, у типовому використанні тут зовсім немає операцій із відкритим ключем. Завдяки цьому транзакції стають надзвичайно швидкими

й «легкими», що ідеально підходить для мікроплатежів. Недолік полягає в тому, що MicroMint не надає таких же сильних криптографічних гарантій, як, наприклад, системи з підписаними монетами (на кшталт електронної готівки Чаума). Монети не прив'язані до конкретних особистостей, що певною мірою забезпечує анонімність, подібну до готівки, але при цьому брокер може вести журнал відповідності між користувачами та виданими їм монетами. Крім того, монети MicroMint зазвичай мають строк дії або «обертаються» (оновлюються партіями), змушуючи користувачів періодично замінювати їх, щоб зменшити ризики довгострокового накопичення та потенційного фальшування. Автори схеми зазначали, що підробку можна виявити за характерними патернами при погашенні, а будь-яких продавців чи користувачів, викритих у шахрайстві, можна виключити із системи. Тим часом чесні користувачі й продавці користуються практично миттєвими платежами: досить просто передати монету.

6.3 Головні відмінності систем

Схема PayWord використовує геш-ланцюги та один цифровий підпис, щоб підтримувати повторювані платежі одному продавцеві, мінімізуючи витрати на один платіж, але зберігаючи криптографічні гарантії (за умови довіри до брокера). Натомість MicroMint застосовує карбування монет на основі геш-колізій і створює токеноподібні монети, повністю усуваючи пер-платіжні операції з відкритим ключем і залишаючи лише гешування, за рахунок дещо слабшої моделі безпеки. Обидві схеми свідомо обирають примітиви, які відповідають їхнім цілям: PayWord спирається на швидкі геш-функції й одноразовий підпис для амортизації вартості, тоді як MicroMint використовує виключно геші й комбінаторну складність колізій. Ці конструкції вплинули на подальші платіжні системи та дослідження, демонструючи, як зміна балансу між криптографічною «силою» та ефективністю відкриває нові можливості для мікроплатежів, які були б непрактичними в рамках стандартних платіжних протоколів.

7 Протоколи електронних грошей: цифрова готівка Чаума (eCash)

На відміну від рахункових платіжних систем, таких як PayPal чи WebMoney, протоколи електронних грошей (e-money) часто мають на увазі схеми *цифрової готівки* — системи, у яких цифрові токени безпосередньо представляють гроші, і користувач може володіти ними та витрачати їх аналогічно до фізичної готівки. Визначальною рисою багатьох таких протоколів є застосування криптографії для забезпечення анонімності платника при одночасному захисті від подвійної витрати. Найвідомішим раннім прикладом є схема eCash Чаума (Chaumian eCash), розроблена Девідом Чаумом у 1980-х роках.

Цифрова готівка Чаума (eCash), реалізована компанією DigiCash у 1990-х роках, базувалася на проривній криптографічній техніці, відомій як сліпий підпис (blind signature). В eCash користувач «знімає» цифрові монети з банку, по суті змушуючи банк підписати випадковий серійний номер монети, не маючи змоги його побачити (значення «засліплюється»). Протокол схематично працює так:

1. **Зняття коштів за допомогою сліпих підписів.** Програмне забезпечення користувача генерує набір випадкових серійних номерів майбутніх монет і «засліплює» їх, застосовуючи спеціальний множник (blinding factor) — це математичне перетворення, яке приховує значення номера. Користувач надсилає засліплені серійні номери до банку разом із запитом на зняття, скажімо, N цифрових монет (кожна на фіксовану

суму, наприклад, 1 долар). Банк списує з рахунку користувача N доларів і за допомогою свого закритого ключа підписує засліплені значення, фактично створюючи підписи, які засвідчують згод у банку випустити ці монети, але без знання їх справжніх (незасліплених) серійних номерів. Підписані засліплені монети повертаються користувачу, який роззасліплює їх: завдяки властивостям сліпих підписів (наприклад, у схемі RSA) після зняття засліплення підпис $Sign(Blind(x))$ перетворюється на дійсний підпис $Sign(x)$ на початковому значенні x. У результаті користувач має N цифрових монет, кожна з яких складається із серійного номера та підпису банку над цим номером, причому банк не знає, який саме підпис відповідає якому випадковому числу (гарантується *розв'язність* дій банку та поточних монет).

2. **Платіж (витрата монети).** Щоб витратити монету, користувач надсилає продавцеві серійний номер монети та підпис банку. Продавець може негайно перевірити цифровий підпис банку, використовуючи його відкритий ключ, і таким чином переконатися, що монета дійсно випущена банком і не була змінена. Оскільки підпис накладено на сам серійний номер, будь-яка спроба змінити номінал чи ідентифікатор монети зробить підпис недійсним. Далі продавець надсилає монету до банку (в онлайн-сценарії) або тимчасово зберігає її для пакетної обробки (в офлайн-сценарії), щоб у підсумку погасити її й отримати реальні гроші.
3. **Запобігання подвійній витраті.** В онлайн-версії eCash продавець, отримавши монету, відразу звертається до серверів банку, щоб перевірити, чи не була ця монета вже витрачена (банк веде базу даних серійних номерів «спалених» монет). Якщо монета «нова», банк позначає її як витрачену й підтверджує транзакцію, забезпечуючи захист від подвійної витрати ціною необхідності мережевого доступу для кожного платежу. В офлайн-системі eCash (Чаум і Наор запропонували першу таку схему в 1990 році) продавець не звертається до банку негайно. Щоб стримати подвійну витрату, в монету вбудовується додаткова криптографічна інформація так, що якщо користувач витратить одну й ту саму монету двічі в різних продавців, то під час подальшого клірингу (звірки) банку двох транзакцій з одним серійним номером з математичних співвідношень буде розкрито особу користувача. Зазвичай це досягається шляхом кодування деяких секретів, пов'язаних із користувачем, у такий спосіб, що при одному коректному платежі вони не розкриваються, а при подвійній витраті — стають доступними. Таким чином, користувач зберігає анонімність у разі чесної поведінки, але втрачає її, якщо намагається шахраювати.

Основним криптографічним примітивом у схемі eCash Чаума є сліпі підписи RSA (або подібні алгоритми). RSA забезпечує цифрові підписи, завдяки яким тільки банк може випускати дійсні монети: підпис банку є криптографічною гарантією того, що монета має певну грошову вартість. Властивість «засліплення» гарантує конфіденційність: банк підписує значення, не знаючи його конкретного значення, тож не може пов'язати факт видачі монети з моментом її витрати. Це надає eCash важливу властивість — анонімні транзакції: банк (та сторонні спостерігачі) не може легко зіставити конкретний акт зняття коштів із подальшим платежем однією з монет. Лише в разі подвійної витрати протокол навмисно руйнує анонімність користувача як засіб криптографічного стримування шахрайства.

З погляду безпеки, eCash Чаума є криптографічно дуже сильною схемою: за умови стійкості RSA користувач не може підробити підпис банку на монеті, тобто не здатен створити гроші «з повітря». За відсутності подвійної витрати транзакції не розкривають особу платника, забезпечуючи справжню готівкову анонімність. Завдяки строгій криптографії ризик непомітного фальшування монет вкрай малий — набагато нижчий, ніж, скажімо, у MicroMint, що базується лише на складності геш-колізій. Водночас така висока безпека має свою ціну — обчислювальну складність та конструктивну складність. Для кожної

монети банк має сформувати підпис RSA, а продавець — перевірити його; для великої кількості дрібних платежів це було суттєвим навантаженням для програмного та апаратного забезпечення 1990-х років. Додатково система потребує підтримки бази серійних номерів витрачених монет (для онлайн-режиму) або виконання складних криптографічних процедур для розкриття особи при виявленні подвійної витрати (в офлайн-режимі).

Підсумовуючи, протоколи електронних грошей на зразок eCash Чаума ставлять у центр приватність і безпеку, спираючись на інтенсивне застосування криптографії — насамперед сліпих цифрових підписів, що дають змогу користувачеві витрачати «монети» анонімно. Вибір криптографічних примітивів (підписи RSA, сліпі підписні протоколи, односторонні функції для забезпечення анонімності) зумовлений прагненням відтворити властивості готівки (анонімність, відсутність централізованого відстеження кожної витрати) при одночасному захисті від шахрайства. Ці протоколи досягають основних цілей безпеки (невідтворюваність, невідстежуваність, захист від підробки) ціною значних обчислювальних витрат та складності реалізації.

8 Висновки

Сучасні електронні платіжні системи поєднують різноманітні криптографічні примітиви та протоколи, щоб задовольнити вимоги одночасно до безпеки й зручності використання. Такі системи, як PayPal, спираються на стандартні протоколи (TLS, HTTPS) та централізовану модель довіри, доповнюючи їх додатковими криптографічними механізмами (шифровані платіжні кнопки, підписані вебхуки) для посилення захисту. WebMoney обирає протилежний підхід — децентралізує керування обліковими даними, передаючи користувачам контроль над особистими криптографічними ключами; це забезпечує сильну автентифікацію та цілісність транзакцій за допомогою цифрових підписів, але водночас покладає відповідальність за зберігання ключів на самих користувачів. Спеціалізовані схеми мікроплатежів, такі як PayWord і MicroMint, демонструють, як модифікація «сили» криптографічного захисту (наприклад, використання односторонніх геш-ланцюгів або геш-колізій замість підписів для кожної транзакції) може зробити економічно доцільними дуже малі платежі. У свою чергу, протоколи електронної готівки, на кшталт eCash Чаума, показують, що за допомогою криптографії (сліпі підписи) можна досягти властивостей, подібних до готівки, — насамперед анонімності платника — й створити цифрові «монети», що функціонують аналогічно фізичним.

Кожна система та протокол свідомо обирають відповідні криптографічні примітиви, балансуючи між безпекою, продуктивністю та функціональністю. Наприклад, і PayPal, і WebMoney забезпечують шифрування й автентифікацію транзакцій, однак перша централізує управління для максимального спрощення користувацького досвіду, тоді як друга розподіляє контроль задля посилення автономності користувачів. Схеми мікроплатежів жертвують частиною «жорстких» криптографічних гарантій (допускаючи мінімальний ризик шахрайства або покладаючись на довіру до брокера), щоб радикально знизити вартість обробки одного платежу завдяки використанню геш-функцій. Електронна готівка, навпаки, максимізує безпеку та приватність, застосовуючи складні математичні конструкції, які зменшують потребу в централізованому контролі кожної транзакції, але ціною суттєвої обчислювальної та реалізаційної складності.

Критично важливо, що кожна така система має обґрунтовувати свій криптографічний дизайн тим рівнем безпеки, який він фактично надає. Наприклад, використання WebMoney цифрових підписів RSA та персональних сертифікатів виправдане високим рівнем упевненості в тому, що лише законний власник ключа може ініціювати переказ. Геш-ланцюг у PayWord виправданий практичною неможливістю прегеш-атак у відповідному контексті,

а один цифровий підпис поширюється на всю серію мікроплатежів. У MicroMint вартість монет обґрунтовується економічно: для зловмисника витрати на підробку монети мають перевищувати потенційний зиск. Сліпі підписи в eCash виправдані тим, що забезпечують одночасно сильну анонімність і нездатність підробити монету, спираючись на десятиліття розвитку теорії криптографії.

Отже, безпечний дизайн цифрової платіжної системи є багатофакторним компромісом, але в його основі завжди лежить використання криптографічних примітивів (шифрування, гешування, цифрові підписи, іноді нульові докази з нульовим розголошенням), підібраних таким чином, щоб відповідати вимогам до довіри, швидкодії та приватності. Розглянуті протоколи — від SET до PayPal/WebMoney, PayWord/MicroMint та eCash — демонструють різні точки цього простору рішень, але всі вони спираються на спільний фундамент криптографії, який дає змогу здійснювати електронні платежі надійно й безпечно в потенційно недовірених мережах.