Метод шифрованої комунікації у стратегічних взаємодіях

Виконав: студент МП ІПЗ-2 Михайленко О. І.
Науковий керівник: доцент, кандидат наук Гороховський С. С.

Постановка задачі

Створення імплементації протоколу наскрізного шифрування для безпечної комунікації в умовах недовіри до сервера. Має компілюватися у WebAssembly для використання у браузері. Демонстрація використання на клієнті.

Актуальність дослідження

- Зростає використання протоколів наскрізного шифрування у всьому світі, проте в першу чергу для текстової комунікації (месенджери, емейл)
- Частішають кібератаки, що можуть скомпрометувати централізовані сервери, через які відбувається шифрована комунікація
- Відсутність WebAssembly-compatible протоколів наскрізного шифрування на Rust

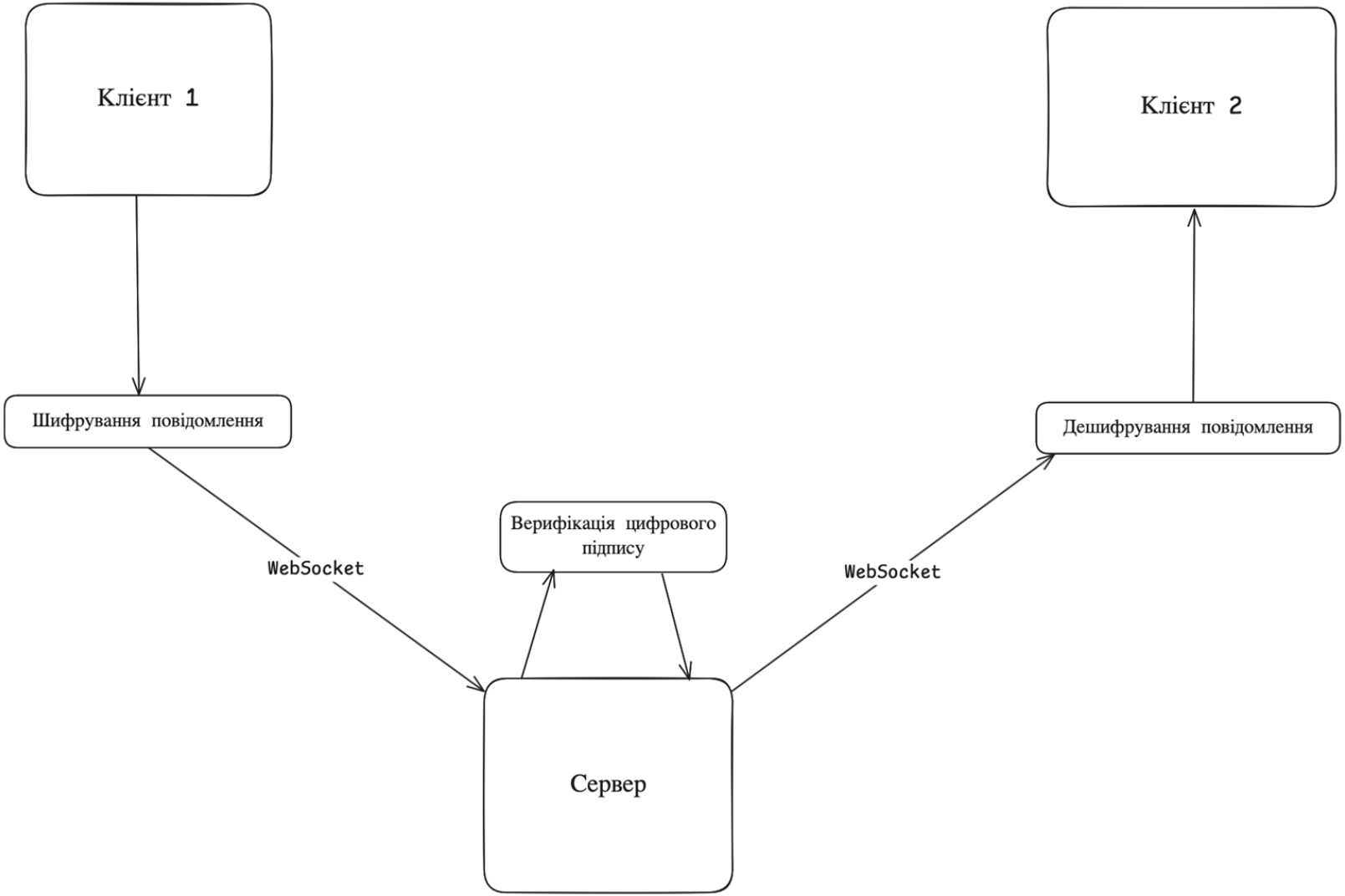
Завдання

- Створити власну реалізацію протоколу Double Ratchet (Perrin et al., 2016), що компілюється у WebAssembly та може використовуватися у браузерних середовищах
- Розробити простий сервер для створення сесій між користувачами з відкритим вихідним кодом
- Застосувати протокол на рівні застосунку, продемонструвати мультиплеєр шашок з шифрованою комунікацією

Архітектура

Архітектура проєкту складається з:

- Імплементації протоколу Double Ratchet без шифрування хедерів мовою Rust
- Імплементації гри у шашки мовою Rust
- Пакетів-обгорток Javascript над протоколом шифрування та шашками
- Сесійного сервера для мультиплеєру
- Веб-застосунку, що реалізує клієнтський мультиплеєр шашок


Клієнт 1
Клієнт 2
Шифрування повідомлення
Дешифрування повідомлення
WebSocket
Верифікація цифрового підпису
WebSocket
Сервер

Реалізація

- Мова програмування Rust (бібліотеки aes-gcm-siv, hkdf, hmac, sha2, x25519-dalek, та ін.) для створення реалізації протоколу Double Ratchet
- Rust проєкт chemkers (Zvirbulis, 2023), що імплементує шашки з можливістю компіляції у WebAssembly
- Rust для створення сесійного серверу
- Typescript для клієнтського застосунку

Ефемерні сесії

- Сервер зберігає дані в оперативній пам’яті
- Пари ключів з клієнта ніколи не перевикористовуються
- Перевага: відпадає необхідність використання безпечного сховища для пар ключів
- Недолік: неможливість перепід’єднання до сесії у випадку втрати зв’язку

Client-friendly API

```rust
#[wasm_bindgen]
pub fn w_init_ratchet_sender(dh_r: &[u8], shared_secret: &[u8]) -> JsValue {}

#[wasm_bindgen]
pub fn w_init_ratchet_receiver(dh_s: &[u8], shared_secret: &[u8]) -> JsValue {}

#[wasm_bindgen(getter_with_clone)]
pub struct HeaderCiphertext(pub Header, pub Vec<u8>, pub JsValue);

#[wasm_bindgen]
pub fn w_ratchet_encrypt(data: JsValue, plaintext: &[u8]) -> HeaderCiphertext {}

#[wasm_bindgen(getter_with_clone)]
pub struct Plaintext(pub Vec<u8>, pub JsValue);

#[wasm_bindgen]
pub fn w_ratchet_decrypt(data: JsValue, header: Header, ciphertext: &[u8]) -> Plaintext {}
```

JSтипи

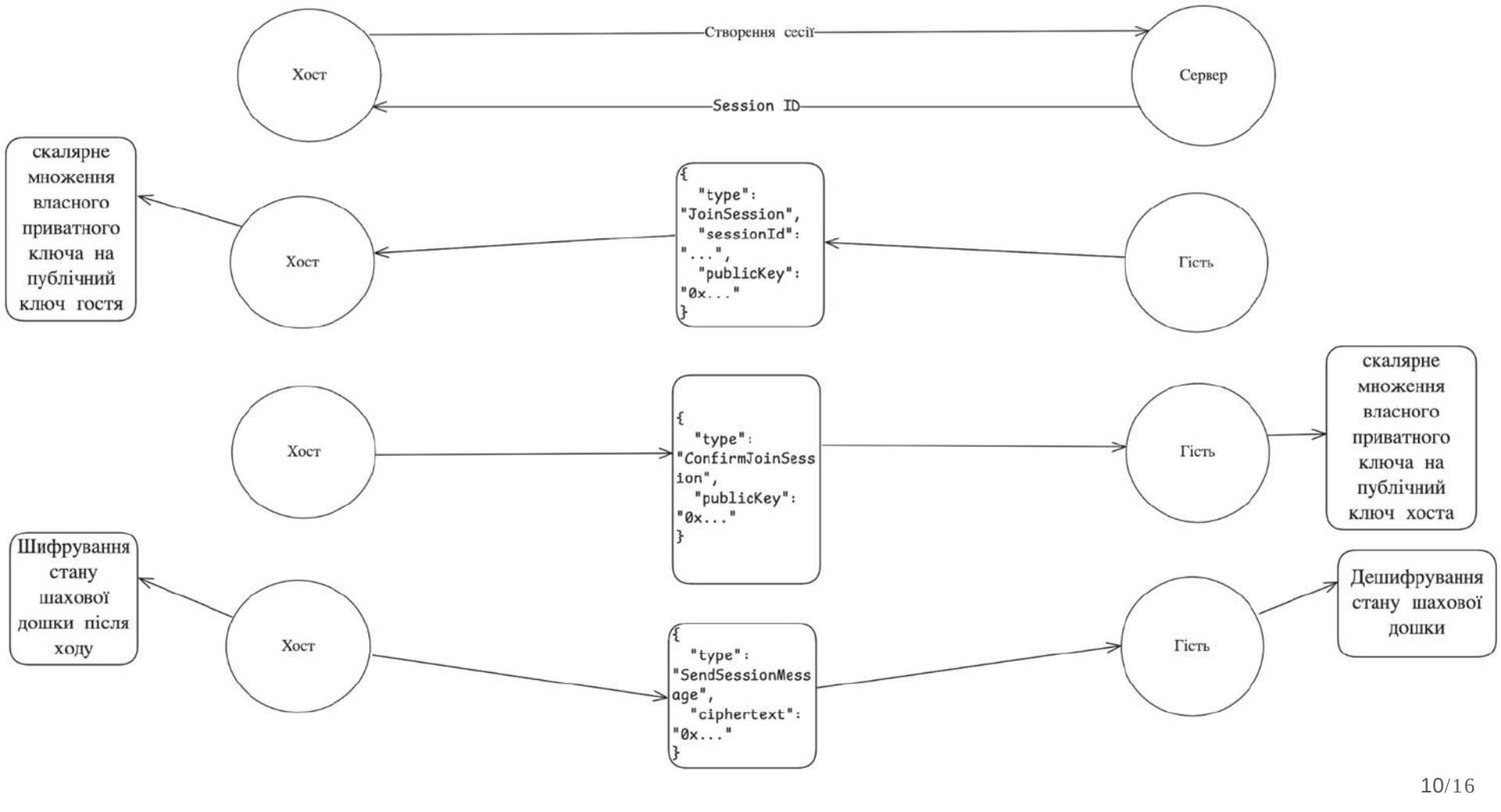
Rust типи

```rust
pub fn init_ratchet_sender(dh_r: PublicKey, shared_secret: [u8; 32]) -> Self {}

pub fn init_ratchet_receiver(dh_s: StaticSecret, shared_secret: [u8; 32]) -> Self {}

pub fn ratchet_encrypt(&mut self, plaintext: &[u8]) -> (Header, Vec<u8>) {}

pub fn ratchet_decrypt(
    &mut self,
    header: &Header,
    ciphertext: &[u8],
) -> anyhow::Result<Vec<u8>> {}
```


Хост
Створення сесії
Сервер
Session ID
скалярне множення власного приватного ключа на публічний ключ гостя
Хост
{ "type": "JoinSession", "sessionId": "...", "publicKey": "0x..." }
Гість
Хост
{ "type": "ConfirmJoinSession", "publicKey": "0x..." }
Гість
скалярне множення власного приватного ключа на публічний ключ хоста
Шифрування стану шахової дошки після ходу
Хост
{ "type": "SendSessionMessage", "ciphertext": "0x..." }
Гість
Дешифрування стану шахової дошки

Приклад зашифрованого повідомлення


```
{
  "sid": 15245,
  "cmd": {
    "type": "SendSessionMessage",
    "args": {
      "public": [
        "..."
      ],
      "header": {
        "sender": [
          "..."
        ],
        "previous_sending_chain_length": 0,
        "num_sent": 0
      },
      "ciphertext": "IdF7KAKukFOSOBZMh39HvfYQDnk+RD17cxURbXk0vpwIj3jhIeGj7DdEo8Keowi/UriVrJ/BHAJ5NXFoNW7F2WDs1T1b2bcBDLmMoL0o/6Vds9Wh0QpD24bOv1OaBDqTZ4FWrFMkA6Sn0LIenCmKvuOONP6xMO24V
YnXM+kxySKOZnV9qZyZVcTh5wT2G9//8d1X+0bsXb9EkXJQrrSgJRX1skSA8TeQ2DimnBxJN5PRH6Yghsh9zVd1mUfs8sA7NF6lTwjg6wndFign3XpBJNY/bKQQlGSqaXzMW6gbNbFusFYhtTU9xvyKJviKuOk2bV2zAnzuQyRY+S68Mh0W4vj
INQ1+rHbnFV1UbV9cpsAsIG2VUz7+ACBs2JkvZeOoPufuMDavd43oWQ9cf1AkVMzMyAhHbCNxQppjGzuZ2p3ZuwAdbHLw4Oaki8BdFinB1VdHkTJtt95ZbL9/mqAE5lXjXld2Hhkd/bwRVMBjfaUxsC7EU1GhAo1yXgvCwH+3uj3ZRQVP2Q+vp
n/07JLZs/J363mouSxvzAIPPlvUrUNYVi7BWWCdrWdPztmGtfxU11ihpvw1cotbvPGbtgVAUU3UCpsmyj/DSIuuShkEBvBBntI7GrZ0Ffqj5rTbJvG321XxRsRQyVNULq0PEtuUGVKVkepRWe83hfTHd7GWsUp+dwPl4WepRNW2pQMc/zBi1nG
1LfIpUoK4VXIj9oTHgmRLRqbyIiRXZFIv3e0pcbyrpWtZmOSQ2gHndwXMBseO+mTAzsn+bHl8ND5hJfql0k9l4nbUapdwnLiPemHtFmyhtLcwb99Ro7tbyH8o32Hm6YJM7/CqMOECtyC3QrdUjnJl3F/eYpcr/1e28X3ocQHS3ydYAyBdWq2JV
g=="
    }
  },
  "sig": "fd7f030272633dd14ab520c40b6b95f34d525f75fe63f1daea98fe2be752f8b8b85e64fb4b04b8b0bebc42639fd1c99051416d5ef48f254703b0e3fcbaa6a60a"
}
```


Create Session
Join Session


Create Session
Join Session

Інші застосування

- Симуляції військових дій з випадковими подіями (покрокові стратегії)
  - Проблема: генерація випадкових значень у WASM

Результати роботи

- Наявність реалізації DR з підтримкою WebAssembly, а отже створена можливість браузерного застосування
- Досліджений підхід ефемерного зберігання сесійних даних на клієнті та сервері
- Запропоновано додаткові client-friendly API для JS пакетів
- Створено сесійний сервер з відкритим вихідним кодом
- Розглянуто та інтегровано використання протоколу наскрізного шифрування у покрокову гру

Покращення

- Інтеграція TLSy протокол (WebSocket Secure замість WebSocket)
- Ефективніший формат серіалізації (Protocol Buffers, Flatbuffers)
- Безпековий аудит, котрий рекомендований усім криптографічним бібліотекам
- Підтримка бази даних для збереження сесій

Подальші дослідження

- Створення механізмів zero-knowledge proofs для запобігання атакам типу Man-In-The-Middle під час створення спільного секрету
- Оптимізація інтеграції з криптографічними модулями апаратної безпеки (HSM)
- Дослідження можливостей масштабування рішення (інші ігри, симуляції військових дій)