Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Кафедра автоматизації проектування

 енергетичних процесів і систем

**Звіт**

**з циклу лабораторних робіт**

**з дисципліни**

**„Безпека та приватність в Інтернеті”**

Лабораторна  робота № 1

Симетричне, асиметричне і гібридне шифрування інформації

Виконала: **Юрченко Богдана**

Студентка групи:   **ТВ-01 мп**

Перевірив викладач: **проф. Гаврилко Є. В.**

Київ - 2021

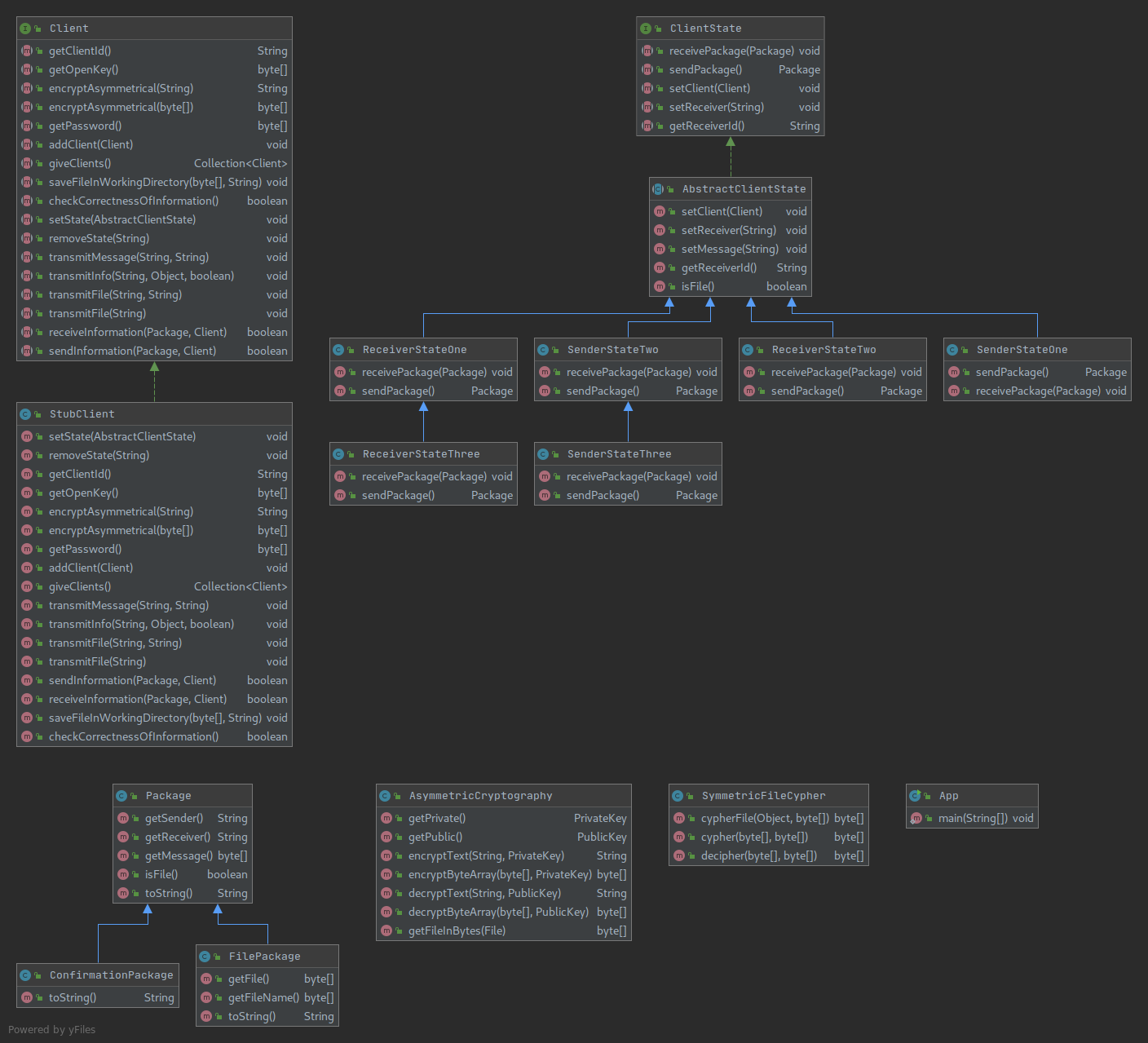
Опис виконаної роботи :

Була розроблена програма для емуляції передачі зашифрованих пакетів в мережі.

В системі є такі базові сутності як Клієнт, СтанКлієнта і Пакет.

Є різні реалізації СтануКлієнта для емуляції протоколу обміну даних (перша фаза, 2 фаза і опціонально 3 фаза у разі передачі файлу). Є також різні реалізації Пакету . Звичайний пакет - для передачі простого повідомлення - зашифрованого або ні. Пакет для передачі Підтвердження (про успішне отримання повідомлення одержувачем). Пакет для передачі файлу.

Усі зазначені класи та різні їх реалізації можна побачити на діаграмі класів :



Завдання :

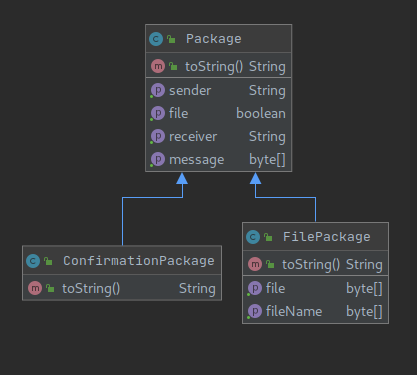
Створити програму, яка імітує процес передачі зашифрованих даних у мережі.

Створіть окремий клас Client, що представляє клієнта мережі. У кожного клієнта повинні бути наявними принаймні наступні компоненти:

1. Власна робоча директорія.
2. Відкритий асиметричний ключ шифрування.
3. Закритий асиметричний ключ шифрування.
4. Пароль для симетричного шифрування файлів.
5. Список клієнтів, підключених до поточного клієнту.
6. Методи для передачі інформації іншим клієнтам і отримання інформації від інших клієнтів.

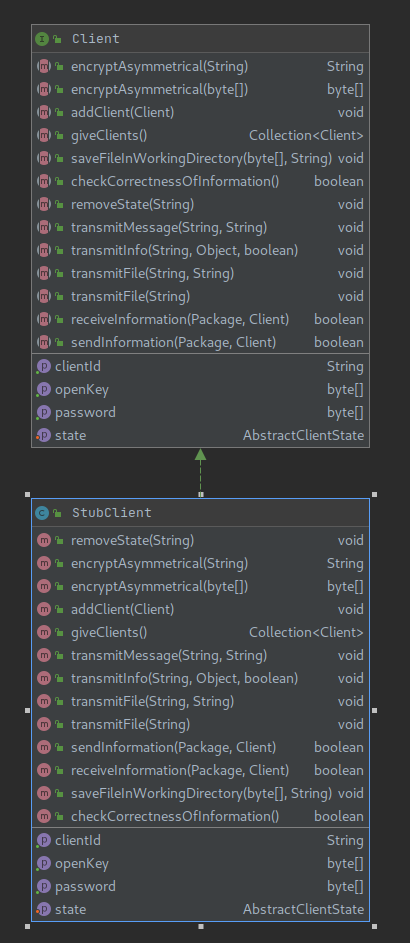
Інформацію, яку передаєте іншим клієнтам, інкапсулюйте в класі Package. Цей клас повинен містити інформацію про відправника, отримувача, і повідомлення або файл. В рамках цієї лабораторної роботи, можете створити два окремих поля – для повідомлення (паролю або звичайного повіпомлення) і файлу (може приймати null, в разі звичайного повідомлення).

Створені класи Клієнту та Пакету, їх методи і Поля можна побачити на діаграмі далі.



Код ініціалізації клієнту виглядає наступним чином :

1. ініціалізація об’єкту для асиметричного шифрування.
2. генерація пари ключів (відкритого і закритого).
3. генерація паролю для симетричного шифрування.
4. ініціалізація робочої директорії для клієнту із назвою “айді клієнту”
5. ініціалізація тестовго файлу txt із спеціально згенерованим текстом.

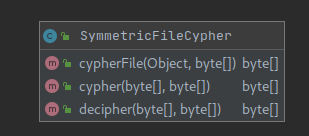


Завдання

Створіть окремий клас SymmetricFileCypher. Клас повинен містити методи для шифрування та дешифрування файлів за отриманим симетричним ключем. Алгоритм симетричного шифрування оберіть самостійно. Для самого процесу симетричного шифрування файлу, можете використовувати готові реалізації, наприклад із javax.crypto.\*.

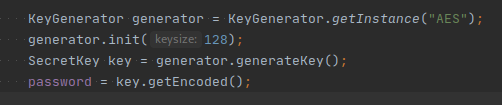
Опишіть у звіті, який алгоритм і реалізацію обрали.

Клас для симетричного шифрування має наступні методи



Для шифрування була використана бібліотека javax.crypto. Алгоритм для шифрування був обраний - “AES”.

Алгоритм AES (алгоритм Рійндаеля) - це симетричний алгоритм блочного шифрування, який приймає звичайний текст у блоки по 128 біт і перетворює їх у зашифрований текст за допомогою ключів 128, 192 і 256 бітів. Оскільки алгоритм AES вважається безпечним, він відповідає світовому стандарту.

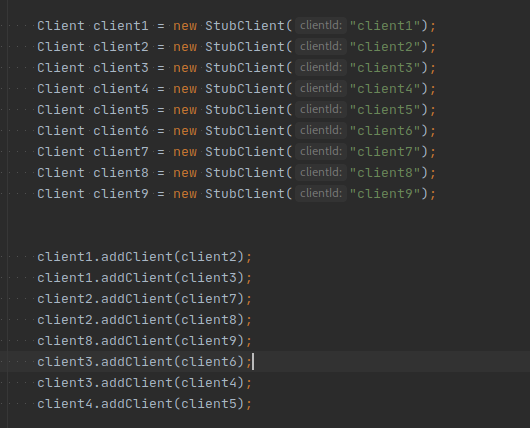


Довжина ключа була обрана 128 біт.

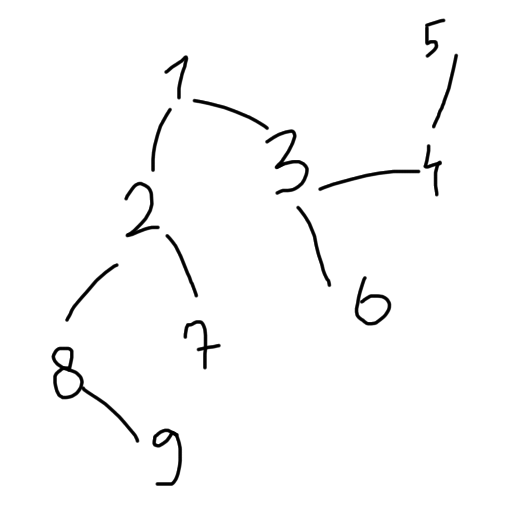
Завдання

У основній програмі, створіть мережу із клієнтів – декілька клієнтів, пов’язаних один з одним, щоб їх зв’язок можна було описати у вигляді простого графу (без петель і без циклів, утворених ребрами) і з більше ніж двома вершинами. Спосіб передачі пакетів від одного клієнта до іншого оберіть самостійно. Опишіть мережу і спосіб передачі пакетів між клієнтами у звіті до лабораторної роботи.

Було створено мережу із обєктів StubClient. Код створення мережі можна побачити на скріншоті нижче.

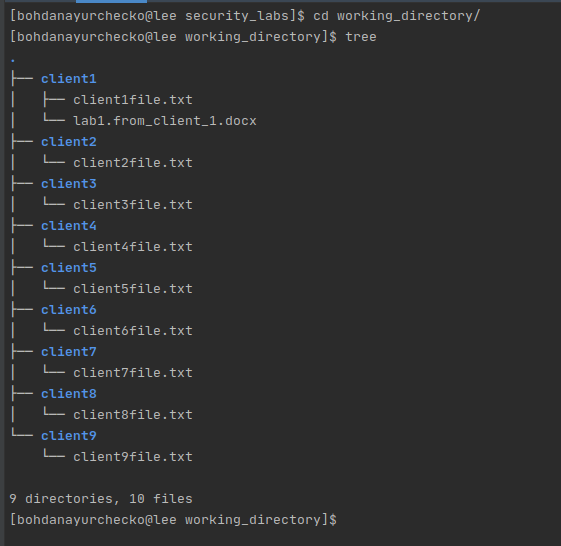


Мережа, створена кодом вижче вигляда наступним чином :



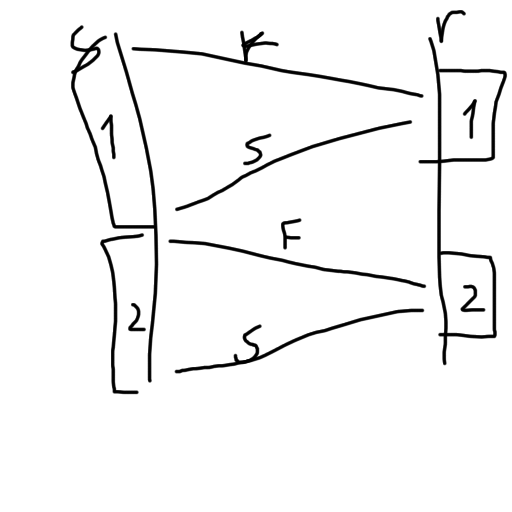
Ордазу після ініціалізації усіх клієнтів батьківська робоча директорія виглядає наступним чином :

Кожен клієнт має свою робочу директорію із своїм тестовим файлом. Клієнт 1 має в своїй директорії також файл із завданням для лабораторної роботи, який буде передаватися між клієнтами в лабораторній роботі.



Процес передачі пакетів між клієнтами проходить або в 4 або в 6 стадій. Відправник і отримувач мають або 2 або 3 стани у процесі передачі даних.

У разі передачі простого повідомлення передача даних виглядає наступним чином :

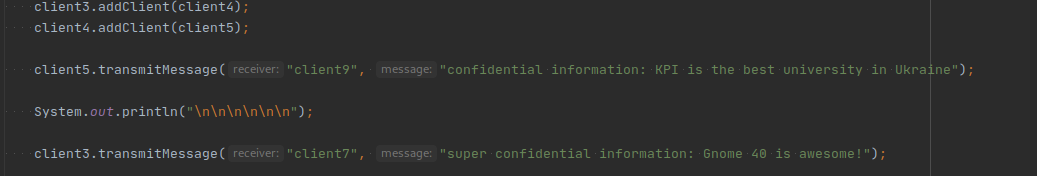


Зліва знаходиться S sender або відправник, справа знаходиться R receiver або отримувач. Кожен із них переходять між 2 станами. Детальніше про стани буде написано далі із прикладом.

Завдання

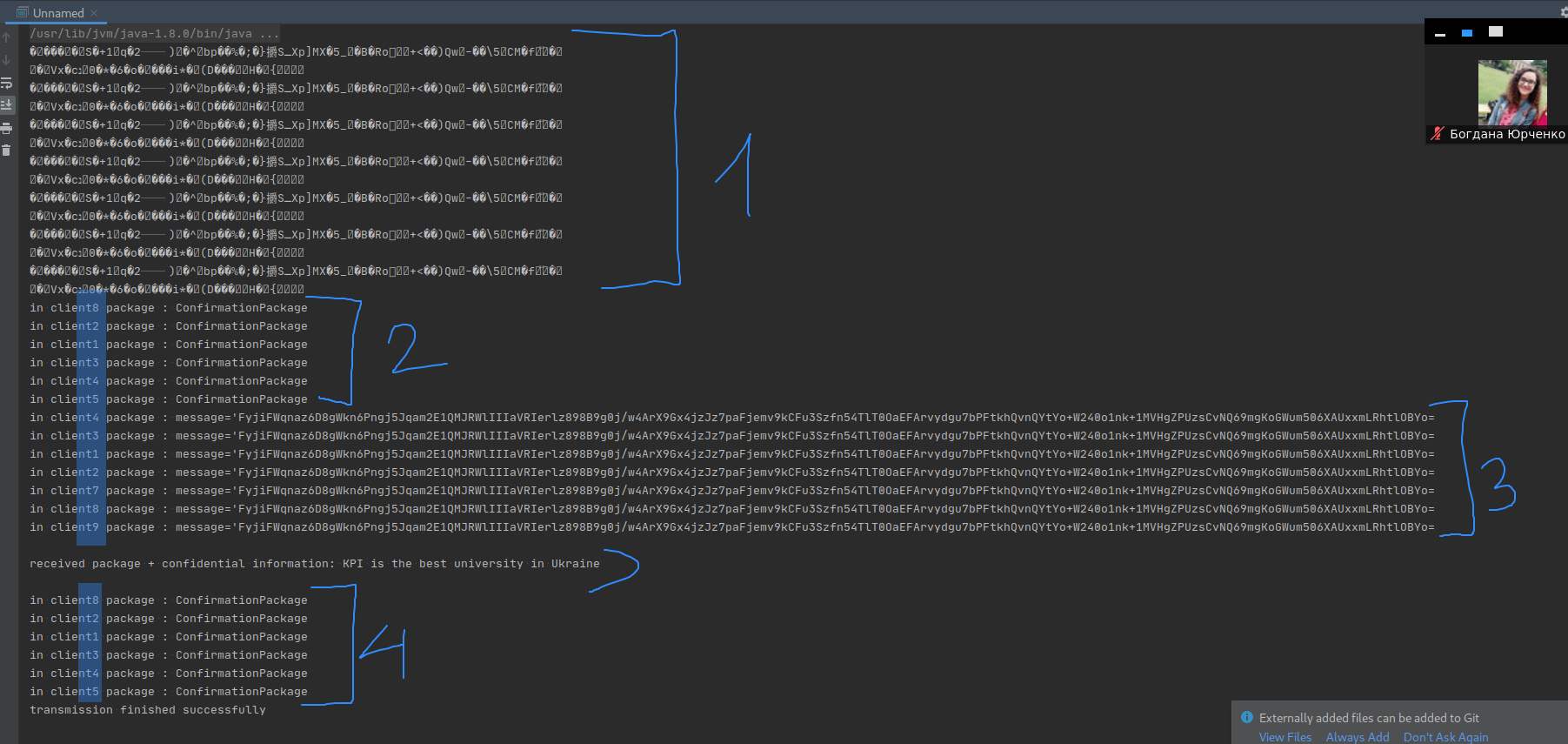
Надішліть декілька повідомлень між двома клієнтами, відстань яких один від одного на графі більше одиниці. Клієнти повинні розшифрувати отримані повідомлення своїм приватним ключем асиметричного шифрування і відобразити їх на екрані. Клієнти, які беруть участь у передачі, але не є відправниками або отримувачами, повинні відображати передаваєме повідомлення, але (очевидно) у зашифрованому вигляді.

Код, що надсилає повідомлення між 2 клієнтами мережі :



Перше повідомлення надсилає клієнт 5 клієнту 9. Із графу мережі видно, що між ними знаходяться клієнти 4, 3, 1, 2, 8 .

Результат виконання передачі повідомлень в консолі :



Із скріншоту видно, що передача проходить в 4 стадії

1. передача відкритого ключа шифрування клієнту 5 до клієнту 9. Ключ - це незашифрований масив байтів.
2. Передача клієнтом 9 клієнту 5 підтвердження про отримання відкритого ключа.
3. Передача повідомлення клієнтом 5 клієнту 9, зашифрованого закритим ключем клієнта 5. На скріншоті видно, що усі клієнти між двома клієнтами мали змогу прочитати повідомлення, але воно зашифроване. Клієнт 9 мав же відкритий ключ клієнту 5 і тому зміг розшифрувати повідомлення і вивести його на екран.
4. Відправлення пакету підтвердження повідомлення від клієнту 9 до клієнту 5.

Після цього з’єднання між клієнтами закривається.

Завдання

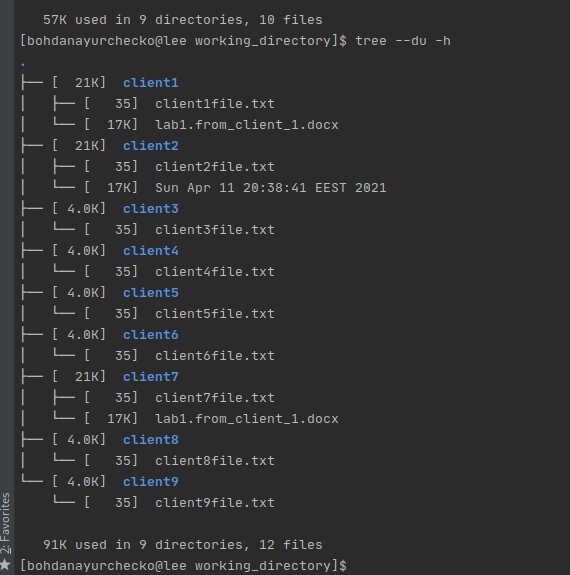
Передайте файл із завданням до лабораторної роботи (цей документ) між цими двома клієнтами. Файл повинен бути зашифрованим симетричним методом шифрування. Симетричний ключ передається у формі повідомлення, зашифрованого асиметричним ключем шифрування. Отримувач повинен розшифрувати файл у своїй директорії. Клієнти, які беруть участь у передачі, але не є відправниками або отримувачами, повинні зберегти файл, який передають, у себе в директорії у (очевидно) зашифрованому вигляді.

Код, що передає файл із між клієнтами 1 та 7 виглядає так :



Із схеми мережі вище видно, що між клієнтами 1 і 7 заходиться всього 1 клієнт номер 2.

Відразу після закінчення передачі робочі директорії усіх клієнтів виглядають так :



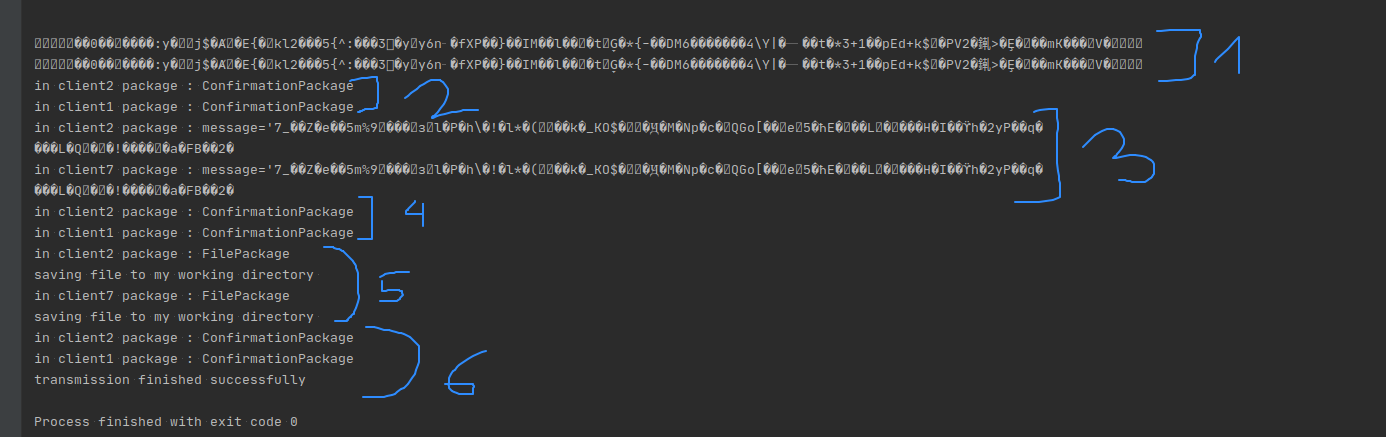
На скріншоті видно, що нові файли зявилися у директоріях клієнтів 7 і 2.

Клієнт 2 взяв учать у передачі даних, але оскільки він не мав переданого паролю для розшифрування файлу - він був збережений із датою передачі і без розширення. Для підтвердження того, що це саме переданий файл, але зашифрований, в дереві директорій було відображено розмір файлу. На скріншоті видно, що розмір файлу в директорії і 2 і 7 відповідає розміру файлу в директорії 1.

Більше детально процес передачі файлу відображений на скріншоті далі. Процес проходить в 6 стадій:

1. передача відкритого ключа від клієнта 1 до клієнта 7
2. пакет-підтвердження від клієнта 7 до клієнта 1
3. передача зашифрованого закритим ключем паролю від клієнта 1 до клієнта 7
4. пакет-підтвердження від клієнта 7 до клієнта 1 про отримання паролю
5. передача пакету із фалом між клієнтами . Клієнти зберігають зашифрований файл в своїй робочій директорії (клієнт 2)
6. пакет-підтвердження від клієнта 7 до клієнта 1 про отримання файлу

Після чого з’єднання між клієнтами закривається.

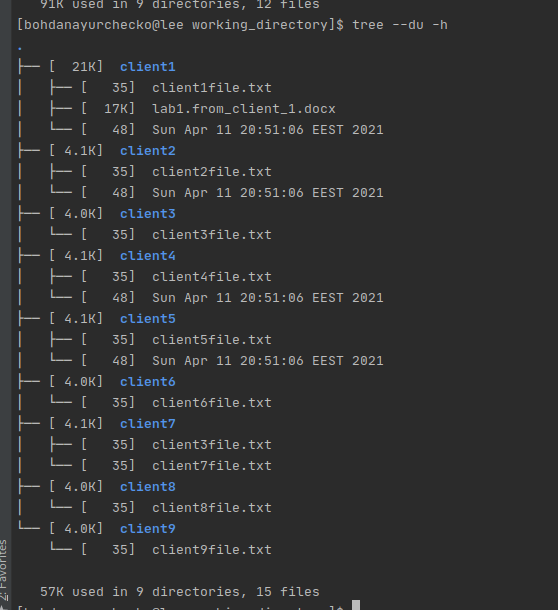


Через природу docx файлів неможливо показати наглядно, що файл зашифрований.

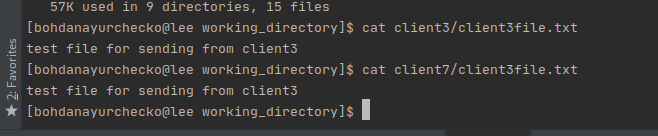
Простіше це зробити за допомогою предачі зегенерованого тектового файлу від клієнту 3 до клієнту 7.



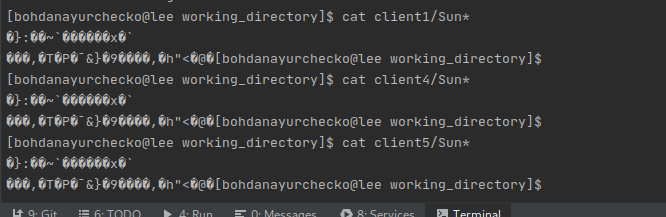
Дерево робочих директорій відразу після передачі виглядає наступним чином (в цій передачі брало участь більше клієнтів) :



Спроба вивести в термінал вміст файлу із директорій клієнтів 3 і 7 дає такий результат :

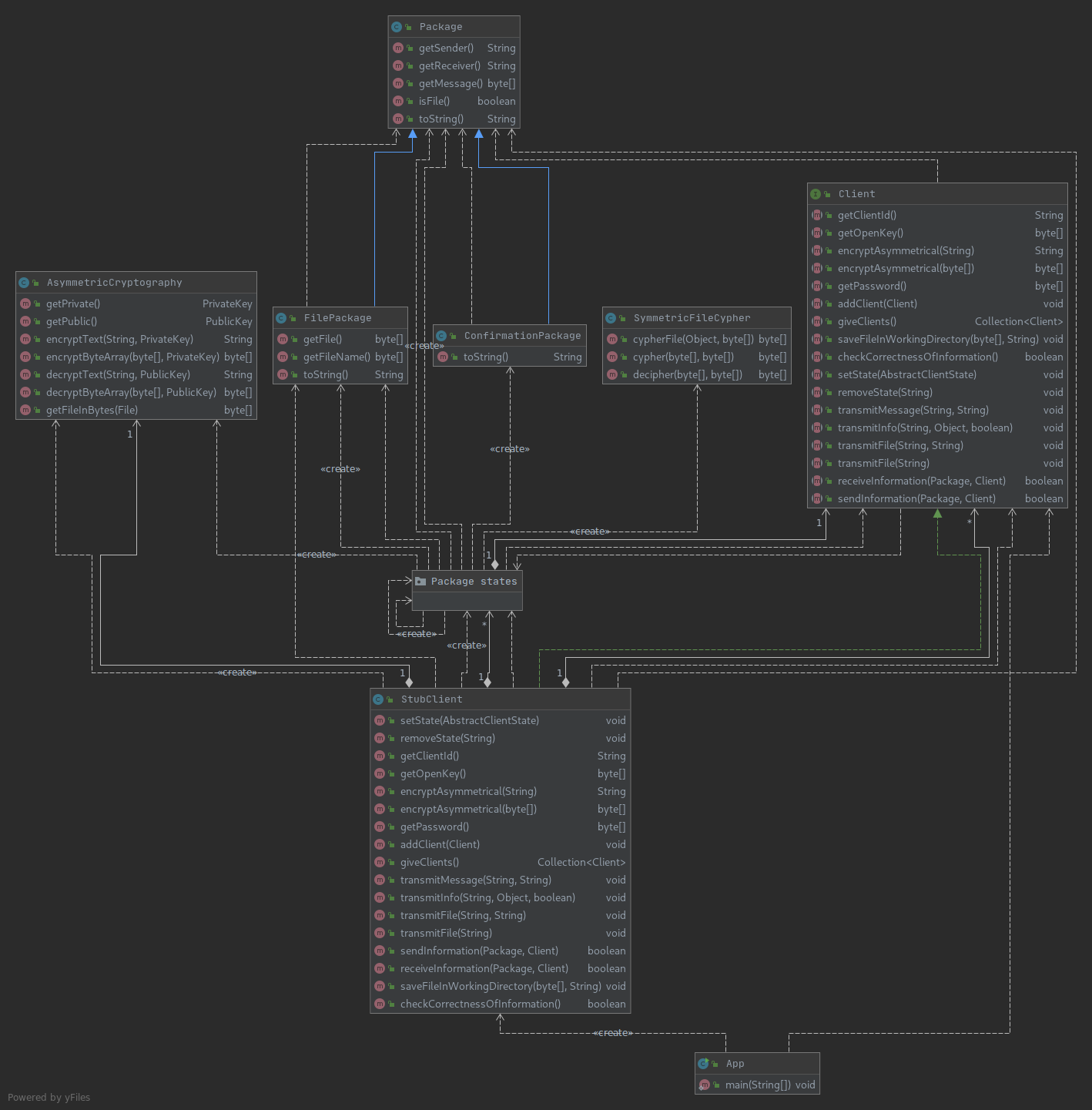


Спроба вивести вміст файлів із директорій проміжних клієнтів дає такі результати :



Текст зашифрований.

Діаграми створених класів програми із відобарженими звязаками асоціації і агрегації :



**Відповіді на контрольні запитання**

**1. В чому полягає принцип симетричного шифрування? Які у такого шифрування є переваги та недоліки.**

Шифрування симетричним ключем - це метод захищення даних за допомогою використання одного і того ж ключа для шифрування та дешифрування. Цей підхід відрізняється від асиметричного шифрування що використовує різні ключі для шифрування та дешифрування.

Використовуючи симертричне шифрування, ключ має зберігатися у таємниці.

Однією із переваг є те, що таке шифрування є надзвичайно безпечним.

Коли він використовує захищений алгоритм, симетричне шифрування ключів може бути надзвичайно безпечним. Однією з найбільш широко використовуваних систем симетричного шифрування ключів є розширений стандарт шифрування, призначений урядом США. Коли ви використовуєте його з найбезпечнішою довжиною ключа 256 біт, знадобиться близько мільярда років, щоб комп’ютер із 10 петафлопами вгадав ключ за допомогою атаки грубої сили. Оскільки станом на листопад 2012 року найшвидший комп’ютер у світі працює на 17 петафлопс, 256-бітний AES по суті незламний.

Також таке шифрування є відносно швидким.

Шифрувати та розшифровувати симетричні ключові дані порівняно просто, що дає дуже хороші показники читання та запису.

Мінусами є те, що для передачі даних треба якось передати ключ між машинами. Втрата ключа не допустима, адже він дозволяє не лише шифрувати а і розшифрувати повідомлення.

Таким чином таке шифрування показує себе найкраще для локального шифрування, коли не треба нікому нічого надсилати.

**2. В чому полягає принцип асиметричного шифрування? Які у такого шифрування є переваги та недоліки.**

Асиметричне шифрування включає два криптографічні ключі для забезпечення безпеки даних: відкритий та приватний ключ. Відкритий ключ доступний кожному, хто бажає надіслати повідомлення. Приватний ключ зберігається у захищеному місці власником відкритого ключа.

Відкритий ключ шифрує інформацію, що надсилається. Тоді як приватний ключ, який є у приймача, розшифровує його. В обох процес використовується один і той же самій алгоритм.

Цей принцип є відносно новим. Використання двох ключей для шифрування і розшифрування робить асиметричне шифрування складною технікою.

Таким чином, цей підхід являється дуже надійним з точки зору безпеки даних.

Перевагами також є :

1. можливість провести аутентифікацію повідомлень.

реалізація цифрового підпису

2. це вирішує проблему розподілу ключів для шифрування.

Кожен публікує свої відкриті ключі, тоді як приватні ключі тримаються в таємниці.

Мінусами є те, що таке шфирування на даний момент є більш повільним.

Шифрування відкритим ключем у цьому методі є повільним у порівнянні з симетричним шифруванням, що означає, що він не підходить для дешифрування великих повідомлень.

Також проблемою є те, що відкриті ключі не автентифіковані.

В основному, ніхто абсолютно не знає, що відкритий ключ належить указаній особі, а це означає, що користувачі повинні будуть перевірити, чи справді їх відкриті ключі належать їм.

**3. Перерахуйте декілька відомих вам алгоритмів симетричного і асиметричного шифрування.**

Алгоритми симетричного шифрування : RC4, AES, DES, 3DES, QUAD, Blowfish.

Алгоритми асиметричного шифрування : RSA, Diffie-Hellman, ECC, El Gamal, DSA.

**4. Опишіть роботу алгоритму Діффі-Геллмана.**

Алгоритм Діффі-Геллмана - це схема обміну ключами. Була розроблена в 1976 році Уітфілдом Диффі і Мартіном Геллманом. Даний алгоримт став першим практичним методом для отримання загального секретного ключа при спілкуванні через незахищений канал зв'язку. Через один рік після розробки даного методу було створено перший алгоритм асиметричного шифрування - RSA, який вирішив проблему передачі даних через незахищений кардинальний канал.

Алгоритм ключає наявність 2 абонентів. Обом абонентам відомі деякі два числа g і p, які не є секретними і можуть бути відомими також іншим зацікавленим особам. Для того, щоб створити невідомий більше нікому секретний ключ, абонент генерує більші випадкові числа: перший абонент - число a, другий абонент - число b. Після цього перший абонент вичисляє значення A = gamod p і надсилає його іношому абоненту , а інший абонент обчислює число B = gbmod p і передає першому. Метод також передбачає те , що зловмисник може отримати обидва ці значення, але не модифікує їх (тому що у нього немає можливості втручатися в процес передачі). На другому етапі першого абонента на основі відомого йому А і отриманого з мережі B вичисляється значення Bamod p = gabmod p, а другий абонент на основі b і отриманого A вичисляє значення Abmod p = gabmod p. Обидва абоненти отримали одне і те же число: K = gabmod p.

Тепер обоє клієнтів можуть використовувати отримане число як секретний ключ.

**5. В чому полягає принципова різниця між алгоритмами Діффі-Геллмана і RSA?**

RSA не вимагає, щоб обідві сторони мали копію одного і того ж секретного числа.