DOI 10.15276/imms.v12.no4.346
УДК 004.633.2

Informatics and Mathematical Methods in Simulation
Vol. 12 (2022), No. 4, pp. 346-351

РОЗРОБКА МУЛЬТИСЕРВЕРНОГО КОРПОРАТИВНОГО МЕСЕНДЖЕРА ЗІ СПЕЦІАЛЬНИМ ЗАХИСТОМ МІЖСЕРВЕРНОГО ТРАФІКУ

М.А. Лісовський, О.А. Стопакевич

Національний університет «Одеська Політехніка»,
просп. Шевченка, 1, Одеса, 65044, Україна; e-mail: mix313131@gmail.com

Запропонована реалізація месенджеру, яка передбачає, що для доступу до даних, пов'язаних з месенджером, користувачу необхідно зв'язатись з сервером локальної корпоративної мережі, в іншому випадку дані будуть знаходитись у шифрованому стані. Логіка з'єднання будується наступним чином: Клієнт підключається до Серверу, що знаходиться в межах Корпоративної Мережі. Після підтвердження версій Клієнта та Сервера, встановлюється зв'язок і Клієнт переходить у робочий режим - дані, збережені в середині Клієнту розшифровуються і Користувач отримує можливість працювати з цими даними. Коли Користувач відправляє повідомлення іншому Користувачу, Клієнт шифрує повідомлення та відправляє його на Сервер з вказанням Відправника та Одержувача. У випадку, коли Одержувач знаходиться в одній мережі з Відправником, Сервер перенаправляє повідомлення без додаткових дій. В іншому випадку - Сервер чекає на ще декілька повідомлень (від будь-яких Відправників), якщо встановлена кількість набирається - повідомлення починають пакуватись в одне велике, після чого шифруються ще раз та відправляються до іншого Серверу. Якщо після встановленого часу кількість не набирається - Сервер додає невистачаючу кількість Повідомлень у вигляді випадкового набору значень, які не матимуть Відправника та Отримувача (для того, щоб другий Сервер відкинув ці повідомлення як "шум").
Ключові слова. Мультісерверний месенджер. Захист міжсерверного трафіку. Шифрування.

В сьогоденні питання інформаційної безпеки особливо важливе, і воно розповсюджується на всі елементи інформаційної системи, у тому числі - засоби з'єднання між окремими філіями бізнесових офісів. Наразі, частіш за все для обміну інформацією використовується електронна пошта, яка має цілу низку вразливостей. Прикладами таких вразливостей можуть бути CVE-2015-0235, що є атакою на сховище електронної пошти з метою переповнення буферу і виконання коду, або CVE-2017-1274, що є атакою на рівні отримання пошти (POP3, IMAP), та ін. Всі ці атаки представляють собою велику загрозу безпеці та можуть призвести до витоку, пошкодження або знищення критичної інформації. Альтернативою використанню пошти є використання месенджерів на кшталт Telegram, WhatsApp, Viber, тощо. Але ці месенджери також становлять велику загрозу безпеці інформації. Прикладом є ймовірність втрати пристрою, на якому є активним обліковий запис месенджеру та зберігається чутлива інформація. У разі втрати, цей пристрій може потрапити у руки зловмисника і, хоча, власник аккаунту може завершити сеанс на цьому пристрої, це не дасть жодного результату, якщо пристрій не буде підключений до мережі Інтернет, бо тоді зловмисник матиме змогу скопіювати всі скачані на пристрій важливі файли, та подивитись всю збережену у хеші переписку, що є менш ніж бажаним. Однак, це не єдина загроза, притаманна месенджерам. Вони, як і будь-яке інше програмне забезпечення, мають якісь вразливості, як незначні, наприклад CVE-2021-41861, який описує баг, присутній версіям Telegram 7.5.0-7.8.0 на ОС Android, при якому фото з функцією самознищення не самовидалялось з пристрою, але відображалось як видалене для

користувачів; так і значні вразливості, як, наприклад, CVE-2019-12569, вразливість десктопної версії мессенджера Rakuten Viber до 10.7.0, при якій зловмисник міг виконати довільні команди, скориставшись небезпечними шляхами, які використовувались для URI програми. Завдяки цій вразливості зловмисник може, переконавши цільового користувача перейти за шкідливим посиланням, завантажити бібліотеки з директорії, на яку вказував URI, та отримати можливість виконувати довільні команди з привілеями цільового користувача.

Звісно, існують месенджери корпоративного призначення, але вони, загалом, працюють виключно у локальній мережі і не можуть використовуватись для з'єднання між віддаленими філіями. Саме тому існує необхідність у створенні месенджера, призначеного для безпечного з'єднання як всередині локальної корпоративної мережі, так і між кількома корпоративними мережами.

Запропонована реалізація такого месенджеру передбачає, що для доступу до даних, пов'язаних з месенджером, користувачу необхідно зв'язатись з сервером локальної корпоративної мережі, в іншому випадку дані будуть знаходитись у шифрованому стані. Логіка з'єднання будується наступним чином: Клієнт підключається до Серверу, що знаходиться в межах Корпоративної Мережі. Після підтвердження версій Клієнта та Сервера, встановлюється зв'язок і Клієнт переходить у робочий режим - дані, збережені в середині Клієнту розшифровуються і Користувач отримує можливість працювати з цими даними. Коли Користувач відправляє повідомлення іншому Користувачу, Клієнт шифрує повідомлення та відправляє його на Сервер з вказанням Відправника та Одержувача. У випадку, коли Одержувач знаходиться в одній мережі з Відправником, Сервер перенаправляє повідомлення без додаткових дій. В іншому випадку - Сервер чекає на ще декілька повідомлень (від будь-яких Відправників), якщо встановлена кількість набирається - повідомлення починають пакуватись в одне велике, після чого шифруються ще раз та відправляються до іншого Серверу. Якщо після встановленого часу кількість не набирається - Сервер додає невистачаючу кількість Повідмлень у вигляді випадкового набору значень, які не матимуть Відправника та Отримувача (для того, щоб другий Сервер відкинув ці повідомлення як "шум").

Розглянемо в деталях як саме влаштовані різні аспекти роботи месенджера. Почнемо з підключення Клієнту до Серверу. Існує декілька сценаріїв, які можуть виникнути під час підключення.

Перший сценарій - успішне підключення, як зображено на рис. 1.

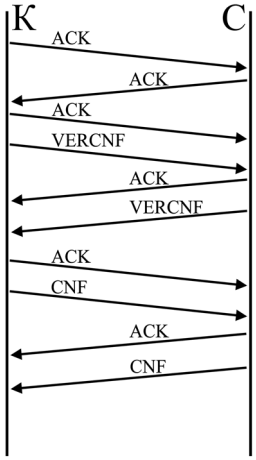

Рис 1. Процес з'єднання Клієнта та Сервера

При такому сценарії Клієнт та Сервер відправляют один одному свої версії у вигляді VERCNF (Version Confirm), та підтверджують вірність версій один

одного, відправляючи CNF (Confirmed) у вигляді відповіді, після чого виконують з'єднання. У цьому випадку наступним кроком для Клієнту буде розшифрування власного вмісту.

Другий та третій сценарії дуже схожі - це сценарії, у яких версія або Клієнту або Серверу виявилась недійсною. При такому сценарії сторона, що виявила недійсну версію відправляє CD замість CNF, що означає Connection Denied (У з'єднанні відмовлено). У випадку такої ситуації Користувач побачить повідомлення "З'єднання невдале" та буде необхідно повідомити про це до свого Системного Адміністратору.

Перед кожним повідомленням відправляється ACK (Acknowledge) від однієї з сторін, для того щоб визначити який з каналів є активним в поточній передачі.

Коли користувач надсилає повідомлення, Клієнт будує запит до сервера наступним чином: ID відправника, ID отримувача, зашифроване повідомлення (може включати до себе субструктуру текст+медіа-вкладення або одне з двох, в такому випадку поле тексту або поле медіа-вкладення буде пустим).

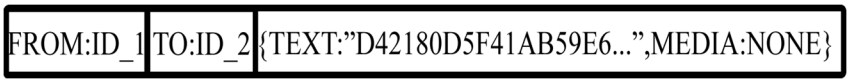

Рис 2. Структура повідомлення Клієнта до Серверу

Сам ID користувача прив'язується до відповідального Серверу, а задля спрощення орієнтації по ID, він може мати форму %server_name%_%username% (наприклад SCS1_TishchenkoG, де SCS1 - Science Company Server 1, TishchenkoG - Тіщенко Г.). Після отримання повідомлення, Сервер перевіряє чи знаходиться Одержувач всередині цієї локальної мережі чи ні, якщо ні - переходить до пакування повідомлення на другий Сервер, якщо Одержувач знаходиться в межах цієї локальної мережі, повідомлення просто пересилається до нього.

У випадку з прикріпленнями, вони посилаються у зашифрованій формі, а медіа мають при собі чек-суму, для перевірки автентичності медіа-вкладень, у випадку коли вони прикладаються до повідомлення, інакше замість медіа знаходиться None.

Коли повідомлення надходить до Серверу, він чекає n часу на надходження ще k повідомлень, після чого або генерує додаткові "мусорні" повідомлення, щоб довести загальну кількість до встановленої, або, якщо встановлена кількість була досягнута, починає процес стиснення повідомлень до одного потоку.

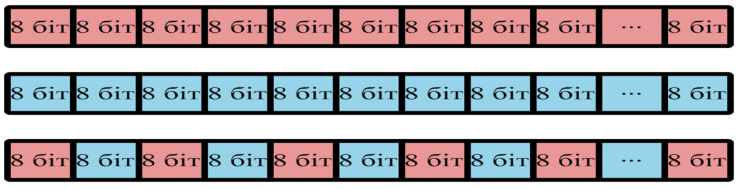

Рис 3. Процес стиснення повідомлень, де червоним кольором зображено повідомлення 1, а блакитним - повідомлення 2, повідомлення з кольорами, що чергуються - результуюче повідомлення.

На рис. 3 зображена узагальнена інтерпретація процесу стиснення, де кожне повідомлення розділяється на блоки розміром 8 біт та поєднуються в один ланцюг. Це дозволяє досягти мінімальну необхідну кількість можливих з'єднань з Сервером ззовні, що значно ускладнює типові атаки на порти з'єднання, які могли бути активними, але не використовуватись.

Після утворення нового ланцюга, він повторно шифрується, з метою приховати внутрішні ідентифікатори користувачів від аналізу потенційними

зловмисниками, після чого Сервер 1 повідомляє Сервер 2 про початок відправки повідомлень та починає пересилання.

Таким чином, навіть якщо хтось зможе продивитись трафік, в нього не буде жодного розуміння стосовно змісту повідомлень, що надсилаються від одного Серверу до іншого.

Тому загальна схема з'єднання двох Серверів буде виглядати наступним чином

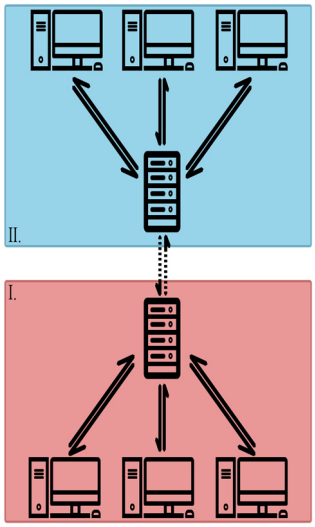

Рис 4. Загальна схема з'єднання Серверів двох філій корпорації.

На рис 4 червоним кольором позначено локальну мережу філії I, а синім - локальну мережу філії II. У кожній з мереж кожен пристрій має двостороннє з'єднання з локальним Сервером, який, в свою чергу, має двостороннє з'єднання з віддаленим Сервером іншої філії через мережу Інтернет. Ця схема передбачає відсутність у користувачів до мережі Інтернет всередині локальної мережі, а єдине, що з'єднує їх з іншою філією - їх локальний Сервер, який, в свою чергу, має доступ до мережі інтернет, але не приймає на свій зовнішній порт інші з'єднання, таким чином значно знижуючи ризики атак ззовні.

Однією з найголовніших частин будь-якого месенджеру є його інтерфейс, оскільки, незалежно від використаних технологій, популярність месенджеру буде залежати від того, наскільки комфортно ним користуватись.

Тому інтерфейс має бути простим для розуміння та ефективно використовувати простір екрану, даючи при цьому можливість змінювати зовнішній вигляд відповідно до побажань користувача.

Спираючись на ці потреби було створено макет інтерфейсу, зображений на рис 5.

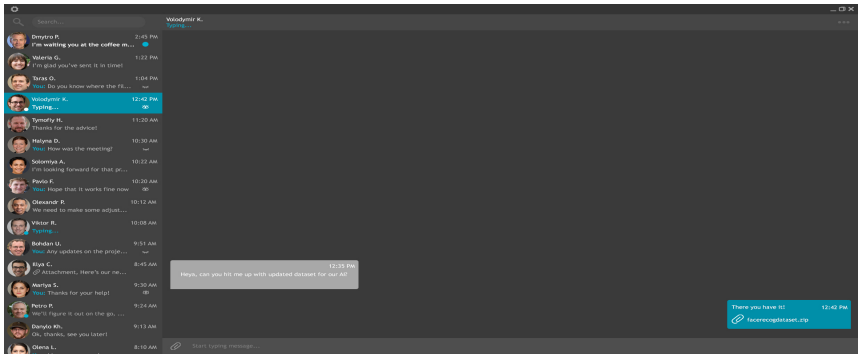

Рис 5. Макет інтерфейсу мессенджеру

У цьому макеті екран розподіллено на дві частини: список чатів та вікно активного чату. Це дозволяє користувачу мати легкий доступ до будь-якого чату навіть при листуванні з іншим користувачем.

Додатково, у лівій стрічці розташовано пошук по чатам, завдяки чому користувач матиме змогу легко та швидко знайти повідомлення та/або файли, які він отримав раніше, без необхідності перевіряти кожен з чатів вручну.

Для більш простого розуміння, кожен чат позначається додатковими символами - це зачинене та відчинене око та блакитна крапка. Перші два позначають чи прочитав інший користувач те повідомлення, яке було йому надіслане. Третій символ позначає непрочитані повідомлення, які отримав користувач, зроблено це для того, щоб візуально звернути увагу користувача на нові повідомлення.

Також у стрічці зліва відображаються: ім’я, зображення профіля, дата останнього повідомлення, стан повідомлення та передпоказ останнього повідомлення або “Typing…”, у випадку коли інший користувач набирає нове повідомлення.

У самому вікні чату користувач бачить свої повідомлення по правому боку, а повідомлення іншого користувача - по лівому. За бажанням користувач може змінити відображення повідомлень по одному боці.

Повідомлення у чаті демонструє час, коли воно було надіслано та його зміст, включаючи зображення та/або інші прикріплені файли. У вікні чату, якщо інший користувач ще не переглянув повідомлення, біля нього буде відображена блакитна крапка, яка буде відображати для користувача цей факт. А у випадку коли повідомлення все ще надсилається - буде відображатись символ годинника.

Також у користувача присутнє меню налаштувань, що зображене на Рис 6.

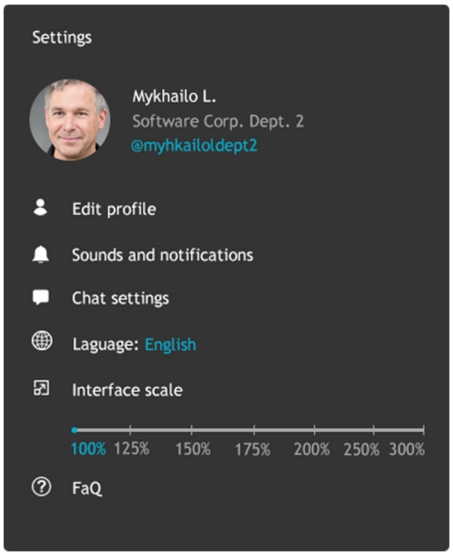

Рис 6. Меню налаштувань

Це вікно відкривається поверх вікна месенджера та має в собі наступні елементи: профіль користувача та можливість налаштувати профіль, звуки, чати, мову та розмір інтерфейсу, а також ЧаПи, які допоможуть користувачам швидше розібратись як саме влаштован месенджер.

У меню Edit profile користувач може змінити: зображення та ім’я профілю. Департамент користувача призначається самим сервером, до якого користувач вперше підключається.

Sounds and notifications дозволяє змінити гучність або повністю вимкнути звуки сповіщень та відображення спливаючих повідомлень, з можливістю вимкнути або повністю або на певний проміжок часу.

Chat settings дозволяє налаштувати фонове зображення чатів, кольорову схему інтерфейсу, а також розташування повідомлень користувача (по правому чи по лівому боку).

Налаштування мови дозволяє обрати мову інтерфейсу або додати власний переклад, у разі коли мова не входить в список мов за замовчуванням.

Розмір інтерфейсу дозволяє вибрати необхідний розмір, згідно з обраним користувачем монітором.

Список літератури

1. Wang X., Wu R., Ma J., Long G. Research on Vulnerability Detection Technology for WEB Mail System. *Procedia Computer Science.* 2018. Vol.131. P.124-130
2. Vulnerability found in top messaging apps let hackers eavesdrop. 2021. https://www.pandasecurity.com/en/mediacenter/security/vulnerability-messaging-apps/
3. CVE-2015-0235 - https://nvd.nist.gov/vuln/detail/CVE-2015-0235
4. CVE-2017-1274 - https://nvd.nist.gov/vuln/detail/CVE-2017-1274
5. CVE-2019-12569 - https://nvd.nist.gov/vuln/detail/CVE-2019-12569

DEVELOPMENT OF MULTISERVER CORPORATE MESSENGER WITH SPECIAL PROTECTION OF INTERSERVER TRAFFIC

M.A. Lisovsky, O.A. Stopakevych

National Odesa Polytechnic University,
ave. Shevchenko, 1, Odesa, 65044, Ukraine; e-mail: mix313131@gmail.com

The proposed implementation of the messenger, which assumes that in order to access the data related to the messenger, the user needs to contact the server of the local corporate network, otherwise the data will be in an encrypted state. The connection logic is built as follows: The Client connects to the Server located within the Corporate Network. After confirming the versions of the Client and Server, the connection is established and the Client goes into working mode - the data stored in the middle of the Client is decrypted and the User gets the opportunity to work with this data. When a User sends a message to another User, the Client encrypts the message and sends it to the Server specifying the Sender and Recipient. If the Recipient is in the same network as the Sender, the Server forwards the message without additional actions. Otherwise - the Server waits for several more messages (from any Senders), if the set number is reached - the messages begin to be packed into one big one, after which they are encrypted again and sent to another Server. If after the set time the number is not dialed - the Server adds an insufficient number of Messages in the form of a random set of values that will not have a Sender and a Recipient (so that the second Server rejects these messages as "noise").

Keywords. Multiserver messenger. Protection of inter-server traffic. Encrypts