УДК 003.26

АСПЕКТИ УПРАВЛІННЯ КЛЮЧОВОЮ ІНФОРМАЦІЄЮ ДЛЯ СИСТЕМ ШИРОКОМОВНОГО ШИФРУВАННЯ

І. А. Мельник[1,a], А. М. Кудін[1]

[1] Навчально-науковий Фізико-технічний інститут

Анотація

У даній роботі наведено протоколи широкомовного шифрування, зокрема протоколи управління ключами для групи абонентів. Було сформульовано основні проблеми забезпечення властивостей прямої і посткомпрометуючої секретності, наведено методики керування ключовою інформацією та можливі вразливості модифікованих систем. У результаті аналізу було показано, що модифіковані схеми широкомовного шифрування мають вразливості, а саме: залежність від ініціатора групи, витік критичної інформації та неможливість виконання умов прямої і посткомпрометуючої секретності.

Ключові слова: широкомовне шифрування, групове шифрування, ключовий розподіл

Вступ

У доповіді розглядається проблема побудови ефективної схеми розподілу ключів для великих груп абонентів. Більшість систем використовують алгоритм Double Ratchet [1], розроблений дослідницькою групою Signal. Однак цей алгоритм суттєво обмежує розмір групи. Asynchronous Ratchet Tree (або ART) [2] є ефективнішим за Double Ratchet, однак має проблеми з синхронізацією, динамічністю та коректністю дерева при оновленні.

Опираючись на одну з ефективних схем широкомовного шифрування, в нашій роботі буде обґрунтовано можливість адаптації схеми до задачі шифрування у групах та проведено її аналіз вразливостей.

1. Основні означення та теоретичні відомості

1.1. Розподіл ключової інформації в мережі

Для шифрування повідомлень в групі абоненти мають встановити відповідні спільні ключі за допомогою деякого алгоритму обміну ключів. Обчисливши ключі, користувачі зможуть обмінюватися повідомленнями таким чином, що якщо деякий користувач $i \in \mathcal{S}$ хоче поширити повідомлення $\boldsymbol{M}$, то він шифрує повідомлення відповідно до алгоритму так, що будь-який інший користувач $j \in \mathcal{S}$ може розшифрувати отримане повідомлення.

Існує безліч підходів, як влаштувати групове шифрування між користувачами: окреме шифрування для кожного учасника групи, шифрування для окремих підмножин множини користувачів $\mathcal{S}$ та єдине шифрування для всіх користувачів. Відповідно, групове шифрування може вимагати встановлення різної кількості ключів: від індивідуального ключа для кожного користувача чи підмножини користувачів до одного ключа на всю групу. Для порівняння цих підходів введемо деяку міру ефективності Δ, за яку приймемо кількість необхідних систем шифрування між користувачами для передачі інформації в групі, кожна з яких має власні задані системні параметри та ключі.

Якщо потужність множини $\mathcal{S}$ є великою, то окреме шифрування для кожного учасника призведе до великої кількості різних каналів зв'язку для обміну. Це неефективний підхід, враховуючи, що сучасні групові чати можуть мати досить високу щільність повідомлень. Таким чином, чим менша міра Δ, тим ефективніше відбувається обмін у групі.

Окрім цього, щоб система залишалася безпечною навіть за умови компрометації поточного секрету, вона повинна відповідати наступним властивостям:

Пряма секретність (англ. forward secrecy) [3] – це властивість криптосистеми зберігати секретність минулих ключів при компрометації поточного.

Посткомпрометуюча секретність (англ. post-compromised secrecy) [4] – це властивість системи зберігати секретність майбутніх ключів після компрометації поточного.

1.2. Широкомовне шифрування

Широкомовне шифрування – це технологія, яка дозволяє одночасно передавати зашифрований контент користувачам, які знаходяться в системі або каналі. Початково, задача широкомовного шифрування була сформульована наступним чином [5]: дозволити централізованому вузлу мовлення транслювати безпечні передачі інформації довільному визначеному набору одержувачів, мінімізуючи при цьому процес,

[a] illmel-ipt23@lll.kpi.ua

пов'язаний з управлінням ключами.

Однак перші реалізації таких схем не вирізнялися ефективністю та не мали унікального розрізнення учасників, сприймаючи всіх користувачів як множину порядкових номерів від 1 до n. Логічним наступним кроком розвитку широкомовного шифрування стали протоколи, що базувалися на основі ідентифікаторів – IBBE протоколи [6, 7]. Це дозволило більш гнучко керувати користувачами системи, виконуючи шифрування лише для визначеного набору ідентифікаторів.

За основу дослідження було обрано схему з [6], яку називають протоколом IBBE з фіксованим розміром шифротекстів і приватних ключів, що дозволяє використовувати лише один канал зв'язку для всіх користувачів, підтримуючи міру $\Delta = 1$.

1.3. Білінійне спарювання

Нехай $\mathbb{G}_1$, $\mathbb{G}_2$ і $\mathbb{G}_3$ це три комутативні групи простого порядку p. Тоді білінійне спарювання [8, 9] – це відображення $e(\cdot,\cdot) : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$, що для будь-яких двох генераторів $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$ та $a, b \in Z_p$:

$$e(g_1^a, g_2^b) = e(g_1, g_2^b)^a = e(g_1^a, g_2)^b = e(g_1, g_2)^{a\cdot b},$$
$$e(g_1 \cdot g_1', g_2) = e(g_1, g_2) \cdot e(g_1', g_2),$$
$$e(g_1, g_2 \cdot g_2') = e(g_1, g_2) \cdot e(g_1, g_2').$$

1.4. Протокол IBBE з фіксованим розміром шифротекстів і приватних ключів

Протокол, запропонований у [6], вирізняється простотою конструкції та вдало поєднує ефективність і безпеку завдяки використанню білінійного спарювання.

`Setup`(λ, m). За заданим параметром безпеки λ та цілого числа m, що визначає максимальну кількість користувачів в системі, група білінійного відображення $\mathscr{B} = (p, G_1, G_2, G_T, e(\cdot,\cdot))$ побудована таким чином, що $|p| = \lambda$. Також два генератори $g \in G_1$, $h \in G_2$ і секретне значення $\gamma \in \mathbb{Z}_p^*$ обираються довільним випадковим чином. Обирається деяка криптографічна геш-функція $\mathscr{H} : \{0,1\}^* \to \mathbb{Z}_p^*$. Зазначимо, що ця версія вимагає роботи $\mathscr{H}$ як випадкового оракула. Головний секретний ключ визначається як MSK $= (g, \gamma)$. $\mathscr{B}$ і $\mathscr{H}$ є публічними параметрами системи. Публічний ключ системи PK $= (w, v, h, h^\gamma, ..., h^{\gamma^m})$, де $w = g^\gamma$ і $v = e(g, h)$.

`Extract`(MSK, ID). За заданим секретним значенням головного секретного ключа MSK $= (g, \gamma)$ та ідентифікатора користувача ID, алгоритм обчислює значення приватного ключа користувача як

$$\mathrm{sk}_{\mathrm{ID}} = g^{\frac{1}{\gamma + \mathscr{H}(\mathrm{ID})}}.$$

`Encrypt`$(\mathcal{S}$, PK). Припустимо, що група користувачів, яким ми хочемо надіслати повідомлення, є деяка множина $\mathcal{S} = \{\mathrm{ID}_j\}_{j=1}^s$, де $s \le m$. За заданим значенням публічного ключа PK $= (w, v, h, h^\gamma, ..., h^{\gamma^m})$, власник широкомовного шифрування обирає деяке випадкове значення $k \leftarrow \mathbb{Z}_p^*$ й обчислює заголовок $\mathrm{Hdr} = (C_1, C_2)$ та ключ K наступним чином:

$$C_1 = w^{-k}, \qquad C_2 = h^{k\cdot\psi_{\mathcal{S}}(\gamma)}, \qquad K = v^k,$$
$$\psi_{\mathcal{S}}(\gamma) = \prod_{i\in\mathcal{S}} (\gamma + \mathscr{H}(\mathrm{ID}_i)).$$

Алгоритм `Encrypt` повертає пару значень (Hdr, K), де K – це ключ, що використовується для шифрування повідомлення M з допомогою деякого симетричного шифру: $E(M, K) = C_M$.

`Decrypt`$(\mathcal{S}, \mathrm{ID}_i, \mathrm{sk}_{\mathrm{ID}_i}, \mathrm{Hdr}, \mathrm{PK})$. Для того, щоб отримати ключ шифрування повідомлення K, інкапсульований в заголовку $\mathrm{Hdr} = (C_1, C_2)$, користувач з ідентифікатором $\mathrm{ID}_i \in \mathcal{S}$ обчислює:

$$K = \left(e(C_1, h^{\phi_{i,\mathcal{S}}(\gamma)}) \cdot e(\mathrm{sk}_{\mathrm{ID}_i}, C_2)\right)^{1/\prod_{\substack{j\in\mathcal{S}\\ j\neq i}} \mathscr{H}(\mathrm{ID}_j)}, \text{ де}$$

$$\phi_{i,\mathcal{S}}(\gamma) = \frac{1}{\gamma} \cdot \left(\prod_{\substack{j\in\mathcal{S}\\ j\neq i}} (\gamma + \mathscr{H}(\mathrm{ID}_j)) - \prod_{\substack{j\in\mathcal{S}\\ j\neq i}} \mathscr{H}(\mathrm{ID}_j) \right).$$

2. Широкомовне шифрування як протокол шифрування у групах

2.1. Адаптований алгоритм IBBE

Нижче буде наведено опис розширеної схеми з [6] для шифрування у групах.

Фаза ініціалізації. Аналогічно до стандартних протоколів, нехай деякий користувач A буде ініціатором створення групи з деякою множиною користувачів $\widehat{S}$, а загальною множиною користувачів групи вважатимемо $S = \widehat{S} \cup \{A\}$. Тоді, з допомогою алгоритму `Setup`(λ, m), аналогічно оригінальній схемі зі статті [6], користувач A генерує необхідні параметри групи. На даному етапі припускається, що користувач A володітиме секретними параметрами як власник групи.

Фаза створення ключів. Цей етап також не буде мати суттєвих змін. Користувач A з допомогою MSK та генератора приватних ключів генерує приватні ключі користувачів системи, у тому числі й для себе: $\mathrm{sk}_{\mathrm{ID}_A} = g^{\frac{1}{\gamma + \mathscr{H}(\mathrm{ID}_A)}}$, де ID_A – його ідентифікатор у групі. Після цього етапу всі користувачі групи зможуть надсилати повідомлення.

Фаза шифрування. Нехай деякий користувач з ідентифікатором ID_i хоче надіслати повідомлення в групу. Для цього він генерує деяке випадкове значення $k \leftarrow Z_p^*$ та обчислює симетричний ключ $K = v^k$, яким зашифрує повідомлення M: $E(M, K) = C_M$. Далі користувач i повинен обчислити значення $\mathrm{Hdr} = (C_1, C_2)$, з допомогою яких кожен користувач групи зможе обчислити значення ключа K. Хоча значення $\psi_s(\gamma) = \prod_{i\in\mathcal{S}} (\gamma + \mathscr{H}(\mathrm{ID}_i))$ є залежним від секрету γ, проте згадаємо, що публічний ключ системи PK містить наступні значення: $h, h^\gamma, h^{\gamma^2}, ..., h^{\gamma^m}$. Розпишемо $\psi_s(\gamma)$ у вигляді поліному

від γ:

$$\prod_{i\in\mathcal{S}}(\gamma+\mathscr{H}(\mathrm{ID}_i)) =$$
$$= (\gamma+\mathscr{H}(\mathrm{ID}_1))\cdot(\gamma+\mathscr{H}(\mathrm{ID}_2))\cdot\ldots\cdot(\gamma+\mathscr{H}(\mathrm{ID}_s)) =$$
$$= \gamma^s+\gamma^{s-1}\cdot\sum_{i=1}^{s}\mathscr{H}(\mathrm{ID}_i)+\gamma^{s-2}\cdot\sum_{i<j}^{s}\mathscr{H}(\mathrm{ID}_i)\cdot\mathscr{H}(\mathrm{ID}_j)+\ldots$$
$$+\gamma\sum_{j_1<j_2<\ldots<j_{s-1}}^{s}\mathscr{H}(\mathrm{ID}_i)\mathscr{H}(\mathrm{ID}_{j_1})\ldots\mathscr{H}(\mathrm{ID}_{j_{s-1}})+\prod_{i\in\mathcal{S}}\mathscr{H}(\mathrm{ID}_i).$$

Таким чином, ми можемо отримати:

$$C_2 = h^{k\cdot\prod_{i=1}^{s}(\gamma+\mathscr{H}(\mathrm{ID}_i))} =$$
$$= (h^{\gamma^s}\cdot h^{\gamma^{s-2}\cdot\sum_{i<j}^{s}\mathscr{H}(\mathrm{ID}_i)\cdot\mathscr{H}(\mathrm{ID}_j)}\cdot\ldots\cdot h^{\prod_{i=1}^{s}\mathscr{H}(\mathrm{ID}_i)})^k,$$

де h^{γ^i} та ID_i для $i=\overline{1,s}$ – публічні параметри. Отже, кожен учасник групи, використовуючи публічні параметри системи, може шифрувати повідомлення.

Фаза розшифрування. Розшифрування відбувається аналогічним чином до оригінальної схеми, використовуючи той самий підхід, що застосовувався при шифруванні повідомлення. Коли користувачі отримали значення Hdr та C_M, вони обчислюють значення ключа K:

$$\widehat{K_l} = e\left(C_1, h^{\frac{1}{\gamma}\cdot\left(\prod_{j=1,j\neq i}^{s}(\gamma+\mathscr{H}(\mathrm{ID}_j))-\prod_{j=1,j\neq i}^{s}\mathscr{H}(\mathrm{ID}_j)\right)}\right) =$$
$$= e(g,h)^{-k\cdot\left(\prod_{j=1,j\neq i}^{s}(\gamma+\mathscr{H}(\mathrm{ID}_j))-\prod_{j=1,j\neq i}^{s}\mathscr{H}(\mathrm{ID}_j)\right)},$$
$$\widehat{K_r} = e(\mathrm{sk}_{\mathrm{ID}_i}, C_2) = \left(g^{\frac{1}{\gamma+\mathscr{H}(\mathrm{ID}_j)}}, h^{k\cdot\prod_{i=1}^{s}(\gamma+\mathscr{H}(\mathrm{ID}_i))}\right) =$$
$$= e(g,h)^{k\cdot\prod_{j=1,j\neq i}^{s}(\gamma+\mathscr{H}(\mathrm{ID}_j))},$$
$$K = (\widehat{K_l}\cdot\widehat{K_r})^{\frac{1}{\prod_{j=1,j\neq i}^{s}\mathscr{H}(\mathrm{ID}_j)}} = (g,h)^k.$$

Після цього користувач може розшифрувати повідомлення M, використавши деякий симетричний алгоритм дешифрування для шифротексту C_M та ключа K: $Dec(C_M, K) = M$.

Фаза оновлення групи. Щоб додати нового користувача з ідентифікатором ID_i, A перевіряє, чи $|\mathcal{S}| \leq m$. Якщо так, то він запускає алгоритм $\mathtt{Setup}(\mathrm{MSK}, \mathrm{ID}_i)$, що поверне секретний ключ користувача i та оновлює множину користувачів $\mathcal{S} := \mathcal{S}\cup\{\mathrm{ID}_i\}$. Інакше, він додатково обчислює новий параметр h^{m+1} та оновлює публічний ключ системи PK, додаючи нове значення:

$$\mathrm{PK} := (w, v, h, \ldots, h^{\gamma^m}, h^{\gamma^{m+1}}).$$

Щоб видалити користувача з ідентифікатором ID_i, A видаляє його з множини користувачів: $\mathcal{S} := \mathcal{S}\backslash\{\mathrm{ID}_i\}$. Видалення може ініціюватися як через запит користувача i, так і за власним рішенням A.

2.2. Проблеми адаптованого протоколу IBBE

Адаптований протокол не має достатньої безпеки, він не забезпечує властивості прямої і посткомпрометуючої безпеки, як, до прикладу, протокол ART. Окрім цього, необхідно вирішити питання організації самих секретів групи.

Керування секретами групи. В загальному випадку, керування секретними значеннями системи MSK має адміністратор A – ініціатор створеної групи. Окрім цього, він має доступ до приватних ключів усіх користувачів, оскільки він володіє усіма параметрами для обчислення цих значень. Щоб убезпечити систему від випадків, коли власника групи можуть скомпрометувати або він почне поводитися зловмисно, можна розподілити володіння приватними параметрами одним із способів:

- Довіреність третій стороні, яка брала участь у створенні групи $\mathcal{S}$ і володіє значеннями секретних параметрів групи. Вимагає додаткових витрат для залучення такої сторони, що змогла б підтримувати систему, будучи постійно в мережі. Крім того, довірена сторона може бути аналогічно скомпрометована або вийти із ладу і бути не в мережі.
- Розподіл секрету на частини між користувачами. Формально розподіл можна виконати як між усіма користувачами так і з деякою обмеженою множиною. Крім цього, може існувати деякий поріг, необхідний для відновлення початкового секрету. Очевидно, що великий поріг може вплинути на затримку в групі, оскільки неможливо гарантувати, що усі користувачі будуть в мережі.

Також нагадаємо, що секрети групи прямо впливають на генерацію усіх ключів в системі. Тому кожне оновлення MSK буде призводити до повної перегенерації параметрів. Це складний і трудомісткий процес, що включає в себе ще й окрему комунікацію з кожним учасником групи для надсилання нового приватного ключа. З одного боку, це спонукає відмовитися від частого оновлення для ефективної роботи, а з іншого – часте оновлення підтримує безпеку.

Керування приватними ключами учасників групи. Інтуїтивно, модифікація ключів повинна вирішити дві проблеми: залежність приватних ключів лише від ініціатора A та введення деякого змінного параметра, що дозволить досягти прямої секретності системи. Згадаємо, як виглядає секретний ключ для користувача з ідентифікатором ID_i, що генерує ініціатор групи. Визначимо його тепер як:

$$\mathrm{presk}_{\mathrm{ID}} := g^{\frac{1}{\gamma+\mathscr{H}(\mathrm{ID})}}. \qquad (1)$$

Припустимо, що користувач i згенерував деяке випадкове значення $r_i \leftarrow Z_P^*$, що є його секретом і яке він зберігатиме. Тоді можна піднести значення ключа (1) до степеня $1/r_i$:

$$\mathrm{sk}_{\mathrm{ID}} := (\mathrm{presk}_{\mathrm{ID}})^{\frac{1}{r_i}} = (g^{\frac{1}{\gamma+\mathscr{H}(\mathrm{ID})}})^{\frac{1}{r_i}} = g^{\frac{1}{(\gamma+\mathscr{H}(\mathrm{ID}))\cdot r_i}}. \qquad (2)$$

Тепер ключ залежить від деякого секрету, що відомий лише користувачу i. Крім того, можна організувати процедуру оновлення, де користувач періодично оновлював би свій приватний ключ. Однак, варто помітити, що за компрометації приватного ключа

(2) користувача i разом із значенням r_i, зловмисник зможе отримати початкове значення ключа (1), що призведе до можливої підміни секретного значення r_i. Отже, необхідно додати до ключа ще одну змінну, яка зберігала б секретність утвореної ключової структури і не зберігалася користувачем. Наприклад, використати акумулююче значення $\epsilon_{i,k}$, де k – це кількість оновлень. Основна ідея акумулюючих значень полягає в наступному: робити засліплення параметру системи під час кожного оновлення, доміксовуючи минуле значення $\epsilon_{i,k-1}$ із деяким новим випадковим $\epsilon_{i,\text{new}}$, що буде видалятися після використання: $\epsilon_{i,k} = \epsilon_{i,k-1} \cdot \epsilon_{i,\text{new}}$. Опишемо цю процедуру наступним чином: нехай відбувається k-тий раунд оновлення ключів користувача i, а його секретний ключ має вигляд:

$$\text{sk}_{\text{ID}} := g^{\frac{\epsilon_{i,k-1}}{(\gamma+\mathscr{H}(\text{ID}))\cdot r_i}}.$$

Тоді він генерує випадкові $r_{i,\text{new}}, \epsilon_{i,\text{new}} \leftarrow Z_p^*$ і оновлює ключ за наступною схемою:

$$\text{sk}_{\text{ID}} := (\text{sk}_{\text{ID}})^{\frac{r_i\cdot\epsilon_{i,\text{new}}}{r_{i,\text{new}}}} = \\ = (g^{\frac{\epsilon_{i,k-1}}{(\gamma+\mathscr{H}(\text{ID}))\cdot r_i}})^{\frac{r_i\cdot\epsilon_{i,\text{new}}}{r_{i,\text{new}}}} = g^{\frac{\epsilon_{i,k}}{(\gamma+\mathscr{H}(\text{ID}))\cdot r_{i,\text{new}}}}.$$

При цьому, старе значення r_i видаляється і перевизначається новим значенням: $r_i := r_{i,\text{new}}$.

Модифікація схеми шифрування. В ході дослідження було виявлено, що схеми, засновані на спарюванні, дуже важко піддаються модифікаціям.

Вводячи нові параметри в одній частині конструкції їх необхідно рівноцінно компенсувати і в протилежній частині схеми, щоб у результаті отримати бажане значення. Крім цього, є велика ймовірність, що для компенсації знадобиться додаткова інформація, яка може розкривати додаткові деталі схеми. Покажемо це на основі розширеного протоколу із секції 2.1, яка визначається наступним чином.

`Setup`(λ, m) визначатиметься таким чином, що публічний ключ обчислюватиметься як

$$\text{PK} := (w, v, h, h^{1/\gamma}, R_{-1}, R_0, R_1, ..., R_m),$$

де $w = g^\gamma$, $v = e(g,h)$, а $R_i = h^{\gamma^i}$.

`Extract`(MSK, ID). За заданим MSK $= (g, \gamma)$ та ідентифікатором ID, алгоритм преобчислює значення приватного ключа користувача ID як в 1.

`KeyGen`$(r_i, R, \text{presk}_{\text{ID}})$. Кожен користувач генерує випадкове значення $r_i \in_R \mathbb{Z}_p^*$ і дообчислює його власний секретний ключ як 2.

Далі кожен користувач $(\forall i \in \mathcal{S})$ переобчислює значення PK як $\forall j = \overline{-1, m} : R_{j,\text{new}} = R_j^{r_i}$. Потім, вони надсилають ініціатору A

$$\text{PK}_{\text{new}_i} = (w, v, h, h^{1/\gamma}, R_{-1,\text{new}}, ..., R_{m,\text{new}}).$$

A приймає це значення як новий публічний ключ $\text{PK} := \text{PK}_{\text{new}_i}$ і поширює в системі як оновлену версію публічного ключа. Варто помітити, що для коректності ця процедура має виконуватися під чітким контролем і почергово, де A буде довіреною стороною.

`Encrypt`$(\mathcal{S}, PK)$. Припустимо, що група $\mathcal{S} = \{\text{ID}_j\}_{j=1}^s$, де $s \le m$. Тоді користувач обирає випадкове значення $k \leftarrow \mathbb{Z}_p^*$ і обчислює заголовок $\text{Hdr} = (C_1, C_2)$ і K, де

$$C_1 = w^{-k}, \qquad C_2 = h^{k\cdot\psi_{\mathcal{S}}(\gamma)\cdot\prod_{j\in\mathcal{S}} r_j}, \qquad K = v^k,$$
$$\psi_{\mathcal{S}}(\gamma) = \prod_{j\in\mathcal{S}}(\gamma + \mathscr{H}(\text{ID}_j)).$$

Алгоритм `Encrypt` повертає значення (Hdr, K). Далі K використовується як симетричний ключ для шифрування повідомлення.

`Decrypt`$(\mathcal{S}, ID_i, sk_{ID_i}, Hdr, PK)$. Для того, щоб отримати ключ шифрування повідомлення K, інкапсульований в заголовку $\text{Hdr} = (C_1, C_2)$, користувач з ідентифікатором $\text{ID}_i \in \mathcal{S}$ обчислює:

$$K = \left(e(C_1, h^{\phi_{i,\mathcal{S}}(\gamma)}) \cdot e(\text{sk}_{\text{ID}_i}, C_2)\right)^{1/\prod_{\substack{j=1\\ j\neq i}}^{s} \mathscr{H}(\text{ID}_j)}, \text{ де}$$

$$\phi_{i,\mathcal{S}}(\gamma) = \frac{1}{\gamma}\cdot\left(\prod_{\substack{j=1\\ j\neq i}}^{s}(\gamma + \mathscr{H}(\text{ID}_j)) \cdot \prod_{\substack{j=1\\ j\neq i}}^{s} r_j - \prod_{\substack{j=1\\ j\neq i}}^{s} \mathscr{H}(\text{ID}_j)\right).$$

`KeyUpdate`(PK, r_i, sk_i). Користувач A визначає правила оновлення секретних параметрів приватних ключів r_i для користувача i наступним чином: згенерувавши нове значення $r_{i,\text{new}} \in \mathbb{Z}_p^*$:

1. $\forall j = \overline{1, m}: \quad R_{j,\text{new}} := R_j^{r_{i,\text{new}}/r_i}$;
2. $sk_i := sk_i^{r_i/r_{i,\text{new}}}$;
3. Користувач i видаляє старе значення r_i;
4. $r_i := r_{i,\text{new}}$.

Хоча ця система не буде мати повну безпеку, однак є прикладом того, як можна зробити, щоб ініціатор групи не мав знання приватних ключів користувачів. Однак така схема має вразливість серед нових параметрів. Якщо зловмисник перехопить значення PK та C_1, то він зможе отримати:

$$K' = e(w, C_1) = e(g^{-\gamma\cdot k}, h^{1/\gamma}) = e(g,h)^{-k}.$$

Наступним кроком йому залишається отримати обернене значення до цього елемента, що і буде ключем, який шифрує повідомлення: $K = (K')^{-1}$.

Висновки

В даній роботі було продемонстровано адаптацію широкомовного шифрування для групового шифрування. Було показано, що схема IBBE з фіксованим розміром шифротекстів і приватних ключів може піддаватися такій адаптації з мінімальними модифікаціями. Проте запропонована схема не може забезпечити виконання умов прямої чи посткомпрометуючої секретності. За модифікації розширеної конструкції необхідно розкривати додаткову інформацію для обчислення основних перетворень, яка може компрометувати систему, як було показано в наведеній атаці. Крім цього, залишається висока залежність від ініціатора групи. Однак, широкомовне шифрування

може знайти застосування у сьогоднішньому сегменті групових чатів як деякий полегшений протокол, що може шифрувати деякі службові повідомлення.

Перелік використаних джерел

1. *Marlinspike M.*, *Perrin T.* The Double Ratchet Algorithm // Signal. — 2016.
2. On ends-to-ends encryption asynchronous group messaging with strong security guarantees / K. Cohn-Gordon, C. Cremers, L. Garratt, J. Millican, K. Milner // Proceedings of the ACM Conference on Computer and Communications Security. — 2018. — DOI: 10.1145/3243734.3243747.
3. *Boyd C.*, *Gellert K.* A Modern View on Forward Security // Computer Journal. — 2021. — Т. 64, вип. 4. — ISSN 14602067. — DOI: 10.1093/comjnl/bxaa104.
4. *Cohn-Gordon K.*, *Cremers C.*, *Garratt L.* On post-compromise security // Proceedings - IEEE Computer Security Foundations Symposium. 2016— August. — 2016. — DOI: 10.1109/CSF.2016.19.
5. *Fiat A.*, *Naor M.* Broadcast Encryption //. Т. 773. — 01.1993. — С. 480—491. — ISBN 978-3-540-57766-9. — DOI: 10.1007/0-387-30038-4_15.
6. *Delerablée C.* Identity-based broadcast encryption with constant size ciphertexts and private keys // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 4833 LNCS. — 2007. — DOI: 10.1007/978-3-540-76900-2_12.
7. *Sakai R.*, *Furukawa J.* Identity-Based Broadcast Encryption. — 2007. — URL: https://eprint.iacr.org/2007/217. Cryptology ePrint Archive, Paper 2007/217.
8. *Menezes A.* An introduction to pairing-based cryptography //. — 2009. — DOI: 10.1090/conm/477/09303.
9. *Zhang F.*, *Safavi-Naini R.*, *Susilo W.* An efficient signature scheme from bilinear pairings and its applications // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). — 2004. — Т. 2947. — ISSN 16113349. — DOI: 10.1007/978-3-540-24632-9_20.