Технологія захисту автентифікаційних даних користувачів комп'ютерної мережі

Юрій Хлапонін[1], Володимир Вишняков[2], Олег Комарницький[3]

[1,2] Київський національний університет будівництва і архітектури
пр-т Повітрофлотський, 31, Київ, Україна, 03680
[1] y.khlaponin@gmail.com, https://orcid.org/0000-0002-9287-0817
[2] volodymyr.vyshniakov@gmail.com, https://orcid.org/0000-0003-4668-712X
[3] Київська міська державна адміністрація
вул. Хрещатик, 36, Київ, Україна, 01044
[3] komarnitskiy2012@gmail.com, https://orcid.org/0000-0003-4830-919X

Received 16.10.2023, accepted 07.12.2023
https://doi.org/10.32347/uwt.2023.13.1204

Анотація. Стаття присвячена проблемі досконалого захисту автентифікаційних даних користувачів комп'ютерних систем, особливо у разі великої кількості різних прав та повноважень, які персонально надаються користувачам. Важливість цієї проблеми особливо зростає коли кількість таких користувачів у системі дорівнює сотням або тисячам. При цьому кожному з них необхідно створити умови для доступу лише до своїх даних і забезпечити захист від будь-якого позаштатного втручання, як з боку інших користувачів, так і з боку штатного персоналу. Типовим прикладом таких умов є системи таємного електронного голосування. Програмно-технічні рішення, що описані у даній роботі, пройшли багаторічну перевірку та продовжують практично використовуватись у системі електронного голосування Київського національного університету будівництва і архітектури. Ця система регулярно використовується для проведення виборів до органів студентського самоврядування та для опитувань серед студентів щодо якості викладання дисциплін. За допомогою цієї системи було проведено вибори керівників Товаристві Червоного Хреста України під час пандемії, яка були пов'язана з вірусом Covid-19. Математичною основою описаної технології захисту автентифікаційних даних є теорія алгебраїчних груп, а саме, задача дискретного логарифмування над полями Галуа великого розміру. Завдяки використанню криптографічних перетворень над цими полями, замість відомих геш-функцій, вдалося позбутись можливості колізій і нейтралізувати розкриття паролів зловмисниками, які мали змогу це робити за допомогою спеціалізованих Інтернет ресурсів. Також вдалося підвищити стійкість до розкриття паролів методом Brute-force, оскільки перетворення на полями Галуа потребують у десятки разів більше часу ніж обчислення геш-функцій.

Ключові слова: захист

Юрій Хлапонін
завідувач кафедри кібербезпеки та комп'ютерної інженерії
д.т.н., професор, академік Української академії наук

Володимир Вишняков
доцент кафедри кібербезпеки та комп'ютерної інженерії
к.т.н., доцент

Олег Комарницький
Chief Information Officer in Diit.digital

автентифікаційних даних, захист комп'ютерної інформації, захист паролів від розкриття, захисту даних користувачів комп'ютерних систем, використання полів Галуа.

ОПИС ПРОБЛЕМИ

Забезпечення конфіденційності зберігання автентифікаційних даних є найважливішою умовою для захисту комп'ютерної інформації від несанкціонованого доступу. Тому прийнято зберігати паролі у вигляді геш-функцій, наприклад MD-5, PBKDF, scrypt, bcrypt, argon2, серед яких MD-5 є найбільш розповсюдженим рішенням, а argon2 – найбільш ефективним [1]. Але усі вони мають недоліки, до яких слід віднести таке:

– колізії геш-функцій, що означає надання однакового результату, як на вірний пароль, так і на деяку множину схожих паролів;

– можливість розкриття паролів за допомогою відомих спеціалізованих ресурсів, наприклад, таких як md5decrypter.com або msurf.ru.

Перелічені недоліки було виявлено авторами під час розробки та тестування системи електронного голосування для студентів у закладі вищої освіти. Проблема полягала в тому, що деякі різні, але схожі між собою, паролі мали однакові геш-функції, а це надавало можливість проголосувати не тільки за себе, але й за іншого виборця. При цьому виборець, за якого проголосували, за правилами системи голосування, втрачає право голосу. Ідентифікатори призначались виборцям за відкритими правилами, тому, проголосувавши за себе, зловмисник робив спроби голосування з чужими ідентифікаторами та зі своїм паролем. Така ситуація може мати місце у разі коли паролі випадково співпадають, але під час спеціальної перевірки було виявлено, що паролі були різні, але співпадали їх геш-функції. Було вирішено відшукати технічне рішення, яке б дозволило позбутись цієї проблеми, а також позбавити можливості зловмисників, користуючись спеціалізованими ресурсами, розкривати паролі. Обраному технічному рішенню присвячено цю статтю.

МЕТА РОБОТИ

Метою даної роботи є розробка технології автентифікації, яка була б позбавлена колізій геш-функцій, а також унеможливлювала розкриття даних автентифікації за допомогою спеціалізованих ресурсів, наприклад, таких як md5decrypter.com або msurf.ru..

ОСНОВНА ЧАСТИНА

Нова технологія, що дозволяє позбутись вказаних недоліків під час захисту автентифікаційних даних користувачів мережі, побудована на засадах, які описані у роботі [2]. Такий захист базується на відомій задачі дискретного логарифмування з використанням у якості алгебраїчної групи поля Галуа. Хоч в усіх схожих системах захисту неможливо усунути ризик від розкриття методом Brute-force (грубої сили), що означає перебір усіх можливих варіантів, але цей ризик у нашому випадку мінімізується до значень, що виходять за межі реальності. Для цього показник степені n у полі Галуа $GF(2^n)$ обирається з множини достатньо великих безпечних простих чисел. Під достатньо великими слід розуміти такі числа, для яких розв'язати задачу дискретного логарифмування на рівні сучасної комп'ютерної техніки не уявляється можливим. Крім того, для формування показника степені над полем Галуа обрано конкатенацію пароля з ідентифікатором, а зберігання цих криптографічних перетворень реалізовано у масиві персональних даних користувача. Адміністратор сервера через свій інтерфейс не має можливості отримати зашифровані автентифікаційні дані користувачів, які формуються і зберігаються у базі даних на окремому сервері. У разі використання методу Brute-force, затрачувати час на обчислення степені над полем Галуа, потрібно у десятки разів більше, ніж на

знаходження геш-функцій.

Найпростішим варіантом використання поля Галуа GF(2^n) замість геш-функції є перетворення паролю у степінь твірного елементу алгебраїчної групи, де показником степеню обирається пароль. Наприклад, якщо обрати для перетворень поле GF(2^{503}), де 503 є безпечним простим числом (бо 503 = 251 * 2 +1, де 251 – просте число), то значення степені можна отримати з виразу

$$A^P = \prod_{i=1}^{503} \left(A^{2^{P_i}}\right)^{b_i}, \qquad (1)$$

де A – елемент поля GF(2^{503});

P – пароль, що перетворений у двійкове число (рядок бітів, де i – номер біта);

P_i – дорівнює 0, якщо i-тий біт нульовий, чи дорівнює 2^{i-1}, якщо i-тий біт одиничний:

b_i – значення i-того біта,

Множення у виразі (1) виконується за правилами множення елементів скінченого поля за допомогою функції MU();

Принцип перетворення паролю P у двійкове число полягає у тому, що кожен символ латинського регістру за стандартною кодовою таблицею ASCII замінюється послідовністю з семи бітів. Максимальна послідовність бітів у отриманому таким чином рядку, за умовами обраного поля Галуа, не повинна перевищувати 503. Це означає, що кількість символів паролю не повинна перевищувати 71, що є достатнім для будь-яких реальних значень паролів.

Початковою процедурою для реалізації піднесення до степеню за виразом (1) є утворення масиву M для степенів елементу A, що має такий вигляд

$$M = \left[A^{2^0}, A^{2^1}, A^{2^2}, A^{2^3}, \ldots, A^{2^{502}}\right] \qquad (2)$$

Програмну реалізацію виразів (1) та (2) показано на рис. 1 та на рис. 2.

```
const { workerData, parentPort } = require('worker_threads');
var A = [504]; // Елемент поля для піднесення до степені
var B = [504]; // Показник степені
var M = new Array(504); // Масив для степенів елементу A
for(var i=0; i<504; i++) M[i]=new Array(504);//Доповнення виміру M
var M1=[504], M2=[504]; // Множники для функції MU()
var R=[504]; // Добуток для функції MU()
let STR=''; // Символьний рядок для введення та виведення даних
STR=workerData; // Введення значень A та B у символьний рядок
for (var i=1;i<=503;i++) A[i]=B[i]=0; // Спочатку заносимо нулі
for (var i=1;i<=503;i++) {if (STR[i-1]=='1') A[i]=1;//Ввели A
if (STR[i+502]=='1') B[i]=1;} // Вводимо значення B
for (var i=1;i<=503;i++) M[1][i]=A[i];//Занесли A у перший елем. M
for (var I=2;I<=503;I++) // Цикл заповнення решти елементів M
{ // Кожний наступний елемент масиву буде квадратом попереднього
for (var J=1;J<=503;J++) M1[J]=M2[J]=M[I-1][J]; MU();
for (var j=1;j<=503;j++) M[I][j]=R[j];
} // Завершено заповнення елементів M
// Для початку циклу множень заносимо у A значення одиниці
for (var i=1;i<=503;i++) A[i]=0; A[1]=1;
for (var J=1;J<=503;J++) // Цикл множення елементів M
if (B[J]==1)//Обираємо для множення одиниці з показнику степені,
{ // бо множення на нуль дає 0.
for (var I=1;I<=503;I++) {M1[I]= M[J][I]; M2[I]=A[I];}
// Кожне наступне множення відбувається на результат попереднього,
MU(); // а перше – на одиничний елемент поля
for (var I=1;I<=503;I++) A[I]=R[I]; // Переносимо результат для
} // наступного множення у циклі, або для виводу у кінці циклу
// Результат піднесення A до степеня B занесено в A
STR=''; // Спорожнили символьний рядок для виводу результату
for (var i=1;i<=503;i++){if (A[i]==1)STR=STR+'1';else STR=STR+'0';}
// Перенесли результат піднесення A до степеня B у рядок STR
parentPort.postMessage(STR);//Відправили результат у головну про-
граму
```

Рис. 1. Програма піднесення елементу A до степені B (на мові JavaScript).

```
function MU()
{ var i,j,r,r1,r2,r3; for (i=1;i<=503;i++) R[i]=0;
// Спочатку занесли нулі у масив бітів результату
for (i=1;i<=503;i++) // Початок циклу множення
if (M1[i]==1) // Обираємо лише одиниці, бо множеня на 0
дає 0
{for (j=1; j<=503; j++)
if (M2[j]==1) // Обираємо лише одиниці другого множника
{r=i+j-1; // Степінь результату є сумою степенів множників
if (r>503) // У разі перевищення степені результату
{r=r-503; // віднімаємо 503
if (r>=501) // У разі другого перевищення
{r=r-501; r1=1+r; r2=4+r; r3=501+r;
if (R[r3]==0) R[r3]=1; else R[r3]=0;
} // Додавання у разі другого перевищення
else {r1=r; r2=r+3;}
if (R[r1]==0) R[r1]=1; else R[r1]=0;
if (R[r2]==0) R[r2]=1; else R[r2]=0;
} // Додавання за модулем 2 у разі перевищень
else {if (R[r]==0) R[r]=1; else R[r]=0;}
} // Додавання за модулем 2 у разі відсутності перевищень
}
} // End of function MU()
```

Рис. 2. Функція множення елементів поля GF(2^{503}) з використанням поліному $x^{503} = x^3 + 1$.

Слід зауважити, що використання даної технології для захисту паролю адміністратора не є доцільним, бо лише вони мають доступу до файлу, де зберігаються зашифровані паролі. Для досконалого захисту адміністратору достатньо обирати пароль, який складно вгадати. Лише у разі наявності декількох адміністраторів сервера за умов недовіри між ними, використання такої технології може бути доцільним. При цьому бажано також періодично змінювати пароль, щоб за час підбирання попереднього паролю було вже встановлено новий.

Технологія захисту автентифікаційних даних, якій присвячена ця стаття, впроваджена в системі таємного електронного голосування, але вона може бути корисною і в інших системах масового користування, де існує ризик заподіяння шкоди у разі розкриття паролів. Опишемо дану технологію у вигляді послідовності наступних дій.

Дія 1. Кожному користувачу системи призначається унікальне значення ідентифікатора, яке співпадає з початковим паролем. Ідентифікатори, а також персональні дані виборців, заносяться у базу даних на сервері управління паролями. Також туди заносяться криптографічні перетворення за допомогою виразу (1) конкатенацій ідентифікаторів з паролями. Довжина цих перетворень становить 72 байти, де розміщується елемент поля Галуа з 503 бітів.

Дія 2. Кожен користувач, для одержання прав доступу, повинен у головному меню системи обрати режим перевірки або заміни паролю та заповнити отриману форму так, як показано на рис. 3.

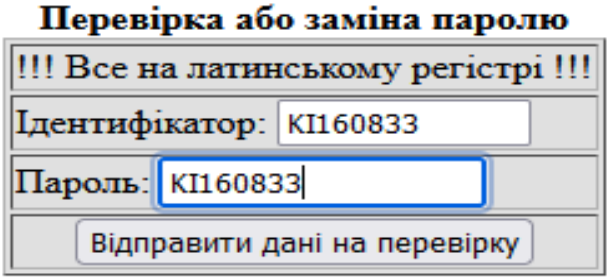

Рис. 3. Початок діалогу з сервером управління паролями.

Цей діалог реалізується клієнтською частиною підсистеми управління паролями, яка у вигляді html документу завантажується на пристрій користувача з сервера управління паролями після обрання відповідного режиму у головному меню системи .

Дія 3. Відправка даних на сервер управління паролями відбувається у вигляді послідовності з 72 байт криптографічного перетворення, які формуються за допомогою виразу (1) з конкатенації ідентифікатора з паролем. Ці 72 байти являють собою ключову інформацію для пошуку записів у базі даних. Оскільки дискретне логарифмування для обраного поля Галуа є надскладною задачею, то дешифрувати дані, у випадку їх перехоплення зловмисником, не уявляється можливим.

Дія 4. У разі позитивного результату пошуку інформації в базі даних, у користувача з'являється форма, яку показано на рис. 4.

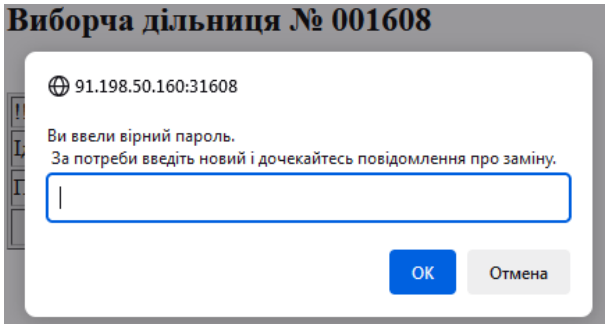

Рис. 4. Форма запиту для перевірки та заміни пароля.

Отримавши цю форму, користувач може бути впевненим, що його пароль не перехоплено зловмисником. Через цю форму після першої відправки даних на сервер управління паролями користувач повинен ввести і зберігати в таємниці свій персональний пароль.

У разі негативного результату пошуку інформації в базі, користувач отримає повідомлення про помилковий ідентифікатор або пароль. У цьому випадку користувач повинен звернутись до адміністратора сервера управління паролями для відновлення початкового пароля з подальшим поверненням до першої дії.

Дія 5. Пересилання нового паролю на сервер управління паролями відбувається у вигляді 72 байт криптографічного перетворення через захищений наскрізним шифруванням канал зв'язку, після чого відбувається поновлення у базі даних цього перетворення з 72 байт.

Дія 6. Копіювання інформації з бази даних сервера управління паролями на сервер голосування (або іншої прикладної задачі) відбувається безпосередньо у символьний масив прикладної програми через захищений наскрізним шифруванням канал зв'язку. Копіюються лише дані, що зашифровані у 72 байти та ознаки повноважень користувачів.

Дія 7. Користувачі звертаються до сервера голосування (або іншої прикладної задачі) через головне меню системи, обираючи потрібний режим, що призводить до завантаження клієнтської частини у вигляді html документу. Ця клієнтська частина одразу відправляє на сервер запит для утворення захищеного з'єднання між серверною та клієнтською частинами прикладного програмного забезпечення. При цьому на екрані клієнта з'являється форма для введення ідентифікатора і пароля.

Дія 8. Значення ідентифікатора і пароля відправляються на сервер через утворений захищений канал.

Дія 9. З конкатенації отриманих значень ідентифікатора і пароля на сервері за допомогою виразу (1) формується криптографічне перетворення у вигляді 72 байтів, які в точності співпадають з єдиним елементом масиву даних про клієнтів. Такий елемент у масиві може бути лише один, бо не може бути двох однакових ідентифікаторів, а обране криптографічне перетворення є однозначним.

Дія 10. Користувачу надаються права для виконання дій на сервері згідно з ознакою про його повноваження. Ця ознака знаходиться у сусідньому елементі масиву зі знайденим значенням індексу.

Проаналізуємо наявність слабких місць щодо можливості розкриття паролів. У даній системі передбачено, що усі паролі є індивідуальними, а єдиним місцем їх

збереження повинна бути лише пам'ять користувача або його особисті записи. Шифрування паролів відбувається безпосередньо у комп'ютерах користувачів одразу після введення (див. Дію 4). Оскільки обране криптографічне перетворення є досконалим, що забезпечується надзвичайною складністю задачі дискретного логарифмування, розв'язання якої для поля $GF(2^{503})$ є невідомим, то реалізація зворотного перетворення не є реальною. Навіть для більш простих полів Галуа, для яких розв'язана задача дискретного логарифмування, це потребує багато годин роботи на суперкомп'ютері. Тому єдиною реальною можливістю розкриття паролів є використання методу Brute-force. Для цього треба мати криптографічне перетворення паролів (їх у зашифрованому вигляді). У такому вигляді паролі пересилаються з комп'ютера користувача на сервер управління паролями, а також з сервера управління паролями на сервер голосування. Для цього пересилання утворюється абсолютно захищений канал, що досягається завдяки використанню шифру Вернама (one-time pad**),** що є єдиним, для якого математично доведена абсолютна криптографічна стійкість. У таблиці 1 надано перелік умов для абсолютного захисту даних під час передавання.

Таблиця 1. Умови забезпечення абсолютного захисту даних під час передавання

Опис умови	Виконання умови
Слід генерувати випадкові (не псевдо випадкові) бітові послідовності	Використано метод генерування випадкових бітів, який є працездатним на будь-якому комп’ютері [4].
Кожну випадкову бітову послідовність можна використовувати для шифрування тільки один раз	Для кожного сеансу зв’язку генеруються випадкові бітові послідовності незалежно одна від одної
Для передачі випадкових бітових послідовностей слід використовувати абсолютно захищений канал зв’язку	Обмін випадковими послідовностями бітів відбувається за алгоритмом Діффі-Геллмана з такими параметрами, для яких у сучасних умовах не існує можливості розкриття даних.

Для виконання останньої умови з табл. 1 у реалізації алгоритму Діффі-Геллмана обрано те саме поле Галуа $GF(2^{503})$, яке забезпечує досконалий захист завдяки надзвичайній складності дискретного логарифмування, що було описано вище. Технічно таке рішення виконано за допомогою тих самих комп'ютерних програм, що показані на рис. 1 та рис. 2. Таким чином, перехоплення даних під час передавання зашифрованих паролів не уявляється можливим через досконалий захист. Це означає, що єдиним місцем, де існує можливість отримати зашифровані паролі, є база даних на сервері управління паролями. Слід зауважити, що на сервері голосування ці дані заносяться одразу у символьний масив робочої програми, звідки їх добути неможливо. Навіть за умов розкриття змісту цього масиву практично неможливо буде дізнатись паролі, бо часу на їх пошук методом Brute-force у найсприятливіших умовах потрібно буде не

менше ніж десять років. Пояснюється це тим, що шифрується конкатенація ідентифікаторів разом з паролями, а це означає, що у кожному разі слід буде перебирати усі варіанти ідентифікаторів, що уповільнює пошук у кількість разів, що дорівнює кількості ідентифікаторів.

Залишається проаналізувати єдиний найбільш реальний шлях для розкриття паролів через втрату даних з сервера управління паролями. Таке можливо лише за участі адміністратора сервера, усі дії якого можуть бути проконтрольовані спеціалізованими засобами аудиту. Але якщо це трапиться, то припускаючи, що пароль буде з десяти символів на латинському регістрі, а для перевірки кожного варіанту за виразом (1) з використанням одного сучасного комп'ютера потрібно близько секунди, то повний час однієї перевірки буде приблизно 10^{20} секунд. Це дорівнює $3*10^{12}$ років. У разі, якщо одночасно використовувати 1 млн. комп'ютерів, а їх швидкодію підвищити у 1000 разів, то на розкриття кожного пароля буде потрібно витратити лише 3000 років. Для спрощення розрахунків було прийнято, що кількість варіантів символів дорівнює 100, що приблизно дорівнює тій кількості, яка може бути введена на латинському регістрі. Навіть, якщо дозволити скорочення паролів з десяти до восьми символів, то цей час буде зменшено до декількох місяців на пароль кожного користувача.

ВИСНОВКИ

Вважаючи, що забезпечення конфіденційності зберігання автентифікаційних даних є важливою умовою для захисту комп'ютерної інформації від несанкціонованого доступу, а у традиційних технологіях захисту було виявлено суттєві недоліки у разі значної кількості різних індивідуальних прав користувачів, запропоновано та впроваджено у діючій системі технологію, яка позбавлена цих недоліків. Прийняте за основу технології криптографічне перетворення з використанням операцій піднесення до степеню елементів поля Галуа, дозволило усунути виявлені недоліки та забезпечити досконалий захист автентифікаційних даних на усіх ділянках роботи системи за рахунок доволі простих та ефективних програмних рішень. Ці рішення є відкритими і докладно описані у роботі.

Детально проаналізована наявність слабких місць у запропонованій технології і показана висока досконаліть прийнятих технічних рішень, а також забезпеченість неможливості розкриття паролів будь-якими технічними засобами у реальних умовах.

ЛІТЕРАТУРА

1. **Biryukov Alex, Daniel Dinu, and Dmitry Khovratovich** (2017) Argon2: the memory-hard function for password hashing and other applications, University of Luxembourg, March 24, 2017. https://github.com/P-H-C/phc-winner-argon2/blob/master/argon2-specs.pdf.
2. **Khlaponin, Y., Vyshniakov, V., & Komarnytskyi, O.** (2023). Proof of the possibility for a public audit of a secret internet voting system. EUREKA: Physics and Engineering, (1), 189-200. https://doi.org/10.21303
3. **Hassan Mohamed Muhi-Aldeen, Yurii Khlaponin & et al.** (2021) Technology of secure data exchange in the IoT system. 3rd International Conference on Information Security and Information Technologies (ISecIT 2021) co-located with 1st International Forum "Digital Reality" (DRForum 2021) Odesa, Ukraine, September 13-19, 2021. CEUR Workshop P.115-121. http://ceur-ws.org/Vol-3200/.
4. **Чуприн В.М., Вишняков В.М., Пригара М.П.** (2016) Генерування випадкових чисел штатними засобами хостів мережі Інтернет. Захист інформації. – 2016. – Т. 18, №4. – С. 323-335. https://jrnl.nau.edu.ua/index.php/ZI/article/view/11085.

Technology of protection of authentication data of computer network users

Yuri Khlaponin, Volodymyr Vyshnyakov,
Oleg Komarnytskyi

Abstract. The article is devoted to the problem of perfect protection of authentication data of users of computer systems, especially in the case of a large number of different rights and powers that are personally granted to users. The importance of this problem increases especially when the number of such users in the system is hundreds or thousands. At the same time, each of them must create conditions for access only to their data and ensure protection against any outside interference, both by other users and by staff. A typical example of such conditions are secret electronic voting systems. The software and technical solutions described in this work have passed many years of testing and continue to be practically used in the electronic voting system of the Kyiv National University of Civil Engineering and Architecture. This system is regularly used for conducting elections to student self-government bodies and for surveys among students regarding the quality of teaching subjects. With the help of this system, elections of leaders of the Red Cross Society of Ukraine were held during the pandemic, which was associated with the Covid-19 virus. The mathematical basis of the described authentication data protection technology is the theory of algebraic groups, namely, the problem of discrete logarithmization over large Galois fields. Thanks to the use of cryptographic transformations over these fields, instead of known hash functions, it was possible to get rid of the possibility of collisions and neutralize the disclosure of passwords by attackers who were able to do it with the help of specialized Internet resources. It was also possible to increase resistance to brute-force password cracking, since conversions to Galois fields take ten times more time than hash function calculations.

Keywords: protection of authentication data, protection of computer information, protection of passwords from disclosure, protection of user data of computer systems, use of Galois fields.