Cryptography in decentralized systems

Practice 1

“Робота з пароль та геш-значеннями”

1. **Аргументація стійкості паролю**

Складність паролю визначається середньою кількістю спроб та загальною кількістю часу, який необхідно витратити на знаходження пароля або його підбору з використанням методів, алгоритмів, функцій, наприклад методу брутфорсу. Відомо, що від загальної кількості символів в алфавіті залежить кількість можливих комбінацій підбору. Наприклад, візьмемо варіант знаходження паролю який складається з 8 символів. Для першого варіанту де алфавіт складається тільки з цифр від 0 до 9 (усього 10), кількість можливих варіантів перебору або підбору буде 10^8? тобто 100 мільйонів. Якщо розглядати варіант з літерами англійського алфавіту (abcdefghijklmnopqrstuvwxyz - 26 літер), то кількість комбінацій складатиме 26 ^ 8, тому складність підбору паролю з 8 символів більша ніж в порівнянні з цифрами у 2088 разів. Якщо до літер англійського алфавіту додати верхні та нижні регістри то комбінацій буде 52 ^ 8 - це 5345 \* 10 ^ 10. Звичайно, якщо додати ще й цифри або спецсимволи, то кількість можливих комбінацій збільшиться ще більше. У випадку з цифрами буде 62 ^ 8, зі спецсимволами - 95 ^ 8. Отже, останній варіант паролю який складається з цифр, літер англійського алфавіту з верхнім та нижнім регістрами та спеціальними символами буде найнадійніший, оскільки потребуватиме більшої кількості комбінацій для можливого знаходження правильної послідовності.

1. **Пошук прообразу геш-функції**

SHA-3, як і ряд інших алгоритмі гешування представляють собою однонаправлені функції перетворення масиву вхідних даних довільної довжини у бітовий рядок фіксованої довжини. При цьому одним з ключових властивостей гещ-функцій є незворотність, тобто відсутність ефективного зворотного алгоритму для відновлення вихідного тексту за відомим геш-значенням, за винятком спроб повного перебору (брутфорсу). При цьому, навіть якщо після перебору вдалося підібрати значення яке співпадає, не можна точно стверджувати що початкове вихідне повідомлення було саме те, яке вийшло знайти, оскільки можуть бути присутні випадки колізій, бо вхідний масив даних він необмежений, а у випадку зі SHA-3 масив даних геш значення обмежений розміром у 256 бітів. Усього кількість можливих комбінацій для SHA-3 на довжині 256 бітів буде складати 16 ^ 64 = 2 ^ 256 варіантів.

Отже, в результаті перебору значень та логічного та дедуктивного мислення прообразами від геш-значень для виконання завдання є такі рядки тексту: 123, cryptography, Distributed Lab.

1. **Дослідження райдужних таблиць (rainbow table)**

Райдужна таблиця — це велика попередньо обчислена таблиця, призначена для кешування виводу криптографічних геш-функцій для виявлення прообразу гешованих паролів та повідомлень у форматі відкритого тексту. Райдужні таблиці були винайдені ІТ-фахівцем Філіпом Оексліном, який опублікував статтю про свою роботу в 2003 році. Сам метод базується на дослідженнях компромісу продуктивності між часом обробки та пам’яттю, що необхідні для криптоаналізу. «Райдуга» означає кольори, які використовуються в таблиці для представлення різних функцій і кроків зменшення. Вони створюють барвисту веселку з відповідною кількістю ітерацій.

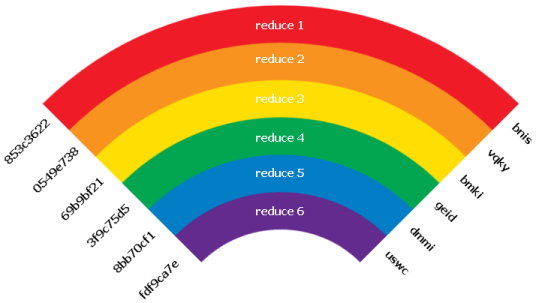


Рисунок 1 - Приклад райдужної таблиці

З міркувань безпеки протягом кількох років організації зазвичай зберігали паролі користувачів за допомогою гешів. Алгоритми гешування маскують паролі та роблять їх недоступними для перегляду без інвертування геш-функції. Технічно цього можна досягти за допомогою брутфорсу, але в міру колізій та того, як набір можливих результатів паролю може зростати, використання цього методу стає все більш неможливим та довгим за виконанням.

Райдужні таблиці значно зменшують складність злому великої кількості паролів за допомогою попередньо згенерованого набору даних гешів паролів. Через цей набір даних можна пропустити величезну кількість гешованих паролів і після кількох етапів редукції (зменшення), за допомогою яких геш-значення розбиваються на менші компоненти, обчислюються та пов’язуються зі словами та літерами, щоб виявити цілі паролі у відкритому тексті.

Райдужна таблиця зберігає всі однонаправлені геші відкритих текстових паролів від 1 до 7 та, можливо, 8 символів. (8-символьна райдужна таблиця для простого алгоритму гешування займе до 32 Петабайт простору бази даних для зберігання). Коли зловмисник викрадає кеш із простими гешами паролів, він може шукати геш-значення в райдужній таблиці, щоб знайти прообраз.

Райдужна таблиця працює, виконуючи криптоаналіз, достатньо швидко та ефективно, на відміну від повного перебору, який працює шляхом обчислення геш-функції кожного наявного рядка, обчислення їх геш-значення, а потім порівняння його з іншим на кожному кроці. Атака райдужної таблиці усуває цю потребу, оскільки вже обчислює геші великого набору доступних рядків. Усього виділяють два основні кроки:

* Створення таблиці. Тут береться геш рядка, який потім зменшується, щоб створити новий рядок, який потім зменшується повторно;
* Злом пароля. Починаючи з гешованого тексту (паролю), перевіряється його наявність у базі даних. Якщо так, то переходять до початку ланцюжка ітерацій та починають гешувати, доки не знайдеться збіг. Щойно збіг знайдено, процес успішно припиняється.

Наступна блок-схема пояснює ці два основних кроки:

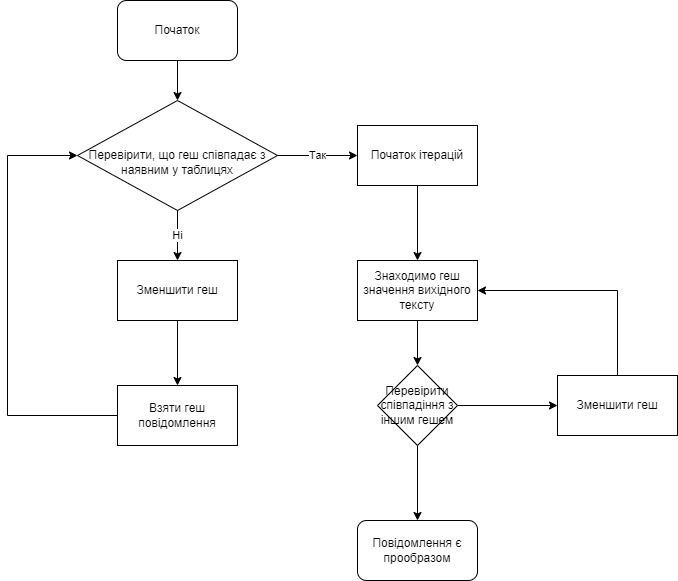


Рисунок 2 - Блок схема райдужних таблиць

Переваги:

* На відміну від брутфорсу, обчислення геш-функції не містить проблем (оскільки значення вже попередньо обчислено). Тому, увесь процес спрощений до простої операції пошуку та порівняння в таблиці.
* Точний рядок паролю знати не потрібно. Якщо геш збігається, це не має значення, якщо рядок не є самим паролем. Він вже буде аутентифікований.

Недоліки:

* Для зберігання таблиць необхідний великий обсяг сховища.

На наступному рисунку наведено специфікації радужних таблиць за алгоритмом гешування, довжини тексту, дискового простору, розміру таблиці та рівнем успішного знаходження.

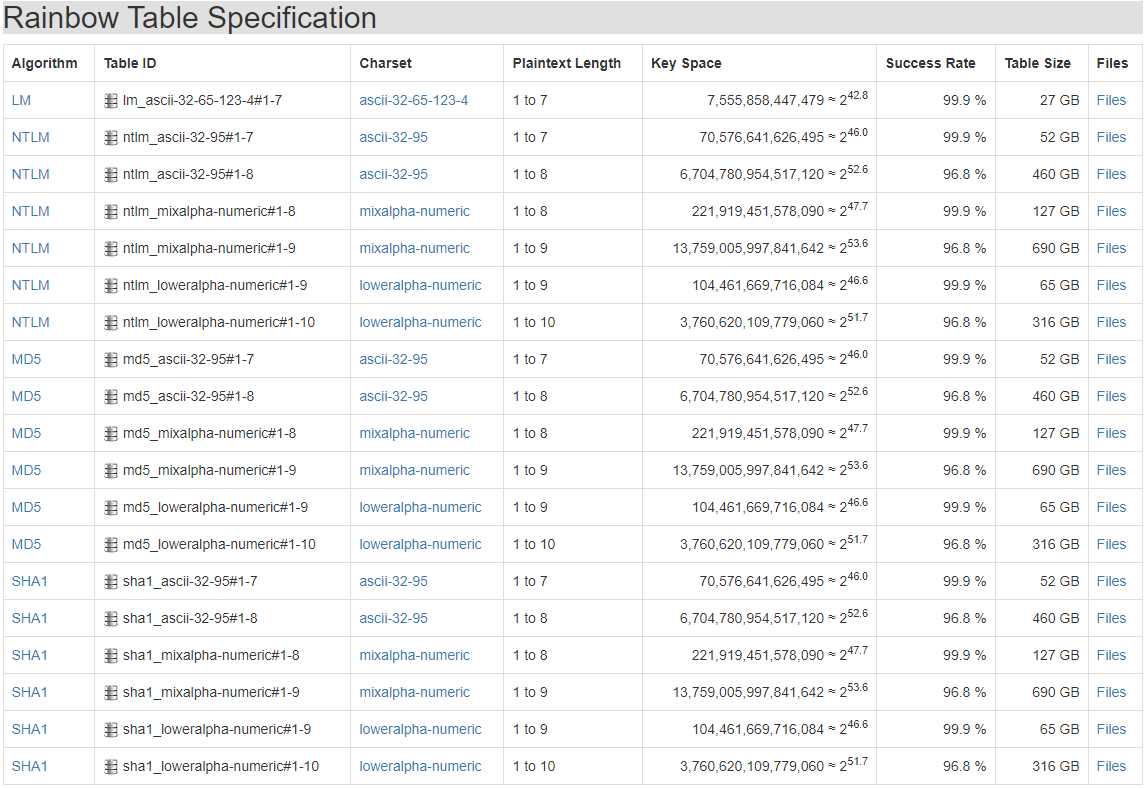


Рисунок 3 - Специфікації райдужних таблиць

Для захисту від атак райдужної таблиці можна використовувати різні методи, зокрема використання надійних, випадкових паролів, використання унікальної солі (рядку даних, модифікатору) для кожного пароля та багаторазове ітераційне гешування пароля. Ці методи ускладнюють для зловмисника попереднє обчислення райдужних таблиць, які можна використовувати для злому геш даних повідомлень та паролів.