Петров Д.М. КІТ-М120Б

Лабораторна робота №2

Зберігання паролів

Мета: Дослідити і порівняти існуючі механізми зберігання паролів.

Індивідуальне завдання

Дослідити існуючі механізми зберігання паролю. Зробити порівняльну характеристику кожного механізму. Реалізувати механізм зберігання паролю та продемонструвати процес аутентифікації. Довести що даний метод оптимальний.

Хід роботи

Хеш-функція - функція, що виконує одностороннє перетворення. По заданій строці (паролю) вона отримує деяке хеш-значення (шістнадцяткове позначення: зазвичай послідовність із цифри та букв a - f), таке, що за ним дуже сложно отримує яку-небудь інформацію про вихідну строку-пароль.

1. MD5

MD5 (Message Digest 5) — 128-бітний алгоритм хешування, розроблений професором Рональдом Л. Рівестом в 1991 році. Призначений для створення «відбитків» або «дайджестів» повідомлень довільної довжини. Прийшов на зміну MD4, що був недосконалим. Описаний в RFC 1321. З 2011 року відповідно RFC 6151 алгоритм вважається ненадійним.

Вибірки повідомлень MD5 широко використовувались у світі програмного забезпечення для гарантування правильної передачі файла. Наприклад, файлові сервери часто надають попередньо підраховану контрольну суму MD5 (відому як md5sum) для файлів, так що користувач може порівняти контрольну суму завантаженого файла з нею.

Оскільки дуже легко генерувати колізії у MD5, особа, яка створила файл, може створити другий файл із такою ж контрольною сумою, тому цей прийом не може захистити від деяких форм зловмисного втручання. У деяких випадках контрольній сумі не можна довіряти (наприклад, якщо вона була отримана за тим самим каналом, що і завантажений файл), у цьому випадку MD5 може забезпечити лише функціональність перевірки помилок: він розпізнає пошкоджене або неповне завантаження, що часто трапляється при завантаженні великих файлів.

Історично MD5 використовувався для зберігання одностороннього хешу пароля, часто з розтягуванням ключів. NIST не включає MD5 у свій список рекомендованих хешів для зберігання паролів.

2. Радужні таблиці

Райдужна таблиця (англ. rainbow table) — спеціальний варіант таблиць пошуку (англ. lookup table) для звернення криптографічних геш-функцій, які використовують механізм розумного компромісу між часом пошуку таблицею і займаною пам'яттю (англ. time-memory tradeoff).

Райдужні таблиці використовуються для розкриття паролів, перетворених за допомогою важкозворотної геш-функції, а також для атак на симетричні шифри на основі відомого відкритого тексту. Використання функції формування ключа з застосуванням солі робить цю атаку неможливою.

Райдужні таблиці є розвитком ідеї таблиці геш-ланцюгів. Вони ефективно вирішують проблему колізій шляхом введення послідовності функцій редукції R1, R2, …, Rk. Функції редукції застосовуються по черзі, перемежаючись з функцією гешування: H, R1, H, R2, …, H, Rk.

При такому підході два ланцюжки можуть злитися тільки за умови збігу значень на одній і тій же ітерації. Отже, достатньо перевіряти на колізії тільки кінцеві значення ланцюжків, що не вимагає додаткової пам'яті.

На кінцевому етапі складання таблиці можна знайти всі злиті ланцюжки, залишити з них тільки один і згенерувати нові, щоб заповнити таблицю необхідною кількістю різних ланцюжків. Отримані ланцюжки не є повністю вільними від колізій, тим не менш, вони не зіллються повністю.

Радужні таблиці підходять для пошуку паролів, але чим складніше чи довше пароль, кількість ланцюжків збільшується до величезних розмірів, понад 100 Гб.

3. Сіль (криптографія)

Сіль (також модифікатор) — рядок даних, який передається геш-функції разом з паролем.

Головним чином використовується для захисту від перебору за словником і атак з використанням райдужних таблиць, а також приховування однакових паролів. Однак, сіль не може захистити від повного перебору кожного окремого пароля.

Завдяки повному перебору можна знайти необхідні значення, але для зберігання словника для повного перебору, необхідно мати величезну кількість вільної пам’яті. Тому сіль – це один із варіантів покращити якість зберігання паролів.

4. X.509

X.509 — в криптографії стандарт ITU-T для інфраструктури відкритого (публічного) ключа (англ. public key infrastructure (PKI)) та інфраструктури управління привілеями (англ. Privilege Management Infrastructure (PMI)).

X.509 стандартизує формати представлення та кодування:

- сертифікатів відкритого ключа (англ. public key certificates);
- списку відкликаних сертифікатів (англ. certificate revocation lists);
- атрибутів сертифікатів (англ. attribute certificates);
- алгоритм перевірки методу сертифікації (англ. certification path validation algorithm).

5. **Bcrypt**

Bcrypt — адаптивная криптографическая хеш-функция формирования ключа, используемая для защищенного хранения паролей. Разработчики: Нильс Провос и David Mazières. Функция основана на шифре Blowfish, впервые представлена на USENIX в 1999 году. Для защиты от атак с помощью радужных таблиц bcrypt использует соль (salt); кроме того, функция является адаптивной, время её работы легко настраивается и её можно замедлить, чтобы усложнить атаку перебором.

Радужні таблиці являються універсальним способом підібрати необхідний пароль, але якщо використовувати хеш-функцію і зберігати паролі в хеш-значені, то шанси підібрати пароль значно зменшуються, але це все ще можливо. Для того щоб пароль не можливо було підібрати використовують або сіль для паролів перед хешуванням, наприклад звичайні хеш-функції, sha0, sha1, sha2, sha3, md5 або зразу хеш-функції з автоматичною сіллю у алгоритмі, такий як bcrypt.

Для безпечної передачі даних по мережі, використовують протокол https. Оскільки HTTPS це фактично HTTP, який передається через SSL або TLS, то майже всі його основні елементи шифруються: URL-запити, включаючи шлях та назву ресурсу (сторінки), параметри запиту, заголовки та кукі, які часто містять ідентифікаційні дані про користувача. Не шифруються: назва або адреса хоста (веб-сайту) та порт, оскільки вони використовуються транспортним протоколом TCP/IP для встановлення з'єднання.

Шифрування гарантує помірний захист від підслуховування та від нападу «людина посередині» (man-in-the-middle), за умови це коректних налаштувань та підпису сертифікату авторизованим центром сертифікації.

TLS або SSL використовую публічні та приватні ключи для обміну даними. Публічні ключи передаються за допомогою стандарту X.509.

В ході лабораторної роботи була розроблена програма в форматі клієнт-сервер. Клієнт проходить ідентифікацію, а сервер перевіряє логін, а потім і пароль який зберігається в хеш-форматі. Навіть якщо пароль буде легким, завдяки хеш-функції і солі справжній пароль підібрати буде складно, тому навіть радужній таблиці не допоможуть, а звичайний перебор потребує величезні розміри вільної пам’яті.

```
[danilpetrov@MacBook-Pro bin]$ md5 -s "passwords5a5l5t"
MD5 ("passwords5a5l5t") = 71019d290df19a9f6459d62623195e4e
```

Так пароль може бути password + s5a5l5t, де s5a5l5t – це сіль за допомогою якої пароль вже не підібрати.

Як можна бачити пароль був правильне відправлений:

```
[danilpetrov@MacBook-Pro bin]$ ./server 10100
Connection: 127.0.0.1:51090
No certificates.
Client msg: admin:password
admin
password
71019d290df19a9f6459d62623195e4e
admin
Authorization successful.
```

Клієнт підключається до сервера та отримує його сертифікат.

```
[danilpetrov@MacBook-Pro bin]$ ./client 127.0.0.1 10100
Enter the User Name :
admin
Enter the Password :
password
Connected with TLS_AES_256_GCM_SHA384 encryption
Server certificates:
Subject: /C=UA/ST=Name/L=Kharkiv/O=Name/OU=NTU KHPI/CN=Name/emailAddress=server@gmail.com
Issuer: /C=UA/ST=Name/L=Kharkiv/O=Name/OU=NTU KHPI/CN=Name/emailAddress=server@gmail.com
Received: Authorization successful.
```

Можна зробити висновок що використання солі з одною із хеш-функцій допомагає підвищити безпеку від взлому. Якщо навіть зловмисник вломає та отримає хеш-значення та зможе підібрати до них паролі, після відправки їх на сервер і додавання солі до паролів, хеш-значення буде іншим від того що зберігається на сервері.

Висновок

Дослідив існуючі механізми зберігання паролю. Зробив порівняльну характеристику кожного механізму. Реалізував механізм зберігання паролю та продемонстрував процес аутентифікації. Довів, що даний метод оптимальний.