Пірог О. В., канд. техн. наук, доцент кафедри КІтаКБ
Державний університет «Житомирська політехніка»

СПОСОБИ «СОЛІННЯ» ТАБЛИЦІ ПАРОЛІВ КОРИСТУВАЧІВ WEB-СЕРВЕРУ

В цілях протидії розголошенню таблиці паролів вона як правило зберігаються на сервері в хешованому вигляді. Хешування відноситься до складно оборотних функцій. У випадку потрапляння цієї таблиці до зловмисників їм потрібно підбирати паролі та зіставляти їх хешовані значення з даними в отриманій таблиці. Тому атака методом повного перебору (brute-force attack) ускладнюється часом хешування. За допомогою вибору способу хешування можна настроювати цей час та трудомісткість кодування. На практиці паролі мають нерівномірний розподіл (паролі типу 123, qwerty і такі подібні). 20% користувачів використовують 5000 найбільш популярних паролів. Тому задля 20% паролів можна зробити rainbow tables їх хешів та зіставляти вже ці таблиці.

Задля протидії цьому було вигадано метод «соління» паролів (Salt).

Приклад створення хешу з сіллю:

$$\$saltedPassword = md5(\$password.\$salt) \quad (1)$$

Переваги соління: унікальні навіть однакові паролі; додавання більше біт до псевдопаролю збільшує трудомісткість розгадування паролю; пароль буде відрізнятися навіть, якщо пароль замінять точно таким же паролем.

Соління може бути як статичним, коли домішується однакова сіль до усіх паролів, так і динамічною, коли для кожного паролю використовується своя сіль. Але якщо статична сіль або таблиця динамічних сілей потрапляє до зловмисника, то стійкість цього хешованого паролю значно понижується. З точки зору злому методом повного перебору (brute-force attack) стійкість пароля до хакерських атак сильно залежить як від його довжини, так і від використовуваного набору знаків. Ентропія пароля (складність пароля, яка вимірюється в бітах), згенерованого випадковим чином, знаходиться за такою формулою:

$$H = L \times \frac{\ln(N)}{\ln(2)}, \quad (2)$$

де

L – набір символів в паролі;

N – кількість символів у алфавіті, що використовується;

ln – натуральний логарифм.

Розглянемо гіпотетичний випадок, коли в нас пароль складається з 6 літер латинського алфавіту і сіль – з 6 літер, наприклад:

$$PPPPPPSSSSSS, \quad (3)$$

де

P – один символ паролю,

S – один символ солі.

Таким чином, якщо нам відома сіль, то ентропія знов створеного паролю після соління буде дорівнювати ентропії самого паролю, тобто:

$$H = 6 \times \frac{\ln(26)}{\ln(2)} = 28{,}2\,. \quad (4)$$

Також, якщо це паролі вигадані користувачем, то до них можна застосувати атаку за словником. Тому їх ентропія в залежності від частоти вживання цього паролю або слова може знижуватися до значень в діапазоні від 3 до 12.

Але якщо ми застосуємо нестандартний метод соління, коли символи солі перед хешуванням будуть перемішані з символами паролю, наприклад:

$$PPSPPSSSPPSS, \quad (5)$$

то зловмиснику, навіть якщо йому відома сіль, доведеться перебирати усі символи знов створеного паролю після соління. Та навіть окремі ділянки цього паролю не будуть розпізнаватися за словником. Ентропія такого паролю буде:

$$H = 12 \times \frac{\ln(26)}{\ln(2)} = 56{,}4\,. \quad (6)$$

Таким чином запропонований спосіб соління паролів дозволяє значно збільшити стійкість таблиці паролів до злому.

Список використаних джерел

1. Sriramya P., Karthika R.A. Providing password security by salted password hashing using bcrypt algorithm. Asian Research Publishing Network (ARPN) Journal of Engineering and Applied Sciences, 2015, pp. 5551–5556.

2. Robertiello A.G., Bandla K.A. Attacks on MD5 Hashed Passwords: Technical Report. George Mason University, USA, 2005.

3. Al-Fayoumi M., Aboud S. Blind decryption and privacy protection. Am. J. Appl. Sci, Vol. 2, 2005, pp. 873–876.

4. Dürmuth M., Güneysu T., Kasper M., Paar C., Yalçin T., Zimmermann R. Evaluation of Standardized Password-Based Key Derivation against Parallel Processing Platforms. Computer Security – ESORICS, 2012, pp. 716–733.

5. Rouse M. Salt. SearchSecurity. URL: http://searchsecurity.techtarget.com/definition/salt.