Using hashing algorithms for user authentication in an online testing and education platform

Boychenko O.[1], Gavrikov I.[2] (Russian Federation)

Использование алгоритмов хеширования для аутентификации пользователей в системе онлайн тестирования и обучения

Бойченко О. В.[1], Гавриков И. В.[2] (Российская Федерация)

[1]Бойченко Олег Валерьевич / Boychenko Oleg – доктор технических наук, профессор;
[2]Гавриков Илья Владимирович / Gavrikov Ilya – студент,
кафедра бизнес-информатики и математического моделирования,
Институт экономики и управления,
Крымский федеральный университет им. В. И. Вернадского, г. Симферополь

Abstract: *the article describes the principles behind the functioning of hashing algorithms and presents an example of their integration in an online testing and education platform developed for use in a university environment.*
Аннотация: *в статье описываются принципы работы алгоритмов хеширования и рассматривается их практическое применение на примере их интеграции в систему онлайн тестирования и обучения, разработанную для применения в университете.*

Keywords: *authentication, hashing, cryptography, information security.*
Ключевые слова: *аутентификация, хеширование, криптография, информационная безопасность.*

Today, the vast majority of web applications use some form of user authentication for purposes of separating access and privileges between different users and user groups, and especially for purposes of keeping confidential and personal information safely out of reach for third parties [1]. However, the risk of a user account falling victim to hacking is present in any authentication system, and different countermeasures exist to mitigate it. Among the most widely used techniques used to protect user account data is hashing applied to user passwords.

The process of 'hashing' is the use of a hash function to produce a unique value, also called a digest or hash value, for a given message. The digest can be used later for verification and authentication purposes. A hash function is a one-way function, which means that although the process of generating a digest is fast and inexpensive, the reverse process of reconstructing the original data is practically impossible. Also notable is the so-called 'avalanche effect'—cryptographically strong hashing algorithms will produce vastly different results for inputs that differ even very slightly, with one different or missing letter completely changing the function's result.

These properties of hash functions make them particularly useful for storing passwords. Because a password only needs to be checked at the time of user login, it is inexpensive for an authentication system to store the hash value for the user's password at the time of registration and verify the digest of the entered password against the stored value every time the user needs to authenticate. Any value other than the password entered by the user originally will produce a different digest, and through simple comparison, the system will be able to deny entry to a potential attacker.

Today, a variety of different hashing algorithms exists, the most popular among them being SHA-1, SHA-2 (particularly SHA-256 and SHA-512), RIPEMD, Whirlpool, and bcrypt. A number of hashing algorithms were also popular once but have since been proven to be susceptible to attacks and fallen into disuse. One notable hashing algorithm that once enjoyed widespread use but has since been 'broken' is MD5 [2].

Hashing was chosen as a safeguard against attacks on user accounts during development of an online testing and education system for the Crimean Federal University. Different hashing algorithms were analysed and compared in safety and ease of implementation. The SHA-512 algorithm was chosen for the system as the safest and arguably most widely used. Although SHA-1 was also considered, recent studies have shown it to be potentially unsafe for use [3].

Despite the fact that digests cannot be 'decrypted' as such and can only be retrieved through brute force attacks, such attacks also present a risk. A brute force attack against a hashed password theoretically involves generating hash values for every possible combination of characters and checking them against the given digest until a match has been found. Although such an approach is not practically feasible in itself, certain techniques make it more viable, such as using dictionaries of commonly used passwords. An especially dangerous technique involves the use of so-called 'rainbow tables'—lists that contain pre-generated digests for commonly used passwords. This reduces the workload significantly, only requiring the attacker to perform simple lookup and comparison operations for the given password digest. Paired with specialised hardware, rainbow tables become a dangerous and effective tool against any system that stores hashed passwords [4].

An effective countermeasure against the use of rainbow tables is so-called 'salting', which involves appending an arbitrarily long string of random characters—called a 'salt'—to a user password and using the result as the input for the hashing function. This means that even two identical passwords will be stored as different digests, since the salts used for them will be vastly different. This negates the advantage provided by rainbow tables by requiring the attacker to generate digests again, and potentially makes the task even more infeasible if the

attacker does not have access to the salt. Salting was used in addition to simple hashing during development of the university online education system due to the significant increase in data security it offers at a low implementation cost [5].

References

1. *Bykov D. V., Luk'janov V. S., Prohorov I. V., Skakunov A. V.* Sposoby autentifikacii i razgranichenija dostupa k bazam dannyh v servis-orientirovannyh prilozhenijah. // Izvestija Volgogradskogo gosudarstvennogo tehnicheskogo universiteta (№ 6, t. 6). Volgograd, 2009. S. 127.
2. *Xiaoyun Wang*, Dengguo Feng, Xuejia Lai, Hongbo Yu Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD. Universitet Shan'don / Kitajskaja Akademija Nauk / Universitet Shanhaj Dzjaoton, 2004.
3. *Bruce Morton* SHA-1 Freestart Collisions. [Jelektronnyj resurs]: Entrust. URL: https://www.entrust.com/sha-1-freestart-collisions/ (data obrashhenija: 03.06.2016).
4. Kostas Theocharoulis Implementing Rainbow Tables in High-End FPGAs for Super-Fast Password Cracking. // 2010 International Conference on Field Programmable Logic and Applications (Milano, August 31—September 2, 2010). Milan: IEEE, 2010. S. 145-150.
5. *Karen Scarfone* Murugiah Souppaya Guide to Enterprise Password Management: Recommendations of the National Institute of Standards and Technology. Gaithersburg, MD, 2009. 38 s.