JOURNAL OF APPLIED SCIENCE AND SOCIAL SCIENCE
eISSN 2229-3113 pISSN 2229-3205

Volume 15 Issue 10, October 2025

Impact factor: 2019: 4.679 2020: 5.015 2021: 5.436, 2022: 5.242, 2023: 6.995, 2024 7.75

RAINBOW TABLE THREAT AND DEFENSE MECHANISMS

Ibragimov Ulugbek Muradilloyevich

Associate professor in Asia international University

Abtract: Rainbow tables are pre-computed correspondence tables used in cybersecurity to recover passwords and sensitive information. This article explains in detail the concept of rainbow tables, how they work, what capabilities attackers have, and effective defenses against these attacks — in particular, the use of "salts" and strong, slow hash functions. The article explains the dangers of storing passwords in plain text through an example: if a system stores passwords without hashing or with a weak hash algorithm, an attacker can easily take over the database and break into accounts. Rainbow tables, on the other hand, provide much faster results by pre-computing a large number of passwords and their hashes, and then searching the table for the target hash; this significantly reduces the computation time compared to brute-force methods. The article also describes two types of salting — static and dynamic — and discusses their advantages and limitations, as well as what can happen if the salt is compromised. It also provides practical recommendations on using modern KDF (key derivation function) such as Argon2, bcrypt and scrypt, using pepper, strengthening password policies, encrypting files and adding additional layers of protection such as multi-factor authentication (MFA). The article also clearly shows the consequences of an attack - password cracking, data disclosure and credential stuffing, which increases the responsibility for protecting information systems.

Keywords: Rainbow table attack, Salting, Argon2 / bcrypt / scrypt, password security, credential stuffing, multi-factor authentication, data breaches, brute-force and offline attacks, static and dynamic salting, password policy and migration.

Аннотация: Радужные таблицы — это предварительно вычисленные таблицы соответствий, используемые в кибербезопасности для восстановления паролей и конфиденциальной информации. В этой статье подробно объясняется концепция радужных таблиц, принципы их работы, возможности злоумышленников и эффективные методы защиты от этих атак, в частности, использование «солей» и стойких, но медленных хеш-функций. В статье на примере объясняются опасности хранения паролей в виде открытого текста: если система хранит пароли без хеширования или со слабым алгоритмом хеширования, злоумышленник может легко получить доступ к базе данных и взломать учётные записи. Радужные таблицы, с другой стороны, дают гораздо более быстрые результаты благодаря предварительному вычислению большого количества паролей и их хеш-значений, а затем поиску в таблице целевого хеша; это значительно сокращает время вычислений по сравнению с методами полного перебора. В статье также описываются два типа «соли» — статическая и динамическая — и обсуждаются их преимущества и ограничения, а также возможные последствия компрометации «соли». В статье также даны практические рекомендации по использованию современных KDF (функций формирования ключей), таких как Argon2, bcrypt и scrypt, использованию Pepper, усилению политик паролей, шифрованию файлов и добавлению дополнительных уровней защиты, таких как многофакторная аутентификация (MFA). В статье также наглядно

JOURNAL OF APPLIED SCIENCE AND SOCIAL SCIENCE
eISSN 2229-3113 pISSN 2229-3205

Volume 15 Issue 10, October 2025
Impact factor: 2019: 4.679 2020: 5.015 2021: 5.436, 2022: 5.242, 2023: 6.995, 2024 7.75

показаны последствия атаки – взлом паролей, раскрытие данных и подмена учётных данных, что повышает ответственность за защиту информационных систем.

Ключевые слова: Атака с помощью радужных таблиц, «соль», Argon2 / bcrypt / scrypt, безопасность паролей, подстановка учетных данных, многофакторная аутентификация, утечки данных, атаки методом подбора и офлайн-атаки, статическое и динамическое «соль», политика паролей и миграция.

Annotatsiya: Kamalak jadvallari (rainbow tables) kiberxavfsizlik sohasida parollar va maxfiy ma'lumotlarni qayta tiklash uchun ishlatiladigan oldindan hisoblangan moslik jadvallaridir. Ushbu maqola kamalak jadvallari tushunchasini, ular qanday ishlashini, hujumchilar qanday imkoniyatlarga ega bo‘lishini va ushbu hujumlarga qarshi samarali himoya choralari — ayniqsa "tuzlash" (salt) va kuchli, sekin xesh-funksiyalardan foydalanish — haqida batafsil tushuntiradi. Matnda oddiy matnda parol saqlashning xavfi misol orqali ochib berilgan: agar tizim parollarni xesh qilmasdan yoki zaif xesh algoritmi bilan saqlasa, hujumchi ma'lumotlar bazasini egallab, hisoblarni osongina buzishi mumkin. Kamalak jadvallari esa oldindan ko‘p sonli parollar va ularning xeshlarini hisoblab, keyinchalik maqsadli xeshni jadvalda qidirish orqali ancha tez natija beradi; bu esa qo‘pol kuch (brute-force) usuliga nisbatan hisoblash vaqtini sezilarli darajada kamaytiradi. Shu bilan birga, maqolada tuzlashning ikki turi — statik va dinamik — ta'riflangan, ularning afzalliklari va cheklovlari, hamda tuz ochib berilganda ham nima sodir bo‘lishi mumkinligi muhokama qilingan. Shuningdek, Argon2, bcrypt va scrypt kabi zamonaviy KDF (key derivation function) ishlatish, pepper qo‘llash, parol siyosatini kuchaytirish, fayllarni shifrlash va ko‘p faktorli autentifikatsiya (MFA) singari qo‘shimcha himoya qatlamlari haqida amaliy tavsiyalar berilgan. Maqola hujum oqibatlari — parolni buzish, ma'lumotlarning oshkor bo‘lishi va hisoblarni to‘ldirish (credential stuffing) kabi xatarlarni ham aniq ko‘rsatadi, bu esa axborot tizimlarini himoya qilish bo‘yicha javobgarlikni oshiradi.

Kalitli so’zlar: Rainbow table hujumi, Tuzlash (salt), Argon2 / bcrypt / scrypt, parol xavfsizligi, credential stuffing, ko‘p faktorli autentifikatsiya, ma'lumotlar buzilishi, brute-force va oflayn hujumlar, statik va dinamik tuzlash, parol siyosati va migratsiya.

In cybersecurity, the Rainbow table attack is one of the most advanced methods used by cybercriminals to crack passwords and confidential data. Understanding this technique is essential for anyone tasked with protecting information systems. This article explores the concept of the Rainbow table attack, its mechanisms of operation, consequences, and methods of protection against it.

In this article, I want to explain what Rainbow tables are, how they are used to attack users, and how we can prevent this attack.

Let me start the article with some interesting facts:

1) According to public data, 55% of Internet users use the same type of password for most websites.

JOURNAL OF APPLIED SCIENCE AND SOCIAL SCIENCE
eISSN 2229-3113 pISSN 2229-3205
Volume 15 Issue 10, October 2025
Impact factor: 2019: 4.679 2020: 5.015 2021: 5.436, 2022: 5.242, 2023: 6.995, 2024 7.75

2) There are more than 30% of websites that store your passwords in plain text, including well-known sites like Google (Google has been storing some passwords in plain text for fourteen years).

As we have seen, many users and some websites do not follow standard practices, which can easily lead to a hacker gaining access to your account[1].

Let's look at how a typical login process works. Let's say a user has already registered on a website where the password is stored in plain text. The next time the user visits the website and enters their credentials, the following process occurs (Figure 1).

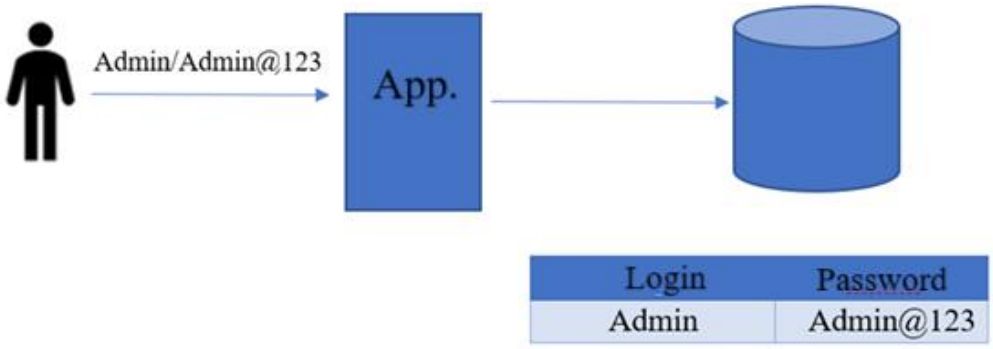

Figure 1. Plain text password scheme stored in a database.

As you can see in the image above, the user enters the credentials (username: Admin/password: Admin@123) and then the application checks the database to see if the credentials the user entered match. If it matches, the user is authenticated, otherwise not[2].

For more security, we can hash the password before storing it in the database. Since we are storing the hashed password in the database, it cannot be directly matched to the plain text password entered by the user. So, we hash the password entered by the user and match the result with the password in the database.

Hashing passwords provides some security, but hackers can attack the user using Rainbow Tables (Figure 2).

JOURNAL OF APPLIED SCIENCE AND SOCIAL SCIENCE
eISSN 2229-3113 pISSN 2229-3205
Volume 15 Issue 10, October 2025
Impact factor: 2019: 4.679 2020: 5.015 2021: 5.436, 2022: 5.242, 2023: 6.995, 2024 7.75

Figure 2. A simple scheme for attacking a user using Rainbow Tables.

Rainbow tables are pre-computed tables used in cryptography to crack passwords. They are basically lists of commonly used passwords and their hashes.

There is a wealth of information about leaked passwords on the Internet. This can be used to create a rainbow table[3].

First, an attacker gains access to password hashes from the target system (SQL injection attack, etc.).

Rainbow table attacks are much more efficient than brute force attacks. This efficiency comes from the fact that the computationally intensive process of hashing possible passwords is performed ahead of time. Instead of computing the hash for each guess, the attacker looks up the hash in a pre-computed rainbow table. This pre-computing dramatically reduces the time it takes to crack passwords, especially passwords that are common or relatively short[4].

The consequences of a rainbow table attack include:

1) Password cracking: Once an attacker has access to clear-text passwords, they can gain unauthorized access to sensitive data and systems.

2) Data breach: Compromised passwords can expose sensitive personal and organizational information, leading to a large-scale data breach. Unauthorised data collection can occur when compromised passwords give attackers access to sensitive databases, leading to the misuse of personal or organizational data.

3) Credential stuffing: Attackers can use compromised passwords for credential stuffing attacks, where they attempt to use the same password for different accounts, taking advantage of the tendency of users to reuse passwords across multiple platforms[5].

JOURNAL OF APPLIED SCIENCE AND SOCIAL SCIENCE
eISSN 2229-3113 pISSN 2229-3205
Volume 15 Issue 10, October 2025
Impact factor: 2019: 4.679 2020: 5.015 2021: 5.436, 2022: 5.242, 2023: 6.995, 2024 7.75

Several measures can be taken to protect against rainbow table attacks:

1. Salted hashes

Salting involves adding a unique, random value to each password before hashing. This means that even if two users have the same password, their hashes will differ due to the "unique salt." This approach makes pre-computed rainbow tables inefficient, as the attacker would have to create a separate rainbow table for each unique salt value.

2. Strong hashing algorithms

Using strong, slow hashing algorithms such as bcrypt, Argon2, or scrypt can significantly increase the difficulty of creating rainbow tables. These algorithms are designed to be computationally intensive, making pre-computing difficult to do in practice.

3. Password policy

Implementing a strong password policy that requires complex, long passwords can reduce the risk of successful rainbow table attacks. Complex passwords are less likely to be found in pre-computed tables.

4. Encrypt Files

Encrypting files is another important step to ensure the security of sensitive data, even if passwords are compromised.

5. Multi-Factor Authentication (MFA)

Implementing multi-factor authentication adds an extra layer of security. Even if an attacker manages to crack a password, a second factor will still be required to log in[10].

6. Encrypted Email Services

Switching to encrypted email platforms like Atomic Mail can help keep sensitive data safe, even if attackers try to use weak passwords. Encrypting your emails adds an extra layer of protection, protecting sensitive data from being intercepted or compromised.

7. Update passwords regularly

Encouraging users to change their passwords regularly can reduce the risk of rainbow table attacks. Changing passwords frequently limits the amount of time a compromised password is useful to an attacker[6,7].

The first of these protection mechanisms is the most widely used and is considered more effective than the others, so let's take a closer look at this mechanism.

A solution to prevent the Rainbow Table attack is called "salting." Salting is the process of adding a random string of characters to a password before hashing it. For example, if your

JOURNAL OF APPLIED SCIENCE AND SOCIAL SCIENCE
eISSN 2229-3113 pISSN 2229-3205
Volume 15 Issue 10, October 2025
Impact factor: 2019: 4.679 2020: 5.015 2021: 5.436, 2022: 5.242, 2023: 6.995, 2024 7.75

password is Admin@123 and the salt is $Do0Ap#1, the hash result of "Admin@123$Do0Ap #1" ('Admin@123') would be stored in the database instead of hashing it[8].

While this may be a commonly used password, adding a random salt turns it into a different string, making it harder for an attacker to crack the password (since he doesn't know the random salt).

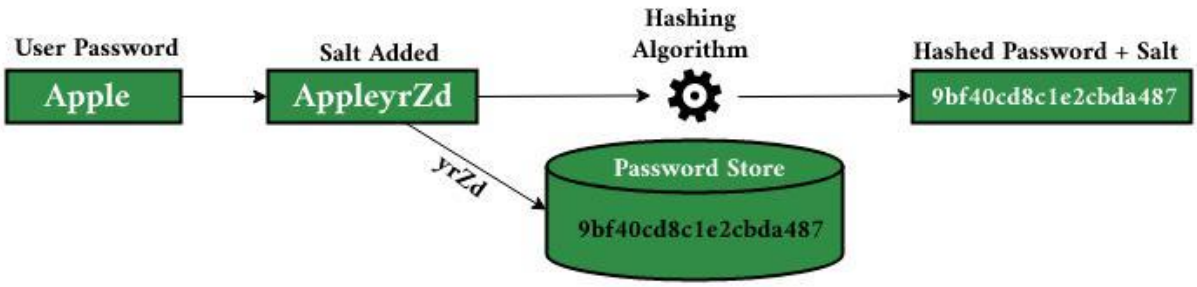

Figure 3. Mechanism for storing "salted" passwords.

There are two most common methods for implementing hashing on "salted" passwords.

The first method is static "salting", where we have a single "salt" for all passwords, which we combine with the password entered by the user, then hash it and store it in the database. Even for common passwords, the hash will be different because we have added a "salt" to the password[9].

But the question arises, what if an attacker has access to and reads this "salt"? The attacker can recreate a new password hash by combining the salt in all the passwords.

The second method is dynamic "salting". In dynamic "salting", we have a separate salt for each user. The "salt" is stored in the database for each user. When the user logs in, the dynamic "salt" for the user is retrieved and combined with the password entered by the user. It is then hashed and compared to the hashed password in the database.

The question remains what if an attacker has access to the password and the dynamic salt. Even if the attacker has this information, he will have to recalculate the Rainbow Table for each user.

Let's say the attacker has a Rainbow Table with 10 million entries. Now, to match a single row, he has to take the salt, combine the 10 million entries, and then calculate the hash for it. With this step, he can only crack one password. This step has to be repeated for each user's password. This is a very expensive and time-consuming operation, making it difficult for an attacker to crack the password.

Salting is a simple but effective way to prevent rainbow table attacks. Therefore, salting is

JOURNAL OF APPLIED SCIENCE AND SOCIAL SCIENCE
eISSN 2229-3113 pISSN 2229-3205
Volume 15 Issue 10, October 2025
Impact factor: 2019: 4.679 2020: 5.015 2021: 5.436, 2022: 5.242, 2023: 6.995, 2024 7.75

recommended as a best practice for storing passwords securely in databases.

Rainbow table attacks are part of modern threats, and their effectiveness depends on weak password practices and poor storage methods in systems. This article presents important conclusions: first, storing passwords in plain text or with fast hash algorithms compromises the security of the entire system and increases the possibility of attacks using rainbow tables; second, creating a pre-computed table when a unique and random salt is stored separately for each user is economically unprofitable, since the attacker would have to perform separate calculations for each record; third, however, a single salt (static salt) or short/predictive salts do not stop the attack, because if the attacker knows this salt, he can use the tables to adapt. Therefore, it is important to combine security layers: using slow and memory-intensive KDFs such as Argon2, bcrypt or scrypt makes offline attacks more expensive, while pepper (server secret key) provides additional protection even in the event of database theft. Additional measures — a strong password policy, regular password updates, two-factor authentication, and encryption — increase the level of protection for user data. Ultimately, a practical recommendation for organizations is to ensure security through several robust layers, rather than relying on just one method: proper configuration, constant monitoring and updating of modern KDF parameters, and rapid response and user notification in the event of an incident. This approach allows you to effectively protect systems from attacks that are as powerful as rainbow tables, but are effective under specific conditions.

References:

1. Oechslin, P. (2003). Making a Faster Cryptanalytic Time–Memory Trade-Off. In D. Boneh (Ed.), Advances in Cryptology — CRYPTO 2003, Lecture Notes in Computer Science, Vol. 2729, pp. 617–630. Springer. DOI: 10.1007/978-3-540-45146-4_36.

2. Provos, N., & Mazières, D. (1999). A Future-Adaptable Password Scheme. Proceedings of the 1999 USENIX Annual Technical Conference (USENIX 1999).

3. Percival, C. (2009). Stronger Key Derivation via Sequential Memory-Hard Functions (scrypt). Technical report / paper (scrypt), 2009.

4. Biryukov, A., Dinu, D., & Khovratovich, D. (2016). Argon2: New Generation of Memory-Hard Functions for Password Hashing and Other Applications. Proceedings of the 2016 IEEE European Symposium on Security and Privacy (EuroS&P 2016), pp. 292–302. DOI: 10.1109/EuroSP.2016.31.

5. Alex Biryukov, Daniel Dinu, Dmitry Khovratovich, Simon Josefsson. (2021). Argon2 Memory-Hard Function for Password Hashing and Proof-of-Work Applications. RFC 9106 (IETF).

6. NIST. (2017; with updates). NIST Special Publication 800-63B: Digital Identity Guidelines — Authentication and Lifecycle Management. National Institute of Standards and

JOURNAL OF APPLIED SCIENCE AND SOCIAL SCIENCE
eISSN 2229-3113 pISSN 2229-3205
Volume 15 Issue 10, October 2025
Impact factor: 2019: 4.679 2020: 5.015 2021: 5.436, 2022: 5.242, 2023: 6.995, 2024 7.75

Technology (SP 800-63B).

7. Hunt, T. (Troy Hunt). (2019). The 773 Million Record “Collection #1” Data Breach (blog post / analysis) — Have I Been Pwned (Pwned Passwords).

8. Akamai Security Intelligence Group. (2021). State of the Internet / SOTI — Reports on Credential Stuffing and Bot Activity (Akamai SOTI reports).

9. U.M. Ibragimov, B. Ergashev. Important aspects of collecting Windows operating system data for the pentest process. Conference: Collection of papers from the international scientific and practical conference on the topic "The role of digital technologies in the economy and education.". Uzbekistan(Samarqand). 2024. p.30-33.

10. U.M. Ibragimov. Effectiveness and efficiency of the PROMETHEUS system. XVI Saginovsky Readings. Integration of Education, Science and Production. Kazakhstan (Karaganda). 2024. p. 237-239.