**Заняття 5**

**Завдання 1**

Для забезпечення безпеки слід використовувати паролі з високою або дуже високою ентропією. Ентропія — це міра випадковості, яка відображає кількість інформації, яку містить пароль. Для визначення рівня ентропії паролів треба оцінити їх складність. Вища ентропія означає більшу складність для зламування пароля за допомогою брутфорс-атак або інших методів.

**Оцінка наданих паролів:**

1. **qwertyuiop**
   * це стандартний рядок, що складається з букв, розташованих поруч на клавіатурі. Такі паролі мають низьку ентропію, оскільки вони легко вгадуються;
   * **ентропія**: низька.
2. **sofPed-westag-jejzo1**
   * цей пароль містить великі та малі літери, цифри, а також роздільники ("-"). Це підвищує його складність порівняно з першим, але ще не є дуже складним;
   * **ентропія**: середня.
3. **f3Fg#Puu$EA1mfMx2**
   * цей пароль має комбінацію великих і малих літер, цифр, а також спеціальних символів (#, $, 1, 2) що значно підвищує складність пароля та його ентропію;
   * **ентропія**: висока.
4. **ТIMCfJDkKBRm9/zwcFbHhE6zaMcSxR7nke1mJKcVqXpvCzg69d7Mf2quanMoAfmPJXyqT4gyGpLoL1lTHoqmwVmaUwrpOPRecB8GAU17eUJJHiksv3qrqcVxhgpMkX/UlKaLdFSwFIr7cVoJmBqQ/buWzxJNCIo7qbtIi3fSi62NwMHh**
   * цей пароль є довгим і включає велику кількість різних символів: великі та малі літери, цифри, спеціальні символи та роздільники. Оскільки він має дуже високу кількість можливих комбінацій і є довгим, його ентропія дуже висока;
   * але довжина паролю може зіграти з ним злий жарт: він може не влізти у зарезервоване поле та бути обрізаний системою;
   * **ентропія**: дуже висока.

**Висновок:**

* **Паролі 1 та 2** мають низьку або середню ентропію і легко можуть бути зламані за допомогою методів, таких як словникові атаки.
* **Паролі 3 та 4** мають високу або дуже високу ентропію, що робить їх набагато складнішими для зламу.

Для ефективного хешування паролів у випадку веб-сервісів, де необхідно перевіряти мільйони паролів користувачів щодня, важливо вибрати хеш-функцію, яка забезпечить баланс між безпекою та ефективністю.

**Ключові вимоги:**

1. **Безпека**: Хеш-функція повинна бути стійкою до атак, таких як брутфорс або атаки з використанням радужних таблиць (rainbow tables).
2. **Швидкість**: Хешування повинно бути досить швидким, щоб не створювати затримок при автентифікації користувачів.
3. **Стійкість до паралельних атак**: Оскільки система буде мати тисячі або мільйони запитів, хеш-функція повинна бути стійкою до атак, що використовують апаратне прискорення, зокрема FPGA або ASIC.

**Можливі алгоритми хешування для паролів:**

1. **PBKDF2 (Password-Based Key Derivation Function 2)**:
   * Це криптографічний стандарт, який використовує багато раундів хешування для уповільнення обчислень.
   * **Переваги**: Захищає від атак методом брутфорс, оскільки можна налаштувати кількість раундів хешування, що робить його обчислення більш повільними і витратними в плані ресурсів.
   * **Вибір параметрів**: Рекомендується використовувати достатньо високу кількість раундів (наприклад, 100 000 або більше), щоб уповільнити процес хешування.
   * **Бібліотека**: В Python можна використовувати hashlib або pycryptodome для реалізації PBKDF2.
2. **Argon2**:
   * Це сучасний алгоритм хешування, який спроектовано спеціально для захисту паролів від атак.
   * **Переваги**: Має налаштування на використання пам'яті (ідеально для захисту від атак за допомогою спеціалізованого обладнання) і є надзвичайно безпечним.
   * **Вибір параметрів**: Argon2 має три варіанти: Argon2d (найбільш стійкий до атак на основі графічних процесорів), Argon2i (стійкий до атак на основі брутфорсу), та Argon2id (поєднує властивості обох). Рекомендується використовувати Argon2id.
   * **Бібліотека**: У Python можна використовувати бібліотеку argon2-cffi для реалізації.
3. **Bcrypt**:
   * Цей алгоритм працює схоже на PBKDF2 і використовує сіль та кількість раундів для підвищення складності хешування.
   * **Переваги**: Добре захищає від атак брутфорс через налаштування складності (т.е. кількість раундів). Широко підтримується у багатьох фреймворках.
   * **Вибір параметрів**: Вибір параметра "work factor" визначає кількість обчислень. Рекомендується вибирати високі значення (наприклад, 12 або більше).
   * **Бібліотека**: Для реалізації в Python можна використовувати бібліотеку bcrypt.

Я обираю **Argon2**, який є найсучаснішим і найбільш безпечним варіантом, особливо в умовах високого ризику атак. Цей алгоритм забезпечує високу стійкість до атак через брутфорс завдяки його параметрам конфігурації, які дозволяють налаштовувати час, пам'ять і паралелізм для кращого балансу між безпекою та продуктивністю.

У бібліотеці **argon2** для Python сіль генерується автоматично під час хешування пароля, тому не потрібно явно задавати сіль, коли викликається метод hash(password). Більш того, сіль зберігається разом з хешем, тому при перевірці пароля вона автоматично використовується для порівняння з оригінальним паролем. Сіль генерується для кожного пароля випадковим чином, і навіть якщо два користувачі мають однаковий пароль, їхні хеші будуть різними через різні солі.

Начало формы

Конец формы