Please check your attendance using Blackboard!

Lecture 2 – User Authentication

[COSE451] Software Security

Instructor: Seunghoon Woo

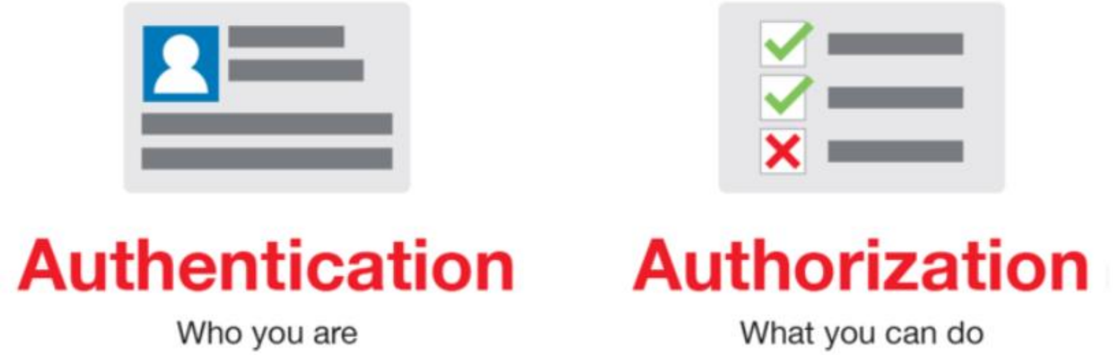
Spring 2024

Overview

- **Authentication**

User Authentication

- **RFC 4949***
  - “The process of verifying an identity claimed by or for a system entity”



* A Request for Comments (RFC): A document awaiting criticism, encompassing new research, innovations, and techniques

https://velog.io/@djaxornwkd12/%EC%9D%B8%EC%A6%9DAuthentication%EC%9D%B8%EA%B0%80Authorization%EB%9E%80-%EB%AC%B4%EC%97%87%EC%9D%B8%EA%B0%80

User Authentication

- **The four methods of authenticating user identity are based on**

Something the individual **knows**

- Password, PIN, answers to pre-arranged questions

Something the individual **possesses**

- Smartcard, electronic key card, physical key

Something the individual **is** (**static** biometrics)

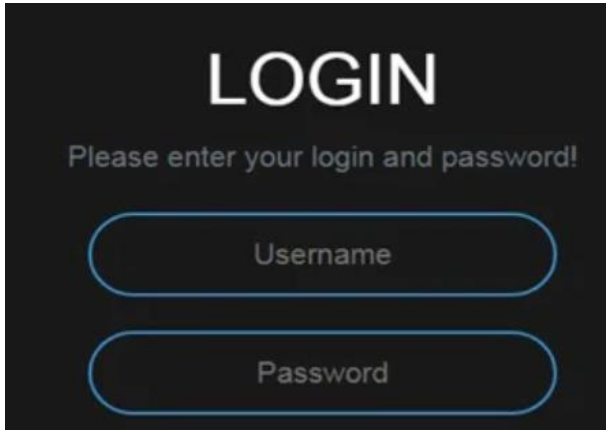
- Fingerprint, retina, face

Something the individual **does** (**dynamic** biometrics)

- Voice pattern, handwriting, typing rhythm

Something the individual knows

- **Password-based Authentication**
  - Widely used authentication method
    - User enter **ID** and **password**
    - System compares the **password** with the one stored for that specified login



LOGIN

Please enter your login and password!

Something the individual knows

- **Password-based Authentication**
  - Widely u
    - User
    - Syste gin

Is this always safe?

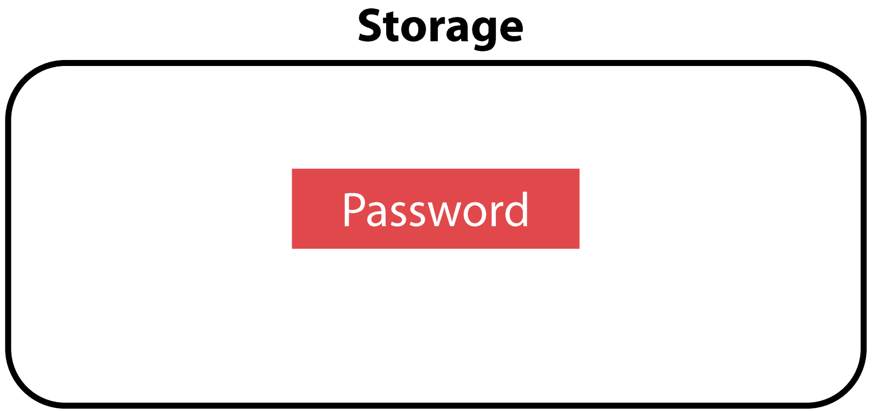
Please enter your login and password.

Something the individual knows ☹

- **Password vulnerabilities**
  1. Offline dictionary attack
  2. Specific account attack
  3. Popular password attack
  4. Password guessing against single user
  5. Workstation hijacking
  6. Exploiting user mistakes
  7. Exploiting multiple password use
  8. Electronic monitoring

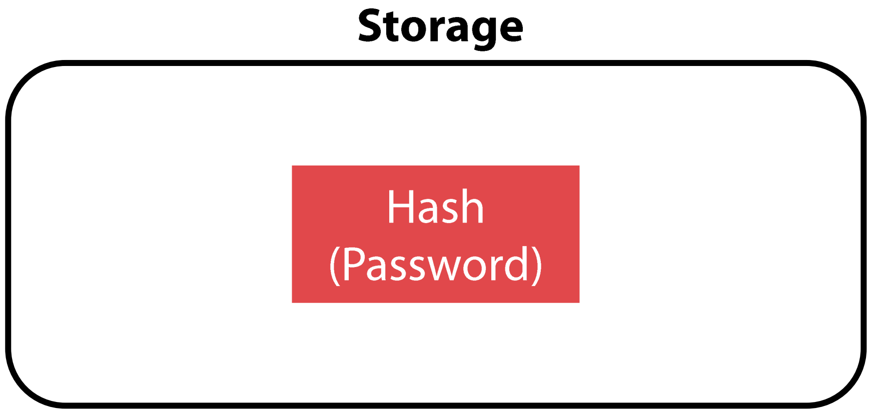
Something the individual knows

1. Offline dictionary attack
   - Simple password storage method (dangerous)



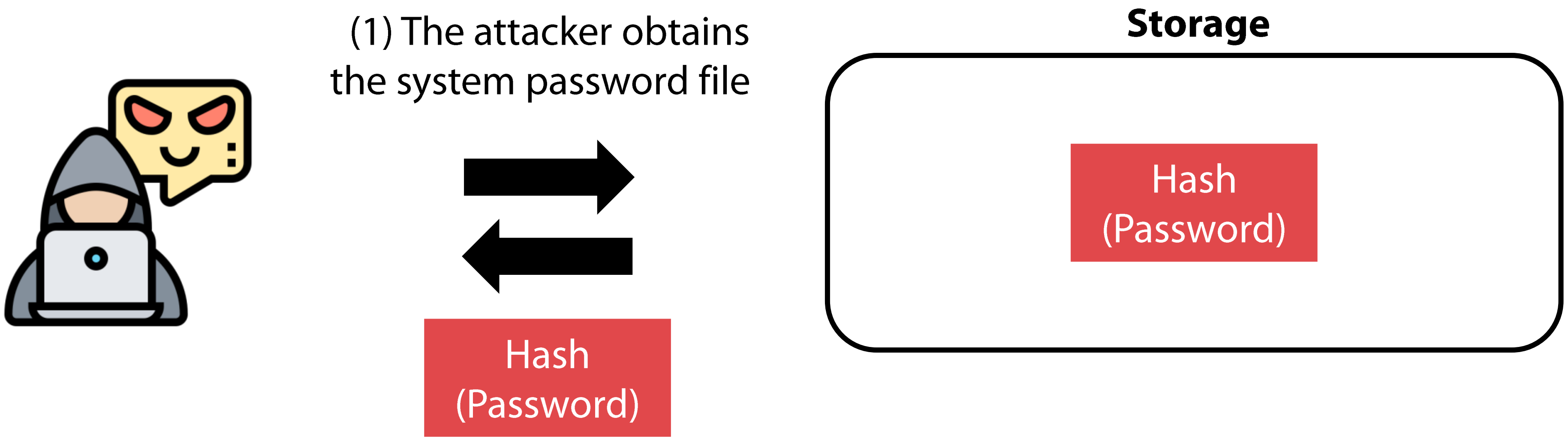
Something the individual knows

1. Offline dictionary attack
    - Using hash function



Something the individual knows

1. Offline dictionary attack
   - Using hash function: still dangerous



(1) The attacker obtains the system password file

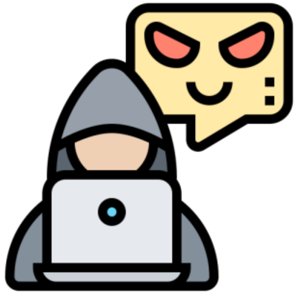
Storage

Hash (Password)

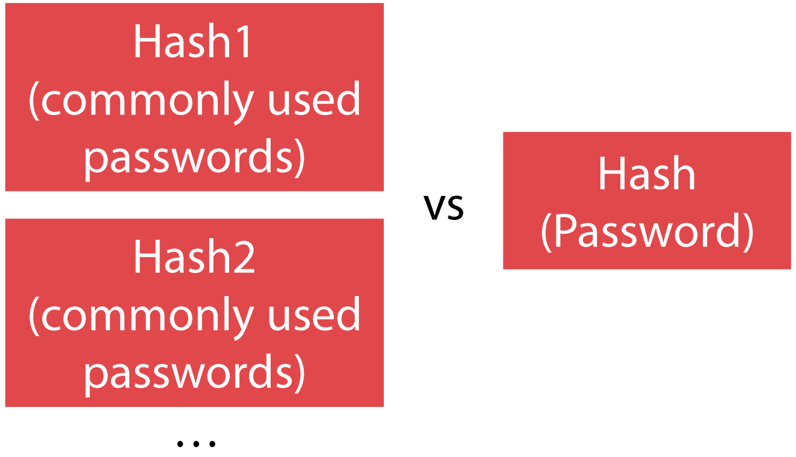
Hash (Password)

Something the individual knows

1. Offline dictionary attack
   - Using hash function: still dangerous



(2) The attacker compares the password hash against hashes of commonly used passwords



Hash1 (commonly used passwords)

Hash2 (commonly used passwords)

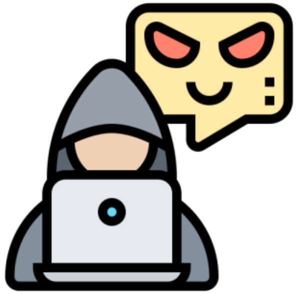
…

vs

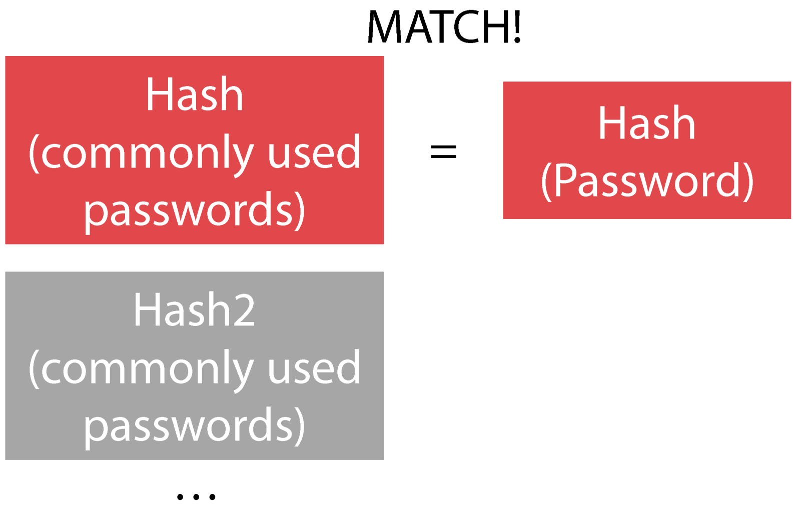
Hash (Password)

Something the individual knows

1. Offline dictionary attack
   - Using hash function: still dangerous



(3) If a match is found, the attacker can gain access by the corresponding ID/password



MATCH!

Hash (commonly used passwords) = Hash (Password)

Hash2 (commonly used passwords)

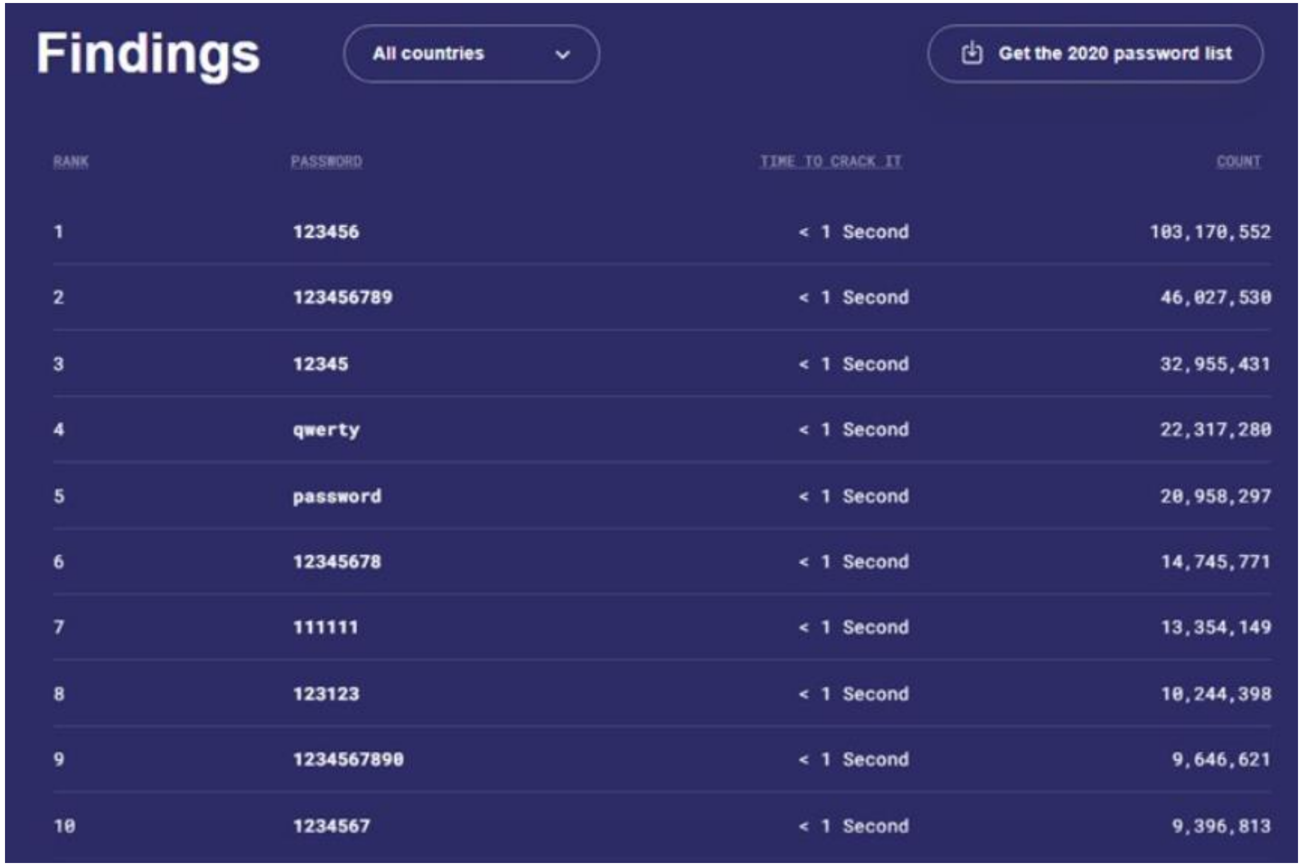
…

Something the individual knows

2. Specific account attack
   - Brute Force attack
     - An attacker submitting many passwords with the hope of eventually guessing correctly
   - The attacker targets a specific account and submits password guesses until the correct password is discovered
3. Popular password attack
   - Similar to the previous attack
   - Attacker uses a popular password and try it against a wide range of user IDs
     - E.g., 123456, 0000, etc.

Something the individual knows

믿기지 않겠지만 가장 흔한 패스워드는 여전히 123456 (EN: Believe it or not, the most common password is still 123456)



Findings

RANK	PASSWORD	TIME TO CRACK IT	COUNT
1	123456	< 1 Second	103,170,552
2	123456789	< 1 Second	46,027,530
3	12345	< 1 Second	32,955,431
4	qwerty	< 1 Second	22,317,280
5	password	< 1 Second	20,958,297
6	12345678	< 1 Second	14,745,771
7	111111	< 1 Second	13,354,149
8	123123	< 1 Second	10,244,398
9	1234567890	< 1 Second	9,646,621
10	1234567	< 1 Second	9,396,813

https://post.naver.com/viewer/postView.naver?volumeNo=32853720&memberNo=967

Something the individual knows

4. Password guessing against single user
   - The attacker first attempts to gain knowledge about the user
   - The attacker then uses that knowledge to guess the password
     - E.g., Birthday, phone number, etc.
5. Workstation hijacking
   - The case where an attacker can directly use the user's device
   - The attacker waits until a logged-in workstation is unattended

Something the individual knows

6. Exploiting user mistakes
   - A user may write down a preconfigured password (difficult to remember)
     - E.g., Storing their passwords into the "password.txt" file
   - User may share a password (share files with colleague)
7. Exploiting multiple password use
   - When a user uses a single password across multiple networks, the burden erased by an attacker's attack becomes smaller, making attacks easier

Something the individual knows

8. Electronic monitoring
   - If a password is communicated across a network to log on to a remote system, it is vulnerable to eavesdropping

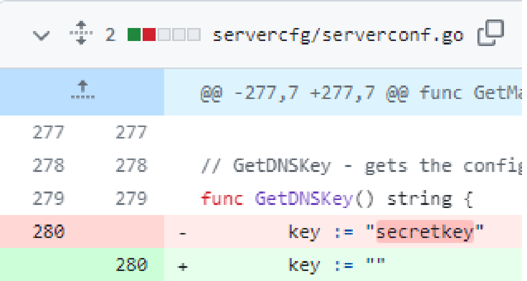
https://www.shiksha.com/online-courses/articles/eavesdropping-how-to-prevent-it/

Something the individual knows

- **Password-related vulnerabilities**
  - CWE (Common Weakness Enumeration)
    - Approximately 1,000 CWEs are defined
  - Several CWEs related to passwords
    - CWE-259: Use of Hard-coded Password
    - CWE-326: Inadequate Encryption of Passwords
    - CWE-521: Weak Password Requirements
    - CWE-522: Insufficiently Protected Credentials
    - CWE-798: Use of Hard-coded Credentials

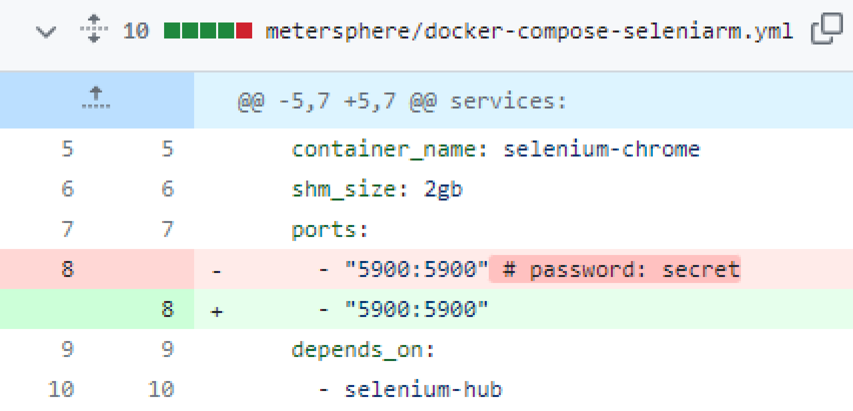
Something the individual knows

- **Password-related vulnerabilities (CWE-798)**



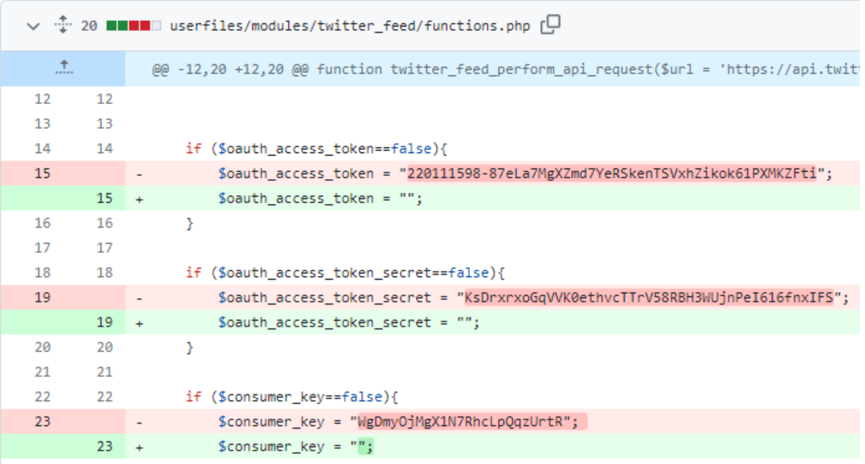
```
2 servercfg/serverconf.go

@@ -277,7 +277,7 @@ func GetMa

    // GetDNSKey - gets the confi
    func GetDNSKey() string {
-           key := "secretkey"
+           key := ""
```

CVE-2023-32077



```
10 metersphere/docker-compose-seleniarm.yml

@@ -5,7 +5,7 @@ services:
        container_name: selenium-chrome
        shm_size: 2gb
        ports:
-         - "5900:5900" # password: secret
+         - "5900:5900"
        depends_on:
          - selenium-hub
```

CVE-2023-41878

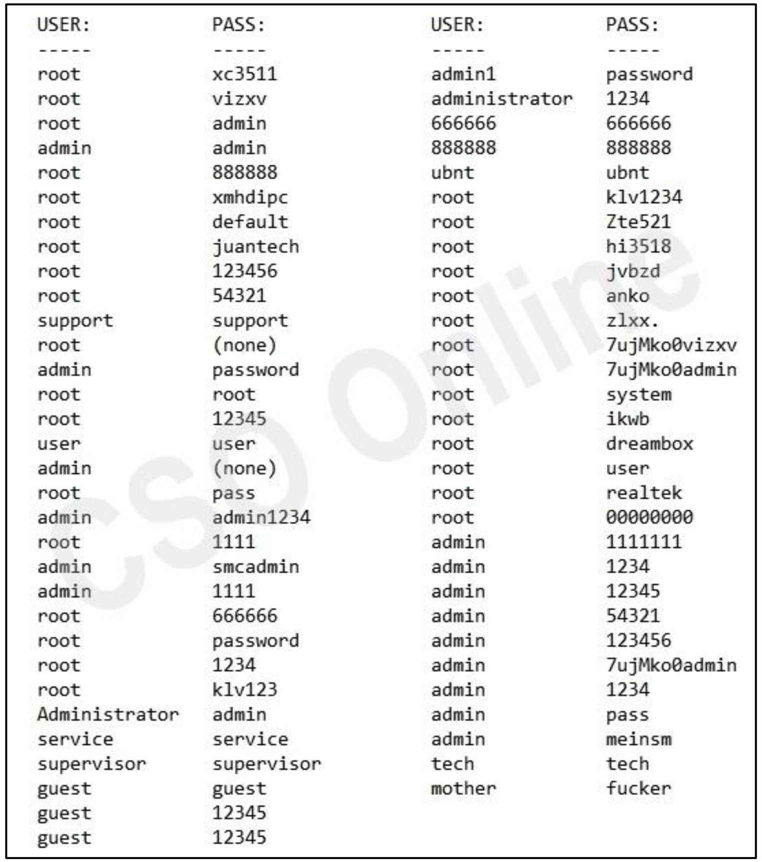


```
20 userfiles/modules/twitter_feed/functions.php

@@ -12,20 +12,20 @@ function twitter_feed_perform_api_request($url = 'https://api.twit


        if ($oauth_access_token==false){
-           $oauth_access_token = "220111598-87eLa7MgXZmd7YeRSkenTSVxhZikok61PXMKZFti";
+           $oauth_access_token = "";
        }

        if ($oauth_access_token_secret==false){
-           $oauth_access_token_secret = "KsDrxrxoGqVVK0ethvcTTrV58RBH3WUjnPeI616fnxIFS";
+           $oauth_access_token_secret = "";
        }

        if ($consumer_key==false){
-           $consumer_key = "WgDmyOjMgX1N7RhcLpQqzUrtR";
+           $consumer_key = "";
```

CVE-2023-5318

Something the individual knows

- **Mirai botnet (2016)**
  - Turns Internet of Things (IoT) devices into zombies and allows hackers to arbitrarily control them on the network
  - Attacked 400K+ IoT devices by using the 60 saved account/password combinations
    - Exploiting the fact that users do not often change their "default passwords"



USER:	PASS:	USER:	PASS:
-----	-----	-----	-----
root	xc3511	admin1	password
root	vizxv	administrator	1234
root	admin	666666	666666
admin	admin	888888	888888
root	888888	ubnt	ubnt
root	xmhdipc	root	klv1234
root	default	root	Zte521
root	juantech	root	hi3518
root	123456	root	jvbzd
root	54321	root	anko
support	support	root	zlxx.
root	(none)	root	7ujMko0vizxv
admin	password	root	7ujMko0admin
root	root	root	system
root	12345	root	ikwb
user	user	root	dreambox
admin	(none)	root	user
root	pass	root	realtek
admin	admin1234	root	00000000
root	1111	admin	1111111
admin	smcadmin	admin	1234
admin	1111	admin	12345
root	666666	admin	54321
root	password	admin	123456
root	1234	admin	7ujMko0admin
root	klv123	admin	1234
Administrator	admin	admin	pass
service	service	admin	meinsm
supervisor	supervisor	tech	tech
guest	guest	mother	fucker
guest	12345		
guest	12345		

https://www.itworld.co.kr/news/101424

Something the individual knows

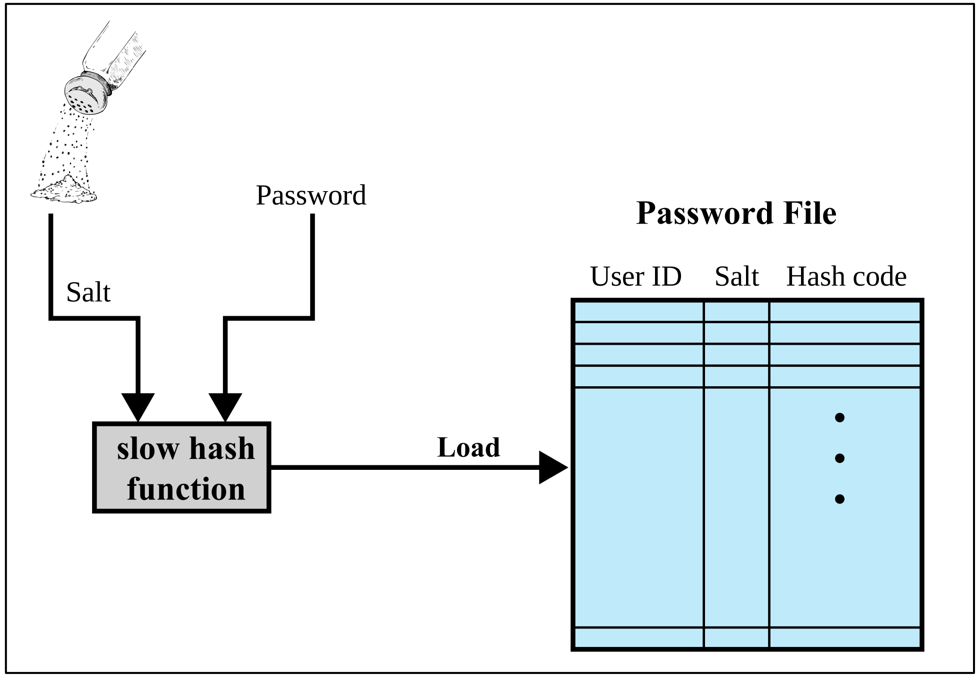
- **How can we respond to password attacks?**
  - Recently, various password policies are applied
    - Locking out access after a number of failed login attempts
    - Prohibiting overly common strings as passwords
    - Avoid storing passwords in local files or source code
    - Minimum length setting
    - Uppercase and lowercase rules
    - Special character rules

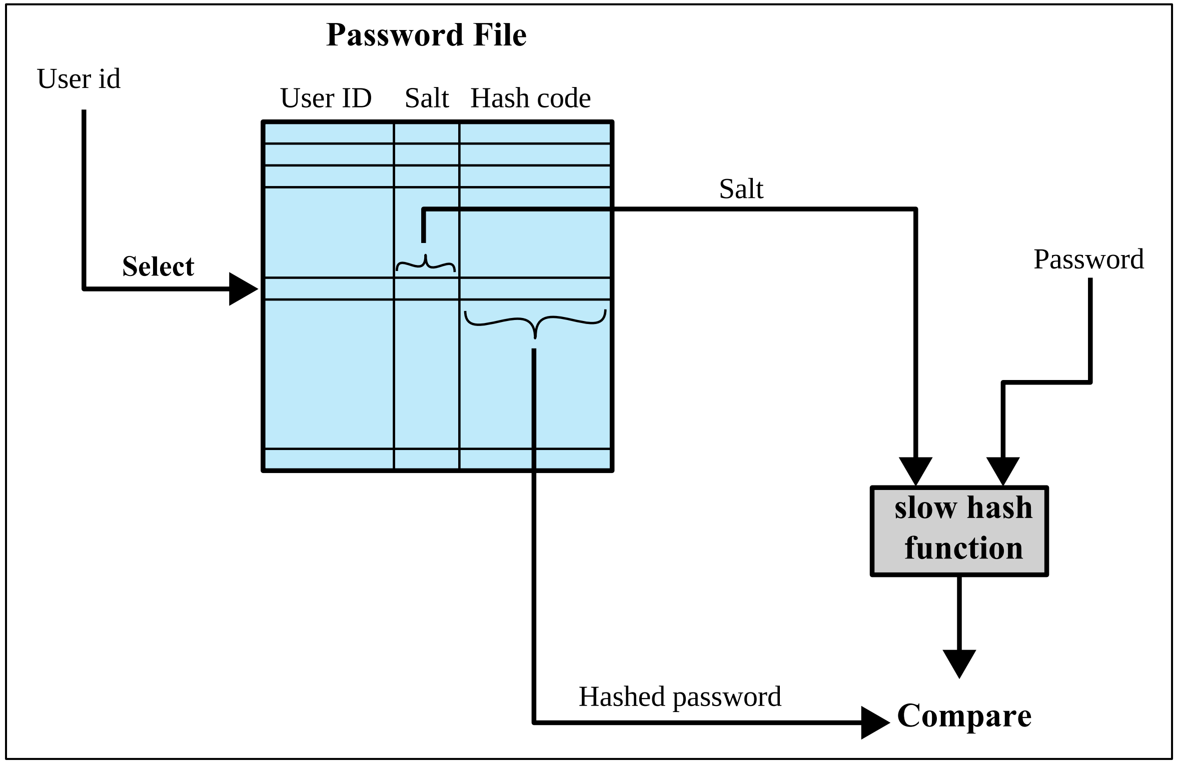
Something the individual knows

- **Enhanced hash-based password management (in UNIX system)**
  - Problem of the previous hash-based approach
    - Same inputs => same hash values
      - Duplicate passwords from being visible in the password file
    - Attackers can relatively easily predict stored passwords

Something the individual knows

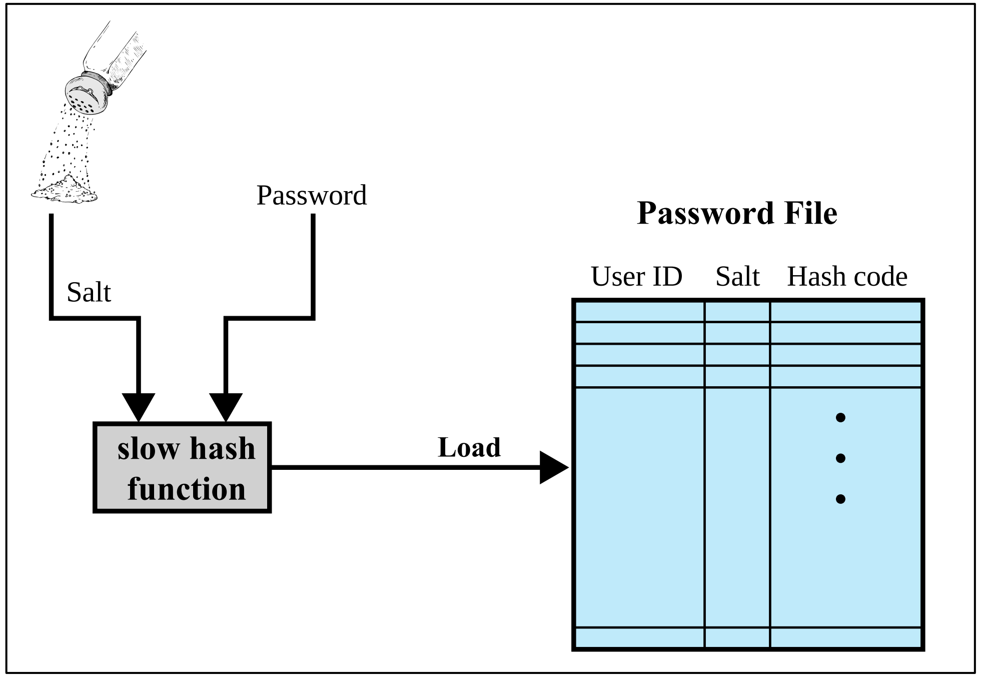
- **Enhanced hash-based password management**
  - Using the concept of “SALT”





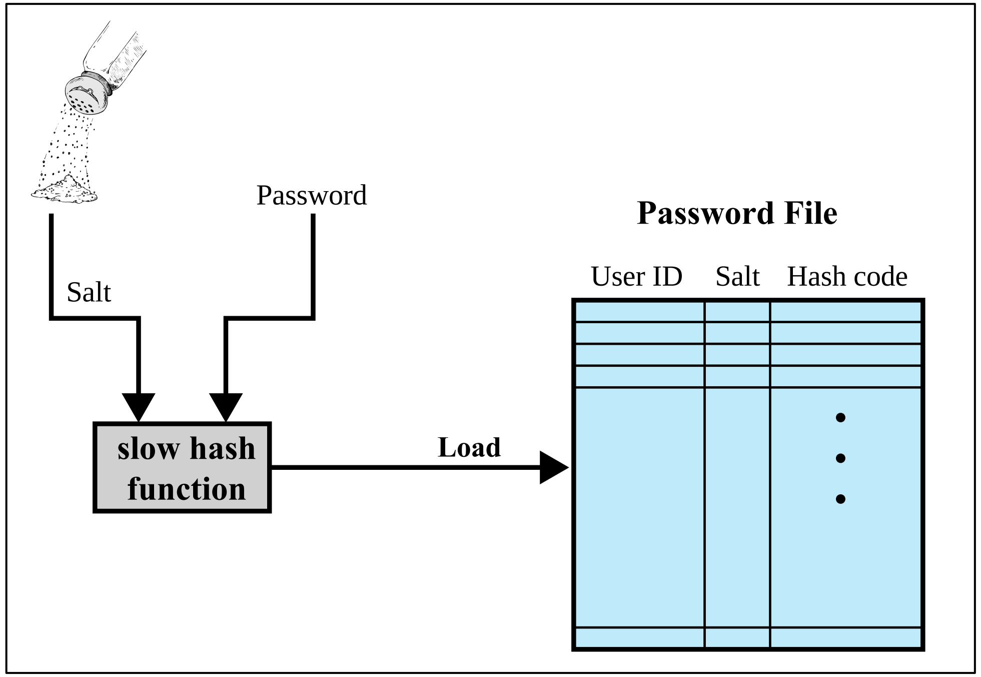
Something the individual knows

- **Enhanced hash-based password management**
  - Using the concept of "SALT"
    - Related to the time
    - Pseudorandom or random number



Something the individual knows

- **Enhanced hash-based password management**
  - Using the concept of "SALT"
    - Related to the time
    - Pseudorandom or random number
  - Greatly increases the difficulty of offline dictionary attacks
    - 128-bit or higher salt is used



Something the individual knows

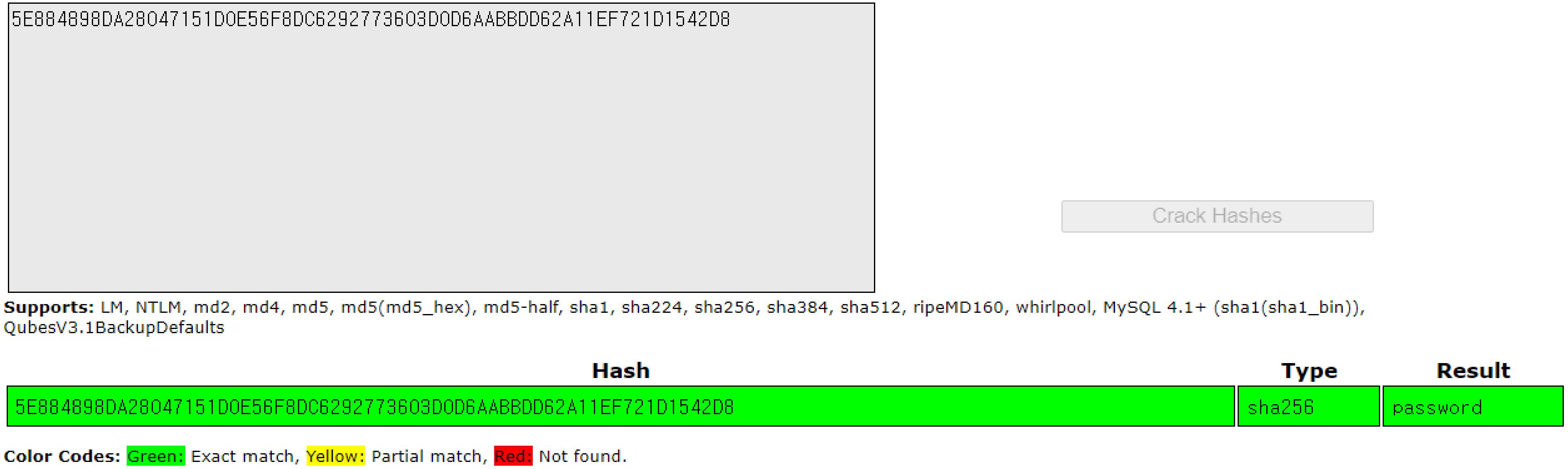
- **Rainbow table attacks**
  - Precomputing potential hash values
  - For each possible password, generate the hash values associated with each possible salt value
  - Create an incredibly large table in advance and use it for attacks
    - An 8-character combination of lowercase letters and numbers is approximately 328GB
    - An 8-character ASCII code combination is approximately 47,225,249,742 TB

Something the individual knows

- **Rainbow table attacks**
  - How can we counter this attack?
    - One of the most efficient and simple way to deal with this is using a sufficiently large salt value and a sufficiently large hash length
    - Using salt multiple times
    - Hashing multiple times

Something the individual knows

- **Rainbow table attacks**
  - Example using the hash of "password"



```
5E884898DA28047151D0E56F8DC6292773603D0D6AABBDD62A11EF721D1542D8
```

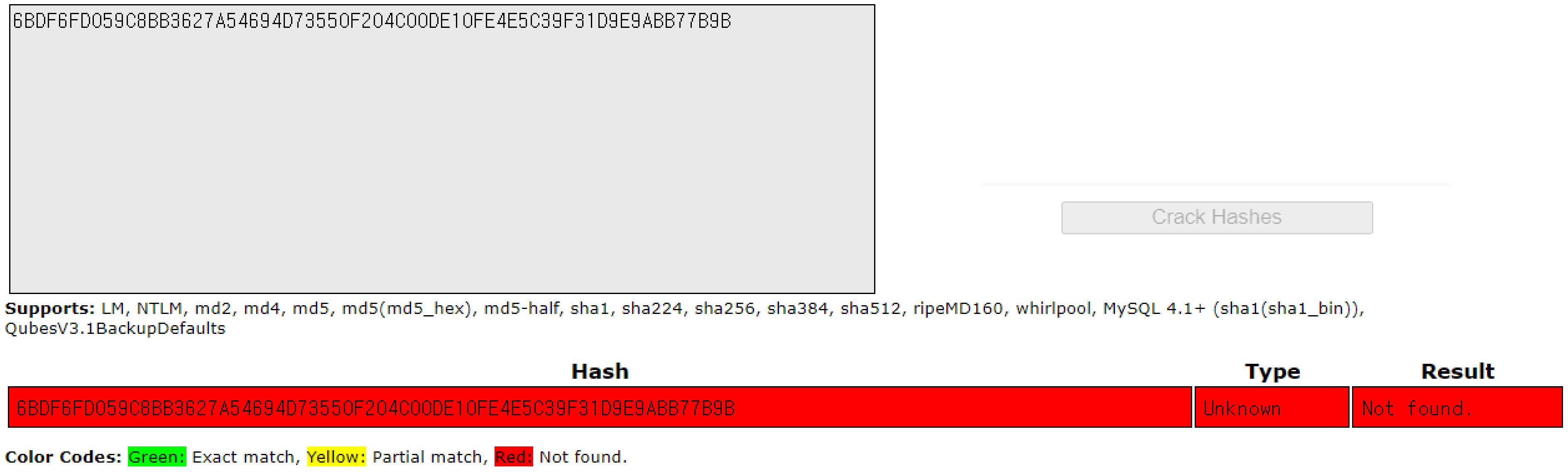
Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
5E884898DA28047151D0E56F8DC6292773603D0D6AABBDD62A11EF721D1542D8	sha256	password

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Something the individual knows

- **Rainbow table attacks**
  - Example using the hash of "passwordthisismysalt"



```
6BDF6FD059C8BB3627A54694D73550F204C00DE10FE4E5C39F31D9E9ABB77B9B
```

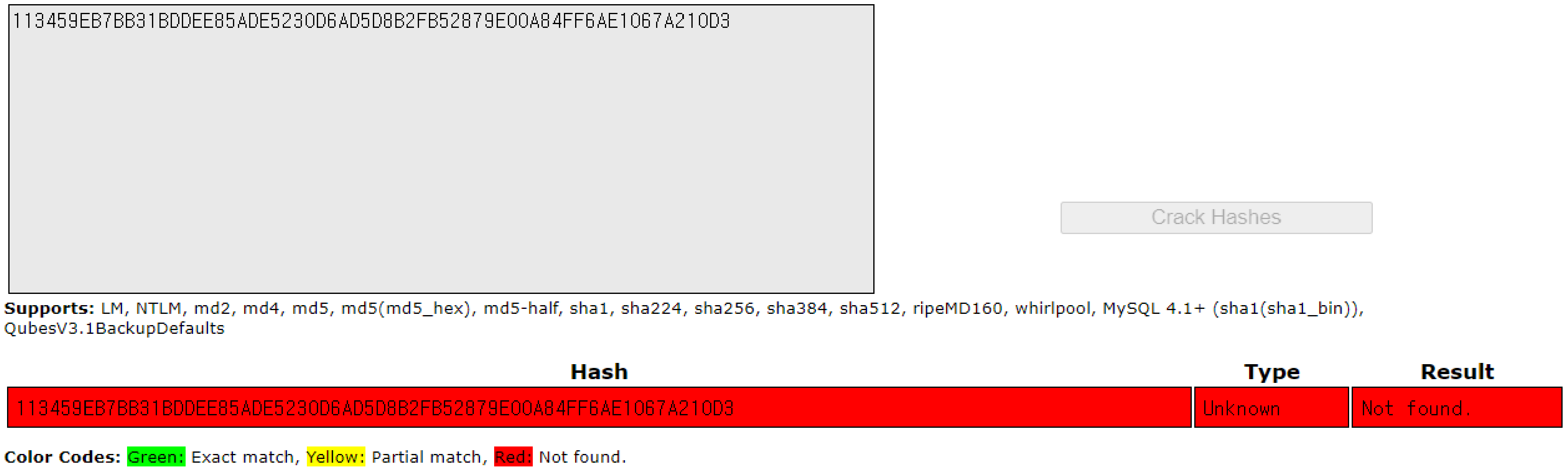
Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
6BDF6FD059C8BB3627A54694D73550F204C00DE10FE4E5C39F31D9E9ABB77B9B	Unknown	Not found.

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Something the individual knows

- **Rainbow table attacks**
  - Example using the hash of the hashed value of "password"



```
113459EB7BB31BDDEE85ADE5230D6AD5D8B2FB52879E00A84FF6AE1067A210D3
```

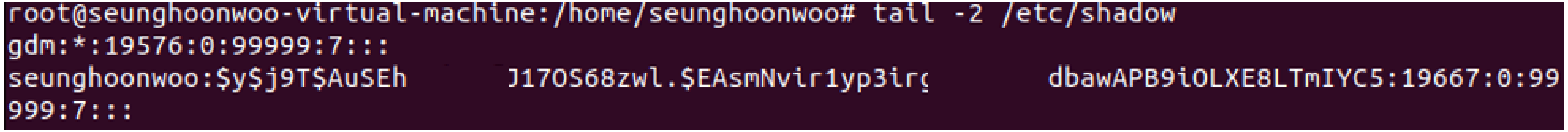
Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
113459EB7BB31BDDEE85ADE5230D6AD5D8B2FB52879E00A84FF6AE1067A210D3	Unknown	Not found.

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

Something the individual knows

- **Password file access control**
  - Can block guessing attacks by denying access to encrypted passwords
    - Make available only to privileged users
  - Often, the hashed passwords are kept in a separate file from the user IDs, referred to as a shadow password file
    - Special attention is paid to making the shadow password file protected from unauthorized access



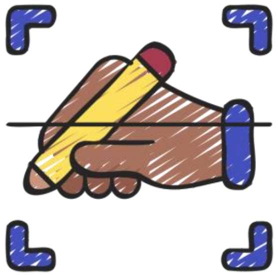
```
root@seunghoonwoo-virtual-machine:/home/seunghoonwoo# tail -2 /etc/shadow
gdm:*:19576:0:99999:7:::
seunghoonwoo:$y$j9T$AuSEh[redacted]J17OS68zwl.$EAsmNvir1yp3irç[redacted]dbawAPB9iOLXE8LTmIYC5:19667:0:99
999:7:::
```

Something the individual possesses

- **Also called "Token-based authentication"**
  - The most common is the magnetic stripe card
    - Smart cards, memory cards, physical key, etc.
  - Can store but do not process data
  - Provides greater security when combined with a password or PIN
  - Drawbacks
    - Requires a special reader
    - Loss of token

Something the individual is/does

- **Also called "Biometric authentication"**
  - Authenticate an individual based on unique physical characteristics
    - Facial characteristics
    - Fingerprints
    - Iris
    - Signature
    - Voice
  - Using pattern recognition
  - Is technically complex and expensive when compared to passwords and tokens



Something the individual is/does

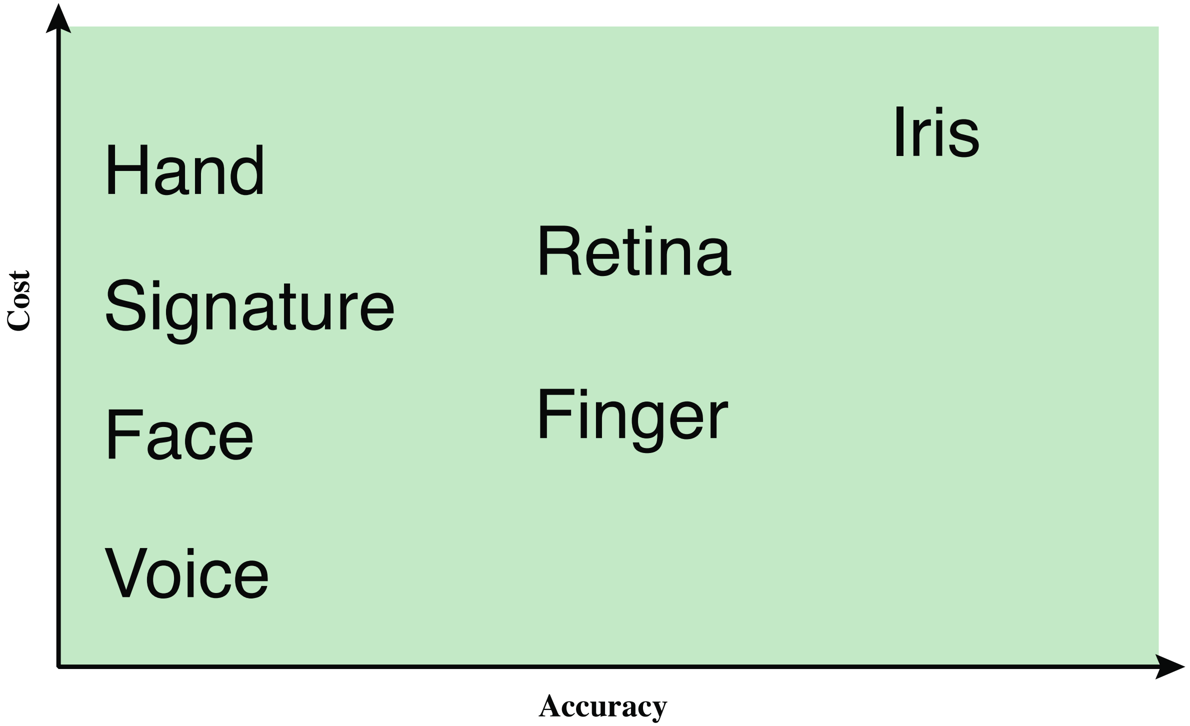


Figure 3.7 Cost Versus Accuracy of Various Biometric Characteristics in User Authentication Schemes.

Next Lecture

- **Access controls**