

Broken access control

Token vulnerabilities

BROKEN ACCESS CONTROL: JWT TOKENS

How do vulnerabilities with (self-contained) JWT tokens arise?

Encoded

PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG91IiwiaWF0IjoxNTE2MzkwMjIu4Lqneuxhy13yAk7-bE_KIXA3QkrDZEKppGe_gmueJ3I
```

Decoded

EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{  "alg": "HS256",  "typ": "JWT"}
```

PAYLOAD: DATA

```
{  "sub": "1234567890",  "name": "John Doe",  "iat": 1516239022,  "exp": 1516240000}
```

VERIFY SIGNATURE

```
HMACSHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
    
)  secret  base64  encoded
```

BROKEN ACCESS CONTROL: JWT TOKENS

How do vulnerabilities with (self-contained) JWT tokens arise?

- Cannot be revoked → risk for compromise
- Insufficient signature validation:
 - Accepting tokens with no signature
 - Accepting arbitrary signatures
- Brute forcing secret key
- JWT header parameter injection

BROKEN ACCESS CONTROL: JWT TOKENS

Can typically not be revoked: server contains no state

→ long expiry means
compromised token
gives long unrestricted
access

Encoded PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG91IiwiaWF0IjoxNTE2MzkwMjIuNjE1MTYyNDAwMDB9.4Lqneuxhy13yAk7-bE_KIXA3QkrDZEKppGe_gmueJ3I
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE
<pre>{ "alg": "HS256", "typ": "JWT" }</pre>
PAYLOAD: DATA
<pre>{ "sub": "1234567890", "name": "John Doe", "iat": 1516239022, "exp": 1516240000 }</pre>
VERIFY SIGNATURE
<pre>HMACSHA256(base64UrlEncode(header) + "." + base64UrlEncode(payload), <input type="text"/>) <input type="checkbox"/> secret base64 encoded</pre>

BROKEN ACCESS CONTROL: JWT TOKENS

Insufficient signature validation: accepting arbitrary signatures

- Straight up forgetting to validate header
- Accidentally doing 'decode()' instead of 'verify()'

BROKEN ACCESS CONTROL: JWT TOKENS

Insufficient signature validation: Accepting tokens with no signature

```
{  
  ...  
  "alg": "none" → unsecured JWT,  
  ...  
}
```

Means the server will accept any claim, as it cannot be validated

BROKEN ACCESS CONTROL: JWT TOKENS

Brute forcing secret key:

Developers might forget to change test / default placeholder key



Crackable using e.g. hashcat and wordlist of well-known secrets
(hashcat creates a signature of JWT and compares with given signature)

DEMO: CRACKING JWT KEY

- Using hashcat, we can bruteforce or use wordlists of common secret keys

BROKEN ACCESS CONTROL: JWT TOKENS

JWT Header injection:

- *jwk* (JSON web key): embedded public key in JWT
- *jku* (JSON web key set URL): URL from which server can fetch public key
- *kid* (Key ID): which key to choose in case there are multiple

BROKEN ACCESS CONTROL: JWT TOKENS

JWT Header injection:

- *jwk:*
- *jku:*
- *kid:*

BROKEN ACCESS CONTROL: JWT TOKENS

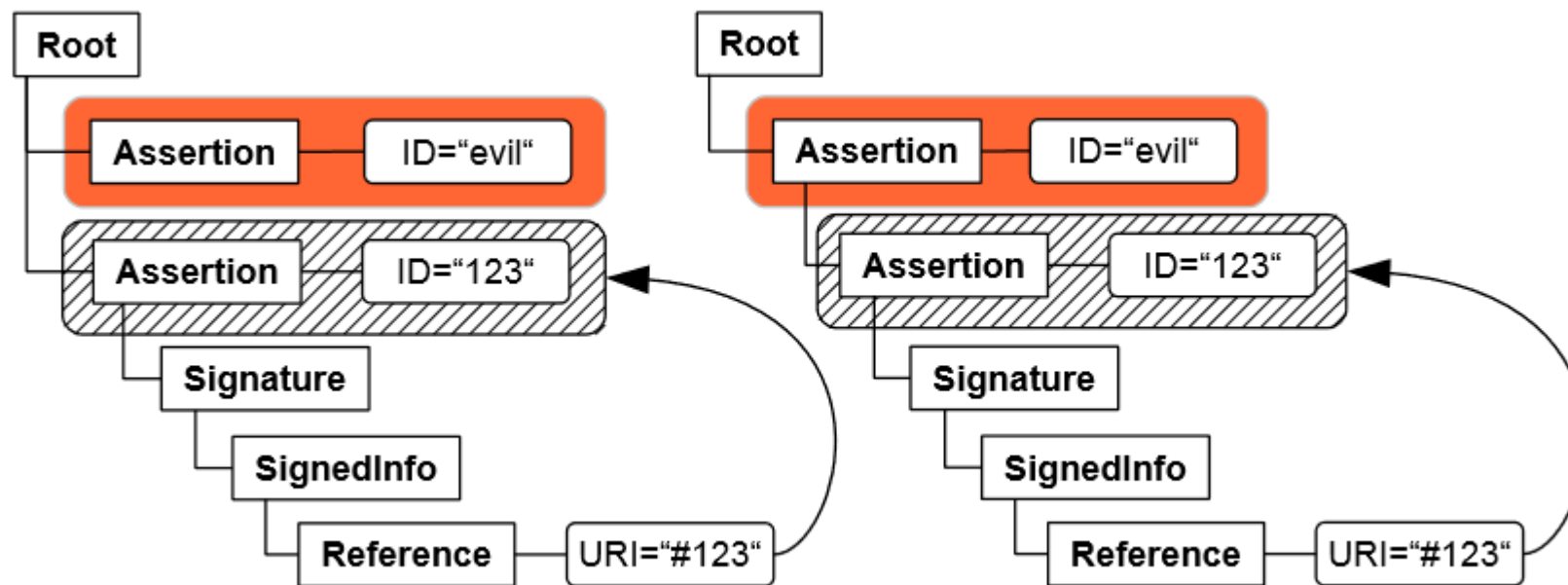
JWT Header injection:

- *jwt*: should only accept whitelisted keys, otherwise one can sign with an arbitrary key
- *kid*: similarly, should only accept trusted domains
- *kid*: No standard, sometimes points to a file. In case of symmetric files, could point to e.g. `/dev/null` (equal to signing the header with empty string)

BROKEN ACCESS CONTROL: SAML TOKENS (ASSERTIONS)

Common vulnerabilities:

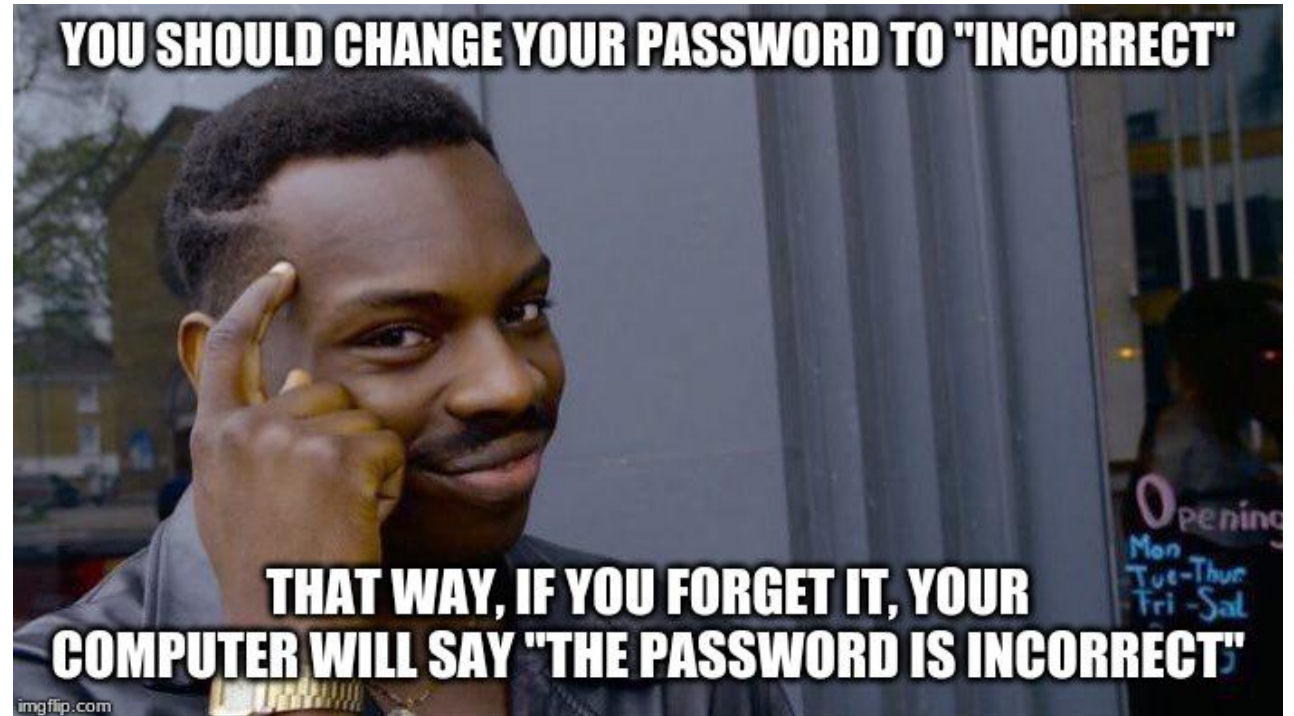
- Only validating signature when present
- XML Signature Wrapping (XSW) attack:



Passwords

AUTHENTICATION

- Three potential methods
 - Something you know



AUTHENTICATION

- Three potential methods
 - Something you know
 - Something you have



AUTHENTICATION

- Three potential methods
 - Something you know
 - Something you have
 - Something you are



SOMETHING YOU KNOW: PASSWORD

;-

Home Notify me Domain search Who's been pwned Passwords API About Donate

';-have i been pwned?

Check if you have an account that has been compromised in a data breach

info@example.com pwned?

Oh no — pwned!

Pwned on 23 breached sites and found 21 pastes (subscribe to search sensitive breaches)

3 Steps to better security [Start using 1Password.com](#)

Step 1 Protect yourself using 1Password to generate and save strong passwords for each website.

Step 2 Enable 2 factor authentication and store the codes inside your 1Password account.

Step 3 Subscribe to notifications for any other breaches. Then just change that unique password.

Why 1Password?

PASSWORDS

What do you do if you forget your password?

Forcing complex passwords leads to frustration for the user

What's your password?

PASSWORD RULES



HOW PASSWORD
LENGTH WINS
THE INTERNET

Passwords 102

PASSWORD RULES

when the hint for your mum's password is
"my favourite" and your sibling's name works



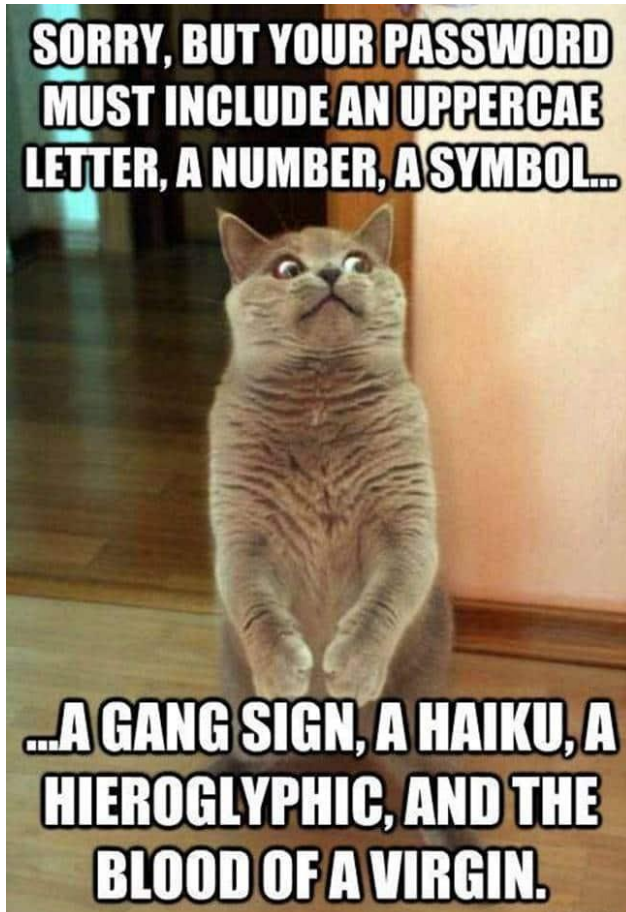
PASSWORD RULES

- Lists of known passwords
 - NordPass
 - SplashData
 - Keeper
 - National Cyber Security Centre
 - https://en.wikipedia.org/wiki/Wikipedia:10,000_most_common_passwords

PASSWORD RULES



PASSWORD RULES



PASSWORD RULES

- ✓ shall be at least 8 characters in length (if chosen by subscriber)
- ✓ do not store 'hints'
- ✓ check passwords against known lists
- × no more periodic changes
- × no more complexity requirements

PASSWORD

For example, a user that might have chosen “password” as their password would be relatively likely to choose “Password1” if required to include an uppercase letter and a number, or “Password1!” if a symbol is also required.

Users also express frustration when attempts to create complex passwords are rejected by online services.

Highly complex memorized secrets introduce a new potential vulnerability: they are less likely to be memorable, and it is more likely that they will be written down or stored electronically in an unsafe manner.

PASSWORD MANAGER



SOMETHING YOU HAVE



SOMETHING YOU ARE

WINDOWS HELLO



**FOR WHEN YOU
LOVE NO ONE
MORE THAN YOURSELF**

imgflip.com

**APPLE INTRODUCES FINGERPRINT
SCANNING**

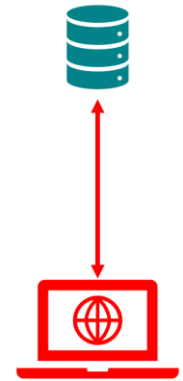


**WORLD'S LARGEST NAME-TO-FINGERPRINT
DATABASE NOW AVAILABLE WITHOUT EVEN TRYING**

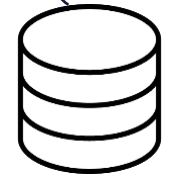
MULTI FACTOR AUTHENTICATION

- Combination of 2 or more methods of authentication
- Makes pretending more difficult

ONLINE BRUTEFORCING

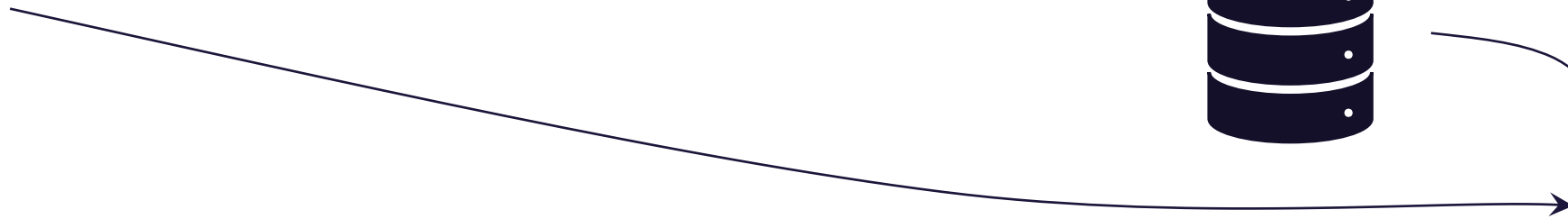


Evil user/browser

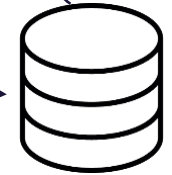


- ✓ Log failed login attempts and send them to your SIEM
- ✓ Ask for a captcha after X number of failed attempts
- ✓ Block account after X + Y number of failed attempts
- ✓ Simply use a good password

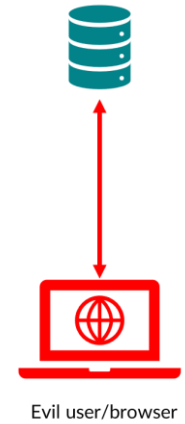
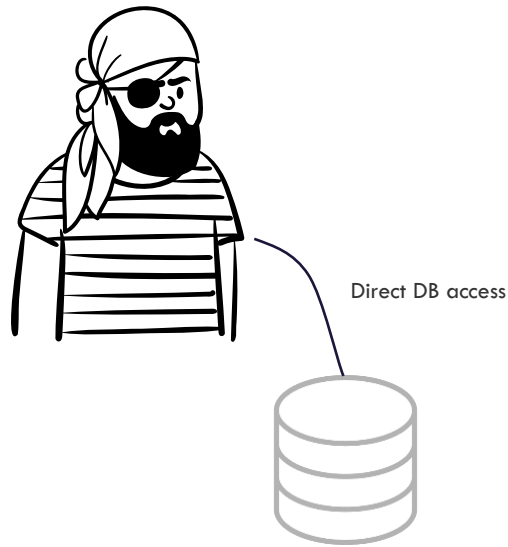
OFFLINE BRUTEFORCING



Evil user/browser



OFFLINE BRUTEFORCING



- ~~✗ Log failed login attempts and send them to your SIEM~~
- ~~✗ Ask for a captcha after X number of failed attempts~~
- ~~✗ Block account after X + Y number of failed attempts~~
- ✓ Secure Password Storage

PASSWORD NIST GUIDELINES

SP 800-63A
Enrollment &
identity proofing

SP 800-63B
Authentication
and lifecycle mgmt

SP 800-63C
Federation and
assertions

PASSWORD STORAGE

HOW WEBSITES STORE YOUR PASSWORD

		
WHAT YOU HOPE THEY DO	WHAT YOU SUSPECT THEY DO	WHAT ATTACKERS HOPE THEY DO
		
WHAT THE SECURITY CONSULTANT SUGGESTED THEY DO	A REASONABLE LEVEL OF SECURITY	WHAT THEY ACTUALLY DO

SECURE PASSWORD STORAGE: SUMMARY

- Plain-text: ...
 - **bad**
- (Cryptographic) Hash: e.g. SHA256
 - **still bad: rainbow tables**
- Hash + salt: e.g. SHA256 + CSPRNG
 - still bad: bruteforcing
- Slow hash + salt: e.g. argon2 (includes logic to generate salt)
 - **standard**

SECURE PASSWORD STORAGE: ARGON2

`$argon2id$v=19$m=195312,t=20,p=1$YCICO9c2tXQj+GHX16YKlg$TxX3dHWIwQc8MTRfDIreybV0E2cWmnpaX7z9XqYDaLE`

argon2
variant

argon2
revision

memory
consumption

iteration count
and
parallalization

Random salt

Hashed password

DEMO

- Demo cracking passwords with hashcat