Cryptography and Certificates

Review

- How is a hash different from both symmetric and asymmetric encryption?
- How can a hash be used for integrity?
- How can a hash be used for confidentiality?
- In order to reset someone's password, we should store it in the database so we can email it to them (true/false)
- Which hash function is stronger? MD5 or SHA-256?
- In the Java example, what was one of the issues in creating our key?

Exercise

- Review the following guide:
  - https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html
- What does a salt protect against?
- How is a salt different from a pepper?
- Which of the following are acceptable ways of storing a password?
  - Argon2id
  - PBKDF2
  - md5

Rainbow Tables

- Used to attack passwords stored in a database or a system
- Pre-calculated hashes with certain characters
- If you find a hash, you can compare the hash in the table to find the original string
- Can get very large
- Using a salt (and good password policies, to come later) helps defend agains this attack

Java Example

- Instead of asking the user for a textual key, we can use PBKDF2
- Let's take a look!

Salt and IVs

- Salt is used to add randomness to a hash
  - Protects against an attack like a rainbow table
  - Salts should be stored with encrypted data
    - You need to use it to derive the key to decrypt data
    - Typically prepended
- Initialization Vector (IVs)
  - Add randomness to symmetric encryption
  - This must be stored as well along encrypted data
    - Encode like a salt

Salt Use on Linux

- Run the following:
  - cat /etc/passwd
  - Find your username, what do you see?
- Run the following:
  - sudo cat /etc/shadow
  - Find your username, what do you see?
  - briankrupp:yj9T$.Sn.07dDYV52SIatfld9G1$tda3czG1HOt/7wVz0ounBZyriXeB3Hi4NImMNtNTnV8:19944:0:99999:7:::
  - username:idsalt$hash
    - id → y is yes crypt
    - salt → The salt added to the hash
    - hash → Hashed password

Certificates

- Fundamentally, they do 2 main things:
  - Provide a mechanism for sharing public keys
  - Provide a mechanism for verifying they are who they say they are (as long as you trust who says they are who they say they are)

Exercise

- Visit the following website: https://duckduckgo.com
  - Firefox works well
- Click on the lock icon
  - What is the common name of the certificate?
  - Who issues the certificate?
  - How long is the certificate valid?
  - How large is the public key?

Certificate Attributes

- X.509 - Standard for formatting a certificate, defines attributes:
- Version - X.509 version
- Public Key - The public key of the holder
- Serial Number - Unique identifier of the certificate
- DN - The distinguished name of the holder
- Validity - When the cert was issued and when does it expire
- Digital Signature - The CA's signature on the certificate
- Signature Algorithm - The algorithm Used
- Other optional fields

Certificate Authorities

- Verisign
- Digicert
- Comodo
- and many more ...
- Let's Encrypt
  - Do a quick search on Let's Encrypt
  - What is it? Who sponsors it?
- Self-sign or Private CAs
  - Sometimes used for test environments

How Certificates are Issued

- Create a certificate signing request (contains public key and information)
- Certificate authority does the following:
  - Verifies it is indeed the identify requesting the certificate
  - Generate a certificate that they sign with a private key
  - Issue the certificate
- Requester then installs the certificate on their web server
- The infrastructure that manages this is known as Public Key Infrastructure (PKI)

Let's Try It Out

- Open the terminal, let's create a certificate singing request and verify the certificates