Passwords

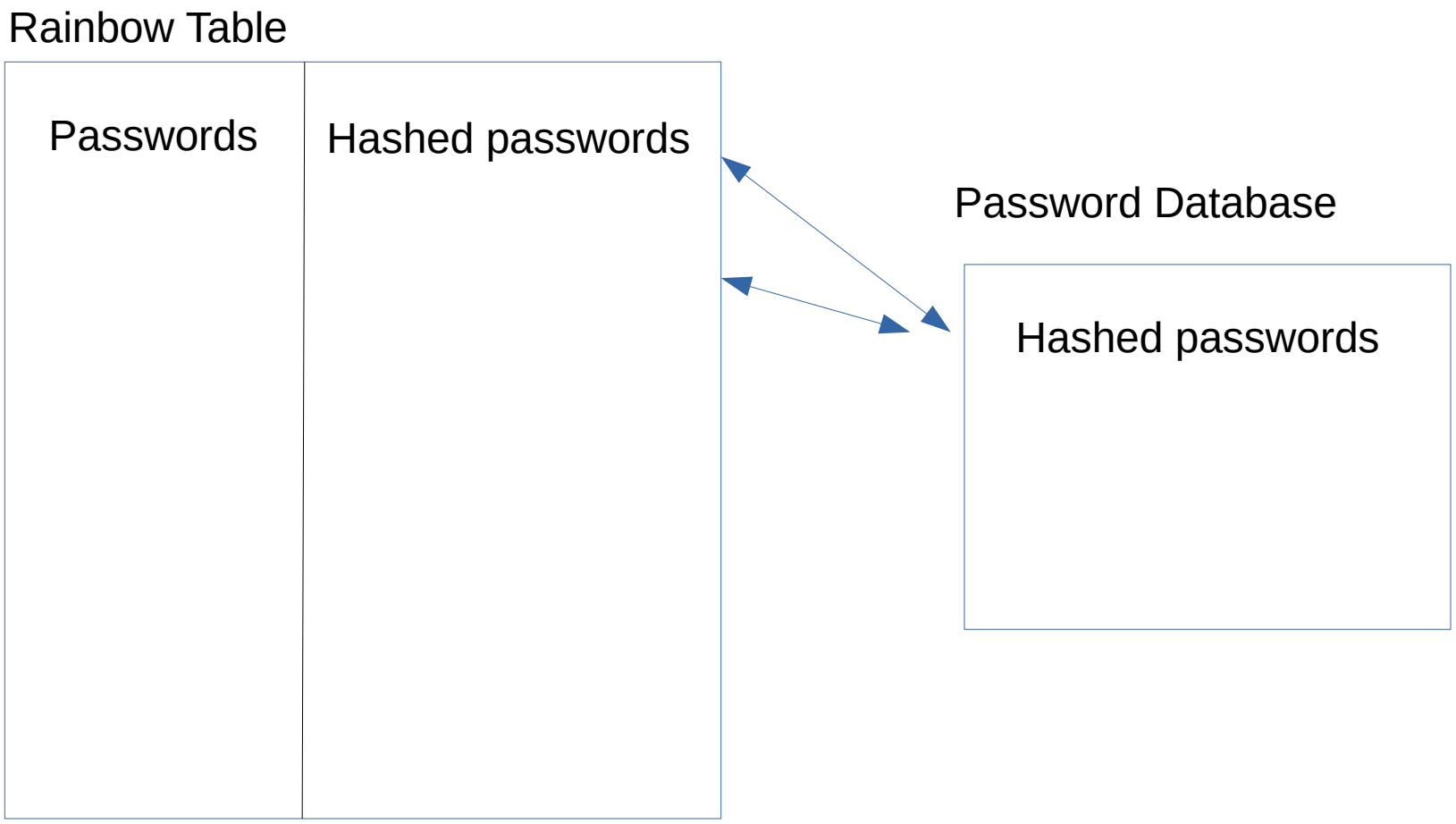
Hashed Passwords

- Obviously storing a username and password in plaintext is a security risk.
- If the database is accessed, an attacker can log on as any user.
- Instead we can hash the password.
- Passwords are checked by hashing supplied password and comparing with password database.
- Passwords can not now be acesses so that if lost the user sets the password.

Dictionary Attacks

- Dictionary Attacks can be run against hashed passwords.
- All entries in a dictionary are hashed and these hashed values are compared with the hashed value stored in the password database.
- [This is called a rainbow table.]

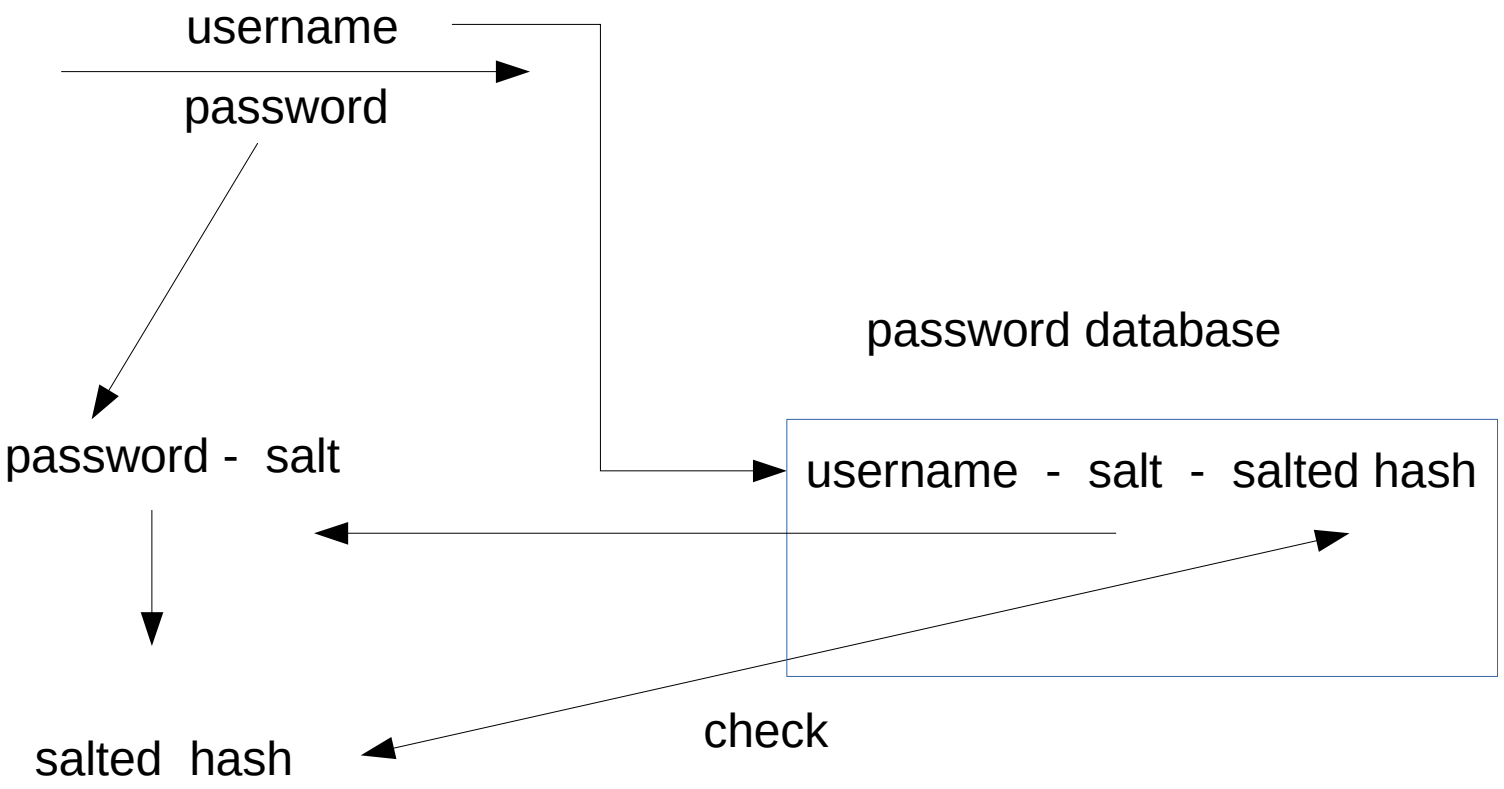
Rainbow Table



Salts

- A random value (the salt) is concatenated to the password.
- The result is hashed.
- Both the hash value and the salt is stored in the password database.
- In all three fields are stored, user name, salt and the "salted hash of the password".

Salted Hash - Login



Salted Hash

- What is the advantage?
- To run a dictionary attack against the password database, the entire dictionary must be rehashed for every password.
- Alternatively an attacker can pre-compute tables for each possible salt value.
- Even for 12 bit salts (old versions of Unix), 4096 tables need to be generated.

Key Stretching

```
key = hash(password + salt)
for 1 to 65536 {
      key = hash(key + password + salt)
}
```

- The overhead is manageable for a login.
- A problem when running a dictionary attack.

Nonce

Nonce

- Number used once.
  - input to block cipher algorithm (initialization vector)
  - prevention of replay attacks

Nonce - Prevention of replay attacks

- Suppose a request is signed to that it can be
  - authenticated
  - integrity checked
- The only attack is a replay attack.
- If the request includes a nonce, then this makes a replay attack impossible.

Nonce - Prevention of replay attacks

- Using a nonce like this requires the server to keep track of all previous nonces.
- But this works as the attacker can't change the nonce as the message is signed.

Nonce + Timestamp

- In practise a nonce can be used with a timestamp with granularity of 1 second say.
- It is then only necessary to make the nonce unique for a second.
- The combination of nonce and timestamp is now unique.
- [Used in OAuth]