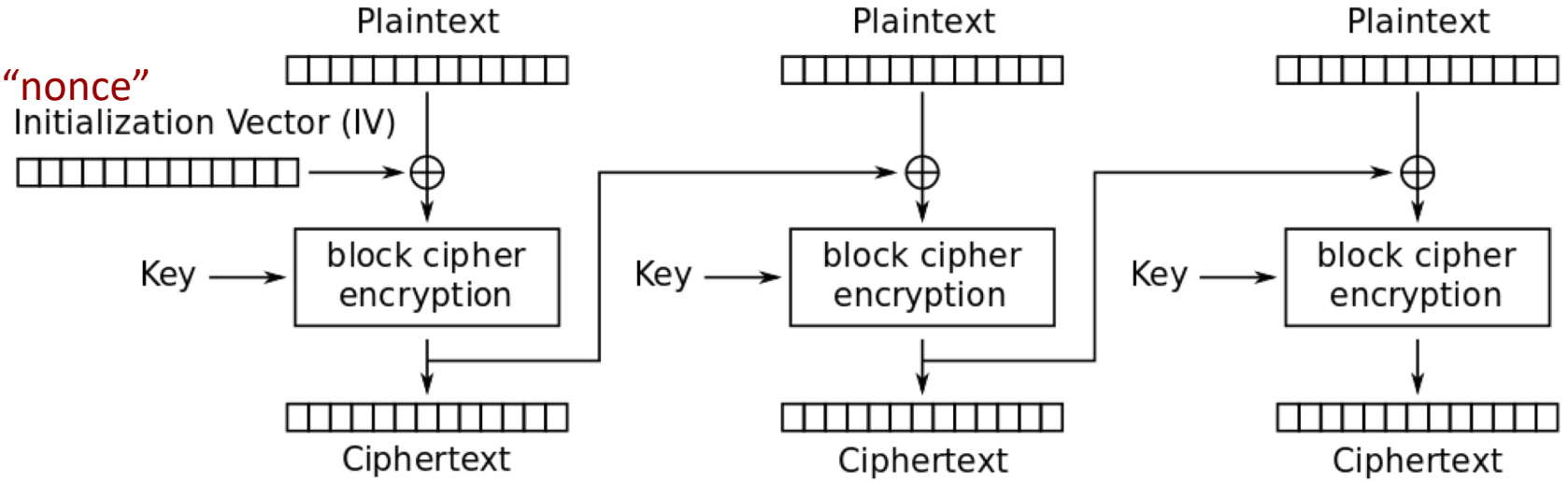
Nonce

Nonce

- Nonce = "**N**umber used **ONCE**"
  - Random or pseudo-random
- Common applications
  - Initialization Vector
  - Authentication Protocol (prevent replay attacks)
  - Cryptographic hash / proof-of-work systems

Nonce / Initialization Vector

- Example 1: **Initialization Vector (IV)** - Fixed input to crypto function that is required to be random or non-repeating (depends on mode of operation)



Cipher Block Chaining (CBC) mode encryption

Nonce / Authentication Protocol

- Example 2: **Authentication Protocol**
  - Goal: prevent replay attacks
  - Example: HTTP Digest Auth – Communicate login password to server *without encryption* plus prevent replay attacks

Nonce / Authentication Protocol

- HTTP Digest Auth

HTTP Client

HTTP Server

1. Client requests protected resource

Request →

← **Nonce**

2. Server creates random nonce and sends it to client with challenge to authenticate (nonce only good once)

3. Client enters user/pw

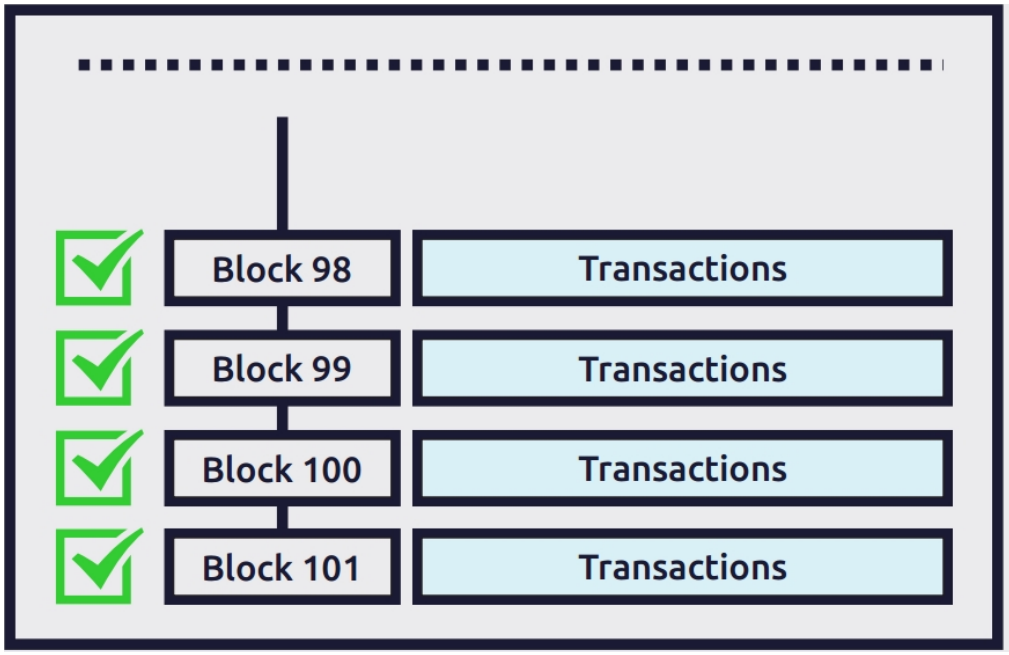
4. Client creates client nonce and hash and sends both to server

Login: Username, client-nonce, Hash(nonce, cnonce, password) →

5. Server validates hash by creating its own hash to see if it matches

← Token ✓

Nonce / Proof-of-Work System



- Example 3: Cryptographic hash / **proof-of-work systems** (e.g. **Blockchain**!)
- Blockchain is database shared by all network users that stores transaction history
- Transaction not recognized until it is added to blockchain

Nonce / Proof-of-Work System

- Example 3: **Cryptographic hash /** proof-of-work systems (e.g. **Blockchain!**)
- Must vary nonce (trial and error) until a desirable cryptographic hash is produced

Find a nonce *x* such that
SHA-256(SHA-256(r+*x*)) < T/d
r = header + transactions

(Simplified)