Secure API

The Report

Objectives

- To secure the API by encrypt the message between client and server (Attackers cannot know the message, cannot re-execute the message (replay attacks))
- Example use-case: Local network weather station, IoT applications

Cryptography (related) libraries

- random
- hashlib
- cryptography.fernet
- time

keygen.py

- Generate a key of **512 letters**, 512 B , and save to a file called ‘key.txt’
- **500** letters randomly
- **10** letters random from seed
- **2** letters from keyboard

Mask

- Apply before encryption
- **<56 – sha><n – massage><r – mask><56 – mask_identifier>**
- sha : *sha224sum* of symmetric key
- massage : massage of length *n* ; can be anything
- mask_identifier : Modification of *sha224sum* to tell the length of the mask (*r*)

Mask Identifier

- To tell the masking length
- Using based-2 number identify as ‘x’
- Example: sha224 = *aaaaa...aa* , mask_ident = *xaaxa...aa* , → length = $2^0 + 2^3 = 1 + 8 = 9$

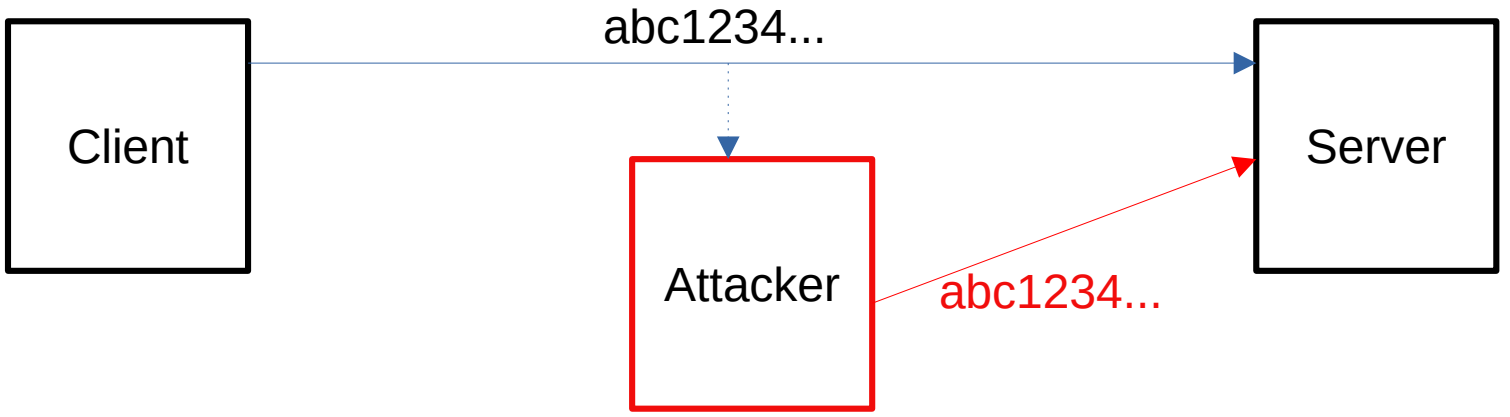
Message structure

- <20 – Time><3 – Code><n – message>
- So the final structure before encrypt is **<56 – sha><20 – Time><3 – Code><n – message><r – mask><56 – mask_identifier>**
- The len of r is random between 1 to 1000
- The message len is *136+n* to *1035+n*

Time format

- Using time.time()
- Formatted to 4 floating points precision
- Is the number of second since the (first time is computer)
- Then convert to string and adding ‘0’ to the front until the len(time) = 20

Replay attack



- Attacker could stored the messages and re-send them to cause disruption
- Example: overwrite the parameters, masquerade and disrupt the communication

Replay attack prevention

- Do not do the operation if time difference is more than 5 seconds
- Do not set the new data if the time of sending is the same or older than the last-saved time
- Time is the critical function to solve replay attacks

(idea) When the time is not exist?

- Client is unable to get accurate time or time is different than server
  - Client send message to server to request the time from it
  - Client use (strong) N-once to ensure that the message is valid (not from replay attack) , (sending and receiving use the different code but check for the same N-once)
  - Send the real message using the time and N-once

(idea) When the time is not exist? (2)

- Both client and server is unable to get time
- Option 1: Do not use the system
- Or
  - Client sends Nonce_1
  - Server sends back Nonce_2 , saves Nonce_1+Nonce_2
  - Client sends the real message using Nonce_1+Nonce_2 (server check if match, and removes it before return)
  - (each of the communication client also sends an extra m_Nonce, to validate the communication)
    - Attacker can replayed send the Nonce_1 as they like, but as long as Nonce_2 is strong enough, attacker cannot get in
    - Attacker could DoS by sending massive requests for N-once, that the server needs to store them in the memory (require another anti-DoS)

TL;DR

- The system may not as secure as you think
- This use weaker Fernet key 16 times
- It take a minute to pick a 1-digit lock 6 times vs. 11 days for 6 digit-key (1 key every second, digit is 0..9)
- The system transfer much more data than non-encrypted counterparts