Final Review

Final Exam

- Next Mon. May 6th
- LH 014
- 8:00 – 10:00 AM
- Topics
  - Everything after crypto(midterm)
  - Slides 10 – 20
- Similar structure as the midterm

Software Security

- Buffer Overflow
  - What makes it possible?
- Code injection
  - 3 Challenges and their countermeasures
- Return to libc vs. code injection
- Race conditions

Mitigation against Software Vulnerabilities

- Defensive coding practices
  - How to program defensively?
- Automated Testing Techniques
  - Static vs. Dynamic
- Fuzzing
  - What
  - Different Types
  - How to measure fuzzing test?
- Symbolic Execution
  - What
  - Goal

Malware

- Understand the different types of malware
  - What
  - Goal
- Virus vs. Worm
- The fight between anti-virus software and virus
  - Different ways to avoid detection

Web Security

- SQL Injection
  - What
  - Countermeasures
- Web Cookies
  - Why?
  - Authentication
- CSRF
  - How it works
  - Goal
  - Countermeasures
- XSS
  - SOP
  - Different types
  - How they work
  - Countermeasures

Authentication

- 3 ways to authenticate human to machine
- Something you know
  - Password
  - Attack on password
  - Countermeasures
- Something you are
  - Identification vs. Authentication
  - Compare different biometrics
- Something you have
  - 2-factor authentication

HW2

- On a particular system, all passwords are 8 characters, there are 128 choices for each character, and there is a password file containing the hashes of 2^{10} passwords. Trudy has a dictionary of 2^{30} passwords, and the probability that a randomly selected password is in her dictionary is 1/4. Work is measured in terms of the number of hashes computed.

- Expected work to crack Alice's password, **unsalted**

- Ans: Hashes for the dictionary is precomputed
- 128 = 2^7; we have 8 characters; Possible password = 2^{56}
- The **expected** work is $(1/4 * 0 + ¾ * 2^{56}) / 2$

HW2

- On a particular system, all passwords are 8 characters, there are 128 choices for each character, and there is a password file containing the hashes of 2^{10} passwords. Trudy has a dictionary of 2^{30} passwords, and the probability that a randomly selected password is in her dictionary is 1/4. Work is measured in terms of the number of hashes computed.

- Expected work to crack Alice's password, **salted**

- Ans: Hashes for the dictionary is precomputed
- $128 = 2^7$; we have 8 characters; Possible password = 2^{56}
- The **expected** work is $(1/4 * 2^{30} + ¾ * 2^{56}) / 2$

HW2

- On a particular system, all passwords are 8 characters, there are 128 choices for each character, and there is a password file containing the hashes of 2^{10} passwords. Trudy has a dictionary of 2^{30} passwords, and the probability that a randomly selected password is in her dictionary is 1/4. Work is measured in terms of the number of hashes computed.

- What is the probability that at least one of the passwords in the password file appears in Trudy's dictionary?

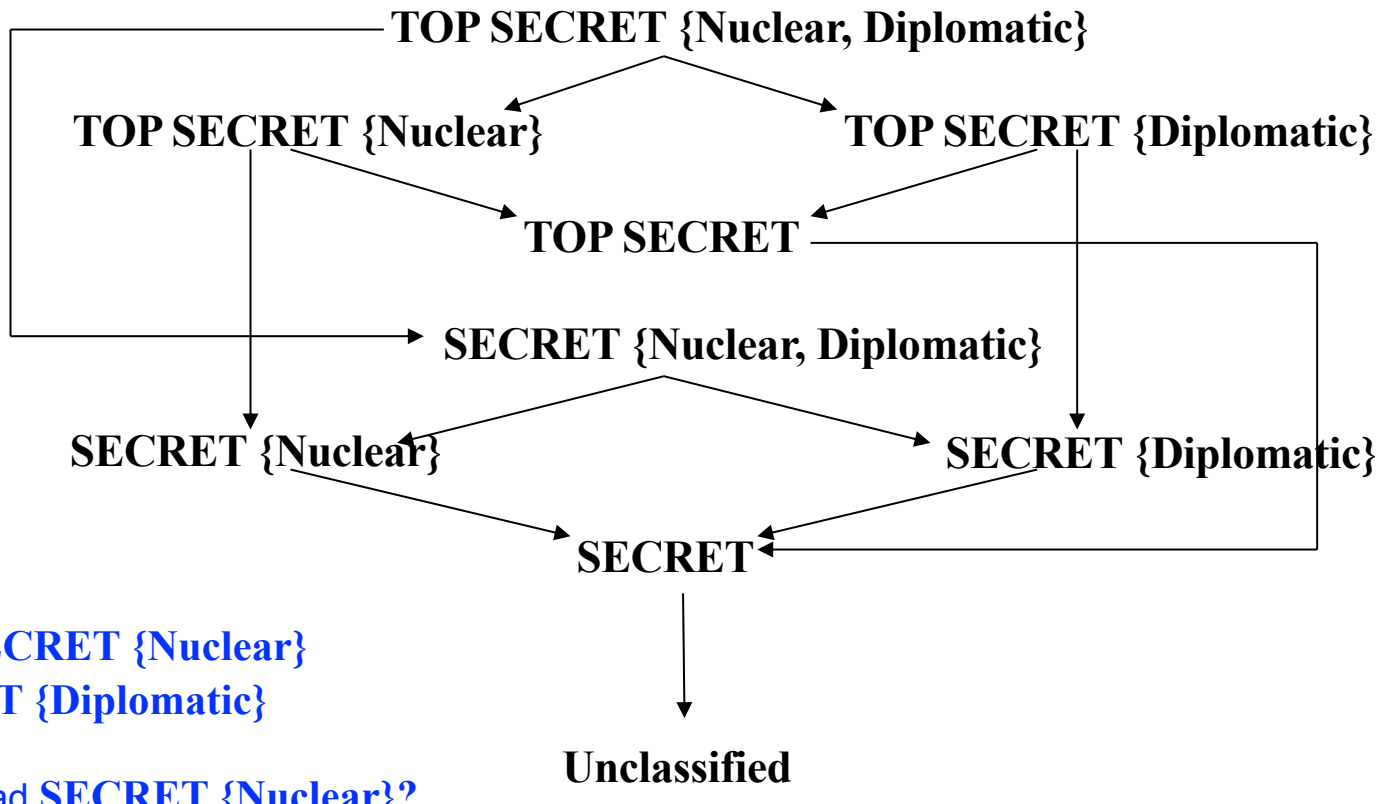
- Ans: $1 - (3/4)^{1024}$
- Here, $1024 = 2^{10}$

Authorization

- DAC vs. MAC
- ACL vs. Capabilities
  - Confused Deputy
- Multilevel Security (MLS) Models
  - Why
  - Classification: objects, clearance: subjects
- Bell-Lapadula(BLP)
  - Read, write rule
- Biba
  - Read, write rule
- Compartments
  - Why
  - How it works
- Covert Channel

HW2: Compartments

- Arrows indicate "≥" relationship



Given:
John: TOP SECRET {Nuclear}
Mary: SECRET {Diplomatic}

1. Can John read **SECRET {Nuclear}?**
2. Can Mary read **TOP SECRET {Diplomatic}?**
3. **TOP SECRET {Nuclear, Diplomatic} file accessibility.**

Trusted Computing

- Trusted Systems
  - Hardware-based, software-based
- SELinux
  - How it works
  - What does it offer
- Trusted Platform Module
  - Attestation Identity Key(AIK) and its usage

Basic Authentication Protocol

- Authentication over network
  - machine to machine
- Attacks on simple protocol
  - Replay and its countermeasures
- Mutual authentication based on symmetric key
  - Attack on simple implementation proposals
  - How to provide a secured implementation
- Mutual authentication based on public key
  - Session Key
  - Timestamps

SSH, SSL

- Simplified SSH
  - Details and questions about it
  - Modify the protocol
  - Any attacks?
- SSL
  - Details of each exchanged messages
  - Authentication
  - Session vs. Connection
  - Attacks

Kerberos and IPSec

- Kerberos
  - What is it designed for
  - How does it achieve a scalability?
  - 3 phases
    - login, request to ”ticket to bob”, use “ticket to bob”
    - Questions about Kerberos
- IPSec
  - IKE(Internet Key Exchange) 2 phases
    - How Phase1 works, different modes, key options, comparisons
    - Phase 2 goal
  - ESP(Encapsulating Security Payload)/AH(Authentication Header)
    - Differences, goal
  - 2 modes: transport, tunnel

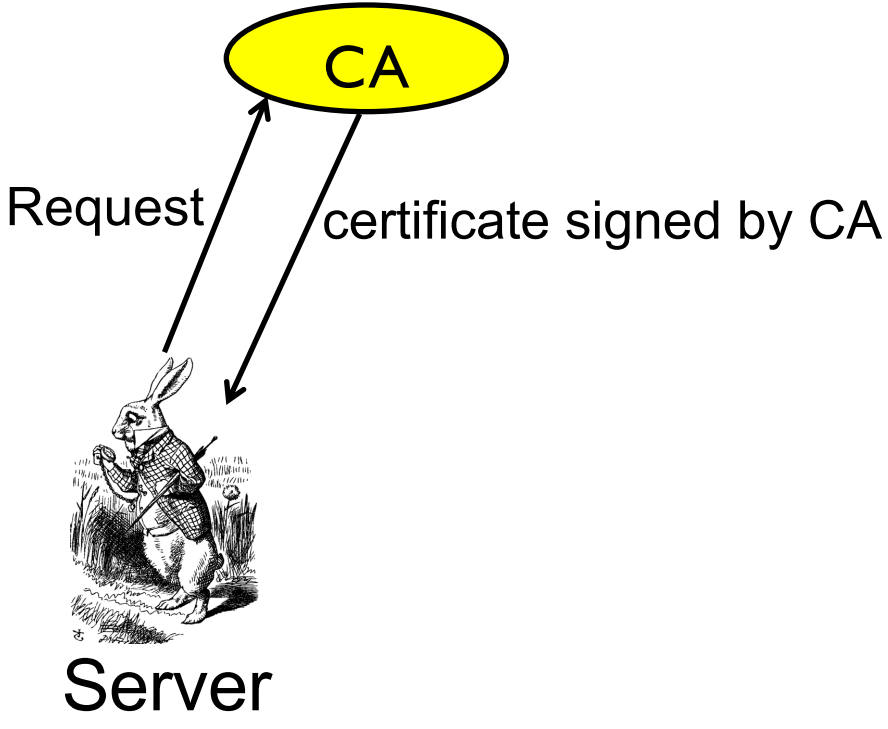
Programming 4 – Base Program

- Client Server programs
  - A non-secure implementation is given
  - Client
    - Receive a number from user input
    - Send it to the server
    - Wait for response
    - Display the received result
    - Repeat until exit
  - Sever
    - Wait for client connection
    - Increase client inputs by 1
    - Send it back to client
    - Close the connection if client disconnect
    - Repeat till exit

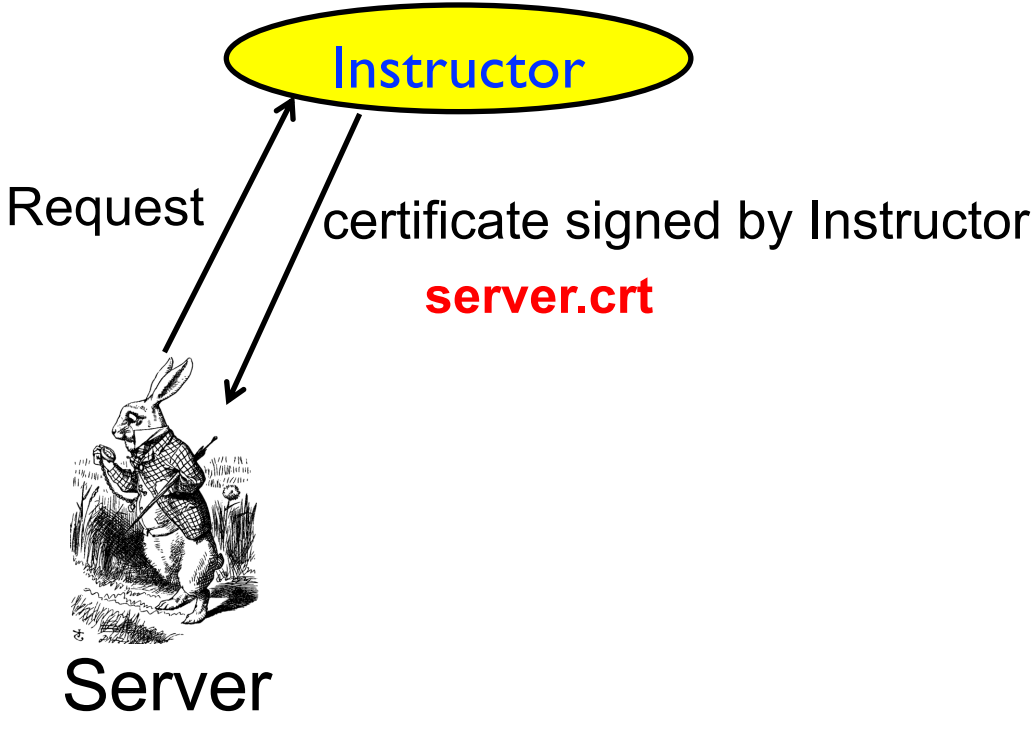
Programming 4 – Base Program

- Goal: provide a secure client server program implementation using OpenSSL
  - Refer to the links, resources in the description to get familiar with OpenSSL APIs
  - Init the secure context
  - Setup the secure context to use the provided certificates for authorization
  - Modify the exiting program to use secured version read/write

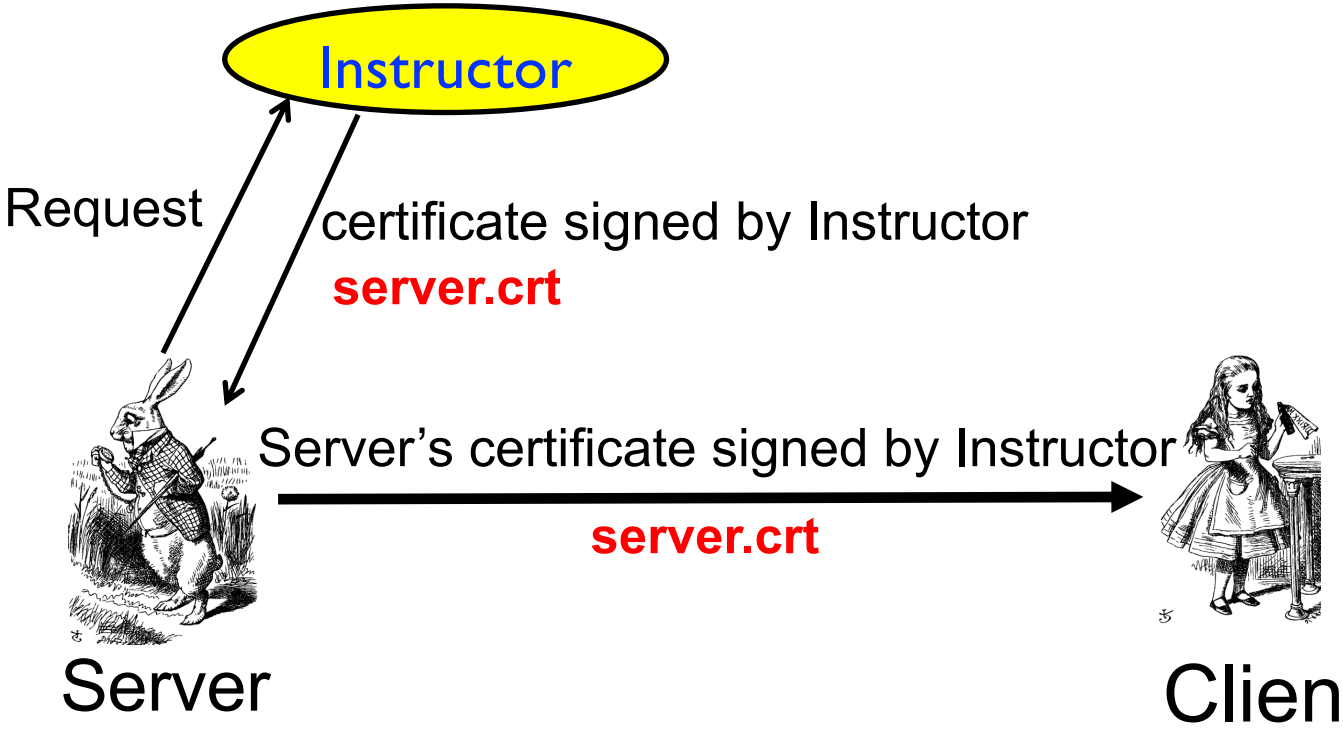
CA and Certificate



CA and Certificate: Instructor as CA



CA and Certificate: Client verify certificate



Instructor's public key: **ca.pem**

Client use this public key to verify server.crt

Self Signed Certificate

Certificate signed by server itself

self_signed_server.crt

Server's certificate signed by itself

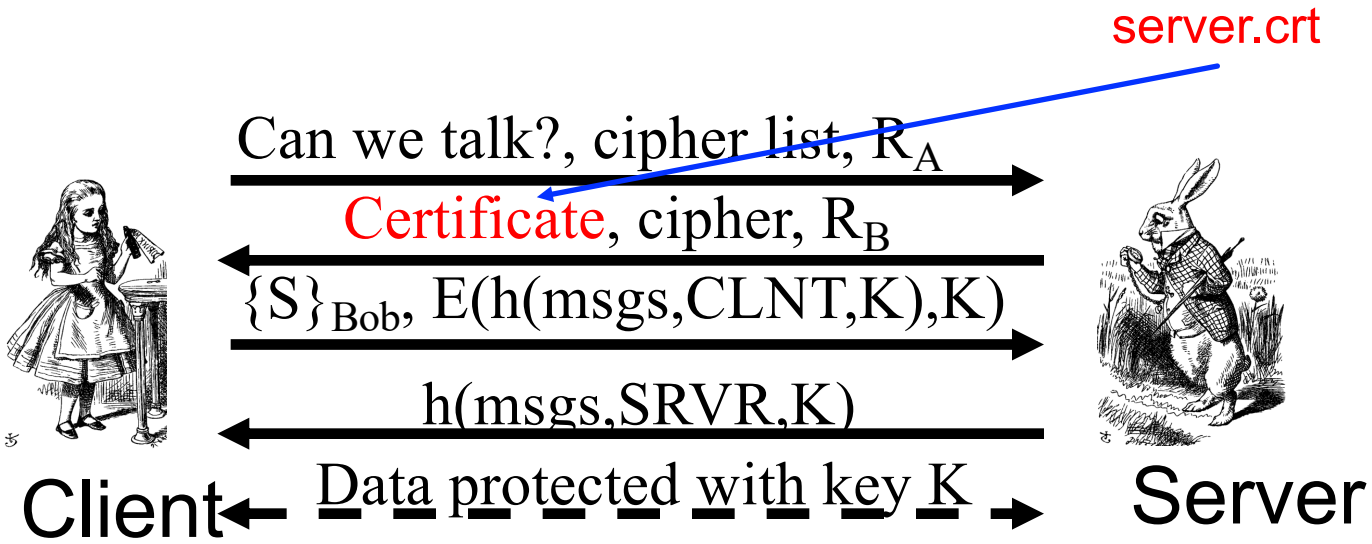
self_signed_server.crt

Server

Client

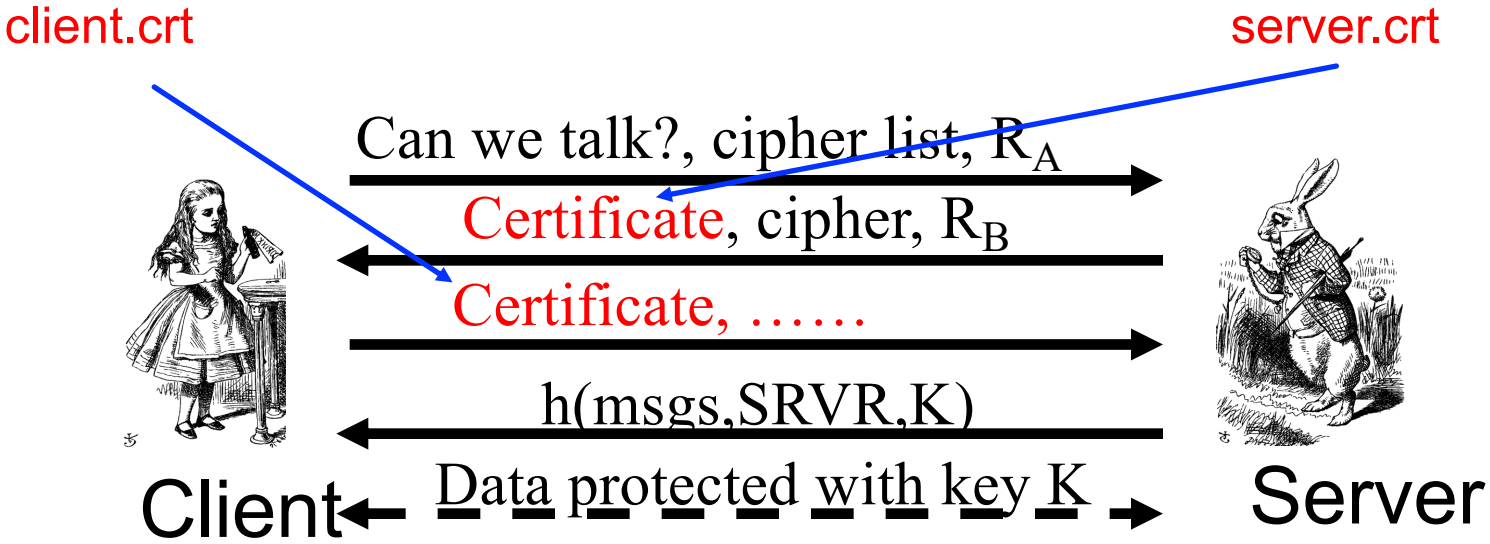
Verification failed with **ca.pem**
since self_signed_server.crt
is not signed by Instructor

Simplified SSL Protocol



- S is known as **pre-master secret,** randomly generated
- $K = h(S, R_A, R_B)$
- "msgs" means all previous messages
- CLNT and SRVR are constants string

Bonus



- S is known as **pre-master secret,** randomly generated
- $K = h(S, R_A, R_B)$
- "msgs" means all previous messages
- CLNT and SRVR are constants string