CS396: Security, Privacy and Society (Fall 2022)
Homework #2, November 29, 2022

Instructor: Abrar Alrumayh

Instructions

Please carefully read the following guidelines on how to complete and submit your solutions.

1. The homework is **due on Wednesday, December 9, 2022, at 11:59pm**. Starting early always helps!
2. Solutions are accepted only via Canvas, where all relevant files should be submitted **as a single .zip archive**. This should include your typed answers **as a .pdf file** and **the source code** of any programming possibly used in your solutions.
3. If asked, you should be able to explain details in your source code (e.g., related to the design of your program and its implementation).
4. You are bound by the Stevens Honor System. For Homework #2, you may work either **by yourself or in pairs**—in this case, your team **names should appear clearly** on your hand-in, and **2 same hand-ins are required**. Any collaboration beyond this is not allowed. You may use any sources related to course materials, but information from external sources must be properly cited. Your submission acknowledges that you have abided by this policy.

Problem 1: Domain-extension MAC implementation (40%)

Given the provided support code in Java, implement a secure secret-key message authentication code that employs only a block cipher (and no other cryptographic primitive) to authenticate messages of any size in a *bandwidth-efficient manner*. In particular, as specified in the provided instructions:

(1) Implement the `mac()` and `verify()` methods. (20%)

(2) Demonstrate that they are correct by providing the MAC tag (in hexidecimal) of the specified default message using the specified default key. (10%)

(3) Explain which algorithm you implemented and why, and what are the domain-extension features of your algorithm in relation to its security. (10%)

> Hint: Does your implementation securely handle messages of fixed size, messages of any size, or messages of any fixed size?

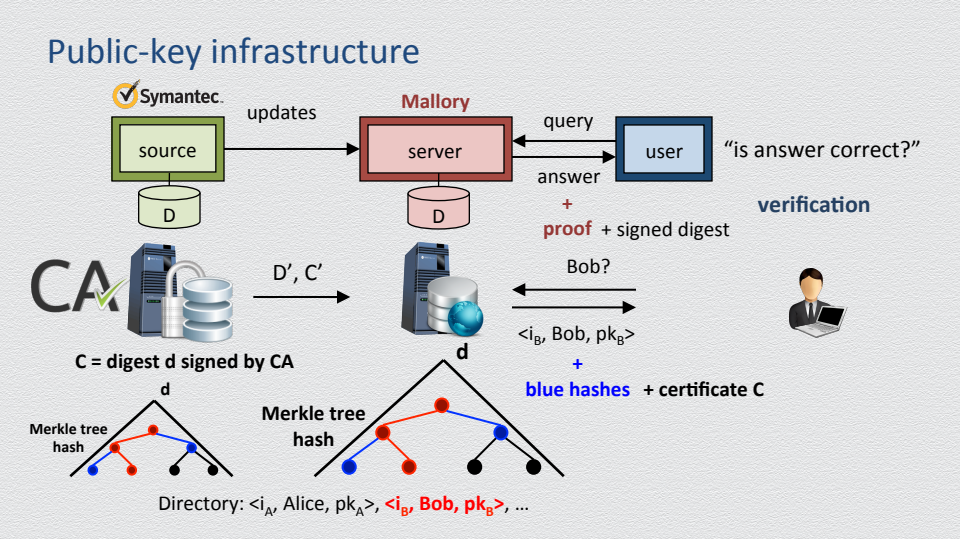

Figure 1: The public-key dictionary-as-a-service model for verifying public keys.

Problem 2: Data outsourcing & public-key infrastructure (30%)

To protect the secrecy of communications amongst its members, Stevens makes use of public-key encryption: Each student, faculty or staff member with Stevens UID i, name n_i and public-key pair (sk_i, pk_i), has their public key registered with a trusted certification authority (CA), e.g., Symantec. For efficiency reasons, the CA makes the directory $D = \{(i, n_i, pk_i) | i \in \text{CS396}\}$ of all such public keys available for users to query via an online service at Stevens that is administered by Mallory. Specifically (see also Figure 1):

- The CA provides Mallory with the public-key directory D along with a *certificate* C, or signed digest, that is the Merkle-tree digest of the directory signed by the CA.
- To send a *new* encrypted message to Bob, Alice asks Mallory for his public key (*even if* Alice had recently sent encrypted messages to Bob, as public keys may be refreshed or revoked).
- Along with Bob's public-key record (i_B, Bob, pk_B) in D, Mallory provides Alice with the current certificate C and a Merkle-tree proof (i.e., hash values) corresponding to Bob's record.
- After any change in the members of Stevens community (e.g., students joining/graduation, new hires, etc.) or any update on users' key pairs, the CA provides Mallory with the newly updated directory D' and its corresponding new certificate C'.

(1) Suppose that Cindy manages to get access to Bob's laptop and steal its secret key sk_B. After Bob becomes suspicious of this, he registers a new public-key pair with the CA. How can Cindy collaborate with Mallory in order for Cindy to be able to decrypt all subsequent encrypted messages that are sent to Bob over an insecure channel controlled by Cindy, without anyone ever becoming suspicious about this? What two named attacks are used in this case? (15%)

(2) Describe, and justify, an efficient and secure countermeasure to the above threat. You may assume that no public key will be updated twice within the same day, and that the CA processes such updates in batch by reporting new directory states (D', C') only every 24 hours. (15%)

Problem 3: On the RSA cryptosystem (40%)

(1) Given the support code in Python that was provided in Lab#8 (ran on November 10), implement all RSA-related algorithms. Namely, submit the completed skeleton code for Lab#8, including:

- The algorithms that you worked (or started working) on during the Lab#8 (methods `modexp`, `RSA_enc` and `RSA_dec`); (15%)
- The RSA key-generation algorithm (method `keygen`). (15%)

(2) The RSA cryptosystem relies on modular exponentiations, as its core operations.

- How are such operations realized more efficiently in practice? (5%)
- Would you recommend choosing a small public exponent (e.g., $e = 3, 5$ or 7) so that at least message encryption and signature verification become much faster, and why? (5%)