Question 1 ***Why do RSA signatures need a hash?*** ()

To generate RSA signatures, Alice first creates a standard RSA key pair: (n, e) is the RSA public key and d is the RSA private key, where n is the RSA modulus. For standard RSA signatures, we typically set e to a small prime value such as 3; for this problem, let $e = 3$.

Suppose we used a **simplified** scheme for RSA signatures that skips using a hash function and instead uses message M directly, so the signature S on a message M is $S = M^d \bmod n$. In other words, if Alice wants to send a signed message to Bob, she will send (M, S) to Bob where $S = M^d \bmod n$ is computed using her private signing key d.

Q1.1 With this **simplified** RSA scheme, how can Bob verify whether S is a valid signature on message M? In other words, what equation should he check, to confirm whether M was validly signed by Alice?

> **Solution:** $S^3 = M \bmod n$.

Q1.2 Mallory learns that Alice and Bob are using the **simplified** signature scheme described above and decides to trick Bob into beliving that one of Mallory's messages is from Alice. Explain how Mallory can find an (M, S) pair such that S will be a valid signature on M.

You should assume that Mallory knows Alice's public key n, but not Alice's private key d. The message M does not have to be chosen in advance and can be gibberish.

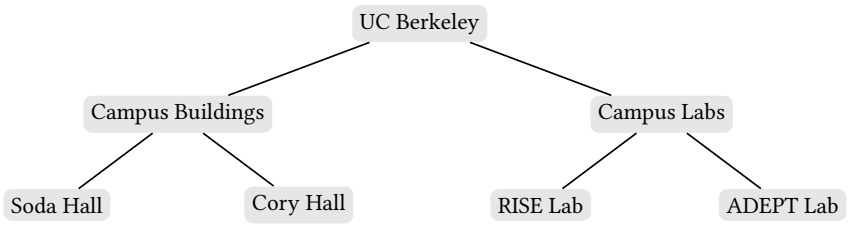
> **Solution:** Mallory should choose some random value to be S and then compute $S^3 \bmod n$ to find the corresponding M value. This (M, S) pair will satisfy the equation in part (a).
>
> **Alternative solution:** Choose $M = 1$ and $S = 1$. This will satisfy the equation.

Q1.3 Is the attack in Q3.2 possible against the **standard** RSA signature scheme (the one that includes the cryptographic hash function)? Why or why not?

> **Solution:** This attack is not possible. A hash function is one way, so the attack in part (b) won't work: we can pick a random S and cube it, but then we'd need to find some message M such that $H(M)$ is equal to this value, and that's not possible since H is one-way.
>
> Comment: This is why the real RSA signature scheme includes a hash function: exactly to prevent the attack you've seen in this question.

Question 2 ***RISELab Shenanigans*** **()**

Certificate authorities of UC Berkeley are organized in a hierarchy as follows:



Alice is a student in RISELab at UC Berkeley and wants to obtain a certificate for her public key. Assume that only RISELab is allowed to issue certificates to Alice.

Q2.1 Which of the following values are included in the certificate issued to Alice? Select all that apply.

- ■ Alice's public key
- ☐ Alice's private key
- ■ A signature on Alice's *public* key, signed by RISELab's private key
- ☐ A signature on Alice's *private* key, signed by RISELab's private key
- ☐ None of the above

Solution: This follows from the definition of certificates: they include a user's public key, and a signature on the enclosed public key, signed by the issuer (which we state in the prologue is RISELab).

Q2.2 Assume that the only public key you trust is UC Berkeley's public key. Which certificates do you need to verify in order to be sure that you have Alice's public key? Select all that apply.

- ■ Certificate for Alice
- ☐ Certificate for Soda Hall
- ■ Certificate for RISELab
- ■ Certificate for Campus Labs
- ☐ None of the above

Solution: To validate Alice's public key, we can follow our way up to our root of trust (which is UC Berkeley's public key). As such, we need certificates for Alice, RISELab, and Campus Labs.

Q2.3 RISELab issues a certificate to Alice that expires in 1 hour. Which of the following statements are true about using such a short expiration date? Select all that apply.

- [x] It mitigates attacks where Alice's private key is stolen
- [ ] It mitigates attacks where RISELab's private key is stolen
- [ ] It mitigates attacks where Campus Labs' private key is stolen
- [x] It forces Alice to renew the certificate more often
- [ ] None of the above

Solution: Short expiration times only mitigate the situation where Alice's private key is stolen. If RISELab's private key is compromised, the attacker can issue certificates with any expiration date, and it is up to the parent CA to revoke RISELab's certificate, not RISELab itself. The same argument applies to Campus Labs' private key.

Question 3 ***Password Storage*** **()**

Bob is trying out different methods to securely store users' login passwords for his website.

Mallory is an attacker who can do some amount of *offline* computation before she steals the passwords file, and some amount of *online* computation after stealing the passwords file.

Technical details:

- Each user has a unique username, but several users may have the same password.
- Mallory knows the list of users registered on Bob's site.
- Bob has at most 500 users using his website with passwords between 8–12 letters.
- Mallory's dictionary contains all words that are less than 13 letters. [*Clarification during exam*: Mallory's dictionary contains all possible user passwords.]
- Mallory can do N online computations and $500N$ offline computations where N is the number of words in the dictionary.
- Slow hash functions take 500 computations per hash while fast hash functions require only 1 computation.[1]

Notation:

- $\mathsf{H_S}$ and $\mathsf{H_F}$, a slow and fast hash function
- Sign, a secure signing algorithm
- uname and pwd, a user's username and password
- k, a signing key known only by Bob

If Bob decides to use signatures in his scheme, assume he will verify them when processing a log-in.

For each part below, indicate all of the things Mallory can do given the password storage scheme. Assume Mallory knows each scheme. **Unless otherwise specified, assume that she can use both offline and online computation**

Q3.1 Each user's password is stored as $\mathsf{H_F}(\mathsf{pwd} \,||\, \texttt{'Bob'})$.

■ (A) Learn whether two users have the same password with only online computation

■ (B) Learn a specific user's password

■ (C) Change a user's password without detection

■ (D) Learn every user's password

☐ (E) None of the above

☐ (F) ——

Solution: Since this is a hash function with the same salt, Mallory can do one full run through of the dictionary with online computation to learn each user's password. Additionally, there are no authenticity checks so Mallory can edit a password.

[1] Keep in mind this is much faster than a real-life slow hash function.

Q3.2 Each user's password is stored as the tuple $(\mathsf{H_S}(\mathsf{pwd} \,||\, \texttt{'Bob'}), \mathsf{Sign}(\mathsf{k}, \mathsf{H_F}(\mathsf{pwd})))$.

■ (G) Learn whether two users have the same password with only online computation

■ (H) Learn a specific user's password

■ (I) Change a user's password without detection

■ (J) Learn every user's password

☐ (K) None of the above

☐ (L) ——

Solution: Because of the slow hash, Mallory can only longer do a full run through of the dictionary using online computation. However, she can do so using offline computation since the salt is the same for all passwords. Since the signature does not include the username, password entries can be swapped without detection.

An earlier version of the solutions incorrectly marked (A) as incorrect. However, since signatures are unsalted, an attacker can learn if two users have the same password by comparing signatures (which requires no computation).

Q3.3 Each user's password is stored as the tuple $(\mathsf{H_F}(\mathsf{pwd} \,||\, \mathsf{uname}), \mathsf{Sign}(\mathsf{k}, \mathsf{uname} \,||\, \mathsf{H_F}(\mathsf{pwd})))$

☐ (A) Learn whether two users have the same password with only online computation

■ (B) Learn a specific user's password

☐ (C) Change a user's password without detection

■ (D) Learn every user's password

☐ (E) None of the above

☐ (F) ——

Solution: Because the salt is now different, Mallory only has enough online computation to bruteforce a single password. However, using offline computation she can still learn all the passwords since she can bruteforce the dictionary 500 times. Since each signature is tied to a specific user and Mallory doesn't know k, she can't edit a user's password.

Q3.4 Each user's password is stored as $(\mathsf{H_S}(\mathsf{pwd} \,||\, \mathsf{uname}), \mathsf{Sign}(\mathsf{k}, \mathsf{H_S}(\mathsf{pwd})))$

[*Clarification during exam*: The expression was missing a leading parenthesis.]

■ (G) Learn whether two users have the same password with only online computation

■ (H) Learn a specific user's password

■ (I) Change a user's password without detection

☐ (J) Learn every user's password

☐ (K) None of the above

☐ (L) ——

Solution: Mallory only has enough total computation to learn a single user's password, denoted as pwd'. She can now edit a different user's password to be this by computing $\mathsf{H_S}(\mathsf{pwd}' \,||\, \mathsf{uname})$ and using the signature $\mathsf{Sign}(\mathsf{k}, \mathsf{H_S}(\mathsf{pwd}'))$. Note this is possible because the signature isn't bound to any specific user.

An earlier version of the solutions incorrectly marked (A) as incorrect. However, since signatures are unsalted, an attacker can learn if two users have the same password by comparing signatures (which requires no computation).