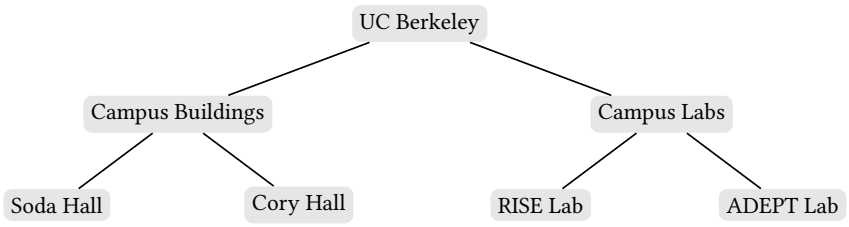
**Question 1** ***RISELab Shenanigans***

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.

Q1.1 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

Q1.2 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

Q1.3 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.

- ☐ 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
- ☐ It forces Alice to renew the certificate more often
- ☐ None of the above

**Question 2** ***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**

Q2.1 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) ——

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

Q2.2 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) ——

Q2.3 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) ——

Q2.4 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) ——

**Question 3** ***Brainf[REDACTED]***

Consider the following code:

```
1  void execute(char* commands, FILE *file) {
2      int buf_ind = 0;
3      int buf_len = 16;
4      char buf[buf_len];
5      size_t comm_ind = 0;
6      while (commands[comm_ind]) {
7          if (commands[comm_ind] == 'C') {
8              buf_ind += 1;
9          } else if (commands[comm_ind] == 'D') {
10             buf_ind -= 1;
11         } else if (commands[comm_ind] == 'E') {
12             printf("%c", buf[buf_ind]);
13         } else if (commands[comm_ind] == 'F') {
14             printf("%x", &buf[buf_ind]);
15         } else if (commands[comm_ind] == 'G') {
16             fread(&buf[buf_ind], sizeof(char), 1, file);
17         }
18         /* assume you are provided two functions: min and max. */
19         buf_ind = max(0, min(buf_len, buf_ind));
20         comm_ind += 1;
21     }
22 }
```

For this question, assume the following:

- You may use `SHELLCODE` as a 52-byte shellcode.
- Stack canaries are enabled, and all other memory safety defenses are disabled.
- If needed, you may use the standard output as `OUTPUT`, slicing it using Python syntax.
- The RIP of `execute` is located at `0xffffabcc`.
- The top of the stack is located at `0xffffffff`.
- `execute` is called from `main` with the proper arguments.

Q3.1 (4 min) Fill in the following stack diagram, assuming that the program is paused after executing **Line 6**, including the arguments of `execute` (the value in each row does not necessarily have to be four bytes long).

**Stack**

| |
|---|
| |
| |
| |
| |
| |
| |
| |
| |
| |

Q3.2 (12 min) We wish to construct a series of inputs that will cause this program to execute `SHELLCODE` that works 100% of the time.

Provide a string input to variable `commands` (argument to `execute`):

| |
|---|
| |

Provide a string for the contents of the file that is passed in as the `file` argument of `execute`:

| |
|---|
| |

Q3.3 (3 min) If ASLR is now enabled, which of the following modifications to the provided code would allow you to execute `SHELLCODE` 100% of the time? Select all that apply.

- ☐ Line 10 is replaced with `scanf("%u", &buf_ind)`.
- ☐ `jmp *esp` is located in your code at `0xdeadbeef`.
- ☐ Line 14 is replaced with `comm_ind = getchar()`.
- ☐ None of the above