CS161 Summer 2026 — Introduction to Computer Security — Discussion 12

Q1 *Potpourri* **(6 points)**

Q1.1 (1 point) Select all true statements

- ☐ To perform a rowhammer attack, the attacker needs write access
- ☐ Memory deduplication allows the attacker to place any code anywhere
- ☐ Rowhammer doesn't require the attacker to have physical access to the machine
- ☐ The repeated reads must bypass the cache to work

Q1.2 (1 point) Select all true statements

- ☐ A password check that returns at the first wrong character leaks how many leading characters are correct
- ☐ Constant-time code should avoid branching on secret values
- ☐ Repeated squaring can leak the Hamming weight of a secret exponent through timing
- ☐ Adding a dummy computation prevents all side channel leaks since our code could be padded to run in a fixed time

Q1.3 (1 point) Select all true statements

- ☐ Spectre abuses speculative execution to run a forbidden path on secret data
- ☐ When probing, a cache hit indicates the location of the speculated execution
- ☐ Separate OS processes mitigate Spectre, but WebAssembly isolation does not
- ☐ Probing attacks can be mitigated with shielded rooms (SCIF) and circuit masking

Q1.4 (1 point) Select all true statements

- ☐ LLMs are trained primarily for next-token prediction
- ☐ LLMs reliably distinguish the user's instructions from third-party data
- ☐ Inferences are deterministic

Q1.5 (1 point) Select all true statements

- ☐ A restaurant review saying "ignore all previous instructions" is an example of prompt injection
- ☐ Prompt injection is fundamentally different from SQL injection and buffer overflows
- ☐ Jailbreaking aims to make the model do something it shouldn't do for anyone
- ☐ The attacker must get the system-instruction syntax exactly right for injection to work

Q1.6 (1 point) Select all true statements

- ☐ Delimiters / data layers guarantee that data is never treated as instructions
- ☐ A quarantined LLM with no tool access limits the damage from an injection
- ☐ LLM defenses prevent the model from relaying misinformation
- ☐ Generative AI can be implemented for defense and never for attack

Q2 *Attacks* **(4 points)**

Q2.1 (1 point)

```
1 def login(email_address, password):
2     password_hash = fetch_password_hash(email_address)
3     if password_hash is None:
4         return False
5     return bcrypt.checkpw(password.encode(), password_hash)
```

Which attack is the code above vulnerable to?

- ◯ Timing/Cache Timing Attack
- ◯ Hamming Weight Attack
- ◯ RowHammer Attack
- ◯ Safe against all above attacks
- ◯ Spectre Attack

Q2.2 (1 point) Which line(s) make this program vulnerable to the attack chosen?

- ☐ Line 2
- ☐ Line 3
- ☐ Line 4
- ☐ Line 5

Why? What information is leaked?

Q2.3 (1 point)

Assume the function below is just one step inside a black boxes block cipher operation that mixes the key with the encrypted data.

- `a` is data that changes per each call.
- `k` is a subkey. It is **the same every call**, and is private
- `r` is a random mask, a new value per each call.

Alice realized that without `r`, an attacker who knows `a`, can simply prime-and-probe to guess each bit of `k` through measuring the power output, since a circuit holding or switching more 1 bits draws more current. By adding a random value `r`, the measurement become unpredictable.

Mallory believes that she can still break it. She can trigger the function as many times as she wants, and can measure power output. However, she never sees the `a`, `r`, or return value.

Is Mallory able to leak information about the subkey `k`?

```
1 def mask(a, k):
2     r = random_32_bits()
3     z = a + r
4     z = z + k
5     z = z - r
6     return z
```

Assume that `z, a, r, k` are all 32-bit values that wrap around in case of overflow.

Which attack is the code above vulnerable to?

- ◯ Timing/Cache Timing Attack
- ◯ Hamming Weight Attack
- ◯ RowHammer Attack
- ◯ Safe against all above attacks
- ◯ Spectre Attack

Q2.4 (1 point) Which line(s) make this program vulnerable to the attack chosen?

- ☐ Line 2
- ☐ Line 3
- ☐ Line 4
- ☐ Line 5
- ☐ Line 6

Why? What information is leaked?