Solutions to Quiz 6 (October 27)

API Design (1 point)

REST stands for REpresentational State Transfer. We learned that REST is a design style that is relevant to the Web and applications using Web technology.

1. In REST, what is the difference between a resource and a representation?
   A. A resource stays at the server
   B. A representation has an invariant
   C. A resource responds to HTTP verbs like PUT and GET
   D. Both A and B
   E. Both A and C

Solution. E. A resource does not move across the network, although a corresponding representation might well do so. A resource also can have HTTP methods invoked on it. Although a representation might have an invariant, and representation invariants are important when we are talking about abstract data types, representation invariants are not a part of what is highlighted as significant in the REST design style.

Security (2 points)

Many buildings have a security checkpoint. Stated policy is usually that everyone must pass through the checkpoint, without exception.

Over time, guards typically become acquainted with many of the regular users of the checkpoint, and even become friendly with some. On occasion, a guard may make an exception for a friend and let them bypass the checkpoint, especially if there’s a long line and the friend is in a hurry.

2. This situation is best understood as a violation of which security principle?
   A. Least privilege
   B. Least common mechanism
   C. Complete mediation
   D. Fail-safe defaults
   E. Open design

Solution. C. Complete mediation is the checking of every request for proper authentication and authorization.
Web Security (2 points)

You have been tasked with fixing the JavaScript calculator that was prone to an injection attack, as demonstrated in lecture. Others have already observed that the vulnerability arises from the line that includes the expression

\[
\text{eval(Calc.input.value)}
\]

which of course evaluates anything that appears in the Calc.input.value field. In lecture, we attacked the calculator by making an assignment to the field Calc.one.value. The plan is to replace the above expression with the expression

\[
\text{eval(sanitize(Calc.input.value))}
\]

The sanitize function will be written by someone else who is a competent JavaScript programmer but doesn’t really understand injection attacks at all. You have the task of telling them what to write.

3. Which of these is the best informal specification for the sanitize function?
   A. Ensure the input does not contain the string “value” anywhere
   B. Ensure the input does not contain any dots used to select values
   C. Ensure the input does not contain the string “Calc” anywhere
   D. None of the above choices is an adequate spec to prevent injection

**Solution. D.** All of these have the flaw that they are talking about the character of the input string to sanitize, when what really matters is the output to be eval'ed. All of these also have the flaw that they can be readily avoided by the use of eval. For example, the string “Calc” could be produced by the expression eval("Ca" + "lc") or similar games. Finally, there may be other injection attacks that have nothing to do with selecting values from Calc, in which case any of these versions of sanitize would not be very useful.

[The following question is unrelated to the previous question.]

Alice says, “If you use a google.com URL inside a <script> tag to load jquery from a Google site, that’s a cross-site scripting attack.”

Bob replies, “No, you can’t even use jquery from Google because of the same-origin policy. So it’s not a problem.”

4. Who is right or wrong?
   A. They are both right.
   B. Alice is right, Bob is wrong.
   C. Alice is wrong, Bob is right.
   D. They are both wrong.

**Solution. D.** It’s a common practice to use an unrelated remote site for fetching code like jquery. And if you deliberately choose to do it, it’s not an attack on you. The same-origin policy attempts to keep data from leaking, but doesn’t try to prevent code sharing... so this usage is fine.