Question 1: Cross-Site Request Forgery

Cross-site request forgery (XSRF) allows a malicious person to make fake HTTP requests from a victim client to a victim server.

Suppose that Eve has a social media account and posts the following image:

```
<img src=‘‘http://www.suchsecurebanking.com/send?amount=9999&beneficiary=Eve’’/>
```

The result is that anyone who loads Eve's account will automatically load the image, which is actually an HTTP request for suchsecurebanking.com. That is, suppose that Alice visits Eve's site, then Alice loads

```
http://www.suchsecurebanking.com/send?amount=9999&beneficiary=Eve
```

1. What are the two arguments / parameters for the "send" HTTP query?
2. If Eve is the beneficiary, who is the person who sends the money?
3. How might suchsecurebanking remember "who" is trying to send money when the send query is issued?
4. Describe in steps the legitimate standard process that occurs when Alice wants to send money to Bob. How does Alice interact with the bank and what information is sent? (You can use concepts in lieu of specifics, e.g., Alice *authenticates* with the bank.)
5. What are defenses against this attack?

Question 2: More Cross-Site Request Forgery

True to their name, suchsecurebanking.com urgently patched their vulnerability. To do so they hired a highly-paid security consultant from Infinity-Plus-One-Percent-Secure (LLC). The consultant cringed seeing that the bank allowed HTTP connections and requires that everything is done over HTTPS. They also "fixed" the XSRF vulnerability by adding a random XSRF token to the sensitive `send` request. It now looks like this:

```
https://www.su....ng.com/send?amount=9999&beneficiary=Eve&token=<random>
```

The XSRF token is chosen randomly and separately for each user and is embedded as a hidden element on the HTML provided by the server to the logged-in user.

Eve plays around with the system and discovered an interesting observation. If she loads the URL

```
https://www.su...ng.com/welcome?name=<script>alert(‘‘a ha!’’);</script>
```

she gets a nice welcome screen without a name written on it but instead an alert pop up on her screen that says "a ha!". She uses this to devise an attack to continue stealing money from those who read her comments.

1. What kind of attack is the welcome page vulnerable to?
2. Eve uses this attack to bypass the XSRF token defense. She replaces the alert script with something else. What should it do? Why does it work?