Chapter 9: Cross-Site Request Forgery Attack

Problems

9.1. Explain why the same-site cookie can help prevent CSRF attacks.

9.2. Explain how a website can use secret token to prevent CSRF attacks, and why does it work?

9.3. These days, most of the websites use HTTPS, instead of HTTP. Do we still need to worry about CSRF attacks?

9.4. Using LiveHTTPHeader, we find out that the following GET request is used to send an HTTP request to `www.example.com` to delete a page owned by a user (only the owner of a page can delete the page).

```
http://www.example.com/delete.php?pageid=5
```

Please construct a simple malicious web page, so when a victim visits this web page, a forged request will be launched against `www.example.com` to delete a page belonging to the user.

9.5. Using LiveHTTPHeader, we find out that the following POST request is used to send an HTTP request to `www.example.com` to delete a page owned by a user (only the owner of a page can delete the page).

```
http://www.example.com/delete.php
...
pageid=5
```

Please construct a simple malicious web page, so when a victim visits this web page, a forged request will be launched against `www.example.com` to delete a page belonging to the user.

9.6. In a request, there is an user id, which is a random number generated by the server. The ID information can be found from the user's page from the server. If attacker does no know this user ID, can he/she still launch an CSRF attack on this service?

9.7. Do browsers know whether an HTTP request is cross-site or not?

9.8. Does servers know whether an HTTP request is cross-site or not?

9.9. Why cannot a web server use the referer header to tell whether a request is cross-site or not?

9.10. Why is it important for a server to know whether a request is cross-site or not?

9.11. Can we simply ask browsers not to attach any cookie for cross-site requests?

9.12. ★★★

If a page from `www.example.com` contains an `iframe`, inside which a facebook page is displayed. If a request is sent from inside the `iframe`, is it considered as a cross-site request or not? If not, how can be this secured?