

Hacking Web App like a pimp

a different approach for web app pentest proxies

@_hugsy_

\$ who

- By day *Christophe Alladoum*
 - Infosec consultant for Sense of Security (we're hiring !!!)
 - web app/external/internal pentests, code review, etc.
- By night *hugsy* (no questions 🤪)
 - Do stuff
 - Low-level, RE, CTF addict, research, tool dev

\$ why

- Burp is THE reference tool for web app pentest proxy
 - Useful modules (Proxy, Repeater, Intruder (\$\$\$), etc.)
 - Well structured for comm. between modules
 - Nice GUI
- But it is limited at MANY levels
 - Heaps and heaps of libs on top of Java VM
 - performance --
 - Works well for traditional web apps, almost useless for unconventional ones

\$ why



Forward Drop Intercept is on Action Comment this item

Raw Params Headers Hex

```
POST /api/parse HTTP/1.1
Host: 10.10.3.47
User-Agent: Mozilla/5.0 (<a href="#" onmouseover="alert('UA2')">UA</a>)
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Cookie: PHPSESSID=elogkvnlh76i4oklbkithlm2bb0
X-Originating-IP: 127.0.0.1
X-Remote-IP: 127.0.0.1
X-Requested-With: XMLHttpRequest
X-Same-Domain: 1
DNT: 1
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache
Content-Length: 234
Content-Type: application/x-www-form-urlencoded

eyJvYmpFaWQiOjA0NSwgInRva2VuIjogIjNhOGNhYTRlNmIzZjZjZWQ3YTgwMmRhODI40TdjMzc5ZWM4NzYyZWI1NTc3M2I0ZkxNjNhZmJlZmE3Zjc0ZDUtLCAic
nvxdwVzdCIzW5kQ29tbWFuZCIsICJhcmdzIjogeyJxIjogIlFVRkJRVUZCUVVGqlFVRkJRVUZCUVVGqlFVRkJRVUZCUVE9PSJ9FQo==
```

Burp does not handle those cases...

... like at all

```
Content-Length: 234
Content-Type: application/x-www-form-urlencoded

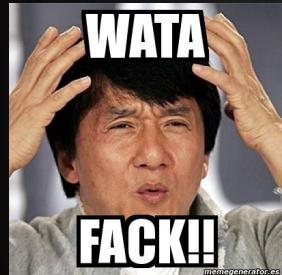
{"obj_id": 45, "token": "3a8caa4e6b3f6ced7a802da82897c379ec8762eb55773b4d9163afbefa7f74d5", "request": "sendCommand",
"args": {"1": "QUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQQ=="}}
```

Ok great, let's write a Burp plugin

\$ why

Because

- Writing plugins for Burp is insanely complicated
 - To make it “simpler”, Burp implements unnatural bindings
 - Ruby over Java
 - Python over Java
- Awesome tools exist (like *burst* or *mitmproxy*), very extensible **BUT** specific to one language
- And (probably most importantly) I like having my own tools



So I wrote a tool...



Introducing *proxenet*

From Ancient Greek, πρόξενος (próksenos, "public guest").

- A negotiator ; a factor ; a go-between.
- A mediator involved in immoral bargains (see pimp).

Goal

- Building a hacker friendly proxy easily pluggable.
- Micro-kernel approach: core does as little as possible, modules do the rest.

\$ info Proxenet

- Plugin driven proxy for pentests
 - Bottom line : proxenet does **NOTHING** to HTTP layer
- 100% pure C code
 - Crazy fast
 - POSIX multi-threading
 - Super low memory use (w/ 5 VMs loaded ⇒ ~ $\frac{1}{5}$ Burp mem use)
- Can view / edit / transform any request via plugins
 - Multiple linked list mechanism prevents overlap between plugins

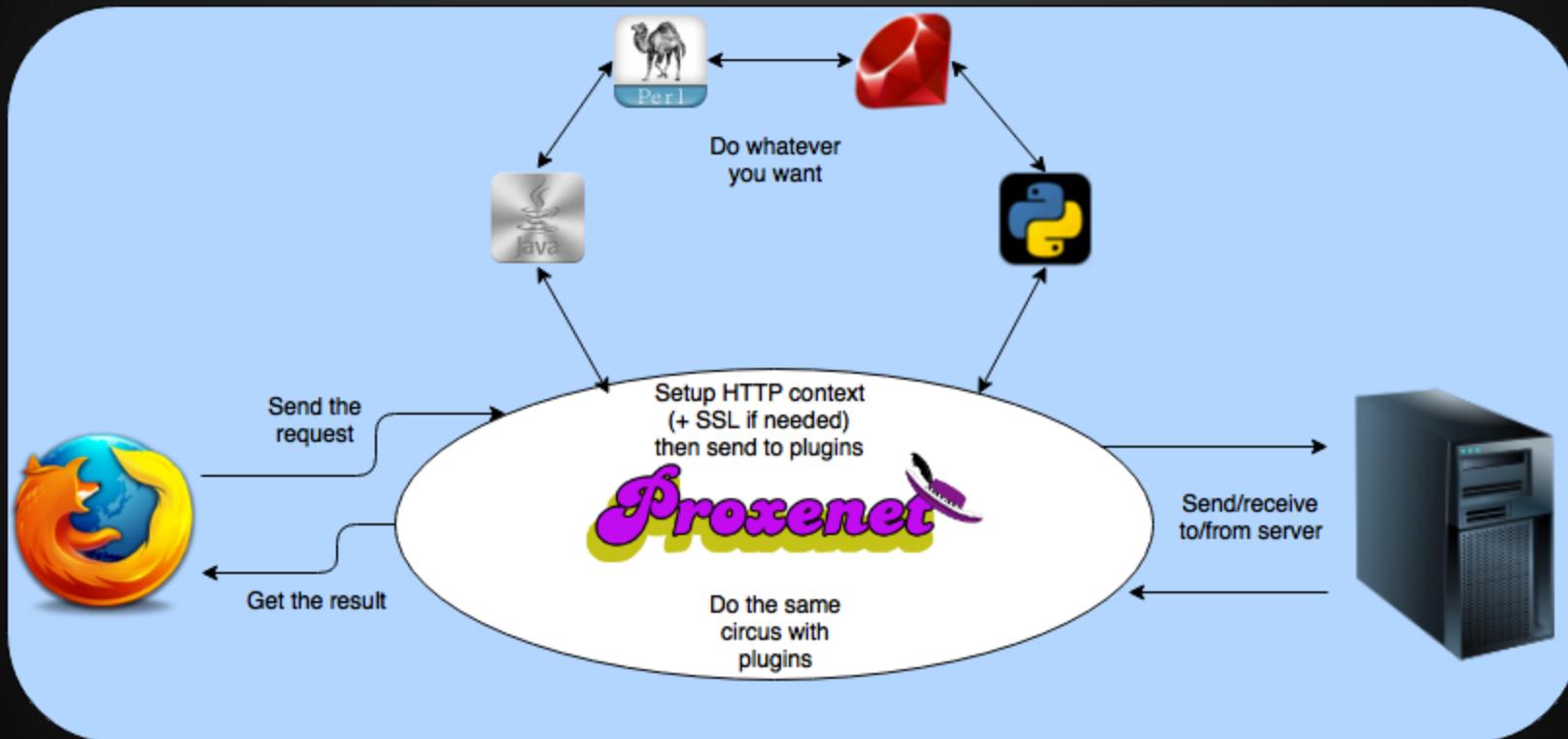


and more...

\$ info Proxenet

- Some of the features
 - Full POSIX (Linux, OpenBSD, FreeBSD, OSX*)
 - Lightweight SSL/TLS
 - Full SSL/TLS interception (internal CA)
 - SSL/TLS client certificate authentication support
 - Future ready (i.e. IPv6 support)
 - HTTP Proxy forwarding
 - Regex filtering (white / black list)
 - Priority mechanism
 - Cool logo and nice '90 style colors
 - etc.
- Can be chained behind your favourite proxy (Burp, Zap, Proxystrike, etc.)

The Big Picture



DIY

- Meant to be ***extremely*** easily to add new plugins
 - If you think it's hard, it means I failed 😞
- To be valid a plugin **must** have:
 - One request function (default: `proxenet_request_hook`)
 - One response function (default: `proxenet_response_hook`)
- Use the rest of the code for whatever you like (other functions, tests, etc.)

DIY

The hook functions are called by the core with 3* arguments:

1. A request/response identifier (Integer)
2. The request/response itself (Byte[] or String)
3. The URL (String)

The hook functions must return:

1. The modified request/response (Byte[] or String)
2. Null/None in case of error

DIY

- Literally as simple as it gets (yeah seriously)
 - Skeletons for each supported language in the RTFD

```
public class MyPlugin
{
    public static String AUTHOR = "";
    public static String PLUGIN_NAME = "";

    public static byte[] proxenet_request_hook(int request_id, byte[] request, String uri){
        return request;
    }

    public static byte[] proxenet_response_hook(int response_id, byte[] response, String uri){
        return response;
    }

    public static void main(String[] args){
        return;
    }
}
```

```
AUTHOR = ""
PLUGIN_NAME = ""

proc proxenet_request_hook {request_id request uri} {
    return $request
}

proc proxenet_response_hook {response_id response uri} {
    return $response
}

# add test cases here
```

```
AUTHOR = ""
PLUGIN_NAME = ""

function proxenet_request_hook (request_id, request, uri)
    return request
end

function proxenet_response_hook (response_id, response, uri)
    return response
end
```

```
module MyPlugin

    $AUTHOR = ""
    $PLUGIN_NAME = ""

    def proxenet_request_hook(request_id, request, uri)
        return request
    end

    def proxenet_response_hook(response_id, response, uri)
        return response
    end

end

if __FILE__ == $0
    # use for test cases
end
```

DIY

```
1 package burp;
2
3 import IBurpExtender.*;
4 import ISessionHandlingAction.*;
5 import IParameter.*;
6 import java.util.Arrays;
7
8 public class BurpExtender implements IBurpExtender, ISessionHandlingAction
9 {
10
11     public void registerExtenderCallbacks(IBurpExtenderCallbacks callbacks)
12     {
13         callbacks.getHelpers().setExtensionName("BurpAddHeader");
14         callbacks.getHelpers().registerSessionHandlingAction();
15         return;
16     }
17
18     public void performAction(IHttpRequestResponse currentRequest,
19                             IHttppRequestResponse[] macroItems)
20     {
21
22         String newHeader;
23         IExtensionHelpers helpers;
24         IRequestInfo reqInf;
25         List<String> headers;
26         byte[] newHttpReq;
27         byte[] body;
28
29         newHeader = "X-Java-Injected: burp";
30         helpers = callbacks.getHelpers();
31         reqInf = helpers.analyzeRequest(currentRequest);
32         headers = reqInf.getHeaders();
33         body = java.util.Arrays.copyOfRange(currentRequest.getRequest(),
34                                           reqInf.getBodyOffset(),
35                                           currentRequest.getRequest().length);
36
37         headers.add( newHeader ); // this is actually the only line that matters
38
39         newHttpReq = helpers.buildHttpMessage(headers,
40                                              currentRequest.getRequest());
41
42         currentRequest.setRequest(newHttpReq);
43         return;
44     }
45 }
```

U:@-- BurpAddHeader.java All of 1.5k (8,0) (Java/l pair Abbrev)

Written in ~1h (API search, test, debug, etc.)

```
1 public class AddHeader
2 {
3     public static byte[] proxenet_request_hook(int request_id, byte[] request, String uri)
4     {
5         String myRequest = new String( request );
6         myRequest = myRequest.replace("\r\n\r\n", "\r\nX-Java-Injected: proxenet\r\n\r\n");
7         return myRequest.getBytes();
8     }
9
10    public static byte[] proxenet_response_hook(int response_id, byte[] response, String uri)
11    {
12        return response;
13    }
14 }
```

U:@-- ProxenetAddHeader.java All of 441 (15,0) (Java/l pair Abbrev)

Written in ~5min (trivial API, relies on primitive types and objects)

DIY

Some plugins already implemented :

- Interceptor GUI (based on PyQt4)
- Interceptor CLI (invoking *burst* from *proxenet*)
- Attacks plugins
 - automatic (stealth) ShellShock detection
 - CVE-2012-1823 (php-cgi code exec)
 - XXE
- Discovery plugins
 - automatic discovery on Directory Listing
 - retrieve all comments in HTML
 - add automatically parameters to all requests (&debug=1, &admin=1)

The screenshot shows the proxenet GUI interface. At the top, there's a header bar with tabs for Raw View, JSON View, XML View, ViewState View, and Options. Below that is a status bar with the message "This frame displays the body content as Raw". The main area contains a text input field with the following content:
a=b&b=c&t=x&_VIEWSTATE=%2fEPDwJkMTQ2OTkzNDMyMWRkOWAfnFeQcY9pkVCluHbdA6WB0%3d

The screenshot shows the proxenet CLI interface. It starts with the command "/code/proxenet = proxenet -v". The session then continues with several commands and their responses:
INFO: Intercepting only matching '*'
INFO: Control interface listening on '/tmp/proxenet-control-socket'
INFO: Control socket: 3
INFO: Control host: localhost:8888
INFO: Bind socket: 4
INFO: Plugins loaded
INFO: Plugin 'proxenet-dlgsBro' is now ACTIVE
INFO: Plain request 1 to http://192.168.56.102:80/ #6
INFO: Established socket to '192.168.56.102:80' #6
INFO: plain request to '192.168.56.102:80'
Loading...
[REDACTED]
Burst 0.6, Copyright (c) 2014 Thibaud Weksteen
Your request RID=1 ('http://192.168.56.102:80/h.php') is in variable 'http'.
To send the modified request, enter 'quit'; to send the original request, enter 'reset'.
>>> print http
GET /h.php HTTP/1.1
Host: 192.168.56.102
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:40.0) Gecko/20100101 Firefox/40.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
Cache-Control: max-age=0

>>> help(http)
>>> http.add_header("X-Added-By", "burst/proxenet")
>>> print http
GET /h.php HTTP/1.1
Host: 192.168.56.102
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:40.0) Gecko/20100101 Firefox/40.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
Cache-Control: max-age=0
X-Added-By: burst/proxenet

>>> quit
INFO: End of request 1, cleaning context

DIY



Demo

\$ apt-get upgrade

- Keep improving code robustness & quality (core)
 - even though it's not too bad right now

Proxenet Scan Report

CHECKMARX

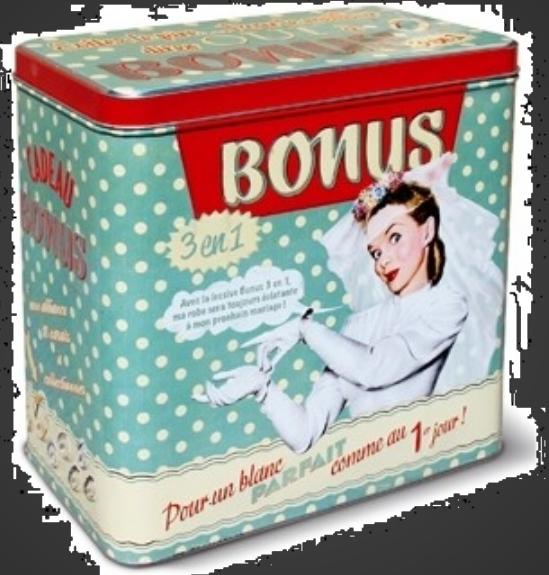
Project Name	Proxenet
Scan Start	Monday, July 13, 2015 3:07:40 PM
Preset	Default 2014
Scan Time	00h:02m:04s
Lines Of Code Scanned	14,121
Files Scanned	38
Report Creation Time	Monday, July 13,

	High	Medium	Low
To Verify	5	80	52
Not Exploitable	0	0	0
Confirmed	0	0	0
Urgent	0	0	0
Total	5	80	52



\$ apt-get upgrade

- Add more features
 - JavaScript (v8) coming up
 - oCaml too (because why not?)
 - WebSockets maybe (because why not?)
- Add more plugins for/from community
 - ~10 publicly released plugins (intercept, XXE detection, dir list, etc.)
 - Feel free to contribute !!
 - Write & submit funky plugins
 - Report bugs (beerz 4 bugz policy)



“proxenet-in-the-middle”© attack

Idea (from @lanjelot):

*Why not use the plugins created
for proxenet for easily view/modify
HTTP streams in MITM attacks?*

CHALLENGE ACCEPTED



“proxenet-in-the-middle”© attack

1. Use Responder

- Poison LLMNR on LAN
- Poison PAC (from WPAD requests) to proxenet

```
[+] HTTP Options:
    Always serving EXE [OFF]
    Serving EXE [ON]
    Serving HTML [OFF]
    Upstream Proxy [ON]

[+] Poisoning Options:
    Analyze Mode [OFF]
    Force WPAD auth [OFF]
    Force Basic Auth [OFF]
    Force LM downgrade [OFF]
    Fingerprint hosts [OFF]

[+] Generic Options:
    Responder NIC [vboxnet0]
    Responder IP [192.168.56.1]
    Challenge set [1122334455667788]
    Upstream Proxy [192.168.56.1:8008]

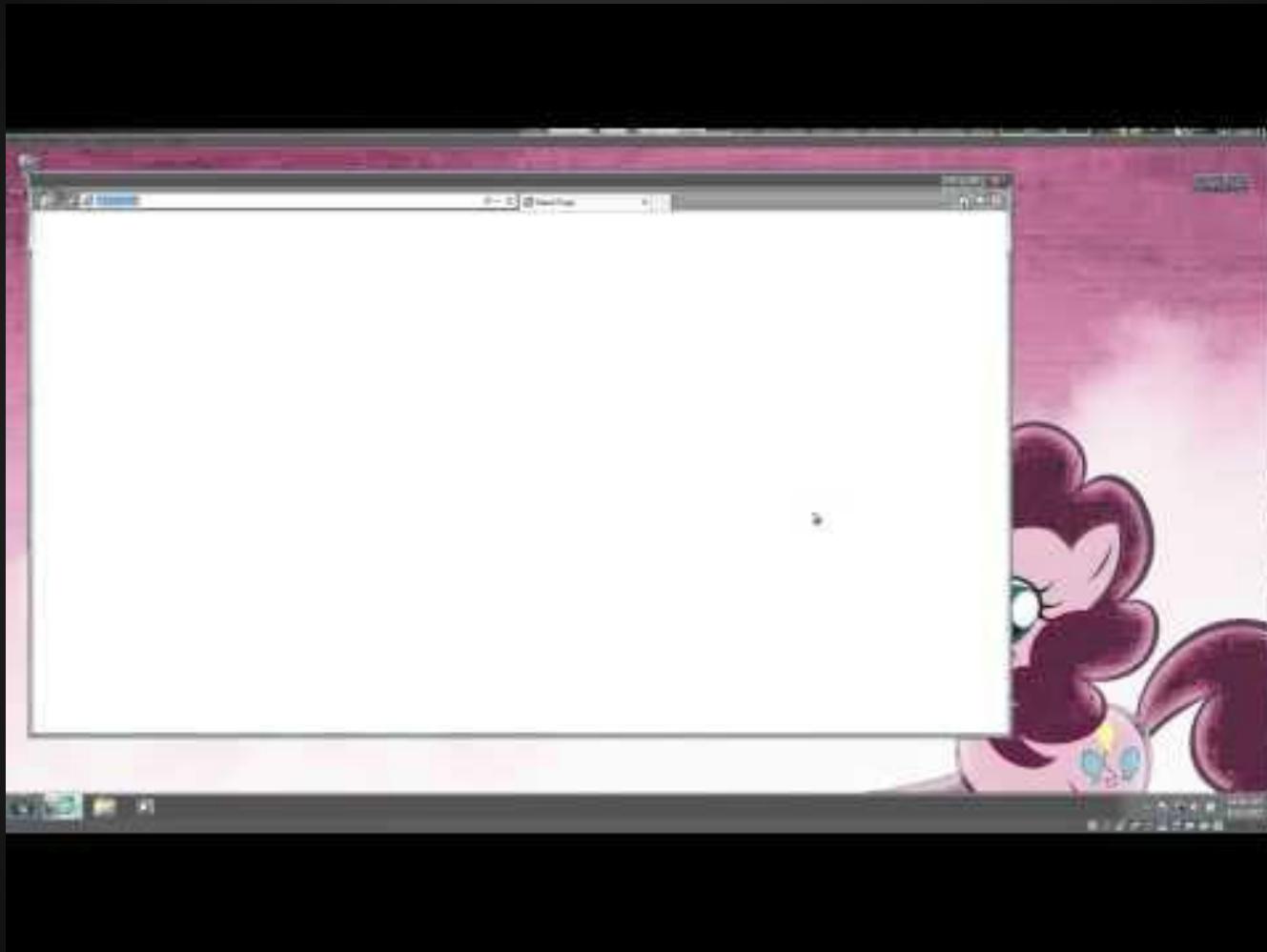
[+] Listening for events...
[*] [NBT-NS] Poisoned answer sent to 192.168.56.101 for name WWW.BING.COM (service: Workstation/Redirector)
[*] [NBT-NS] Poisoned answer sent to 192.168.56.101 for name WPAD (service: Workstation/Redirector)
[*] [NBT-NS] Poisoned answer sent to 192.168.56.101 for name WWW.BING.COM (service: Workstation/Redirector)
[*] [NBT-NS] Poisoned answer sent to 192.168.56.101 for name WWW.BING.COM (service: Workstation/Redirector)
[*] [NBT-NS] Poisoned answer sent to 192.168.56.101 for name YAHOO.COM (service: Workstation/Redirector)
```

2. Use your proxenet plugin to modify HTTP on-the-fly

- Insert JavaScript (BeEF) in HTML body
- Replace on-the-fly documents (zip, doc, xls, etc.) with infected ones

```
INFO: New request 142 to 'http://www.wncchs.org:80/From_New_Nurse_Practitioner_to_Primary_Care_Provider.docx'
INFO: Established socket to 'www.google.com.au:80': #
INFO: plain request to 'www.google.com.au:80'
INFO: Established socket to 'www.wncchs.org:80': #
INFO: plain request to 'www.wncchs.org:80'
INFO: New payload generated for type 'docx' : /tmp/tmp3l4zMG.docx.exe
Poisoning response 142 with format 'docx'
```

Demo



“proxenet-in-the-middle”© attack



- Easy & practical attack for internal pentests
- Totally transparent for victims
- Free shells \o/

Thank you web browser auto-configuration mode



Thanks

proxenet repo:

<https://github.com/hugsy/proxenet.git>

plugins public repo:

<https://github.com/hugsy/proxenet-plugins.git>



@_hugsy_



hugsy



hugsy@blah.cat