

# Burp Suite Extension Writing Workshop

DevSecCon Singapore

March 01, 2019

Sanoop Thomas & Samandeep Singh



Thank you for showing up.

**Thank You**

# About Us

## » Sanoop Thomas

- Organizer – BSides Singapore
  - <https://bsidessg.org/>
- 9+ years in Information Security
- Created Halcyon IDE – an IDE for Nmap Script Developers
  - <https://halcyon-ide.org/>
- Hosting SecTools Podcast at Infosec Campus
  - <https://infosecampus.com/>
- Barely get time to write blogpost, but presentations/talks are updated
  - <https://devilslab.in/>
- Presented at OWASP India, Nullcon Goa, HITBGSEC, ROOTCON, BlackHat Arsenal (USA, Asia) and Defcon
- Tweet me at - @**s4n7h0**

# About Us

- » Samandeep Singh
  - Principal Security Consultant at SEC Consult Singapore
  - OSCE, OSCP, CRT
  - Co-Organizer – BSides Singapore
  - 6+ years in Information Security
  - Author of XVWA – a WebSec learning app
  - Interested in SDR, Windows Exploitation, Mainframe Hacking
  - Licensed scuba and sky diver
  - Adventure sports and traveller in free time
  - Tweet me at - @**samanL33T**

# Agenda

- » Introduction to Burp Suite
- » Burp features:
  - scoping, proxy settings, repeater mode and other options
- » Extender Capabilities
- » Setting up development environment
- » Important/useful methods
- » Debugging tips
- » Packaging tips
- » Extensions will be Developed in Java and Python

## **Schedule**

10:20 - 12:45

1 hour - lunch break

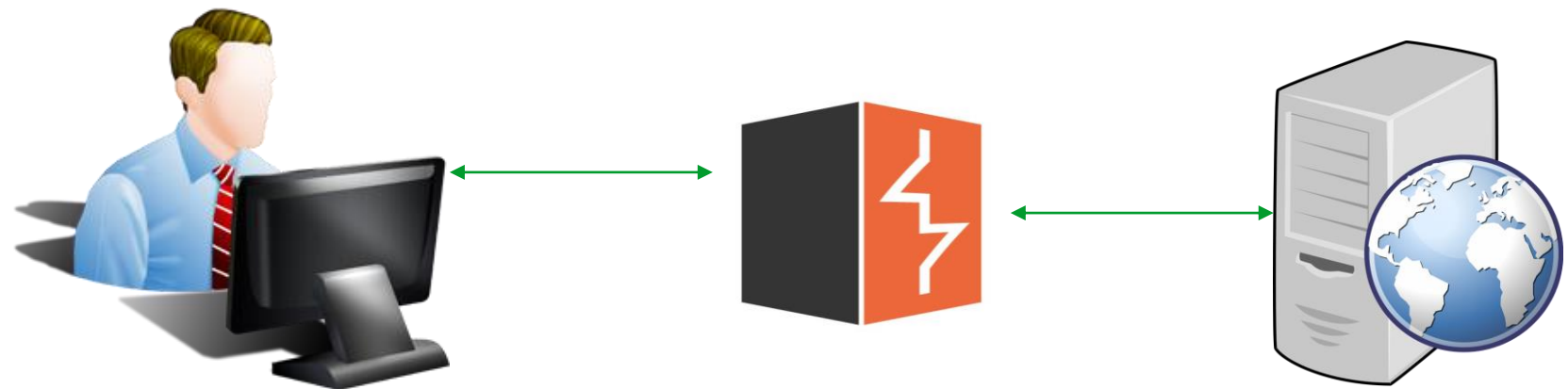
13:45 - 15:00

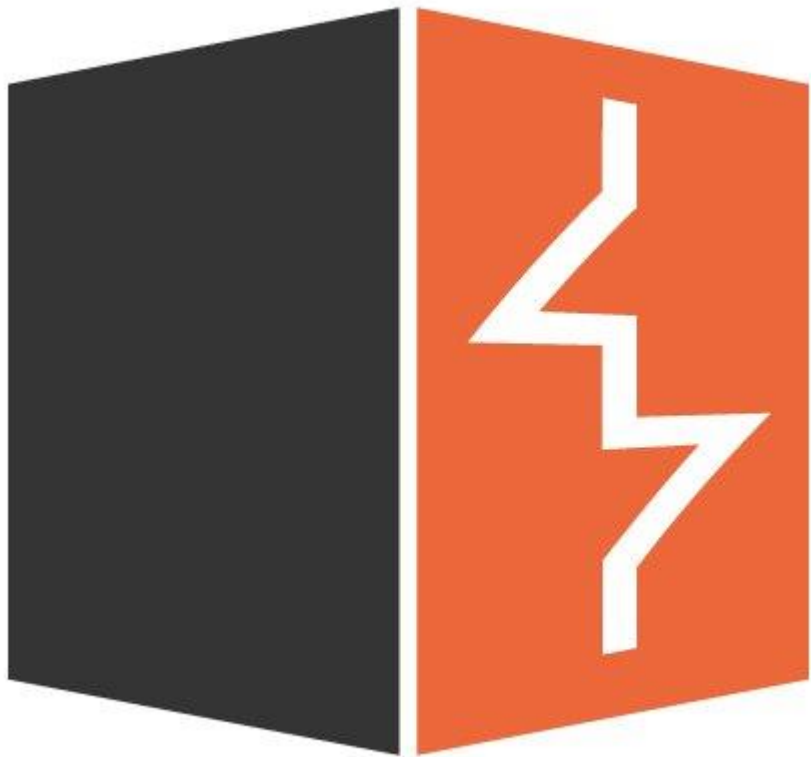
## About the workshop

- » Kickstarter to burp extension development; not a java or python course
- » This is a trimmed version of our 2 day extensive development workshop
- » Hands-on development. i.e., we code and debug together here
- » Improve your code if you see a chance
- » Mistakes are okay. We learn from this more
- » Questions are welcome at any given time. Feel free to interrupt

# Introduction to Burp Suite

- » The de-facto tool for application security testers
- » Application proxy with a bunch of additional capabilities
- » Burp features
  - Proxy
  - Scanner (Pro)
  - Intruder
  - Repeater
  - Sequencer
  - Decoder
  - Extender





## Exercise #1:

Burp Suite crash course

### Objective:

- » Getting started with burp
- » Setting up browser
- » Working with intercept options, repeater and more
- » Know what to automate using burp extensions
- » Analyzing request/response pairs



# Burp Extensions

- » API to Burp functionalities
- » Extensions can be written in:
  - Java
  - Python (JPython)
  - Ruby (JRuby)

## Extender Capabilities

- » Process and modify HTTP requests and responses for all Burp tools.
- » Access key runtime data, such as the Proxy history, target site map, and Scanner issues.
- » Initiate actions like scanning and spidering.
- » Implement custom scan checks and register scan issues.
- » Customize the placement of attack insertion points within scanned requests.
- » Provide custom Intruder payloads and payload processors.
- » Query and update the Suite-wide target scope.
- » Query and update the session handling cookie jar.
- » Implement custom session handling actions.

## Extender Capabilities (cont..)

- » Add custom tabs and context menu items to Burp's user interface.
- » Use Burp's native HTTP message editor within your own user interface.
- » Customize Burp's HTTP message editor to handle data formats that Burp does not natively support.
- » Analyze HTTP requests and responses to obtain headers, parameters, cookies, etc.
- » Build, modify and issue HTTP requests and retrieve responses.
- » Read and modify Burp's configuration settings.
- » Save and restore Burp's state.

# BApp Store

The screenshot shows the Burp Suite Community Edition v1.7.36 interface. The 'BApp Store' tab is active, displaying a list of extensions. The '.NET Beautifier' extension is selected, and its details are shown in the right-hand pane.

Name	Installed	Rating	Popularity	Last updated	Detail
.NET Beautifier		☆☆☆☆☆		23 Jan 2017	
Active Scan++		☆☆☆☆☆		04 Sep 2018	Requires Burp...
Add & Track Custom Iss...		☆☆☆☆☆		16 Jan 2018	Requires Burp...
Add Custom Header		☆☆☆☆☆		18 Sep 2018	
Additional CSRF Checks		☆☆☆☆☆		09 Jan 2018	
Additional Scanner Checks		☆☆☆☆☆		12 Jan 2017	Requires Burp...
AES Payloads		☆☆☆☆☆		28 Aug 2015	Requires Burp...
Attack Surface Detector		☆☆☆☆☆		10 Oct 2018	
AuthMatrix		☆☆☆☆☆		02 Feb 2018	
Authz		☆☆☆☆☆		01 Jul 2014	
Auto Repeater		☆☆☆☆☆		04 Apr 2018	
Authorize		☆☆☆☆☆		09 Jul 2018	
AWS Security Checks		☆☆☆☆☆		18 Jan 2018	Requires Burp...
Backslash Powered Sca...		☆☆☆☆☆		10 Aug 2018	Requires Burp...
Batch Scan Report Gene...		☆☆☆☆☆		03 Oct 2017	Requires Burp...
Blazer		☆☆☆☆☆		01 Feb 2017	
Bradamsa		☆☆☆☆☆		02 Jul 2014	
Brida, Burp to Frida bridge		☆☆☆☆☆		04 Oct 2018	
Browser Repeater		☆☆☆☆☆		01 Jul 2014	
Buby		☆☆☆☆☆		14 Feb 2017	Requires Burp...
Burp Chat		☆☆☆☆☆		23 Jan 2017	
Burp CSJ		☆☆☆☆☆		23 Mar 2015	
Burp-hash		☆☆☆☆☆		28 Aug 2015	Requires Burp...
BurpSmartBuster		☆☆☆☆☆		22 Jan 2018	
Bypass WAF		☆☆☆☆☆		29 Mar 2017	
Carbonator		☆☆☆☆☆		23 Jan 2017	Requires Burp...
Cloud Storage Tester		☆☆☆☆☆		05 Oct 2017	Requires Burp...
CMS Scanner		☆☆☆☆☆		03 Oct 2017	Requires Burp...
CO2		☆☆☆☆☆		20 Jul 2017	
Code Dx		☆☆☆☆☆		06 Jun 2018	Requires Burp...
Collaborator Everywhere		☆☆☆☆☆		21 May 2018	Requires Burp...
Command Injection Attac...		☆☆☆☆☆		27 Jun 2018	
Commentator		☆☆☆☆☆		16 Jul 2018	
Content Type Converter		☆☆☆☆☆		23 Jan 2017	
Copy as Node Request		☆☆☆☆☆		10 Nov 2017	
Copy as PowerShell Req...		☆☆☆☆☆		31 Jan 2018	

**.NET Beautifier**

This extension beautifies .NET requests to make the body parameters more human readable. Built-in parameters like \_\_VIEWSTATE have their values masked. Form field names have the auto-generated part of their name removed.

Requests are only beautified in contexts where they can be edited, such as the Proxy intercept view.

For example, a .NET request with the following body:

```
__VIEWSTATE=%20s1AIHfiohsoiGjKlASgJghajklGjSDGsJdg1SDJg9SDJGsdGjSGJDD
Sasdfja9sdjfasdfja0sdfja
... [1000 lines later] ...
&ctl100%24ctl100%24InnerContentPlaceHolder%24Element_42%24ctl100%24FrmLo
gin%24TxtUsername_intern
al=username&ctl100%24ctl100%24InnerContentPlaceHolder%24Element_42%24ct
100%24FrmLogin%24TxtPass
word_internal=password&ctl100%24ctl100%24InnerContentPlaceHolder%24Elem
ent_42%24ctl100%24BtnLogi
n=Login
```

will be displayed like this:

```
__VIEWSTATE=%&TxtUsername_internal=username&TxtPassword_internal=passw
ord&BtnLogin=Login
```

This is done without compromising the integrity of the underlying message so you can edit parameter values and the request will be correctly reconstructed. You can also send the beautified messages to other Burp tools, and they will be handled correctly.

**Author:** Nadeem Douba  
**Version:** 0.3  
**Source:** <https://github.com/portswigger/dotnet-beautifier>  
**Updated:** 23 Jan 2017

**Rating:** ☆☆☆☆☆

**Popularity:**

<https://portswigger.net/bappstore>

# Lab Setup

- » Burp Suite Community Edition
  - <https://portswigger.net/burp/>
- » Extender APIs
- » Test Application
- » Development Environments
  - Java
  - Python
  
- » Test application : <http://10.191.35.228/xvwa>

# Extender APIs

The screenshot shows the Burp Suite Community Edition v1.7.36 interface. The main window displays the 'Burp Extender APIs' section. At the top, there is a navigation bar with tabs for 'Target', 'Proxy', 'Spider', 'Scanner', 'Intruder', 'Repeater', 'Sequencer', 'Decoder', 'Comparer', 'Extender', 'Project options', 'User options', and 'Alerts'. Below this, there are sub-tabs for 'Extensions', 'BApp Store', 'APIs', and 'Options'. The 'APIs' tab is active, showing a list of interfaces on the left and their corresponding Java code on the right.

**Burp Extender APIs**

You can use the Burp Extender APIs to create your own extensions to customize Burp's behavior.

**IBurpCollaboratorClientContext**

```
package burp;

/**
 * @(#) IBurpCollaboratorClientContext.java
 *
 * Copyright PortSwigger Ltd. All rights reserved.
 *
 * This code may be used to extend the functionality of Burp Suite Community Edition
 * and Burp Suite Professional, provided that this usage does not violate the
 * license terms for those products.
 */
import java.util.List;

/**
 * This interface represents an instance of a Burp Collaborator client context,
 * which can be used to generate Burp Collaborator payloads and poll the
 * Collaborator server for any network interactions that result from using those
 * payloads. Extensions can obtain new instances of this class by calling
 * <code>IBurpExtenderCallbacks.createBurpCollaboratorClientContext()</code>.
 * Note that each Burp Collaborator client context is tied to the Collaborator
 * server configuration that was in place at the time the context was created.
 */
public interface IBurpCollaboratorClientContext
{
    /**
     * This method is used to generate new Burp Collaborator payloads.
     *
     * @param includeCollaboratorServerLocation Specifies whether to include the
     * Collaborator server location in the generated payload.
     * @return The payload that was generated.
     *
     * @throws IllegalStateException if Burp Collaborator is disabled
     */
    String generatePayload(boolean includeCollaboratorServerLocation);

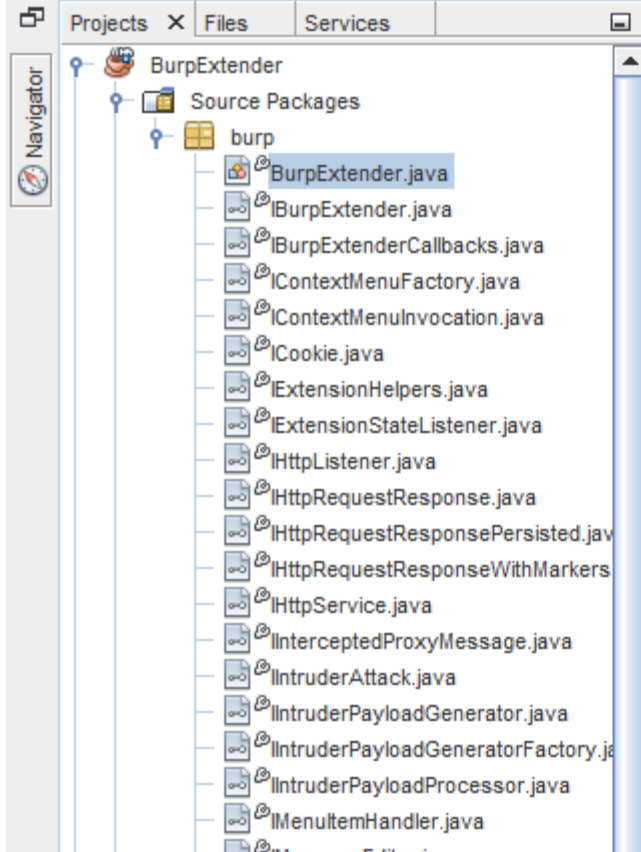
    /**
     * This method is used to retrieve all interactions received by the
     * Collaborator server resulting from payloads that were generated for this
     * context.
     *
     * @return The Collaborator interactions that have occurred resulting from
     * payloads that were generated for this context.
     */
    List<IBurpCollaboratorInteraction> getInteractions();
}
```

0 matches

Save interface files Save Javadoc files

## Setup for Java

- » Create a project in your favourite IDE
- » Create a package "burp" and copy all Extender API files
- » Create a java class "BurpExtender"
- » Download NetBeans project
  - <https://portswigger.net/burp/extender/examples/emptyextension.zip>
- » This workshop will be using NetBeans version 8.2



```

1  package burp;
2
3  public class BurpExtender implements IBurpExtender
4  {
5      @Override
6      public void registerExtenderCallbacks (IBurpExtenderCallbacks callbacks)
7      {
8          // your extension code here
9      }
10 }
11

```



# Setup for Python

- » Download Jython
  - <http://www.jython.org/downloads.html>
- » Sublime Text
  - <https://www.sublimetext.com/>

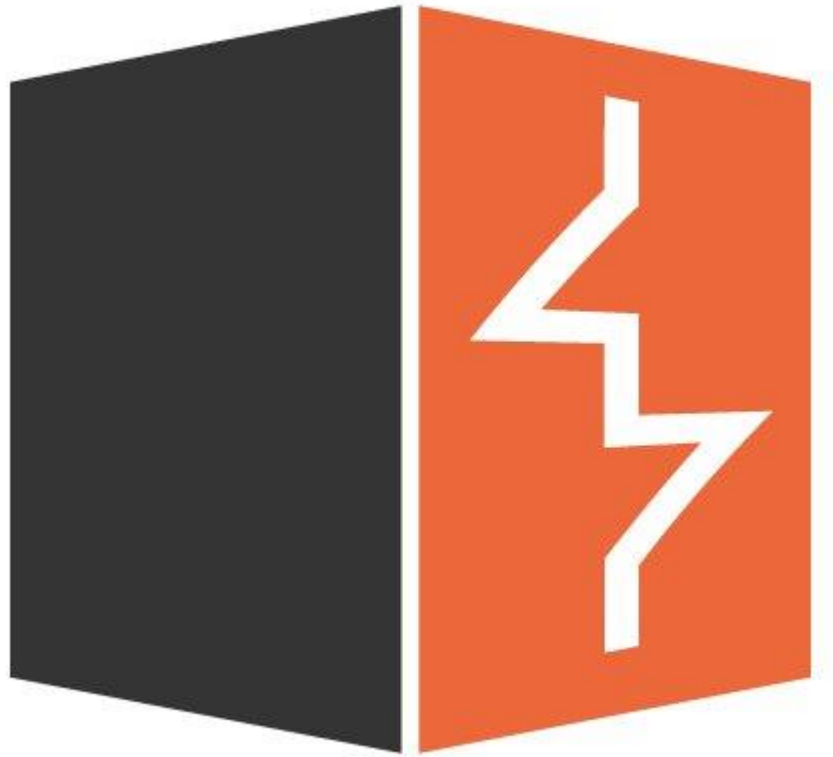
```
1 #Basic Burp Plugin- Python
2 from burp import IBurpExtender
3
4 class BurpExtender(IBurpExtender):
5     def registerExtenderCallbacks(self, callbacks):
6         return
7
```

**Note:** Because of the way in which Jython and JRuby dynamically generate Java classes, you may encounter memory problems if you load several different Python or Ruby extensions, or if you unload and reload an extension multiple times. If this happens, you will see an error like:

```
java.lang.OutOfMemoryError: PermGen space
```

You can avoid this problem by configuring Java to allocate more PermGen storage, by adding a -XX:MaxPermSize option to the command line when starting Burp. For example:

```
java -XX:MaxPermSize=1G -jar burp.jar
```



## Exercise #2:

Setting up development environment

### Objective:

- » Setup environment for Java
- » Setup environment for Python

# IBurpExtender

- » All extensions must implement this interface.
- » BurpExtender must implement this interface with one method
  - `void registerExtenderCallbacks (IBurpExtenderCallbacks callbacks)`
- » Burp invokes registerExtenderCallbacks() method when an extension is loaded

# IBurpExtenderCallbacks

- » Consists a set of callback methods for extensions to perform various actions within Burp
- » Used to register extension settings
- » This interface can handles stdout and stderr

# IEExtensionHelpers

- » Build, analyse and modify HTTP requests and responses
- » Toggle request's method between GET and POST
- » Do encoding/decoding to request/response data
- » Constructs scanner insertion point

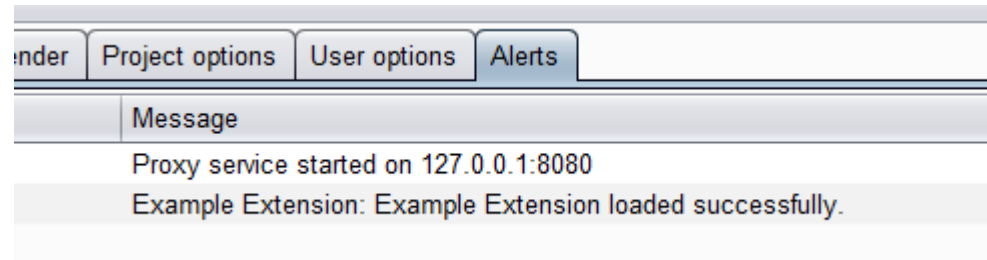
## Handling stdout/stderr

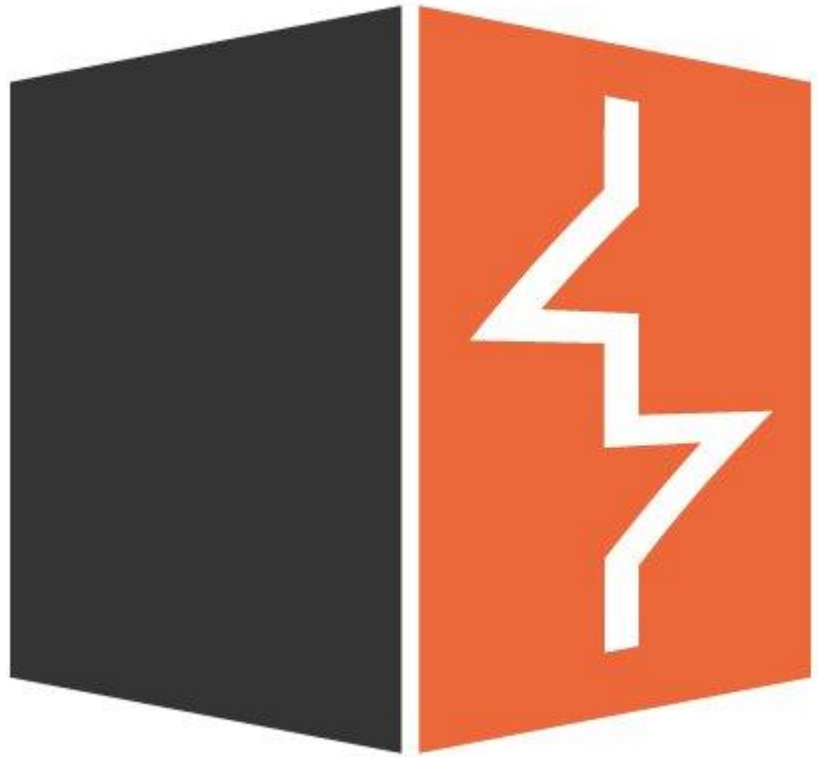
- » `callbacks.getStderr()`
- » `callbacks.getStdout()`

```
private IExtensionHelpers helpers;  
private PrintWriter stdout, stderr;  
  
@Override  
public void registerExtenderCallbacks(IBurpExtenderCallbacks callbacks)  
{  
    this.callbacks = callbacks;  
    this.helpers = callbacks.getHelpers();  
    stdout = new PrintWriter(callbacks.getStdout(), true);  
    stderr = new PrintWriter(callbacks.getStderr(), true);  
}
```

# Extension Settings

- » `callbacks.setExtensionName(String extName)`
- » `callbacks.issueAlert(String msg)`





## Exercise #3:

Using stdout/stdErr for extension with alert on loading

### Objective:

- » Create extension with name
- » Use stdout and stderr to show messages
- » Issue alert on load



## Some useful methods

### » IBurpExtenderCallbacks

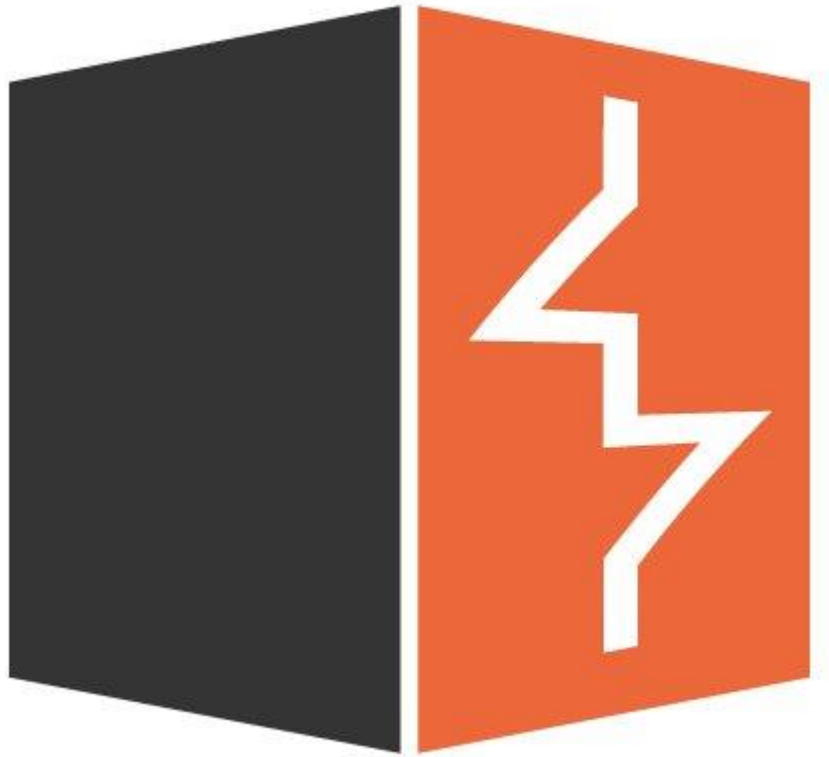
- `IHttpRequestResponse makeHttpRequest(IHttpService httpService, byte[] request);`
- `void registerHttpListener(IHttpListener listener);`
- `void registerProxyListener(IProxyListener listener);`

### » IExtensionHelper

- `IRequestInfo analyzeRequest(byte[] request);`
- `IResponseInfo analyzeResponse(byte[] response);`
- `IParameter getRequestParameter(byte[] request, String parameterName);`
- `byte[] addParameter(byte[] request, IParameter parameter);`
- `byte[] toggleRequestMethod(byte[] request);`
- `byte[] buildHttpMessage(List headers, byte[] body);`

### » IRequestInfo

- `List<String> getHeaders();`
- `List<IParameter> getParameters();`

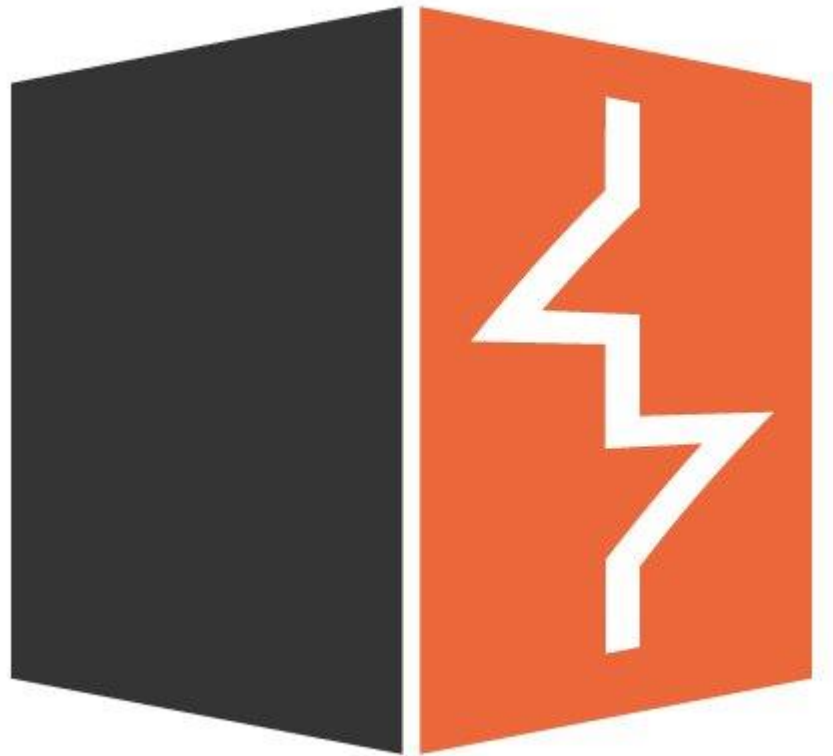


## Exercise #4:

Analyze and match from server response

### Objective:

- » Analyse test application for request and responses
- » Find server header in the responses.
- » Trigger message if server header found
- » Parse server header value and show in messages.



## Exercise #5:

Using iMessageEditorTab to analyze/parse HTTP content

### Objective:

- » Implement iMessageEditorTab
- » Show tab if request or response contain specific keyword
- » Parse and print messages in iMessageEditorTab

# Debugging Tips

- » Version updates may change APIs too. Use the latest
- » Run burp from terminal
- » Memory problems with Jython and Jruby
  - `java.lang.OutOfMemoryError: PermGen space`
- » In case of above error, allocate more PermGen storage
  - `java -XX:MaxPermSize=1G -jar burp.jar`

# Packaging Tips

```
<target name="package-for-release" depends="jar">
    <property name="store.jar.name" value="YourProjectName"/>
    <property name="store.dir" value="store"/>
    <property name="store.jar" value="${store.dir}/${store.jar.name}.jar"/>
    <echo message="Packaging ${application.title} into a single JAR at ${store.jar}"/>
    <delete dir="${store.dir}"/>
    <mkdir dir="${store.dir}"/>
    <jar destfile="${store.dir}/temp_final.jar" filesetmanifest="skip">
        <zipgroupfileset dir="dist" includes="*.jar"/>
        <zipgroupfileset dir="dist/lib" includes="*.jar"/>
        <manifest>
            <attribute name="Main-Class" value="${main.class}"/>
        </manifest>
    </jar>
    <zip destfile="${store.jar}">
        <zipfileset src="${store.dir}/temp_final.jar"
            excludes="META-INF/*.SF, META-INF/*.DSA, META-INF/*.RSA"/>
    </zip>
    <delete file="${store.dir}/temp_final.jar"/>
</target>
```

build.xml > Run target >  
Other Targets > package-for-release



# Thank You

-----

## Reach us at:

Sanoop Thomas

[s4n7h0@infoseccampus.com](mailto:s4n7h0@infoseccampus.com)

[twitter.com/s4n7h0](https://twitter.com/s4n7h0)

Samandeep Singh

[s.singh@sec-consult.com](mailto:s.singh@sec-consult.com)

[twitter.com/samanL33T](https://twitter.com/samanL33T)