

# Burp-Plugin im Eigenbau

## Wie ich lernte die API zu lieben

Bernhard Gröhling

19. März 2013

bernhard.groehling@sicsec.de  
08142 - 4425 037

sic[!]sec GmbH  
Industriestraße 29-31  
82194 Gröbenzell

## Das erwartet euch

### Theorie:

- Was kann Burp seit v1.5.01
- Kleiner API-Rundgang

### Praxis:

- Eclipse-Setup für One-Click Deployment
- Vollständiges Proxy-Plugin mit GUI in 80 Zeilen

## Das erwarte ich von euch

- Schonmal mit Burp gearbeitet
- Minimale Java-Kenntnisse

Burp ist das „Schweizer Taschenmesser“ zum Pentesten von Webanwendungen

## Modularer Aufbau

- Intercepting Proxy
- Spider
- Scanner (aktiv und passiv)
- Intruder
- Repeater
- Sequencer
- Decoder / Comparer
- **Extender**

## Große Änderungen seit v1.5.01

- Extender Tab im GUI
- Mehrere Plugins parallel möglich
- dynamische Laden / Entladen zur Laufzeit
- Unterstützung von Java, Python und Ruby

## Mächtiges API

- Zugriff auf HTTP Request/Responses
- Erweiterungsmöglichkeiten für Burp-Tools wie Intruder und Scanner
- Zugriff auf Laufzeitdaten (Target, History, Scan Issues...)
- Nahtlose Integration eigener GUI-Elemente in Burp-GUI

## Burp Extender API ist Callback-orientiert

- 1 Callbacks registrieren
- 2 Interfaces / Callback-Methoden implementieren
- 3 ...
- 4 Profit

## Callbacks registrieren:

```
/*
 * Implement IBurpExtender
 */
public void registerExtenderCallbacks (IBurpExtenderCallbacks
    callbacks)
{
    callbacks.registerProxyListener (this); // IProxyListener
}
```

## Callback-Methoden implementieren:

```
/*
 * Implement IProxyListener
 */
public void processProxyMessage (boolean messageIsRequest,
    IInterceptedProxyMessage message)
{
    if (messageIsRequest) {
        // Dinge mit der ProxyMessage machen
    }
}
```

## Referenz auf callbacks-Objekt merken:

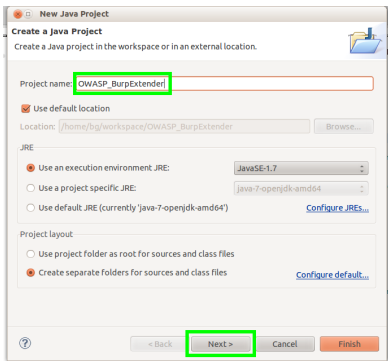
```
private IBurpExtenderCallbacks callbacks;  
public void registerExtenderCallbacks (IBurpExtenderCallbacks  
    callbacks) {  
    this.callbacks = callbacks;  
}
```

## Verfügbare Hilfsmethoden:

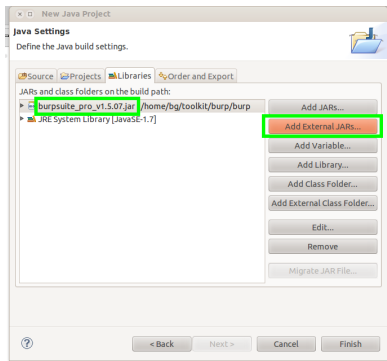
- analyzeRequest
- base64 (De|En) code
- buildHttp (Message|Request|Service)
- (build|update)Parameter
- getRequestParameter
- makeScannerInsertionPoint
- removeParameter
- toggleRequestMethod
- url (De|En) code



# Neues Java Projekt anlegen

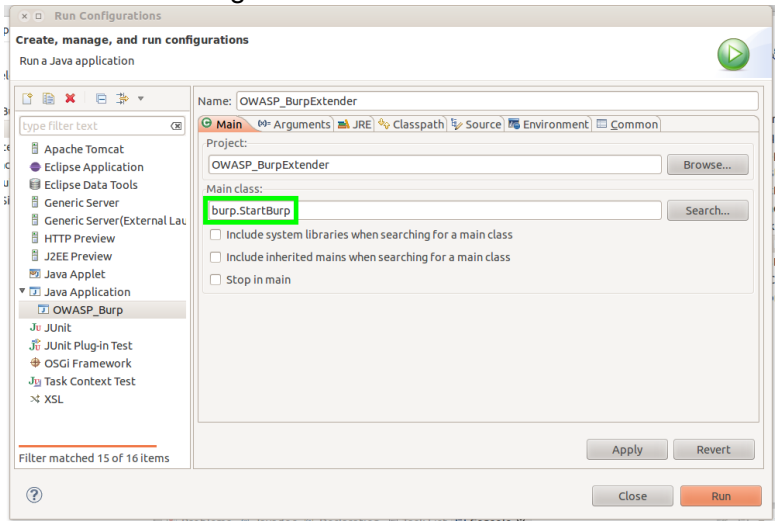


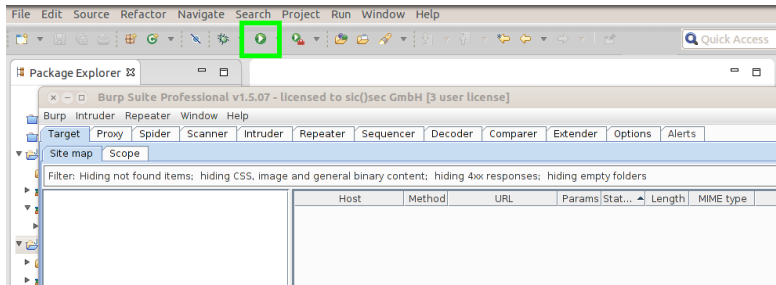
File -> New -> Java Project



Burp als Library hinzufügen

## Run -> Run Configurations...





## Extender -> APIs -> Save interface files

Burp Suite Professional v1.5.07 - licensed to sic()sec GmbH [3 user license]

Burp Intruder Repeater Window Help

Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts

Target Proxy Spider Scanner

Extensions APIs Options

### Burp Extender APIs

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

- IBurpExtender
- IBurpExtenderCallback
- IContextMenuFactory
- IContextMenuInvocation
- ICookie
- IExtensionHelpers
- IExtensionStateListener
- IHttpListener
- IHttpRequestResponse
- IHttpRequestResponseInfo
- IHttpRequestResponseInfo
- IHttpRequestResponseInfo
- IHttpService
- IInterceptedProxyMessage
- IIntruderAttack
- IIntruderPayloadGenerator
- IIntruderPayloadGenerator
- IIntruderPayloadProcessor
- IMenuItemHandler
- IMessageEditor
- IMessageEditorController
- IMessageEditorTab
- IMessageEditorTabFactory
- IParameter

Speichern in: OWASP\_BurpExtender

- bin
- src

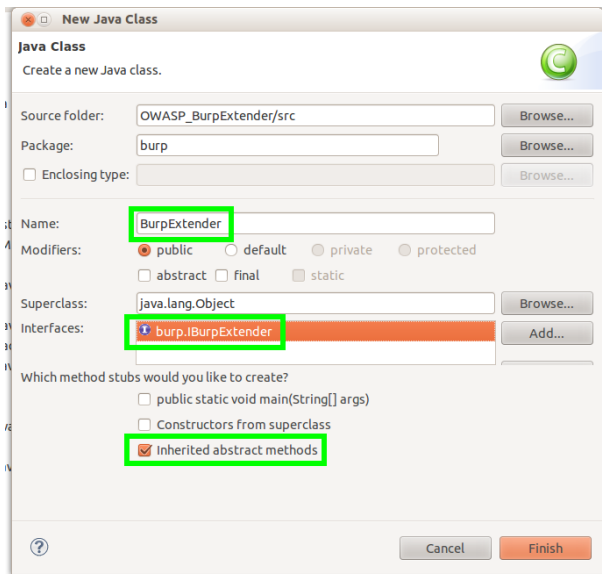
Ordername: /home/bg/workspace/OWASP\_BurpExtender/src

Dateityp: Alle Dateien

Speichern Abbrechen

Save interface files Save Javadoc files

- Burp Projekt aktualisieren (F5)
- Neue Klasse BurpExtender anlegen New -> Class



## Burp mit „Play Button“ starten

The screenshot shows the Eclipse IDE with the BurpExtender.java file open. The 'Run' button (a green play icon) in the toolbar is highlighted with a green box. Below the IDE, the Burp Suite Professional v1.5.07 interface is shown. The 'Extender' tab is active, and the 'Burp Extensions' table is visible. The table has a green border around the 'BurpExtender' entry, which is highlighted in orange. The 'Loaded' checkbox is checked.

Add	Loaded	Type	Name
	<input checked="" type="checkbox"/>	Legacy Java	BurpExtender

Details:  Extension loaded  
Name: BurpExtender

## Simple Statistics

- Einfache Statistik über Requests führen
- Zählt POST- und GET-Requests, Responses
- Stellt Daten in eigenem Tab dar

## Benötigte Interfaces

- IBurpExtender
- IProxyListener
- ITab

## Klasse BurpExtender mit Member Variablen

```
public class BurpExtender implements IBurpExtender,  
    IProxyListener, ITab {  
    private int numRequests = 0;  
    private int numResponses = 0;  
    private int numPostReq = 0;  
    private int numGetReq = 0;  
    private JTextPane textPane = new JTextPane();  
    private IBurpExtenderCallbacks callbacks;
```



## Callbacks registrieren

```
/*  
 * Implement IBurpExtender  
 */  
@Override  
public void registerExtenderCallbacks (IBurpExtenderCallbacks  
    callbacks) {  
    this.callbacks = callbacks;  
    callbacks.setExtensionName ("Simple Statistics");  
    callbacks.registerProxyListener (this); // IProxyListener  
    callbacks.addSuiteTab (this); // ITab  
    callbacks.customizeUiComponent (textPane);  
    this.textPane.setEnabled (false); // Kein UserInput  
}
```

## ProxyMessage verarbeiten

```
/* Implement IProxyListener */
@Override
public void processProxyMessage (boolean messageIsRequest,
    IInterceptedProxyMessage message) {
    if (messageIsRequest) { // Message is Request
        numRequests++;
        IRequestInfo reqInfo = this.callbacks.getHelpers().
            analyzeRequest (message.getMessageInfo());
        if (reqInfo.getMethod().equalsIgnoreCase ("POST")) {
            this.numPostReq++;
        }
        if (reqInfo.getMethod().equalsIgnoreCase ("GET")) {
            this.numGetReq++;
        }
    } else { // Message is Response
        numResponses++;
    }
    this.textPane.setText ("Total GET:           "+numGetReq+"\n"+
        "Total POST :           "+numPostReq+"\n"+
        "Total Requests:       "+numRequests+"\n"+
        "Total Responses:     "+numResponses);
}
```

## ITab implementieren

```
/*
 * Implement ITab
 */
@Override
public String getTabCaption() {
    return "Statistics";
}

/*
 * Implement ITab
 */
@Override
public Component getUiComponent() {
    return this.textPane;
}
```

Einleitung

API  
Benutzung

Eclipse Setup

Beispiel  
Plugin

Referenzen

Portswigger <http://portswigger.net/>

Dokumentation und Beispiele Code

<http://portswigger.net/burp/extender/>

Burp-Extensions im User Forum

[http://forum.portswigger.net/index.cgi?  
board=extensions](http://forum.portswigger.net/index.cgi?board=extensions)