



# OWASP Proxy

An intercepting proxy library,  
so you don't have to

# Background

- OWASP WebScarab
- OWASP WebScarab-NG
- OWASP CSRFTester

```
$ unzip -l CSRFTester-1.0-src.zip | grep java |  
grep webscarab | wc -l
```

**75**

# What good is this?

- Allows visibility into communications
- Allows modification of communications
- Invisible to client and server

# Features

- Flexible
  - compose your own proxy
- Binary clean
  - squeaky clean!
- Performant
  - streams as much as possible
  - buffers only what you tell it to
- Multi-protocol
  - Mostly HTTP-related currently

# The Simplest Proxy

```
requestHandler = new DefaultHttpRequestHandler  
    ();  
httpProxy = new HttpProxyConnectionHandler  
    (requestHandler);  
listen = new InetSocketAddress("localhost", 8008);  
proxy = new Server(listen, httpProxy);  
proxy.start();
```

... isn't very useful

# Message Object Model

```
public interface MessageHeader {  
    byte[] getHeader();  
    String getStartLine() throws MessageFormatException;  
    NamedValue[] getHeaders() throws MessageFormatException;  
    String getHeader(String name) throws MessageFormatException;  
}
```

```
public interface MutableMessageHeader {  
    void setHeader(byte[] header);  
    void setStartLine(String line) throws MessageFormatException;  
    void setHeaders(NamedValue[] headers) throws MessageFormatException;  
    void setHeader(String name, String value) throws  
        MessageFormatException;  
    void addHeader(String name, String value) throws  
        MessageFormatException;  
    String deleteHeader(String name) throws MessageFormatException;  
}
```

# Message Content

```
public interface StreamingMessage  
    extends MutableMessageHeader {
```

```
    InputStream getContent();
```

```
    InputStream getDecodedContent() throws  
        MessageFormatException;
```

```
    void setContent(InputStream content);
```

```
    void setDecodedContent(InputStream  
        content) throws  
        MessageFormatException;
```

```
}
```

```
public interface BufferedMessage extends  
    MessageHeader {
```

```
    byte[] getContent();
```

```
    byte[] getDecodedContent() throws  
        MessageFormatException;
```

```
}
```

```
public interface MutableBufferedMessage  
    extends BufferedMessage,  
        MutableMessageHeader {
```

```
    void setContent(byte[] content);
```

```
    void setDecodedContent(byte[] content)  
        throws MessageFormatException;
```

```
}
```

# Request

```
public interface RequestHeader extends MessageHeader {  
    InetAddress getTarget();  
    boolean isSsl();  
    String getMethod() throws MessageFormatException;  
    String getResource() throws MessageFormatException;  
    String getVersion() throws MessageFormatException;  
}
```

```
public interface MutableRequestHeader extends RequestHeader,  
    MutableMessageHeader {  
    void setTarget(InetAddress target);  
    void setSsl(boolean ssl);  
    void setMethod(String method) throws MessageFormatException;  
    void setResource(String resource) throws MessageFormatException;  
    void setVersion(String version) throws MessageFormatException;  
}
```

similar for Response

# BufferedMessageInterceptor

```
enum Action { BUFFER, STREAM, IGNORE};
```

```
Action directRequest(MutableRequestHeader request);  
void processRequest(MutableBufferedRequest request);  
void requestContentSizeExceeded(BufferedRequest request, int size);  
void requestStreamed(BufferedRequest request);
```

```
Action directResponse(RequestHeader request,  
    MutableResponseHeader response)  
void processResponse(RequestHeader request,  
    MutableBufferedResponse response)  
void responseContentSizeExceeded(RequestHeader request,  
    ResponseHeader response, int size);  
void responseStreamed(final RequestHeader request,  
    BufferedResponse response);
```

# Doing something useful

```
requestHandler = new DefaultHttpRequestHandler();
interceptor = new BufferedMessageInterceptor() {
    public Action directResponse(RequestHeader request,
        MutableResponseHeader response) {
        return Action.BUFFER;
    }
    public void processResponse(RequestHeader request,
        MutableBufferedResponse response) {
        try {
            System.out.println(request.getResource() + " : " +
                response.getDecodedContent().length);
        } catch (MessageFormatException mfe) {
            mfe.printStackTrace();
        }
    }
};
requestHandler = new BufferingHttpRequestHandler(requestHandler,
    interceptor, 10240);
```

# So what about SSL?

```
httpProxy = new HttpProxyConnectionHandler  
    (requestHandler);  
contextSelector = new  
    DefaultServerContextSelector("server.p12",  
    password, password);  
ssl = new SSLConnectionHandler(contextSelector,  
    true, httpProxy);    // true -> autodetect SSL  
httpProxy.setConnectHandler(ssl);  
proxy = new Server(listen, httpProxy);
```

# Bah! Untrusted Connections!



## This Connection is Untrusted

You have asked Firefox to connect securely to **www.fnb.co.za**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

### What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

[Get me out of here!](#)

### ▶ Technical Details

### ▼ I Understand the Risks

If you understand what's going on, you can tell Firefox to start trusting this site's identification. **Even if you trust the site, this error could mean that someone is tampering with your connection.**

Don't add an exception unless you know there's a good reason why this site doesn't use trusted identification.

[Add Exception...](#)

## Untrusted Connection

[General](#) [Details](#)

**Could not verify this certificate because the issuer is not trusted.**

### Issued To

Common Name (CN)	WebScarab
Organization (O)	Open Web Application Security Project
Organizational Unit (OU)	WebScarab
Serial Number	00

### Issued By

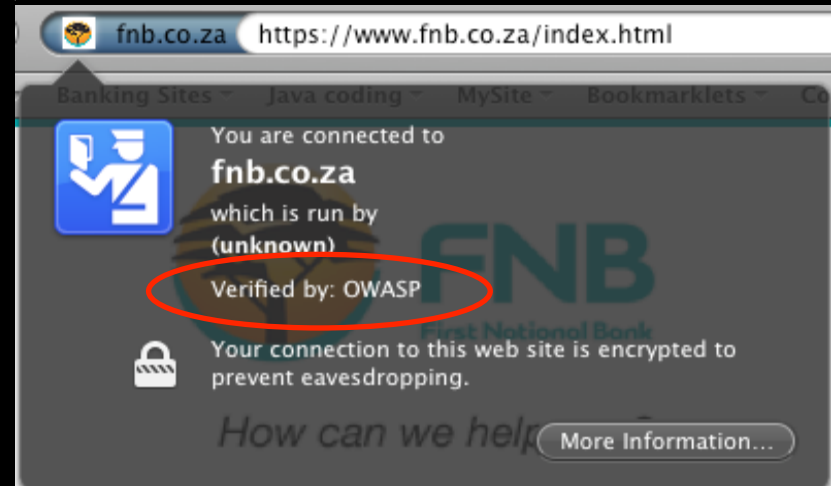
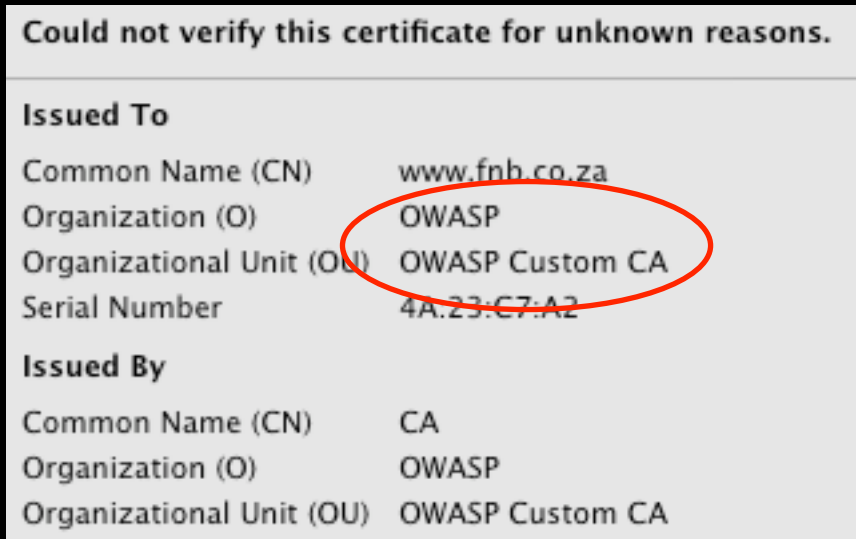
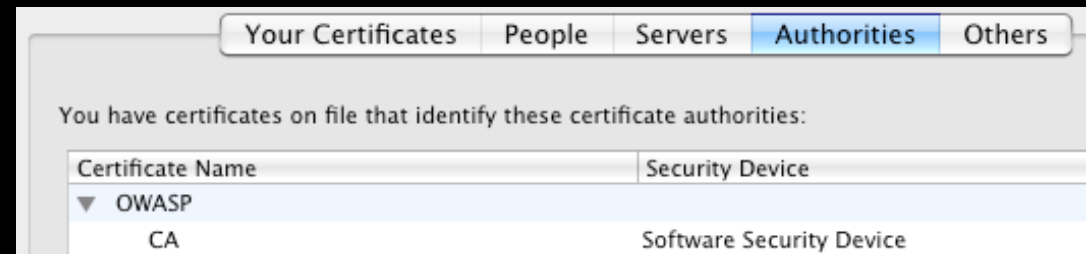
Common Name (CN)	WebScarab
Organization (O)	Open Web Application Security Project
Organizational Unit (OU)	WebScarab

### Validity

Issued On	2004/04/01
Expires On	2014/03/30

# Per server certificates!

```
contextSelector = new  
AutoGeneratingContextSelector("keystore",  
"JKS", password);
```



# Reverse Proxy

```
target = new InetSocketAddress("example.com",  
    80);
```

```
listen = new InetSocketAddress("localhost", 80);
```

```
proxy = new Proxy(listen, httpProxy, target);
```

## and with SSL . . .

```
ssl = new SSLConnectionHandler(contextSelector,  
    true, httpProxy);  
target = new InetSocketAddress("www.fnb.co.za",  
    443);  
listen = new InetSocketAddress("localhost", 443);  
proxy = new Proxy(listen, ssl, target);
```

# How about SOCKS?

```
httpProxy = new HttpProxyConnectionHandler  
    ( requestHandler);  
socks = new SocksConnectionHandler(httpProxy,  
    true);           // true -> autodetect SOCKS  
proxy = new Server(listen, socks);
```

# All together now!

```
httpProxy = new HttpProxyConnectionHandler  
    (requestHandler);  
contextSelector = new AutoGeneratingContextSelector  
    (".keystore", "JKS", password);  
ssl = new SSLConnectionHandler(contextSelector, true,  
    httpProxy);  
httpProxy.setConnectHandler(ssl);  
listen = new InetSocketAddress("localhost", 8008);  
socks = new SocksConnectionHandler(ssl, true);  
proxy = new Server(listen, socks);
```

# But SOCKS redirects EVERYTHING!

```
httpProxy = new HttpProxyConnectionHandler(requestHandler);
ssl = new SSLConnectionHandler(cs, true, httpProxy);
selector = new SelectiveConnectionHandler() {
    @Override
    public TargetedConnectionHandler getConnectionHandler
        (InetSocketAddress target) {
        if (target.getPort() == 80) return httpProxy;
        if (target.getPort() == 443) return ssl;
        return RELAY;
    }
};
httpProxy.setConnectHandler(selector);
socks = new SocksConnectionHandler(selector, true);
listen = new InetSocketAddress("localhost", 8008);
proxy = new Proxy(listen, socks, null);
```

# Upstream proxies?

```
ProxySelector ps = new ProxySelector() {
    private Proxy direct = java.net.Proxy.NO_PROXY;
    private Proxy socks = new java.net.Proxy(Type.SOCKS, socksAddress);
    private Proxy http = new java.net.Proxy(Type.HTTP, httpAddress);
    private List<Proxy> proxies = Arrays.asList(socks, http, direct);

    public void connectFailed(URI uri, SocketAddress sa, IOException ioe) {
        System.out.println("Proxy connection failed! " + ioe.getMessage());
    }

    public List<java.net.Proxy> select(URI uri) {
        return proxies;
    }
};
DefaultHttpRequestHandler requestHandler = new DefaultHttpRequestHandler
();
requestHandler.setProxySelector(ps);
```

# Apache Jserv Protocol

```
requestHandler = new DefaultAJPRequestHandler  
    ();  
tomcat = new InetSocketAddress("tomcat", 8009);  
requestHandler.setTarget(tomcat);  
ajp = new AJPConnectionHandler  
    (requestHandler);  
listen = new InetSocketAddress("localhost", 8009);  
proxy = new Server(listen, ajp);
```

# HTTP -> AJP

```
requestHandler = new DefaultAJPRequestHandler  
    ();  
properties = new AJPProperties();  
properties.setRemoteAddress("127.0.0.1");  
requestHandler.setProperties(properties);  
tomcat = new InetSocketAddress("tomcat", 8009);  
requestHandler.setTarget(tomcat);  
httpProxy = new HttpProxyConnectionHandler  
    (requestHandler);
```

# Other features

- JDBC interface for saving conversations
- “Web Service” `HttpRequestHandler` to expose history
- `LoggingHttpRequestHandler` does CLF logging

# Resources

- <http://dawes.za.net/gitweb.cgi>
- `git clone http://dawes.za.net/rogan/owasp-proxy/owasp-proxy.git/`
- `owasp-proxy@lists.owasp.org`

# Questions?



[rogan@dawes.za.net](mailto:rogan@dawes.za.net)