Notes From The field

OWASP

OWASP tools and usage experiences

Jarkko Holappa
Antti Laulajainen

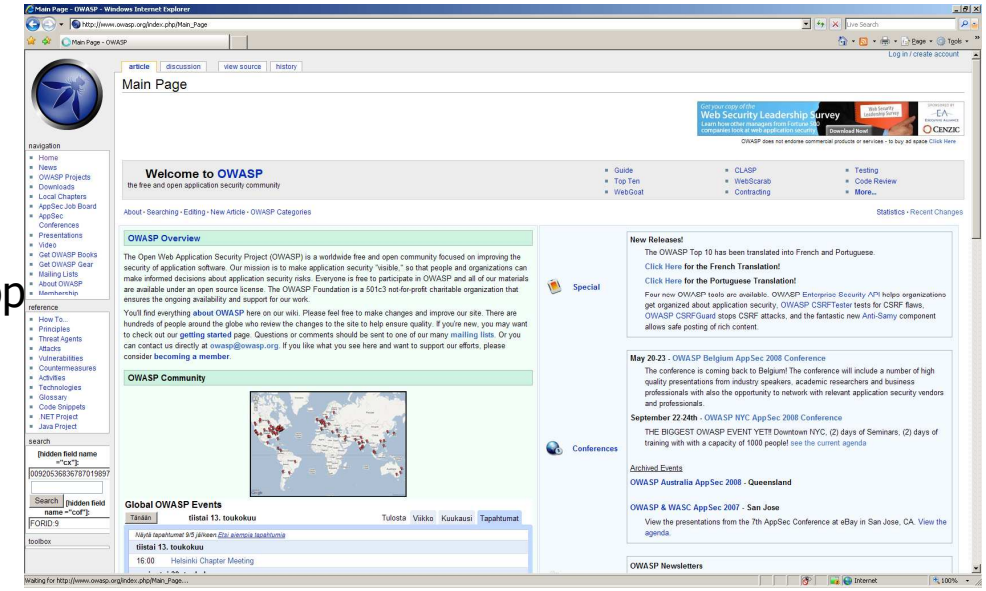
Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document under the terms of the OWASP License.

OWASP Tools project

http://www.owasp.org/index.php/Category:OWASP_Tools_Project

- The OWASP Tools Project's goal is to provide unbiased, practical information and guidance about application security tools.

- Tools are to help:
  - Protect Applications
  - Find Vulnerabilities in Applications
  - Test Other Application Security Tools
  - Educate People on Application Security Top

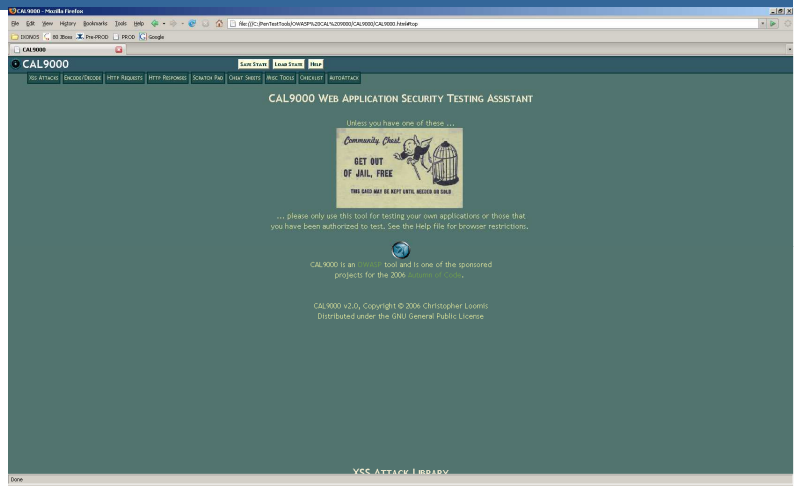


- Categories of tools:
  - Web Application Firewalls (WAFs)
  - Application Vulnerability Scanning Tools
  - Application Penetration Testing Tools
  - Source Code Analysis Tools
  - Test and Educational Applications
  - Application Security Analysis Support Tools

Tools from OWASP

- OWASP .NET Project
- OWASP AntiSamy Project
- OWASP CAL9000 Project
- OWASP DirBuster Project
- OWASP Encoding Project
- OWASP Flash Security Project
- OWASP Insecure Web App Project
- OWASP Interceptor Project
- OWASP JBroFuzz
- OWASP LAPSE Project
- OWASP Live CD Project
- OWASP Orizon Project
- OWASP Pantera Web Assessment Studio Project
- OWASP SQLiX Project
- OWASP Sprajax Project
- OWASP Validation Project
- OWASP WSFuzzer Project
- OWASP WebGoat Project
- OWASP WebScarab Project

OWASP: CAL9000

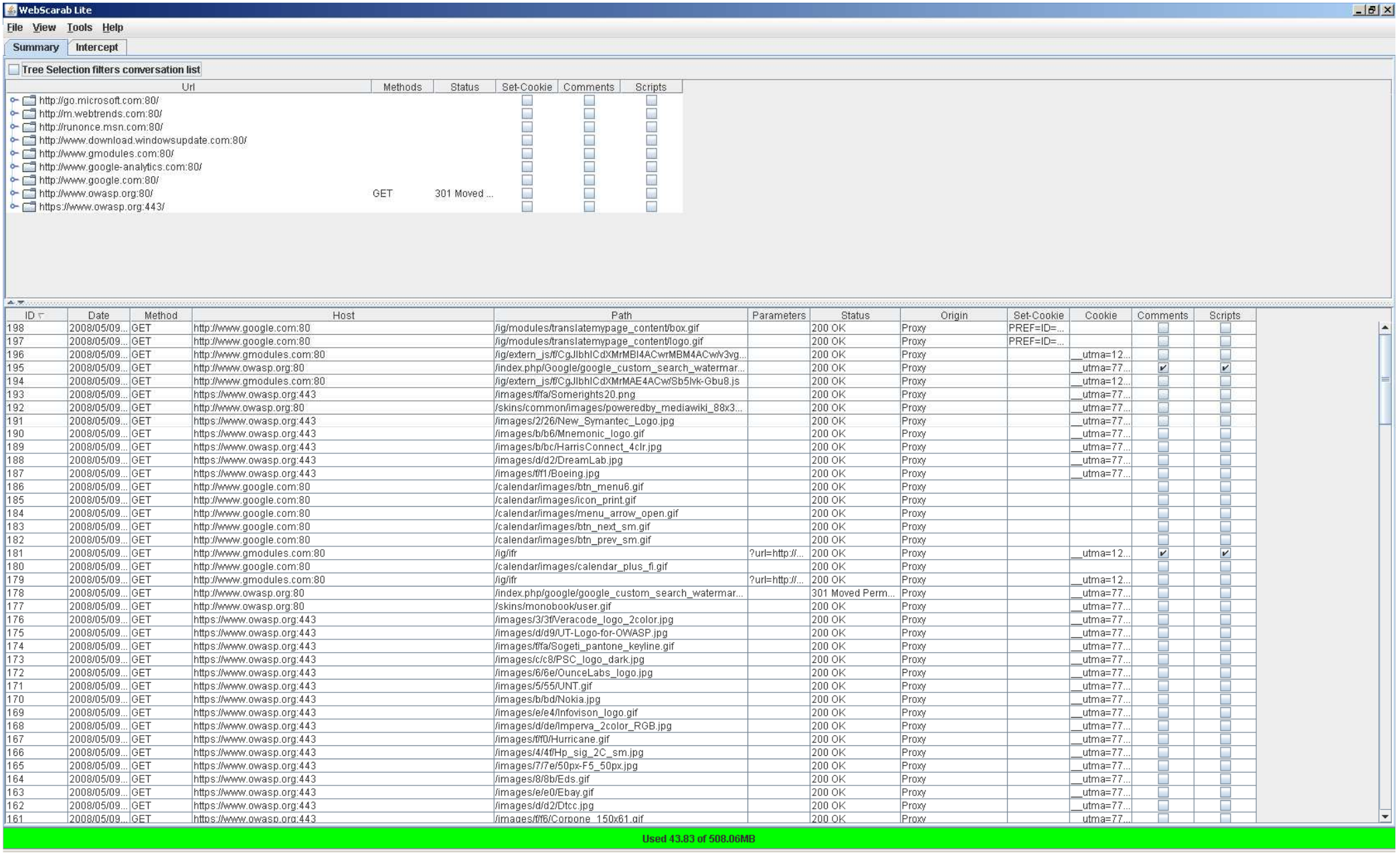


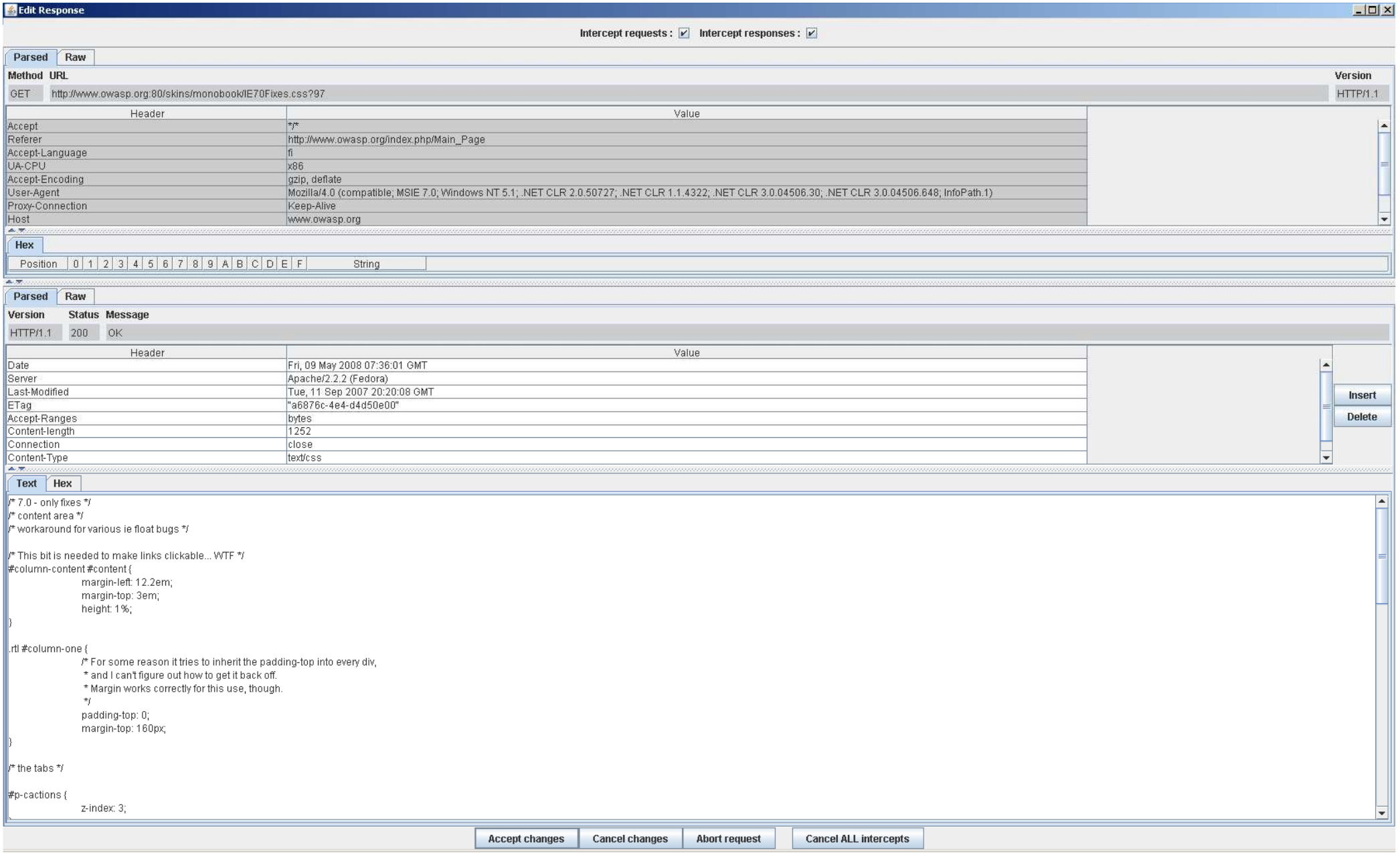
- CAL9000 is a collection of web application security testing tools that complement the feature set of current web proxies and automated scanners.
- CAL9000 is written in Javascript
- Features e.g.
  - XSS Attacks
  - Character Encoder/Decoder
  - Http Requests - Manually craft and send HTTP requests to servers.
  - String Generator - Create character strings of almost any length.
  - Testing Tips - Collection of testing ideas for assessments.
  - Testing Checklist - Track the progress of your testing efforts and record your findings.
  - AutoAttack Editor - Create/edit/save/delete the AutoAttack Lists that are used to drive the automated multiple-request capabilities on the HTTP Requests page.
  - Store/Restore - Temporarily hold and retrieve textarea and text field contents.

CAL9000 Demo

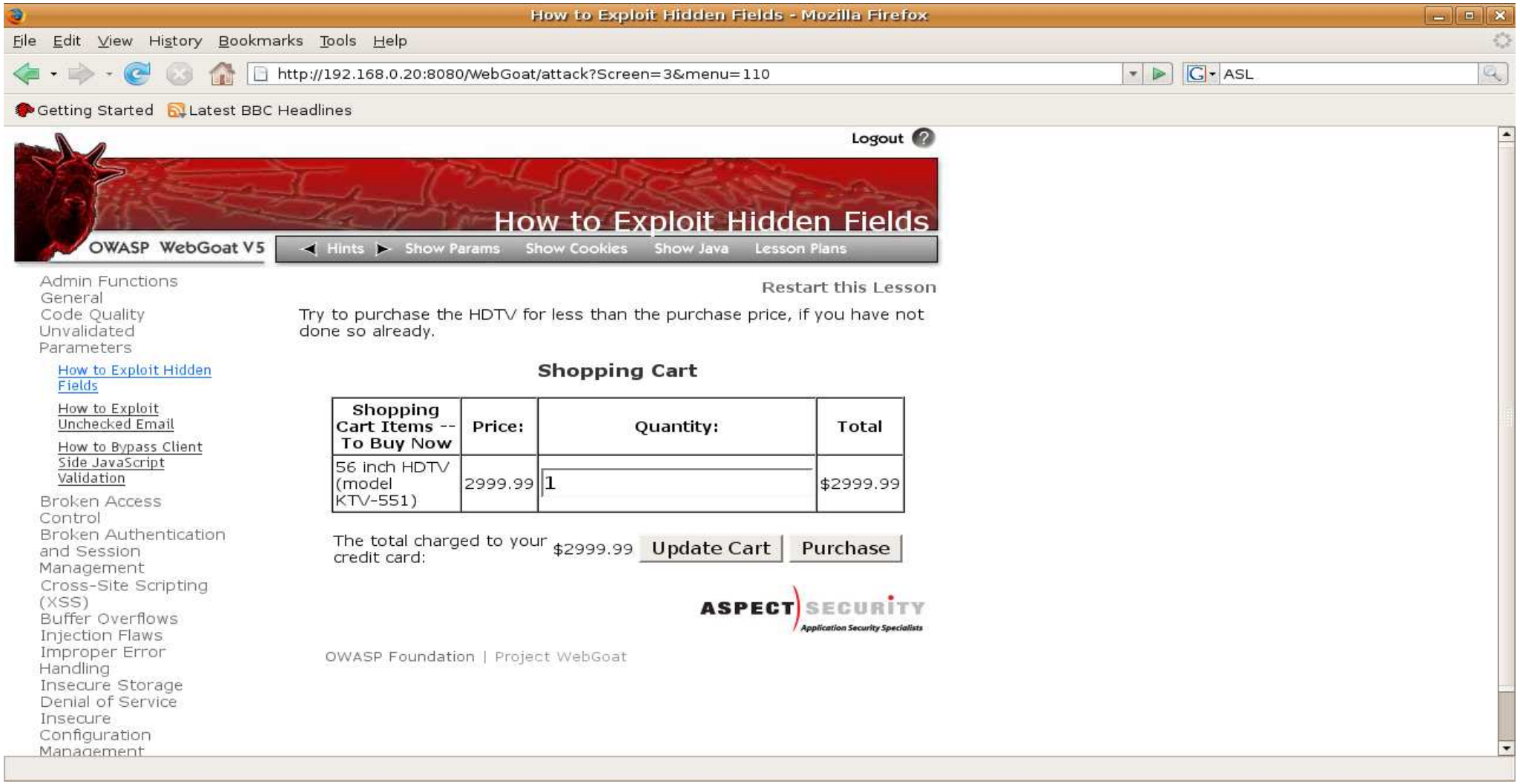
OWASP: WebScarab

- WebScarab is a framework for analysing applications that communicate using the HTTP and HTTPS protocols.
- It is written in Java
- WebScarab operates as an intercepting proxy, allowing the operator to review and modify requests created by the browser before they are sent to the server, and to review and modify responses returned from the server before they are received by the browser.
- WebScarab is able to intercept both HTTP and HTTPS communication. The operator can also review the conversations (requests and responses) that have passed through WebScarab.

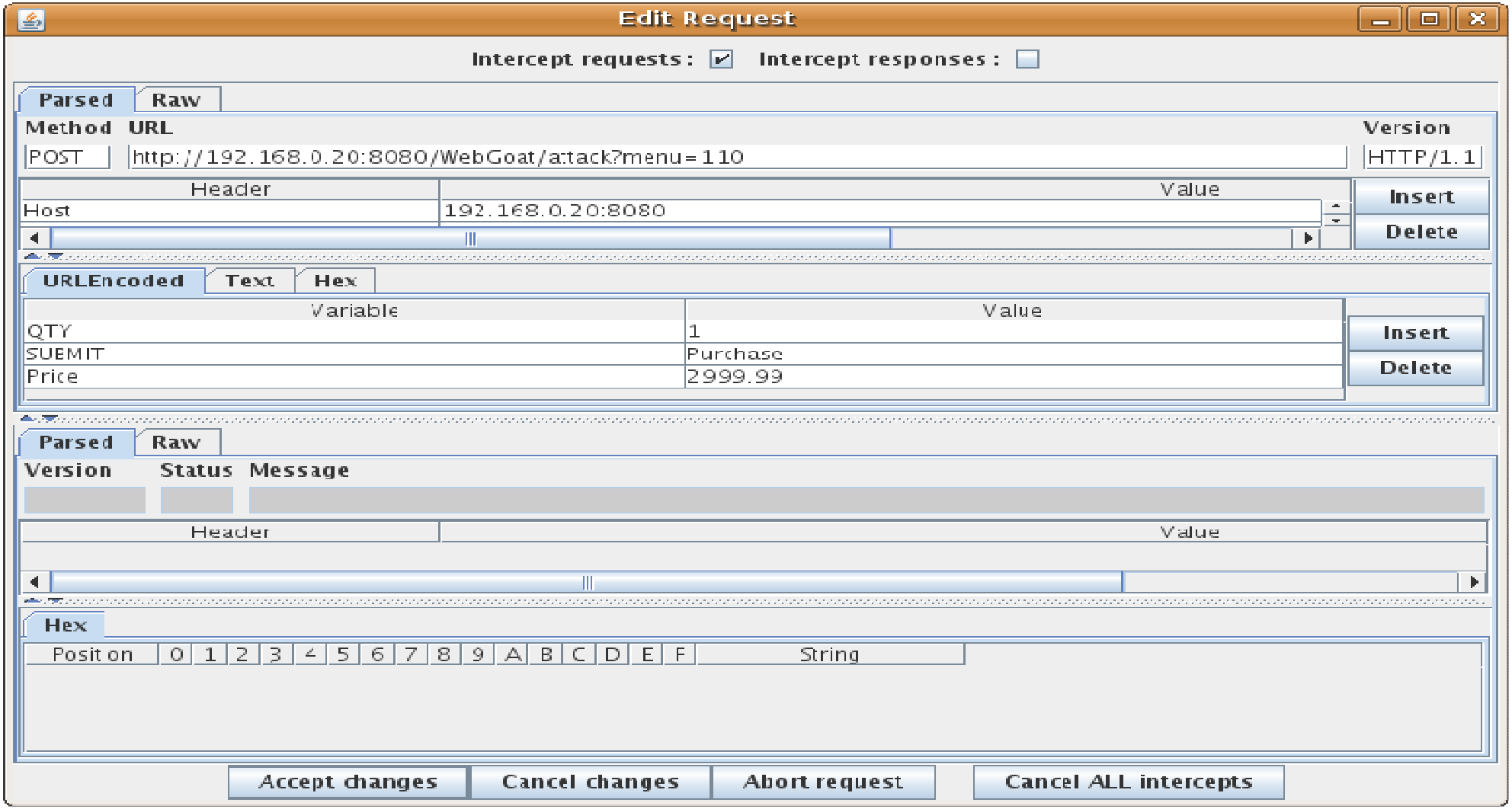

WebScarab Lite
File View Tools Help
Summary Intercept
Tree Selection filters conversation list
Url Methods Status Set-Cookie Comments Scripts
http://go.microsoft.com:80/
http://m.webtrends.com:80/
http://runonce.msn.com:80/
http://www.download.windowsupdate.com:80/
http://www.gmodules.com:80/
http://www.google-analytics.com:80/
http://www.google.com:80/
http://www.owasp.org:80/ GET 301 Moved ...
https://www.owasp.org:443/
ID Date Method Host Path Parameters Status Origin Set-Cookie Cookie Comments Scripts
198 2008/05/09... GET http://www.google.com:80 /ig/modules/translatemypage_content/box.gif 200 OK Proxy PREF=ID=...
197 2008/05/09... GET http://www.google.com:80 /ig/modules/translatemypage_content/logo.gif 200 OK Proxy PREF=ID=...
196 2008/05/09... GET http://www.gmodules.com:80 /ig/extern_js/f/CgJlbhICdXMrMBI4ACwrMBM4ACw/v3vg... 200 OK Proxy __utma=12...
195 2008/05/09... GET http://www.owasp.org:80 /index.php/Google/google_custom_search_watermar... 200 OK Proxy __utma=77...
194 2008/05/09... GET http://www.gmodules.com:80 /ig/extern_js/f/CgJlbhICdXMrMAE4ACw/Sb5lvk-Gbu8.js 200 OK Proxy __utma=12...
193 2008/05/09... GET https://www.owasp.org:443 /images/f/fa/Somerights20.png 200 OK Proxy __utma=77...
192 2008/05/09... GET http://www.owasp.org:80 /skins/common/images/poweredby_mediawiki_88x3... 200 OK Proxy __utma=77...
191 2008/05/09... GET https://www.owasp.org:443 /images/2/26/New_Symantec_Logo.jpg 200 OK Proxy __utma=77...
190 2008/05/09... GET https://www.owasp.org:443 /images/b/b6/Mnemonic_logo.gif 200 OK Proxy __utma=77...
189 2008/05/09... GET https://www.owasp.org:443 /images/b/bc/HarrisConnect_4clr.jpg 200 OK Proxy __utma=77...
188 2008/05/09... GET https://www.owasp.org:443 /images/d/d2/DreamLab.jpg 200 OK Proxy __utma=77...
187 2008/05/09... GET https://www.owasp.org:443 /images/f/f1/Boeing.jpg 200 OK Proxy __utma=77...
186 2008/05/09... GET http://www.google.com:80 /calendar/images/btn_menu6.gif 200 OK Proxy
185 2008/05/09... GET http://www.google.com:80 /calendar/images/icon_print.gif 200 OK Proxy
184 2008/05/09... GET http://www.google.com:80 /calendar/images/menu_arrow_open.gif 200 OK Proxy
183 2008/05/09... GET http://www.google.com:80 /calendar/images/btn_next_sm.gif 200 OK Proxy
182 2008/05/09... GET http://www.google.com:80 /calendar/images/btn_prev_sm.gif 200 OK Proxy
181 2008/05/09... GET http://www.gmodules.com:80 /ig/ifr ?url=http://... 200 OK Proxy __utma=12...
180 2008/05/09... GET http://www.google.com:80 /calendar/images/calendar_plus_fi.gif 200 OK Proxy
179 2008/05/09... GET http://www.gmodules.com:80 /ig/ifr ?url=http://... 200 OK Proxy __utma=12...
178 2008/05/09... GET http://www.owasp.org:80 /index.php/google/google_custom_search_watermar... 301 Moved Perm... Proxy __utma=77...
177 2008/05/09... GET http://www.owasp.org:80 /skins/monobook/user.gif 200 OK Proxy __utma=77...
176 2008/05/09... GET https://www.owasp.org:443 /images/3/3f/Veracode_logo_2color.jpg 200 OK Proxy __utma=77...
175 2008/05/09... GET https://www.owasp.org:443 /images/d/d9/UT-Logo-for-OWASP.jpg 200 OK Proxy __utma=77...
174 2008/05/09... GET https://www.owasp.org:443 /images/f/fa/Sogeti_pantone_keyline.gif 200 OK Proxy __utma=77...
173 2008/05/09... GET https://www.owasp.org:443 /images/c/c8/PSC_logo_dark.jpg 200 OK Proxy __utma=77...
172 2008/05/09... GET https://www.owasp.org:443 /images/6/6e/OunceLabs_logo.jpg 200 OK Proxy __utma=77...
171 2008/05/09... GET https://www.owasp.org:443 /images/5/55/UNT.gif 200 OK Proxy __utma=77...
170 2008/05/09... GET https://www.owasp.org:443 /images/b/bd/Nokia.jpg 200 OK Proxy __utma=77...
169 2008/05/09... GET https://www.owasp.org:443 /images/e/e4/Infovison_logo.gif 200 OK Proxy __utma=77...
168 2008/05/09... GET https://www.owasp.org:443 /images/d/de/Imperva_2color_RGB.jpg 200 OK Proxy __utma=77...
167 2008/05/09... GET https://www.owasp.org:443 /images/f/f0/Hurricane.gif 200 OK Proxy __utma=77...
166 2008/05/09... GET https://www.owasp.org:443 /images/4/4f/Hp_sig_2C_sm.jpg 200 OK Proxy __utma=77...
165 2008/05/09... GET https://www.owasp.org:443 /images/7/7e/50px-F5_50px.jpg 200 OK Proxy __utma=77...
164 2008/05/09... GET https://www.owasp.org:443 /images/8/8b/Eds.gif 200 OK Proxy __utma=77...
163 2008/05/09... GET https://www.owasp.org:443 /images/e/e0/Ebay.gif 200 OK Proxy __utma=77...
162 2008/05/09... GET https://www.owasp.org:443 /images/d/d2/Dtcc.jpg 200 OK Proxy __utma=77...
161 2008/05/09... GET https://www.owasp.org:443 /images/f/f6/Corpone_150x61.gif 200 OK Proxy __utma=77...
Used 43.83 of 508.06MB


Edit Response
Intercept requests :
Intercept responses :
Parsed
Raw
Method URL
Version
GET
http://www.owasp.org:80/skins/monobook/IE70Fixes.css?97
HTTP/1.1
Header
Value
Accept
/
Referer
http://www.owasp.org/index.php/Main_Page
Accept-Language
fi
UA-CPU
x86
Accept-Encoding
gzip, deflate
User-Agent
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 2.0.50727; .NET CLR 1.1.4322; .NET CLR 3.0.04506.30; .NET CLR 3.0.04506.648; InfoPath.1)
Proxy-Connection
Keep-Alive
Host
www.owasp.org
Hex
Position
0 1 2 3 4 5 6 7 8 9 A B C D E F
String
Parsed
Raw
Version
Status
Message
HTTP/1.1
200
OK
Header
Value
Date
Fri, 09 May 2008 07:36:01 GMT
Server
Apache/2.2.2 (Fedora)
Last-Modified
Tue, 11 Sep 2007 20:20:08 GMT
ETag
"a6876c-4e4-d4d50e00"
Accept-Ranges
bytes
Content-length
1252
Connection
close
Content-Type
text/css
Insert
Delete
Text
Hex
/* 7.0 - only fixes */
/* content area */
/* workaround for various ie float bugs */
/* This bit is needed to make links clickable... WTF */
#column-content #content {
margin-left: 12.2em;
margin-top: 3em;
height: 1%;
}
.rtl #column-one {
/* For some reason it tries to inherit the padding-top into every div,
* and I can't figure out how to get it back off.
* Margin works correctly for this use, though.
*/
padding-top: 0;
margin-top: 160px;
}
/* the tabs */
#p-cactions {
z-index: 3;
Accept changes
Cancel changes
Abort request
Cancel ALL intercepts

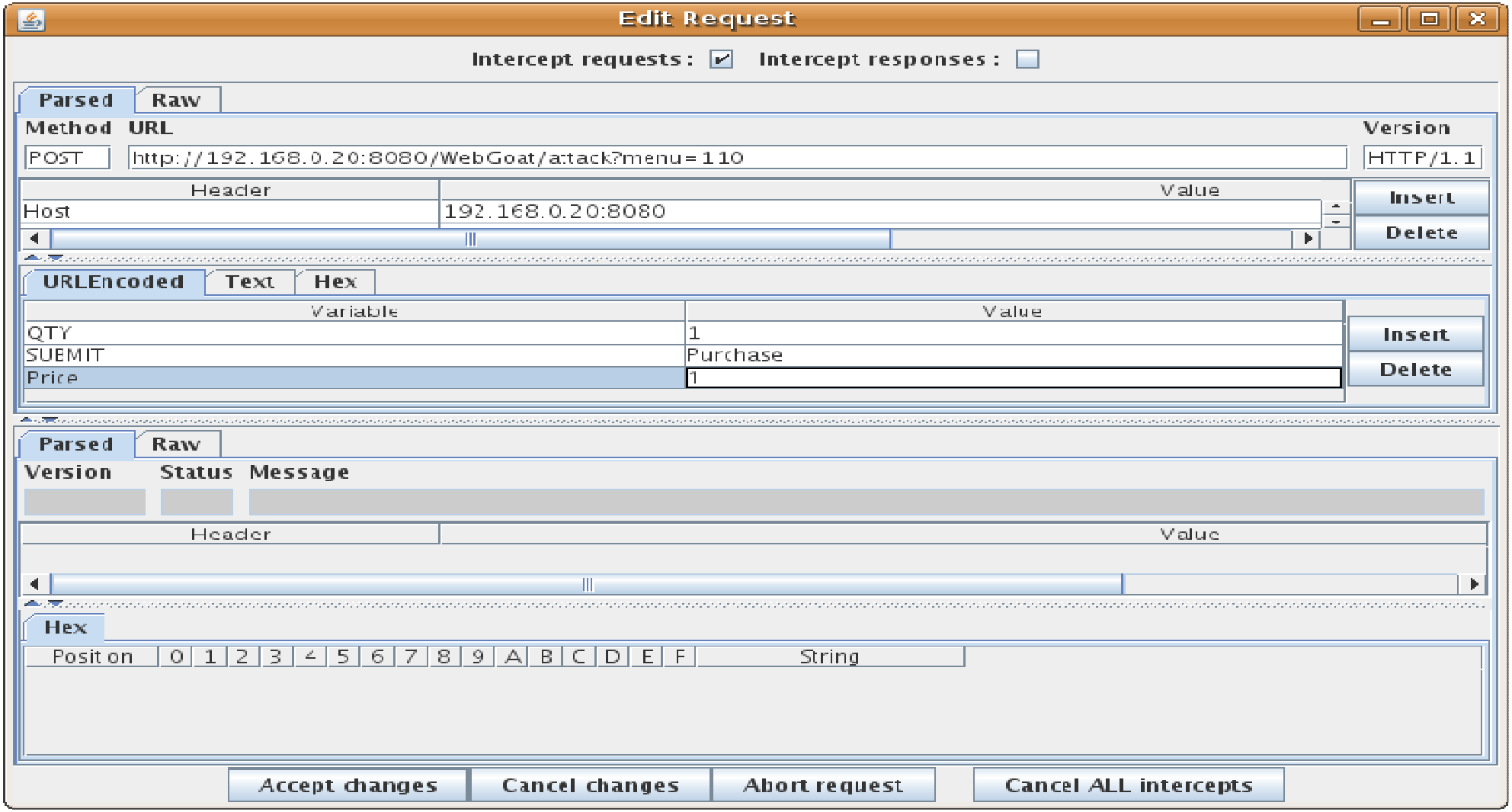
WebScarab - example



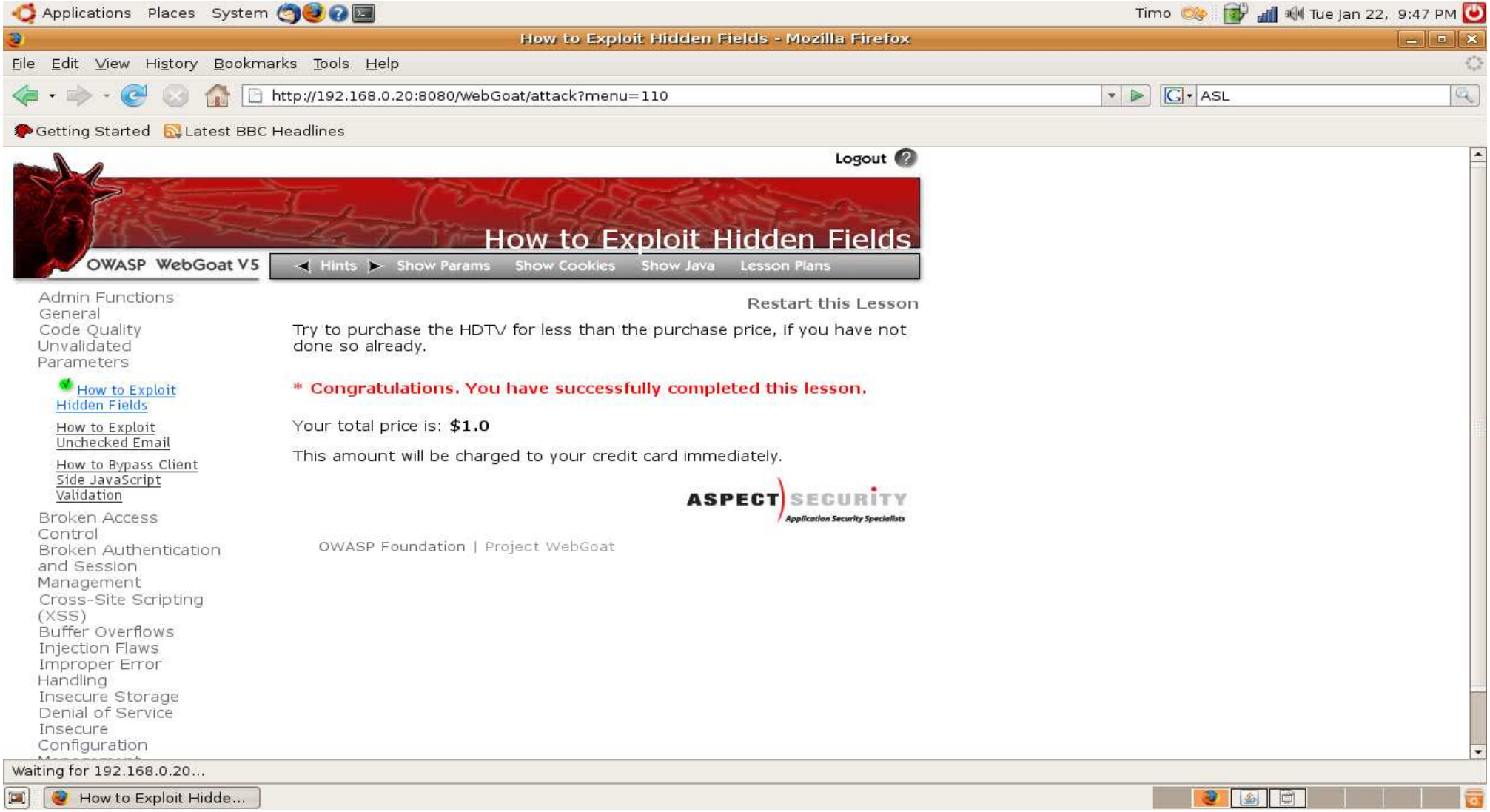
WebScarab - example



WebScarab - example



WebScarab - example



OWASP: WebGoat

- WebGoat is a deliberately insecure J2EE web application
- It is designed to teach web application security lessons. For example, in one of the lessons the user must use SQL injection to steal fake credit card numbers.
- Lessons include e.g.:

  - Cross Site Scripting
  - Access Control
  - Thread Safety
  - Hidden Form Field Manipulation
  - Parameter Manipulation
  - Weak Session Cookies
  - Blind SQL Injection
  - Numeric SQL Injection
  - String SQL Injection
  - Web Services
  - Fail Open Authentication
  - Dangers of HTML Comments

OWASP: Sprajax

- Open source black box security scanner used to assess the security of AJAX-enabled (Asynchronous JavaScript and XML) applications.
- The goal of the Sprajax project is to create a useful tool for assessing the security of AJAX-enabled web application.

OWASP: Insecure Web App

- Web application that includes common web application vulnerabilities.
- Targeted for automated and manual penetration testing, source code analysis, vulnerability assessments and threat modeling.
- The goals of this tool:
  - Demonstrate how dangerous application vulnerabilities can be.
  - Close the gap between the theory of web application security and the actual code that we design and build.
  - Learn how these vulnerabilities can be fixed.

OWASP LiveCD Project: LabRat

- Beta Version 2.1 is focused on providing all of OWASP tools and documents on a bootable CD.
- At this point the OWASP tools and documents are on the CD but they are not all configured.

General notes about testing

OWASP

Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document under the terms of the OWASP License.

Testing: Automated vs. Manual

- Manual testing: a human being *attacks* a web application using his experience, knowledge and tools
- Automated testing: a human being *uses* an automated vulnerability scanner to *attack* a web application
- Many tools generate extensive reports but results need to be verified → Testing can be automated, analysis never

Functional versus Technical Issues

- Different vulnerabilities, different kind of testing:
  - Technical Vulnerabilities
    - Data Handling
    - Dealing with Strong Typing, Encoding, and Validation of Data that the application handles
    - Can be tested fairly effectively by automated tools
  - Functional or Logical Vulnerabilities
    - Deals with issues allowed by design, but not foreseen by the designers as a risk
    - Issues in human-eye visible forms: Comments revealing information, inconsistent business logic, etc.

What are they good for?

- Automated → Saves time and resources.
- In-built knowledge about vulnerabilities and threats against various technologies.
- Quite good reporting features
- Tools provide reliable, repeatable tests that produce accurate quantifiable information about software defects with security implications.

Click & Scan and send report to customer?

- Many tools generate extensive reports but results need to be verified → Testing can be automated, analysis never
- Tools are not short-cut to success, there is still need for security awareness and good insight to software development issues to be able to use these tools efficiently. → ”A Fool with a Tool is still a Fool”
- Testers must be aware of limitations of tools
- There are issues that cannot be automated reliably:

  - business logic flaws
  - workflow bypasses
  - human eye visible errors.

Next step:

Thank You!

OWASP

Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document under the terms of the OWASP License.

The OWASP Foundation
http://www.owasp.org