



WATOBO

The Web Application Toolbox

OWASP
20.10.2010

Andreas Schmidt

SIBERAS

<http://www.siberas.de>

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>

Bio

■ Andreas Schmidt

- ▶ Seit 1998 im Security-Bereich tätig
- ▶ Seit 2001 spezialisiert auf Audits/Penetrationstests
- ▶ Mitgründer von siberas (2009)
 - <http://www.siberas.de>

Agenda

- (Markt-)Überblick
- Motivation
- Hauptkomponenten
- Highlights
- RoadMap
- Demo: WATOBO in action

Überblick

■ Kommerzielle Tools

- ▶ WebInspect, AppScan, NTOSpider, Acunetix,
- ▶ Primär für automatisierte Audits

■ Freie Tools

- ▶ WebScarab, Paros, BurpSuite(+\$), ...
- ▶ Primär für manuelle Penetrationstests

■ 1001+ Script-Tools

- ▶ Nikto, sqlmap, ...

Motivation

- Warum noch ein Tool?

Motivation

■ Kosten/Nutzen-Verhältnis von (kommerziellen) automatisierten Tools zu hoch!

- ▶ Typische Nachteile vollautomatisierter Tools, z.B. Logik-Fehler, ...
- ▶ manuelle „Begehung“ der Applikation trotzdem notwendig

■ Daseinsberechtigung dennoch gegeben!

- ▶ Einfache Bedienung, Reporting, zentrales Management, QA-Schnittstellen, ...

`pay() if pentester.needsFeature?(feature)`

Motivation

- Fehlende Transparenz bei kommerziellen Scannern
 - ▶ Check-Methoden werden meist „geheim“ gehalten
 - ▶ Zuviel „Voodoo“



Motivation

- Manuelle Tools besitzen meist kein Session-Management

- ▶ Erneutes Einloggen notwendig
 - ▶ Mühsames kopieren der SessionID

- Anpassen von (kommerziellen) Tools meist nur schwer möglich

- ▶ Fehlender Source-Code
 - ▶ Entwicklungsumgebung/Compiler notwendig
 - ▶ Oftmals umständlich und unflexibel, z.B. XML,

Motivation

- Manuelle Tools haben oft nur begrenzte automatisierte Funktionen
 - ▶ Ausnahme: BurpSuite Pro (\$\$)
- Vorteile quell-offener Tools
 - ▶ Leistungsfähigkeit und Grenzen können eingeschätzt werden
 - ▶ Können schnell an neue Anforderungen angepasst werden



Ansatz: Vorteile beider „Welten“

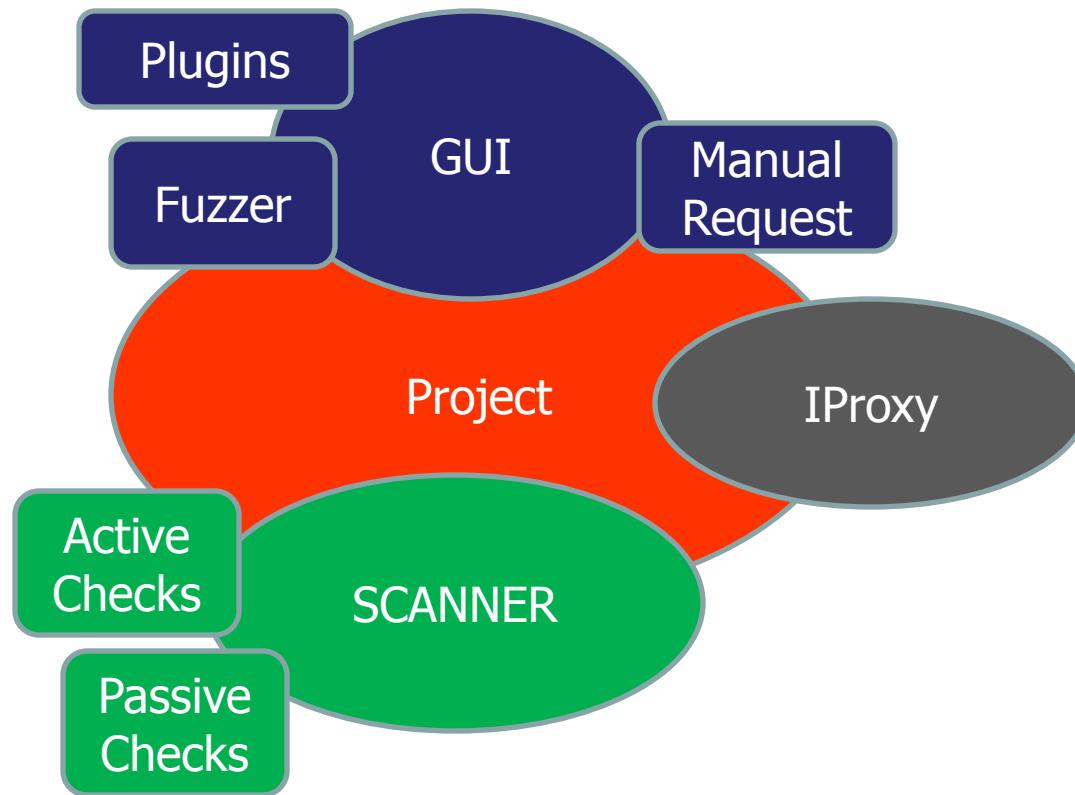
- Fokus: semi-automatisierte Penetrationstests
- Session-Management
- Proxy-basiertes Tool
- Web-Testing-Framework
 - ▶ typische Funktionen, wie Parser, Shaper, ...
 - ▶ einfach zu erweitern!
- Kein Angriffswerkzeug!
 - ▶ Keine Exploitmodule in Open-Source-Version



Zielgruppe

- Primär für professionelle Pentester!
 - ▶ Idealerweise mit Ruby-Kenntnissen
- Aber auch für Entwickler, Admins,...
 - Basis-Checks einfach durchzuführen
 - Kurze Beschreibung der Schwachstellen sowie Maßnahmenempfehlung

Komponentenüberblick



Komponente: GUI

■ GUI ist ein Muss!

- ▶ Web-App-Analyse ohne GUI nicht möglich
- ▶ CLI nicht für alle Bereiche sinnvoll ;)

■ Für manuelle Tests optimiert

- ▶ One-Click En-/Decoder
- ▶ Filter Funktionen
- ▶ Schnelle Analyse der Funktionsweise

Komponente: GUI

WATBOO by siberas (Version: 0.9.5rev202)

File Settings Tools View Help

Findings | Sites |

192.168.72.130

- Vulnerabilities
 - SQL-Injection
 - Reflected XSS [GET]
 - Local File Inclusion
 - [page] - etc/passwd
 - [page] - /etc/passwd500
 - Unencrypted Logins
 - dvwa/login.php
 - Cookie Security
 - Security Options
- Hints
 - HTTP Methods
 - Logins
 - dvwa/login.php
- Info
 - Server Headers
 - Apache/2.2.12 (Ubuntu)
 - PHP/5.2.10-2ubuntu6.1
 - /dvwa/vulnerabilities/upload/
 - Hotspots
 - login.php
 - ./login.php
 - index.php
 - instructions.php

Doc Filter | Text Filter | Options | excl. pics excl. docs excl. javascript excl. style sheets

Apply

	Method	Host	Path	Parameters	Status	Set-Cookie	Comment
1	GET	192.168.72.130	dvwa		301	Movr	
2	GET	192.168.72.130	dvwa/		302	Foun PHPSESSID=1	
3	GET	192.168.72.130	dvwa/login.php		200	OK	
6	POST	192.168.72.130	dvwa/login.php	username=admin&password=302 Foun			
7	GET	192.168.72.130	dvwa/index.php		200	OK	
13	GET	192.168.72.130	dvwa/vulnerabilities/xss_r/		200	OK	
14	GET	192.168.72.130	dvwa/vulnerabilities/xss_r/		200	OK	
15	GET	192.168.72.130	dvwa/vulnerabilities/xss_r/	name=	200	OK	
16	GET	192.168.72.130	dvwa/vulnerabilities/sql/		200	OK	
17	GET	192.168.72.130	dvwa/vulnerabilities/sql/		200	OK	
18	GET	192.168.72.130	dvwa/vulnerabilities/sql/	id=&Submit=Submit	200	OK	
19	GET	192.168.72.130	dvwa/security.php		200	OK	
20	GET	192.168.72.130	dvwa/security.php		200	OK	
22	POST	192.168.72.130	dvwa/security.php	security=low&seclev_submit=302 Foun security=low			
23	GET	192.168.72.130	dvwa/security.php		200	OK	
24	GET	192.168.72.130	dvwa/vulnerabilities/fi/	page=include.php	200	OK	
25	GET	192.168.72.130	dvwa/vulnerabilities/fi/	page=include.php	200	OK	
26	GET	192.168.72.130	dvwa/vulnerabilities/sql/		200	OK	
27	GET	192.168.72.130	dvwa/vulnerabilities/sql/		200	OK	
28	GET	192.168.72.130	dvwa/vulnerabilities/sql/	id=&Submit=Submit	200	OK	
29	GET	192.168.72.130	dvwa/vulnerabilities/xss_r/		200	OK	
30	GET	192.168.72.130	dvwa/vulnerabilities/xss_r/		200	OK	
31	GET	192.168.72.130	dvwa/vulnerabilities/xss_r/	name=	200	OK	
32	GET	192.168.72.130	dvwa/vulnerabilities/upload/		200	OK	
33	POST	192.168.72.130	dvwa/vulnerabilities/upload/		200	OK	
34	GET	192.168.72.130	dvwa/vulnerabilities/exec/		200	OK	
35	GET	192.168.72.130	dvwa/vulnerabilities/exec/		200	OK	
36	POST	192.168.72.130	dvwa/vulnerabilities/exec/	ip=&submit=submit	200	OK	
37	GET	192.168.72.130	dvwa/logout.php		302	Foun	
38	GET	192.168.72.130	dvwa/vulnerabilities/upload/		302	Foun	
39	GET	192.168.72.130	dvwa/logout.php		302	Foun	
40	GET	192.168.72.130	dvwa/vulnerabilities/upload/		302	Foun	
41	GET	192.168.72.130	dvwa/login.php		200	OK	
42	POST	192.168.72.130	dvwa/login.php	username=admin&password=302 Foun			
43	GET	192.168.72.130	dvwa/index.php		302	Foun PHPSESSID=1	
44	GET	192.168.72.130	dvwa/vulnerabilities/sql/		200	OK security=hig	
45	GET	192.168.72.130	dvwa/login.php		302	Foun PHPSESSID=1	
46	GET	192.168.72.130	dvwa/vulnerabilities/sql/		302	Foun PHPSESSID=1	
48	POST	192.168.72.130	dvwa/vulnerabilities/sql/	username=admin&password=302 Foun			
49	GET	192.168.72.130	dvwa/index.php		200	OK	

Finding "Local File Inclusion" [Chat-ID: 24]

Browser-View | Fuzzer | Manual Request

Request:

Text | Hex | Grep Highlight Reset

```
GET http://192.168.72.130/dvwa/vulnerabilities/fi/?page=../../../../etc/passwd HTTP/1.1
Host: 192.168.72.130
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.0; de; rv:1.9.2.8) Gecko/20100722 Firefox/3.6.8 (.NET CLR 3.5.30729)
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: de-de;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Referer: http://192.168.72.130/dvwa/security.php
Cookie: security=low; Milano=0012A9B12goodandy; Brussels=0029A9B1crisp5; Geneva=92BEF345Apecan635; PHPSESSID=b9474c1da707b54646ed6d
Connection: Close
Proxy-Connection: Close
Accept-Encoding: None
```

Response:

Text | Tagless | Hex | Grep Highlight Reset

```
HTTP/1.1 200 OK
Date: Fri, 15 Oct 2010 16:08:38 GMT
Server: Apache/2.2.12 (Ubuntu)
X-Powered-By: PHP/5.2.10-2ubuntu6.1
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 5324
Connection: close
Content-Type: text/html

root@192.168.72.130:/#
```



Komponente: I(nterceptor/)Proxy

- Klassische Proxy-Funktion

- Interceptor
 - ▶ Auffangen und Manipulieren von Requests/Responses

- Pass-Through
 - ▶ Server-Antwort wird direkt an Browser durchgereicht
 - ▶ Einstellbar: Content-Type/Content-Length
 - ▶ Applikation lässt sich flüssig bedienen!

- Pseudo-Server
 - ▶ Z.B für HTML-Preview

Komponente: Scanner

- Multi-Threaded
- Smart-Scan-Funktion
 - ▶ Reduziert Anzahl von Requests
 - ▶ Ähnliche URLs werden zusammengefasst
 - ▶ Berücksichtigt „Non-Unique-Parameter“
 - Z.B. `action=addUser` oder `function=showFile`
- Steuert Active-Checks

Komponente: Scanner

■ Feingranulare Definition des Target-Scopes

- ▶ Site (host:port)
- ▶ Root-Path
- ▶ Exclude-Patterns

■ Session-Management

- ▶ Erkennt Logout
- ▶ Kann (Re-)Login automatisiert durchführen

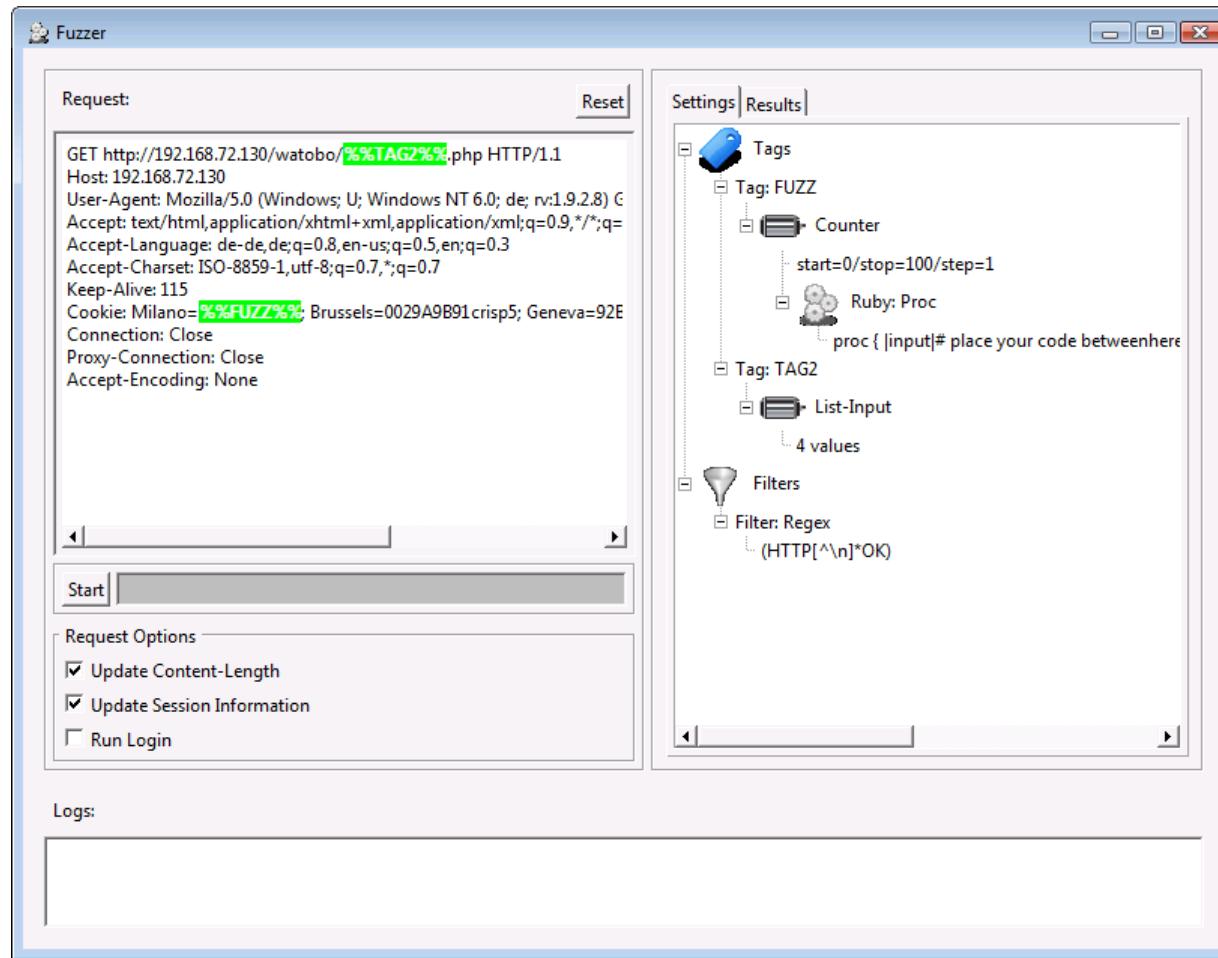
Komponente: Fuzzer

- Multi-Tag
- Multi-Generator
- Multi-Action
- Multi-Filter
- ...



USE THE FORCE, ...

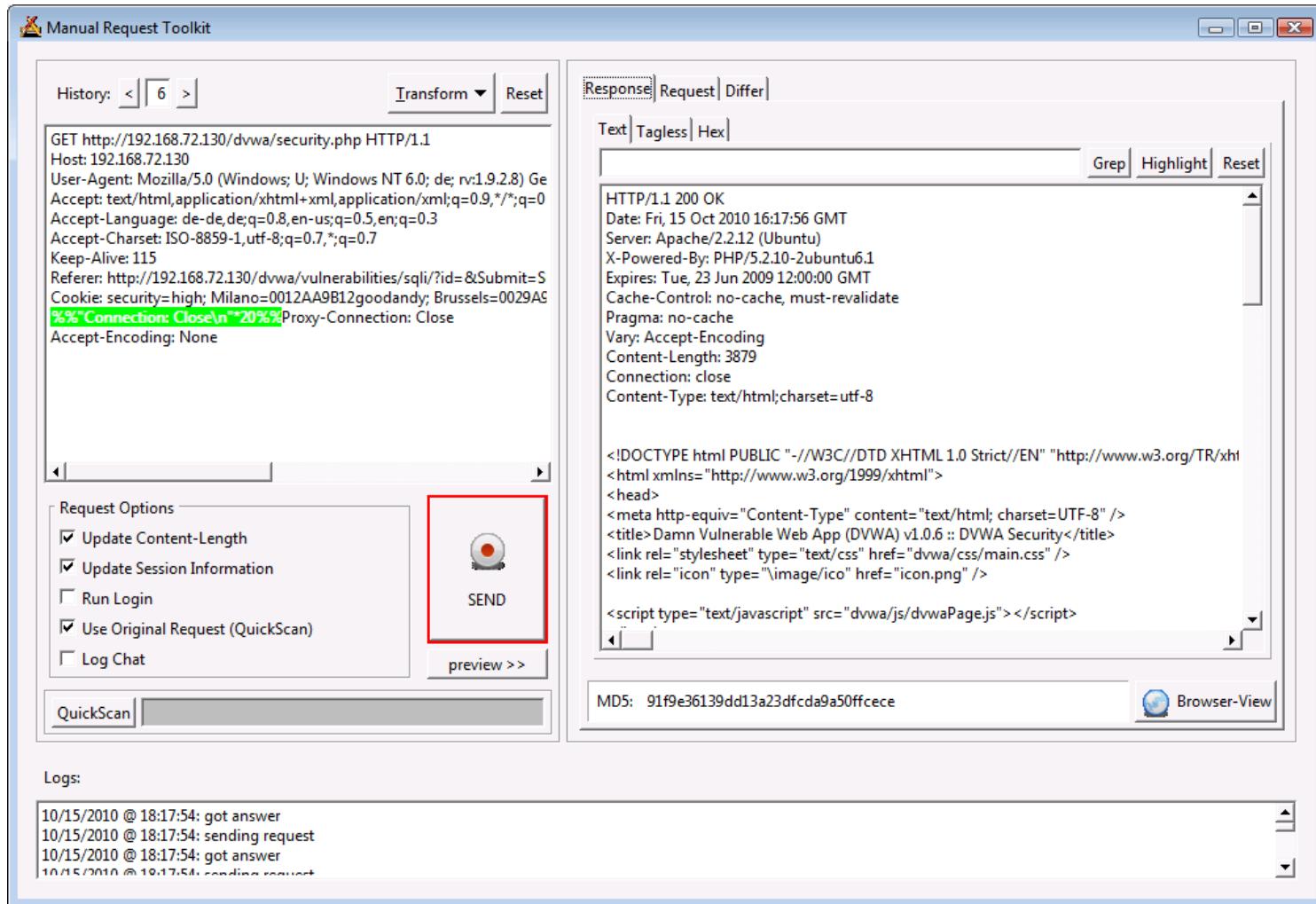
Komponente: Fuzzer



Komponente: Manual Request Editor

- Automatisierter Login
- Update der Session-Informationen
- Request-History
- Differ
- QuickScan
 - ▶ Gezieltes Scannen einer URL

Komponente: Manual Request Editor



Komponente: Active Checks

- Werden über Scanner gesteuert
- Dienen zum aktiven Testen
 - ▶ SQL-Injection
 - ▶ XSS
 - ▶ ...
- Gute Balance zwischen Einfachheit/Flexibilität
 - ▶ Nur mit Skript-Sprachen möglich!
 - ▶ Einige Hersteller haben eigene (Skript-)Sprachen, oder nutzen JavaScript



Komponente: Active Checks

Aktuelle Checkliste (13):

- + Dirwalker
- + Fileextensions
- + Http_methods
- + Domino_db
- + Lfi_simple
- + Jboss_basic
- + Its_commands
- + Its_services
- + Its_service_params
- + Its_xss
- + Sqli_simple
- + Sql_boolean
- + XSS_simple

IN STÄNDIGER
ENTWICKLUNG

Komponente: Passive Checks

- Grep-Style-Checks

- ▶ Pattern-Matching

- Identifiziert Schwachstellen

- ▶ Z.B. Cookie-Security, unverschlüsselte Anmeldung, ...

- Extrahiert hilfreiche Informationen

- ▶ Z.B. HotSpots, Email, IP's...

Komponente: Passive Checks

Aktuelle Checkliste (14):

- + Cookie_options
- + Cookie_xss
- + Detect_code
- + Detect_fileupload
- + Detect_infrastructure
- + Dirindexing
- + Disclosure_emails
- + Disclosure_ipaddr
- + Filename_as_parameter
- + Hotspots
- + Multiple_server_headers
- + Possible_login
- + Redirectionz
- + Redirect_url

IN STÄNDIGER
ENTWICKLUNG

Komponente: Plugins

■ Für individuelle Tests

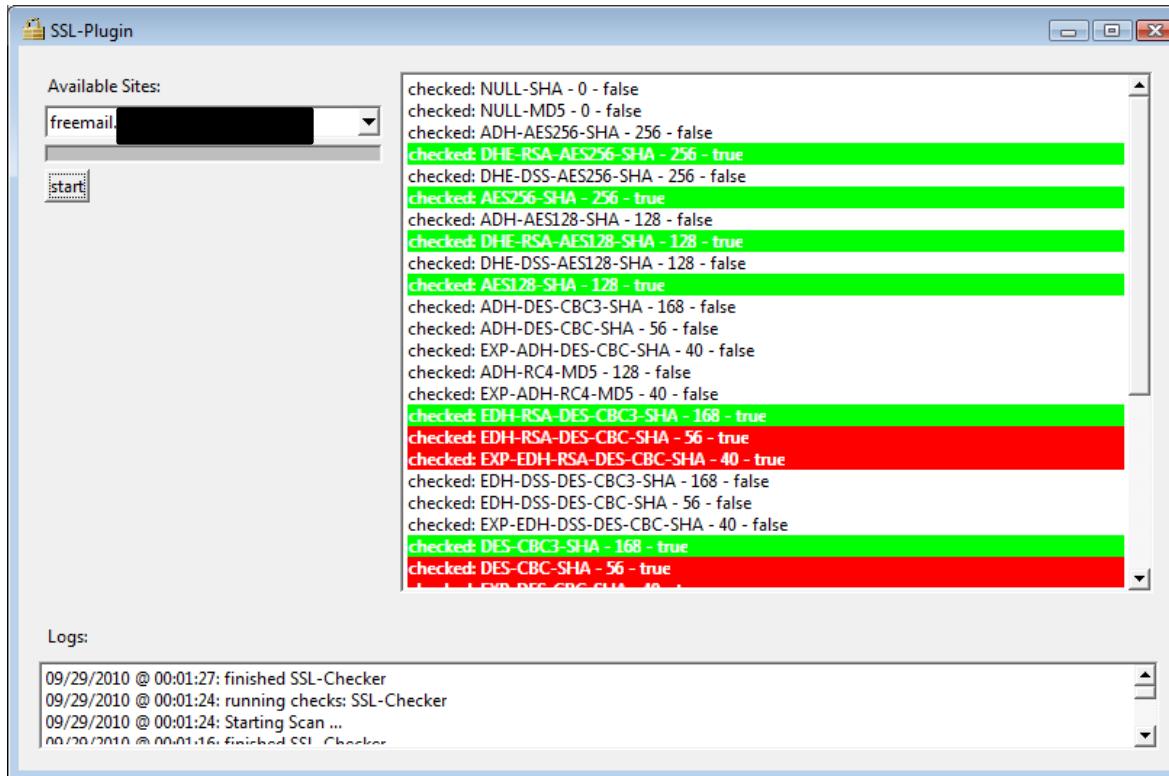
- ▶ Nicht Scanner-kompatibel
- ▶ Z.B. site-spezifische Checks, wie beispielsweise SSL-Cipher

■ Framework-Funktionen und Schnittstellen

- ▶ listSites, listDirs, ...
- ▶ SessionManagement
- ▶ Scanner

Plugin: SSL-Checker

- Prüft unterstützte SSL-Ciphers
 - ▶ Mittels vollständigen HTTP-Requests



Umsetzung

- Ruby, Ruby, Ruby, ...

- ▶ <http://www.ruby-lang.org>

- FXRuby für GUI

- ▶ Ruby-Port von Fox-Toolkit
<http://www.fxruby.org>

- Plattformunabhängig

- ▶ (FX)Ruby für Windows, Linux, MacOS, ...

- Entwicklungsplattform Windows

- ▶ Wird auch unter Linux (Backtrack) getestet



WATOBO Highlights

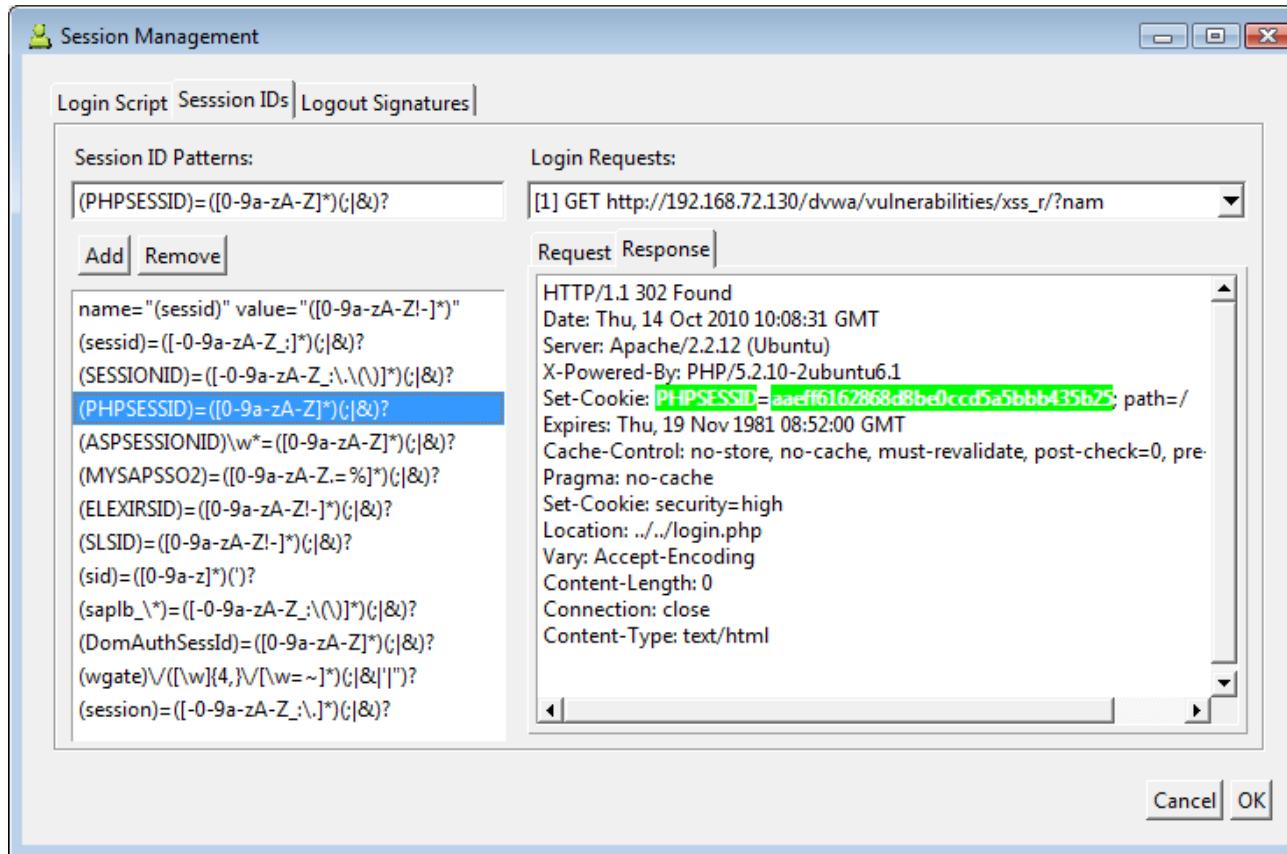
- Session Management
- Ruby-In-Ruby
- HTML-Preview

Highlight: Session Management

- Pattern-basiert
 - ▶ Regular Expressions
 - ▶ Hash[\$1]=\$2
- Header und Body wird analysiert
 - ▶ Nur text/*-Content-Types => Geschwindigkeit
- Session-IDs in Cookie und URLs
- Ca. 15 vordefinierte Patterns
- Regex-Validator

Highlight: Session Management

Beispiel: (PHPSESSID)=([0-9a-zA-Z]*)(;|&)?

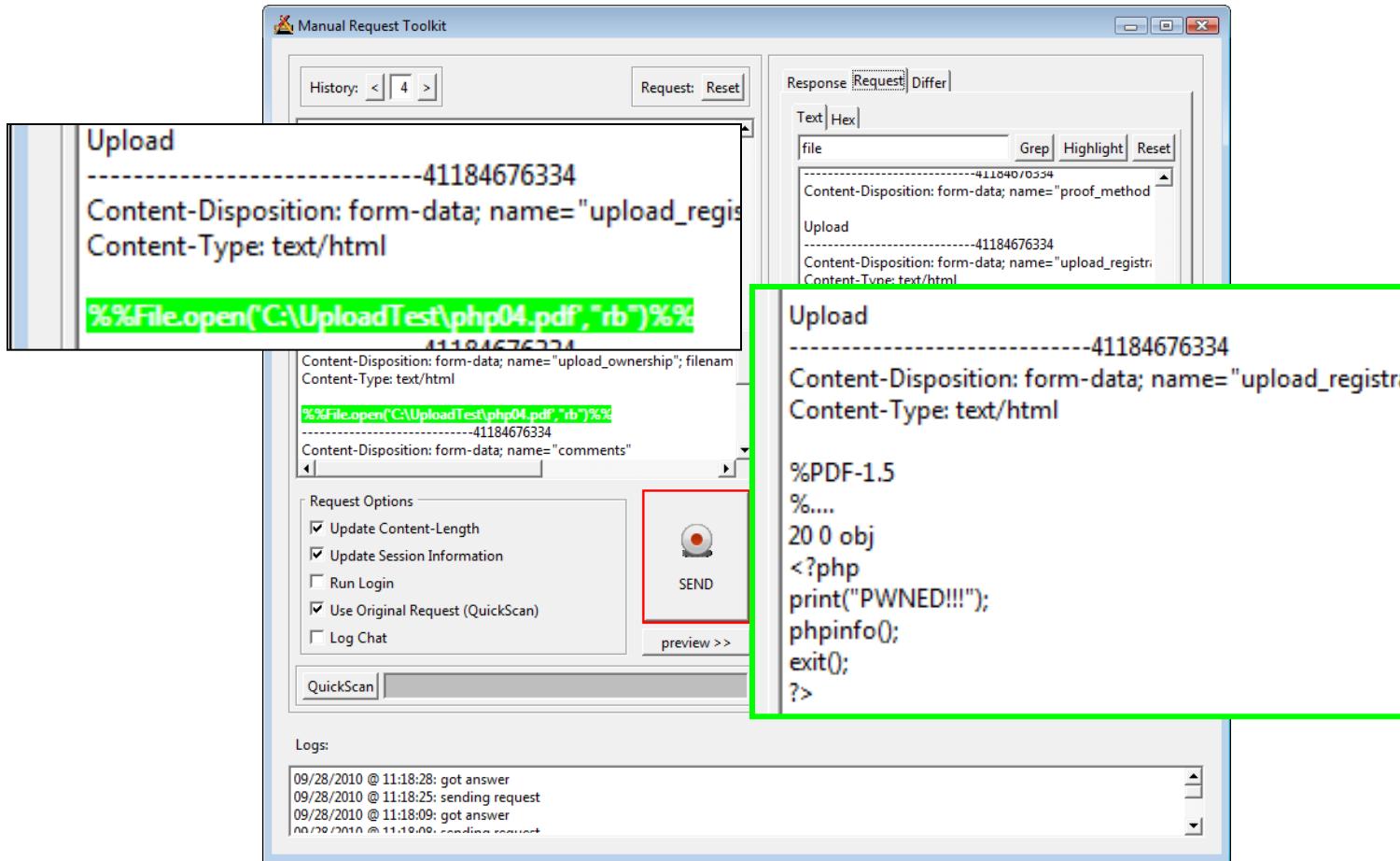


Highlight: Ruby-in-Ruby

- Mittels spezieller Tags (',%%') lässt sich direkt Ruby-Code integrieren
- Nützlich für die Erzeugung von
 - ▶ vielen Zeichen, Headern, ...
 - ▶ Binaerzeichen, Konvertierung, Berechnungen, ...
 - ▶ Daten aus verschiedenen Quellen, z.B. Dateien
- Fuzzer nutzt Ruby (procs) für „Actions“

Highlight: Ruby-in-Ruby

■ Manual Request Editor: Including Binary-Files



Highlight: HTML-Preview

- HTML-Preview sehr hilfreich
 - ▶ Doku-Screenshots, schnelle visuelle Analyse
- FXRuby besitzt kein HTML(WebKit)-Widget ☹
- ..., aber Browser gibt's auf jedem System
 - ▶ IE, Firefox
- Browser-Steuerung mittels JSSH (Firefox) und Win32OLE (IE)
 - ▶ http://www.croczilla.com/bits_and_pieces/jssh/



Road-Map

- CSRF-Token Handling!
- Recheck-Funktion
 - ▶ KB-Diffing
- Neue Module, Plugins, Parser, En-/Decoder
 - ▶ SOAP/XML
- Source-Code-Unterstützung
 - ▶ zum Abgleich der Angriffsfläche

Road-Map

■ Dokumentation

- ▶ Videos, rdoc
- ▶ <http://watobo.sourceforge.net>

■ Installer

■ Schulungen/Trainings/Workshops!

WATOBO - Demo

- <http://watobo.sourceforge.net>