WATOBO
The Web Application Toolbox

Andreas Schmidt

SIBERAS
http://www.siberas.de

OWASP
20.10.2010

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
  - Skript-Sprachen

WATOBO

Web Application Toolbox

Ansatz: Vorteile beider „Welten“

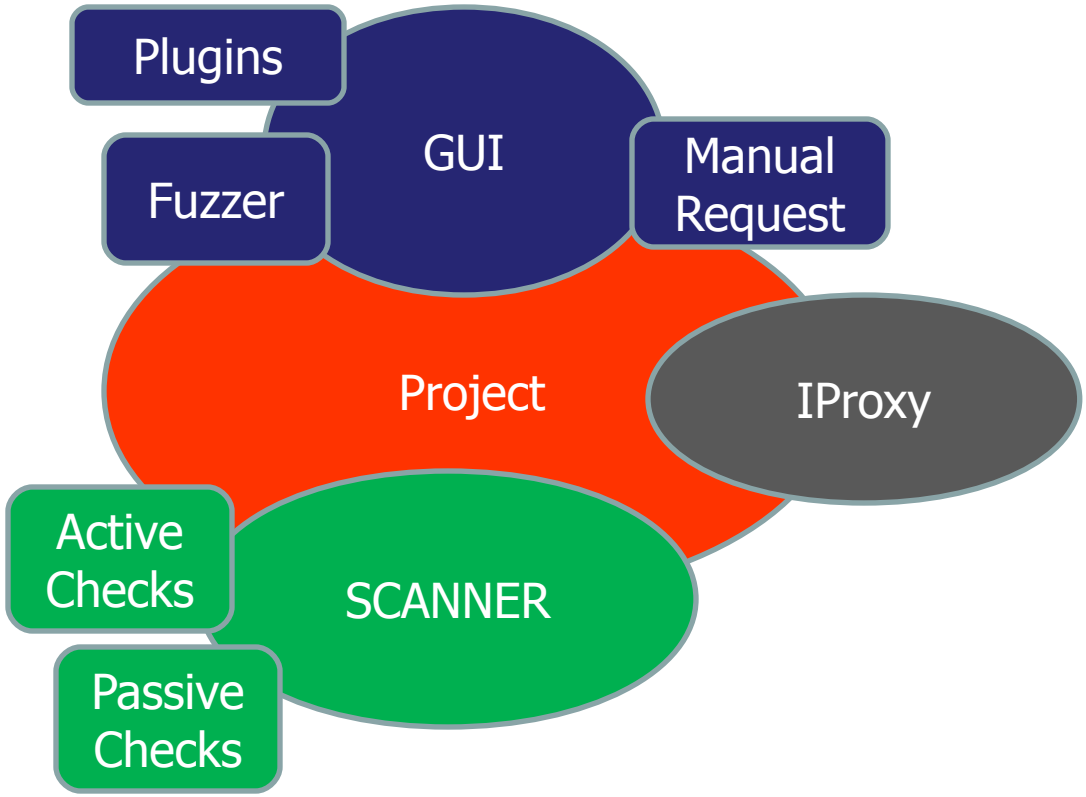
- Fokus: semi-automatisierte Penetrationstests
- Session-Managment
- Proxy-Basiertes Tool
- Web-Testing-Framework
  - typische Funktionen, wie Parser, Shaper, ...
  - einfach zu erweitern!

- 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

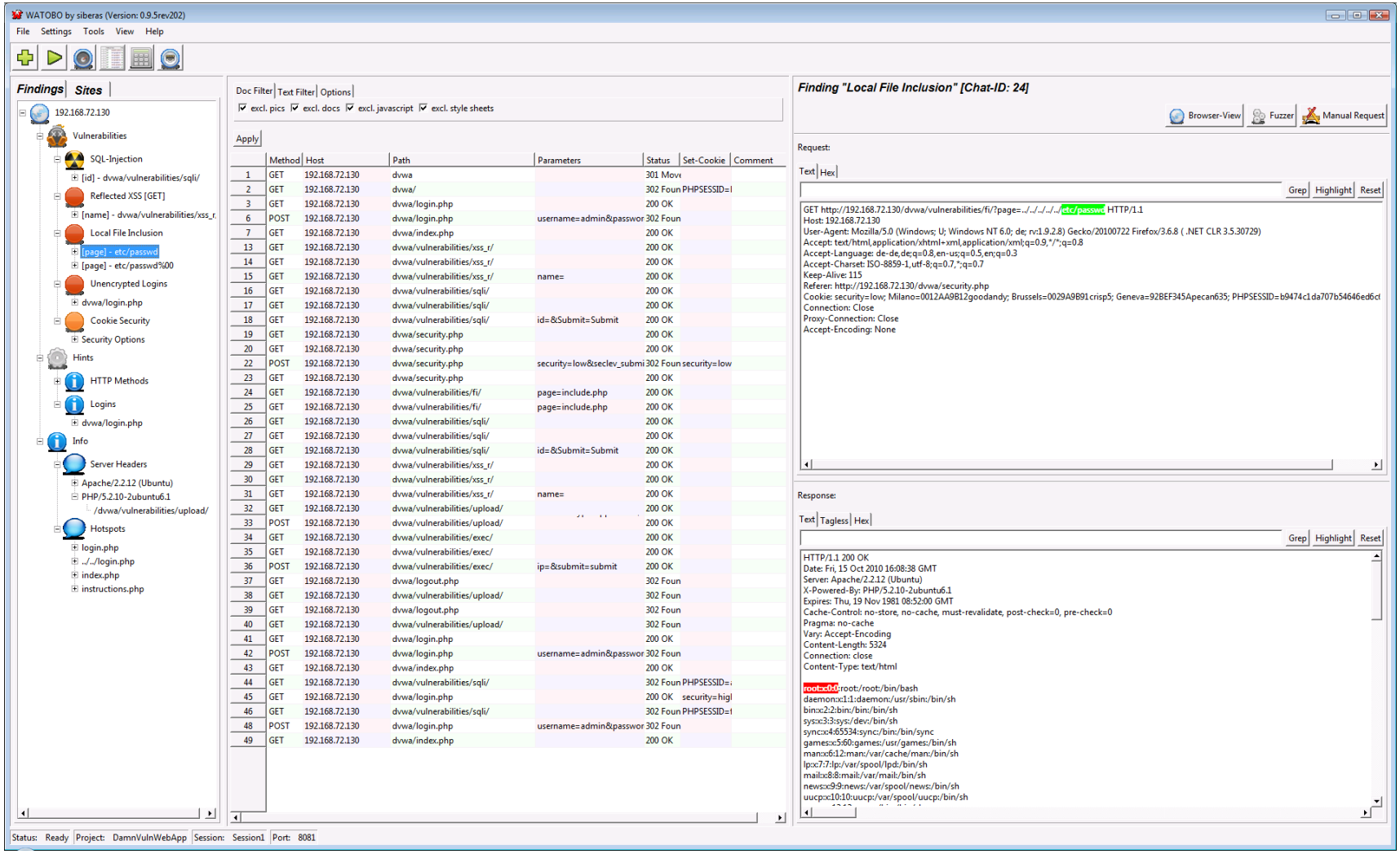


Komponente: GUI

- 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



Komponente: I(nterceptor/)Proxy

- Klassische Proxy-Funktion
- Interceptor
  - Abfangen 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

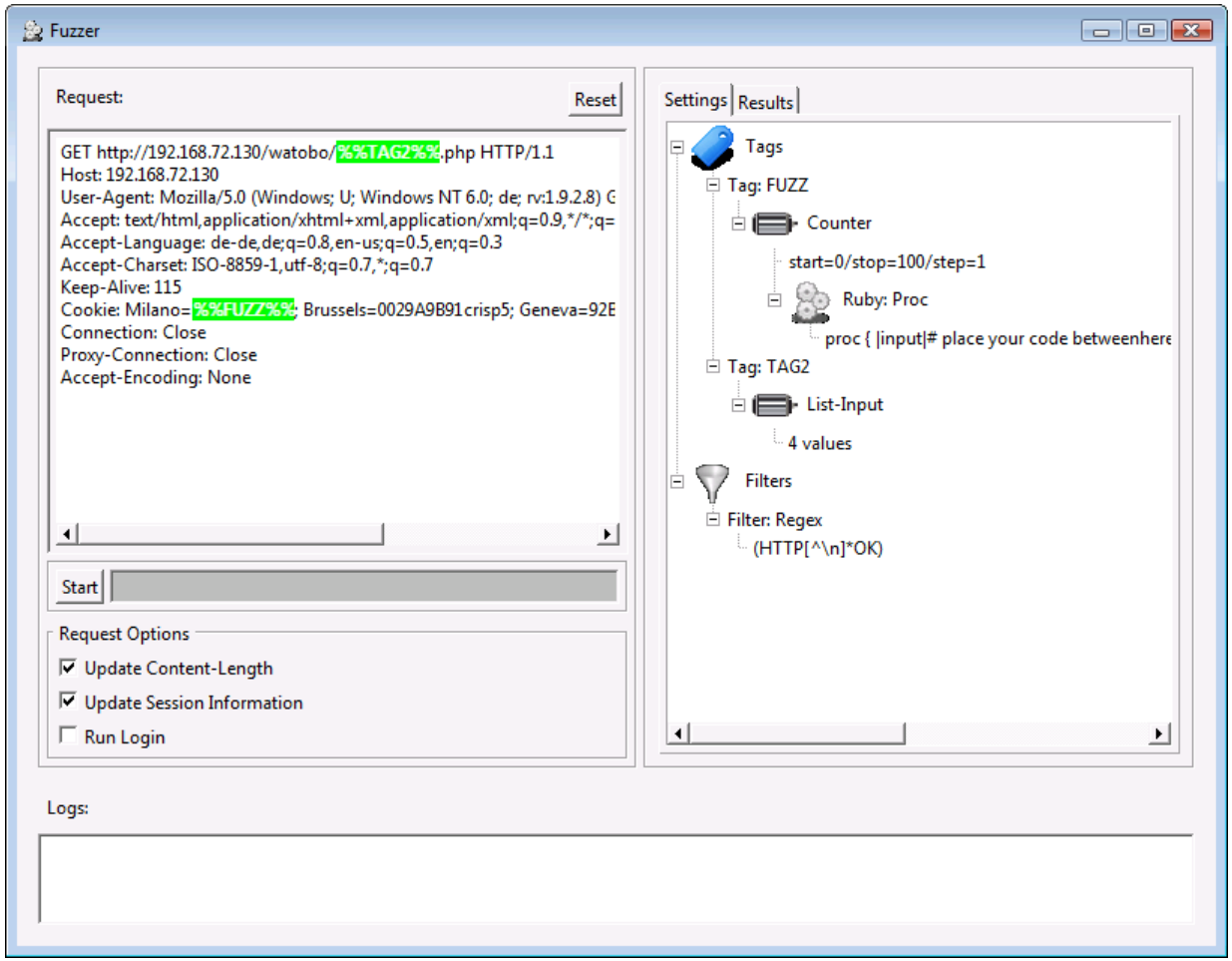
- Feingranulare Definition des Target-Scopes
  - Site (host:port)
  - Root-Path
  - Exclude-Patterns
- Session-Management
  - Erkennt Logout
  - Kann (Re-)Login automatisiert durchführen

Komponente: Fuzzer

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

USE THE FORCE, ...

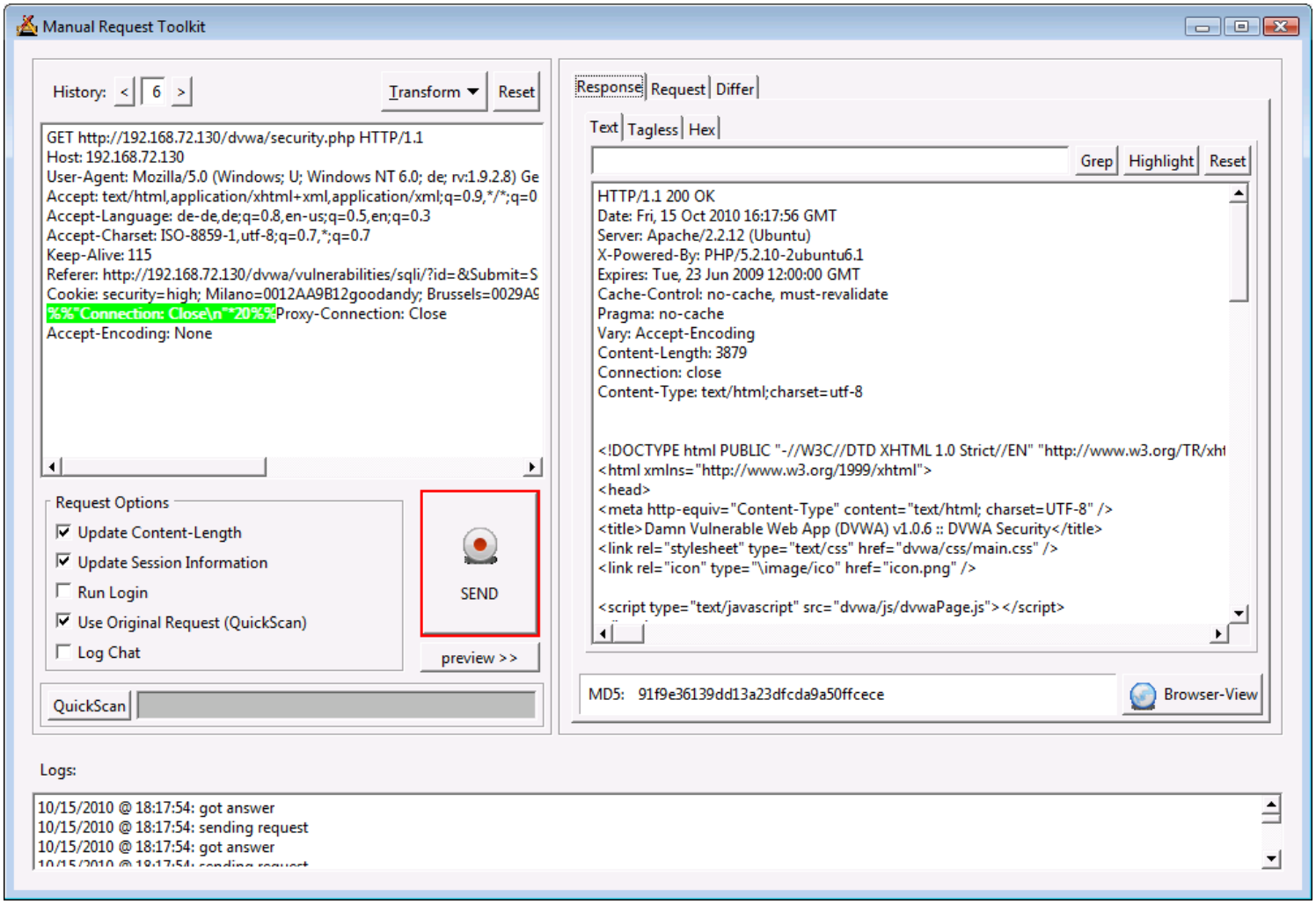
Komponente: Fuzzer



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_parameter
+ 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

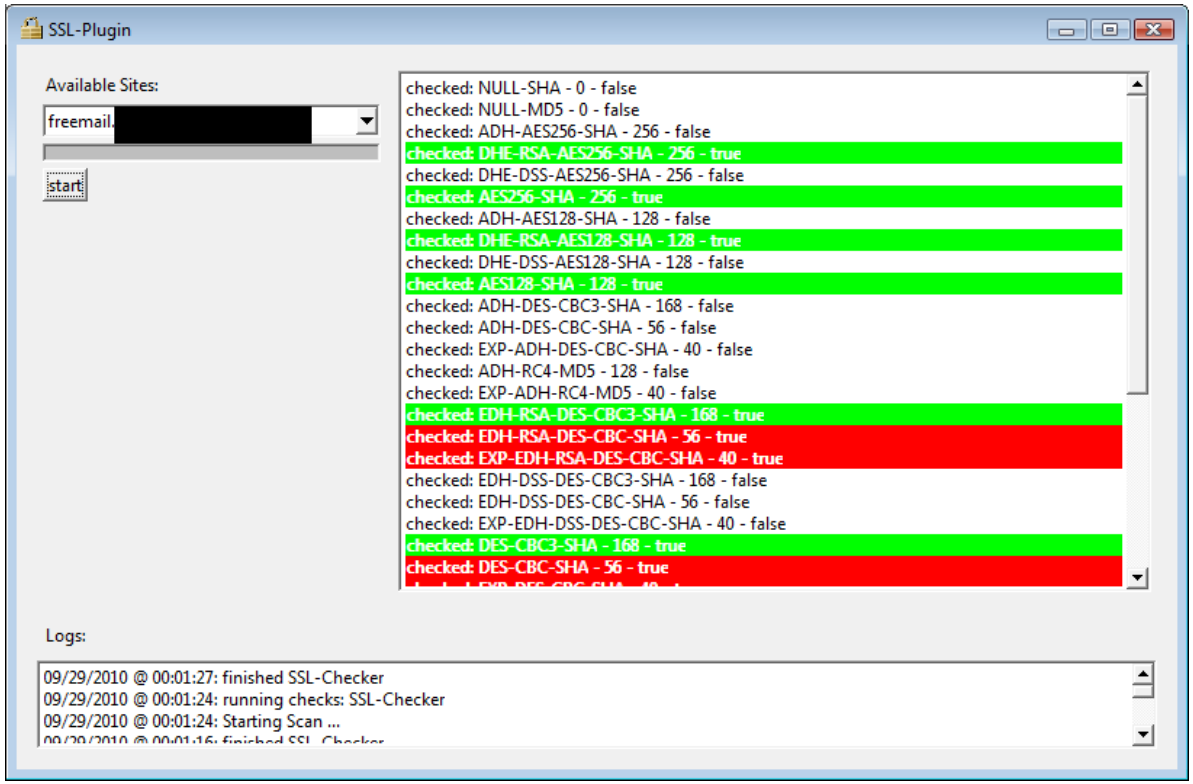
IN STÄNDIGER ENTWICKLUNG

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 unterstütze 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

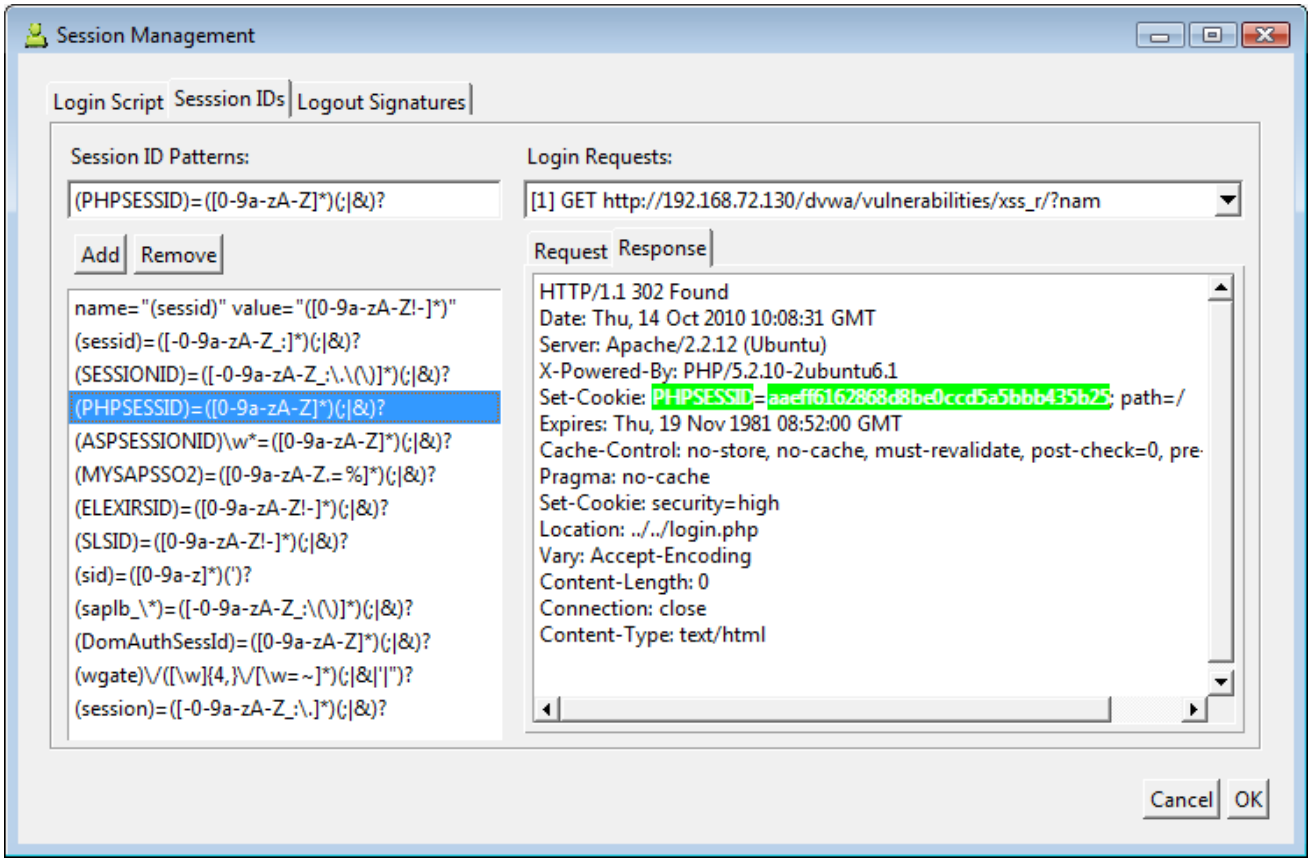
- Session Management
- Ruby-In-Ruby
- HTML-Preview

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

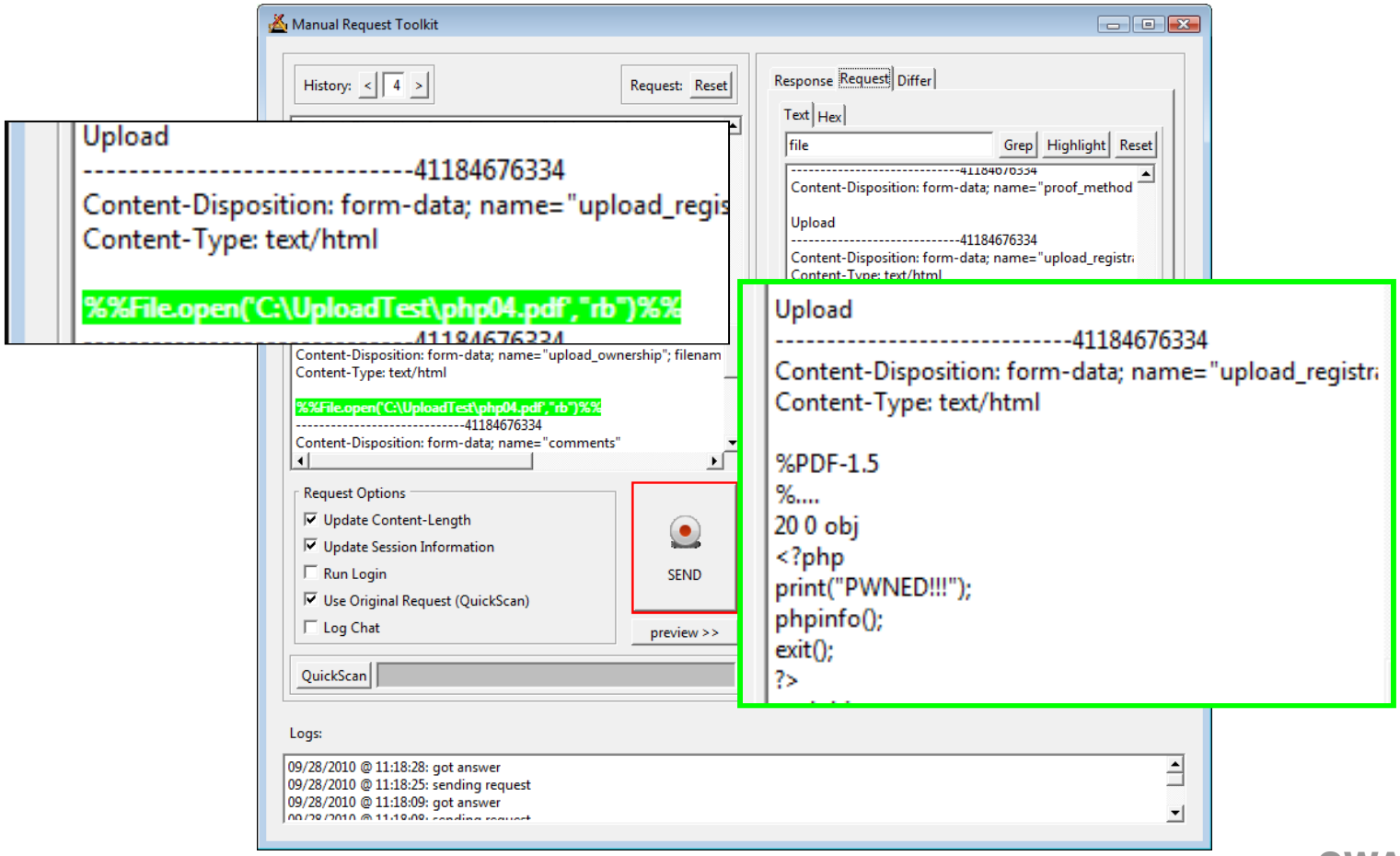
- Mittels spezieller Tags (‚%%‘) lässt sich direkt Ruby-Code integrieren

- 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
- Installer
- Schulungen/Trainings/Workshops!

WATOBO - Demo