

w3af – A framework to own the Web

Andrés Riancho
ariancho <at> cybsec.com

Talk objectives

- Let the security community know about w3af
- Introduce new and interesting ways of exploiting web application vulnerabilities
- Share ideas!

Agenda

- Who am I ?
- Current state of WebAppSec scanners
- What w3af is
- What w3af is not
- Main features
- Architecture
- Plugins
- Tactical exploitation
- Discovery demos
- Interesting features
- Virtual daemon
- Web 2.0
- Web Services
- Known bugs
- Future
- Main contributors
- Conclusions

who am i ?

- Security Consultant
- Programmer
- Open source evangelist
- Web Application security enthusiast
- Background in networking, IPS design and evasion

Current state of Web App Security scanners

- Commercial
 - High price
 - If available, poor extensibility (bad API's, bad programming languages to extend functionality)
 - No user community
 - No exploitation phase
 - Good reporting (cute PDF to send to management)
 - Many paid programmers, thoroughly tested products

Current state of Web App Security scanners

- Open source
 - Small independent tools/scripts that can't communicate with each other
 - Each tool re implements the wheel (authentication, thread management, proxy support, etc)
 - Some tools do audit + exploitation
 - Small or non user community
 - No reporting, they generate large text files
 - Mostly programmed without thinking about extensibility

Current state of Web App Security scanners

- Open source (contd.)
 - Some really cool tools exist and should not be left out of this summary:
 - sqlmap
 - sqlninja
 - wapiti
 - absinthe

Current state of Web App Security scanners

The solution is to implement a web application security framework, where everyone can contribute with his knowledge.

A project where adding a feature is easy and can be done in minutes instead of days!

What w3af is

- w3af stands for **W**eb **A**pplication **A**ttack and **A**udit **F**ramework
- An Open Source project (GPLv2)
- A script that evolved into a serious project
- A vulnerability scanner
- An exploitation tool

What w3af is not

- Stable software
- The solution to all your web application security problems
- A replacement for manual pen-testing
- A point&click hack tool
- A hack tool

Main features

- Finds common and uncommon web application vulnerabilities.
- Cross platform (written in python).
- Uses Tactical exploitation techniques to discover new URLs and vulnerabilities
- Web and console user interface

Main features

- Web Service support
- Exploits [blind] SQL injections, OS commanding, remote file inclusions, local file inclusions, XSS, unsafe file uploads and more!
- WML Support (WAP)
- Really easy to extend
- **Synergy** among plugins

Main features

- Ability to find vulnerabilities in query string, post data, URL filename (`http://a/f00_injectHere_b4r.do`), headers, file content (when uploading files with forms) and web services.
- Number of plugins: 110 and growing
- w3af is **smart**, more on this later ;)

Architecture

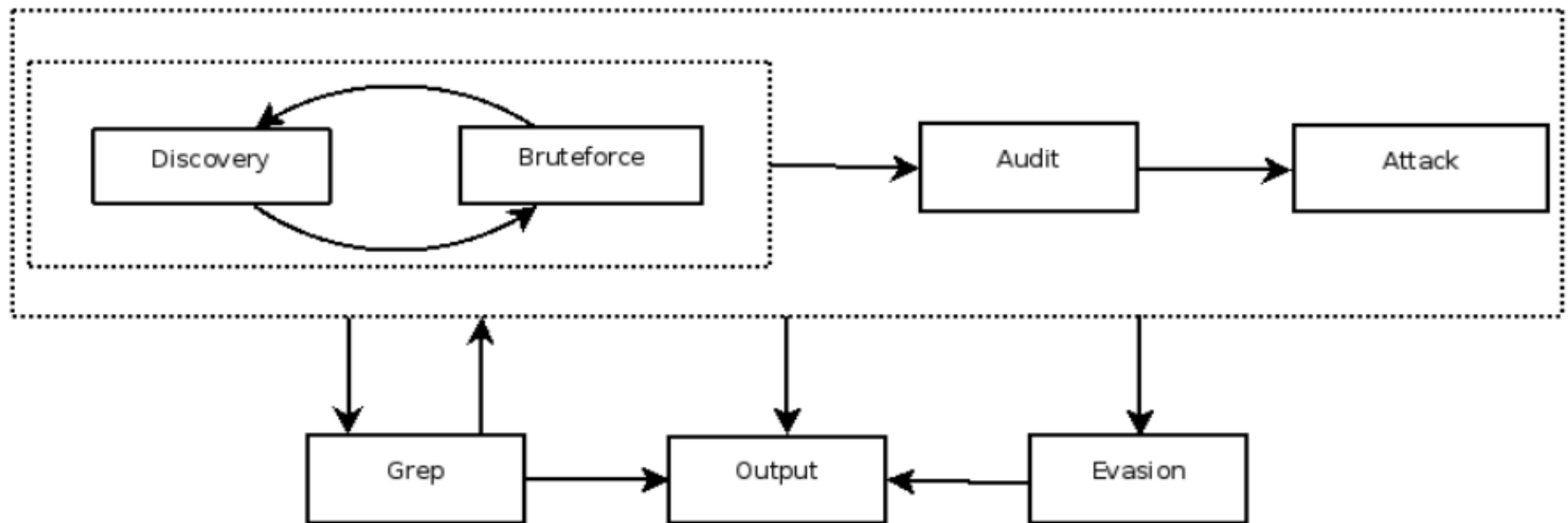
- w3af is divided in two main parts, the **core** and the **plugins**.
- The core coordinates the process and provides features that plugins consume.
- Plugins share information with each other using a **knowledge base**.
- Design patterns and objects everywhere !

Architecture

- 8 different types of plugins exist:
 - discovery
 - audit
 - grep
 - attack
 - output
 - mangle
 - evasion
 - bruteforce

Architecture

- Information flow between plugins:



Plugins | Discovery

They find new URLs and create the corresponding fuzzable requests; examples of discovery plugins are:

- webSpider
- urlFuzzer
- googleSpider
- pykto

Plugins | Discovery

They are run in a loop, the output of one discovery plugin is sent as input to the next plugin. This process continues until all plugins fail to find a new fuzzable request.

Other discovery plugins try to fingerprint remote httpd, allowed HTTP methods, verify if the remote site has an HTTP load balancer installed, etc.

Plugins | Audit

They take the output of *discovery* plugins and find vulnerabilities like:

- [blind] SQL injection
- XSS
- Buffer overflows
- Response splitting.

As vulnerabilities are found, they are saved as *vuln objects* in the knowledge base.

Plugins | Grep

These plugins grep every HTTP request and response to try to find information.

Examples of *grep* plugins are:

- findComments
- passwordProfiling
- privateIP
- directoryIndexing
- getMails
- lang

Plugins | Attack

These plugins read the *vuln objects* from the KB and try to exploit them. Examples of *attack* plugins are:

- mySqlWebShell
- davShell
- sqlmap
- xssBeef
- remote file include shell

Plugins | Others

- *Output:* They write messages to the console, html or text file.
- *Mangle:* They modify requests and responses based on regexs.
- *Evasion:* They modify the requests to try evade IDS detection.
- *Bruteforce:* They bruteforce logins.

Mini demo

Let's see what this is all about...

Tactical Exploitation

- A different way to do pen-test
- Analyze all the available information related to the target and apply it to the pen-test process
- Focus on the applications

Tactical Exploitation

What w3af does about tactical exploitation:

- vhost search in MSN
- mail address searches in Google, MSN, PKS.
- password profiling
- halberd
- archive.org search
- search Google, MSN, Yahoo

Tactical Exploitation

Password profiling plugin:

- Reads every HTTP response and count the word repetitions
- Support for HTML, Text, and PDF (using pyPDF).
- PPP supported file types are extended using plugins (plugins for plugins!)
- Smart HTML counting of word repetitions, knows about `<h[1-3]>` and text size.

Discovery demo

This demo will show:

- fingerPKS, fingerMSN, fingerGoogle
- bruteforce using collected usernames, and dynamically generated passwords:
 - username
 - target site (`www.domain.com` ; `domain.com` ; `domain`)
 - passwords generated by the password profiling plugin

Discovery demo (contd.)

Let's rock...

Interesting features

- Google Sets
 - input:
 - <http://f00/bar?a=football>
 - <http://f00/bar?a=tennis>
 - URLs to be tested:
 - <http://f00/bar?a=golf>
 - <http://f00/bar?a=rugby>

Interesting features

- wordnet
 - input:
 - <http://f00/bar?a=dog>
 - URLs to be tested:
 - <http://f00/bar?a=cat>
 - <http://f00/bar?a=bird>
 - <http://f00/bar?a=animal>

Interesting features

- digitSum
 - input:
 - <http://f00/bar?a=5>
 - URLs to be tested:
 - <http://f00/bar?a=4>
 - <http://f00/bar?a=6>

Interesting features

- digitSum
 - input:
 - <http://f00/bar5-9.htm>
 - URLs to be tested:
 - <http://f00/bar6-9.htm>
 - <http://f00/bar7-9.htm>
 - <http://f00/bar5-8.htm>
 - <http://f00/bar5-10.htm>

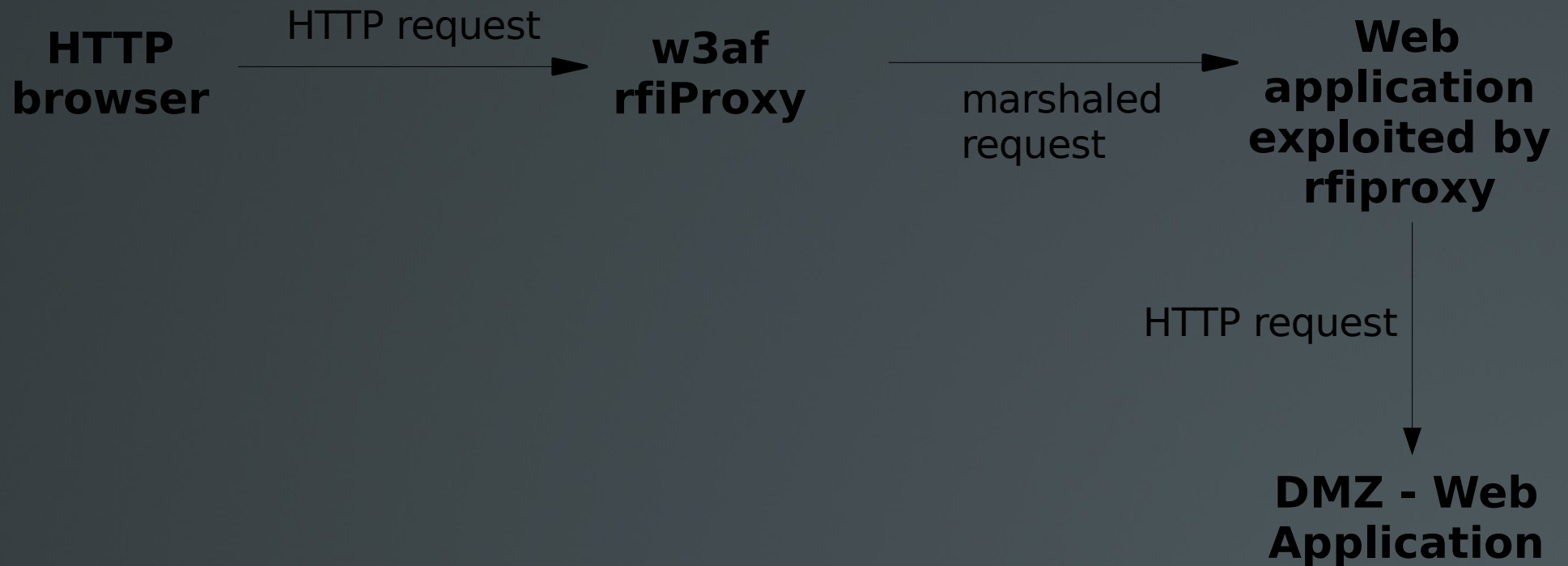
Interesting features

- archiveDotOrg plugin
 - Searches archive.org for older versions of the site, links that were linked somewhere in the past and now are kept in the dark
 - Old and unmaintained sections are prone to vulnerabilities
 - This plugin is a time machine!

Interesting features

- Remote file inclusion proxy
 - Using a remote file inclusion vulnerability, w3af implements an attack plugin that will run a local proxy that uses the remote server as exit point for HTTP requests.
 - Ideal for pivoting into the LAN / DMZ.

Interesting features



Interesting features

- Use of PHP easter eggs to fingerprint the remote PHP version.
 - Old and almost forgotten technique
 - Accurate fingerprinting
 - Almost nobody disables the eggs
(`expose_php = off;`)

Interesting features

- w3af verifies if your server was pwned in the past:
 - phishtank
 - googleSafeBrowsing
 - detectPhishing

Interesting features

- Good Samaritan module
 - A faster way to exploit blind SQL injections!
 - A **funny** way to exploit blind SQL injections!
- A demo will show what I'm talking about...

Good Samaritan Demo

Guiding the blind man

Virtual daemon

- Ever dreamed about using metasploit payloads to exploit web applications ? NOW you can do it !
- How it works:
 - I coded a metasploit plugin, that connects to a virtual daemon and sends the payload.
 - The virtual daemon is runned by a w3af attack plugin, it receives the payload and creates a tiny ELF / PE executable

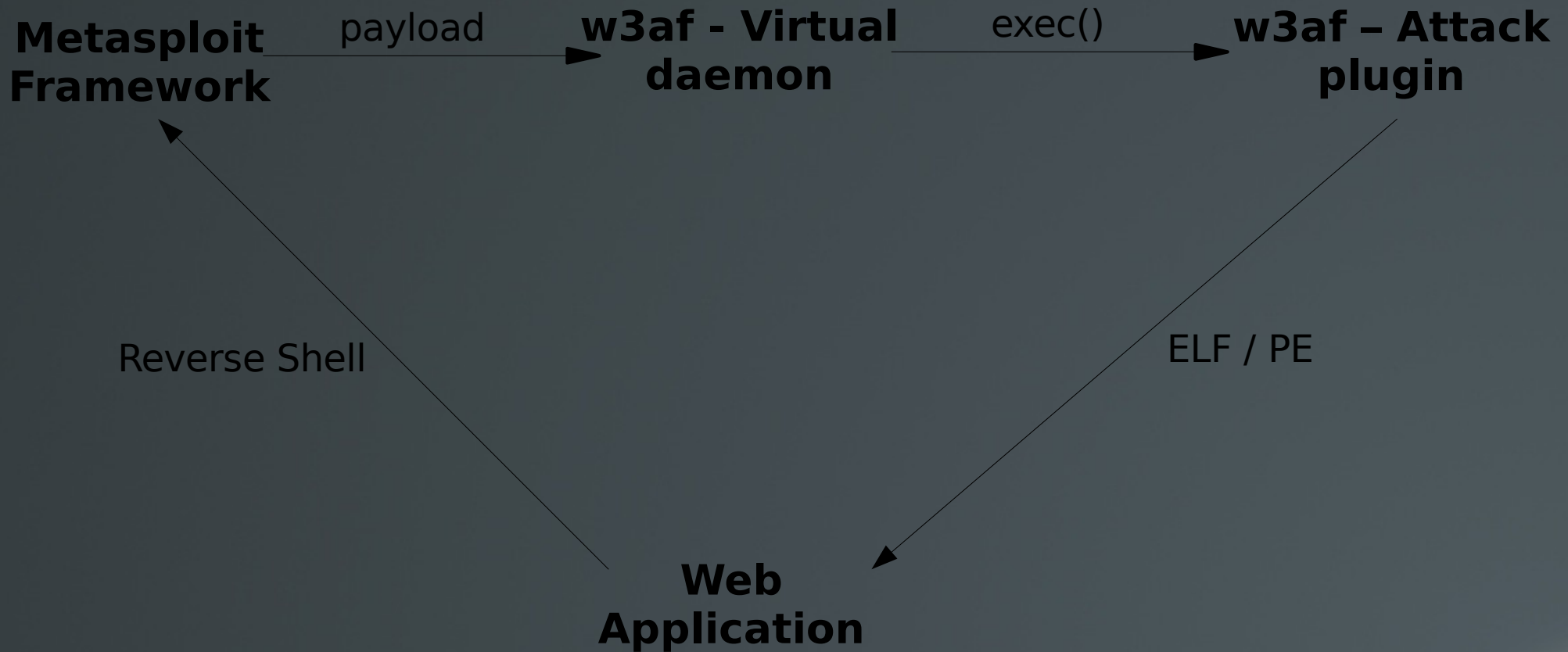
Virtual daemon

- How this works (contd.):
 - The attack plugin knows how to exec remote commands, and the virtual daemon knows how to upload the ELF/PE using “echo” or some other inband method.
 - A new scheduled task is created to run the payload, and the metasploit plugin is ordered to wait
 - The payload is run on the remote server.

Virtual daemon

- How this works (contd.):
 - Normal communication between metasploit and the exploited service follows.

Virtual daemon



Virtual daemon

demo!

Virtual daemon

- Things that won't work:
 - Use of existing connection to transverse NAT / restrictive firewalls

Web 2.0 Support

w3af can analyze pages that make heavy use of JavaScript. The **manual** solution available to achieve this task is the **spiderMan** plugin.

- Local proxy daemon
- Analyzes requests and creates fuzzable requests
- The user needs to navigate the JavaScript sections of the site

Web Services

w3af can find WSDLs using several methods:

- `discovery.wsdlFinder`
- `grep.wsdlGreper`

With the WSDL identified, w3af parses the definition file and creates the corresponding fuzzable requests. These requests are then used by audit and attack plugins.

Web Services

- ALL audit plugins work with web services!
- Exploit plugins *should* work with web services ;)
- w3af uses SOAPpy to parse the WSDL files, this introduces a limitation, because complex WSDL files (mainly doc-style) are not fully supported.

Known bugs

w3af is a work in progress, here is a short bug list:

- Random timeouts in HTTP requests
- Memory consumption, 100% CPU
- Threads “work”
- Fuzzable requests need refactoring
- Attack plugins need refactoring
- and much, much more..

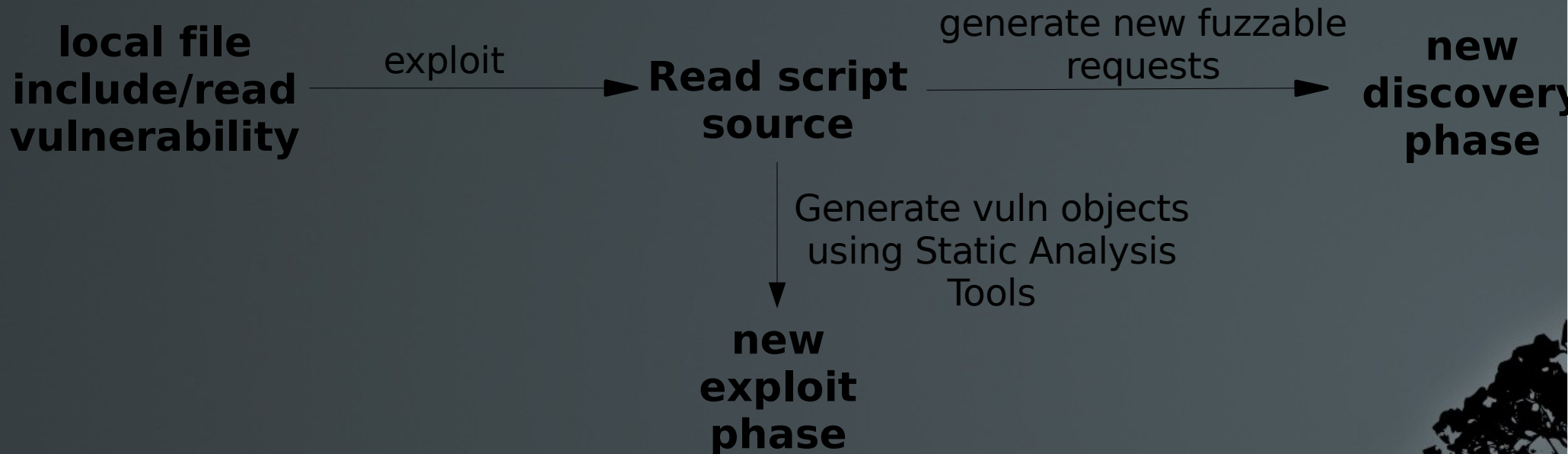
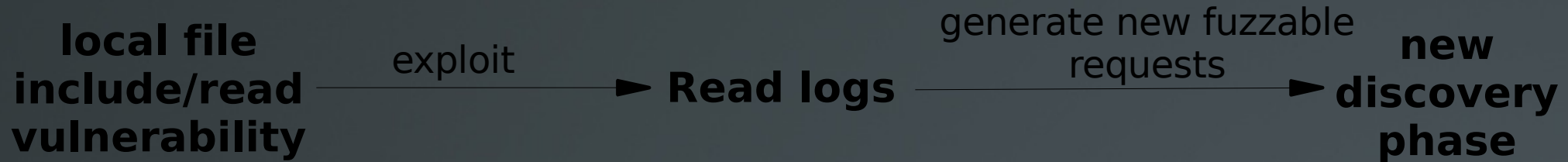
import future

- Some level of Javascript support
- More stable core
- More attack plugins, refactoring of attacks.
- Better webUI
- Better management report generation
- Replace utidy with beautiful soup
- Long descriptions for vulnerabilities

import future

- Replace SOAPpy with ZSI
- And maybe...
 - Static code analysis of scripting languages
 - Apache / IIS log analysis
- Another project by itself, but could be integrated into w3af: “IPS evasion using a userland TCP/IP stack like muXTCP”

import `__future__`



Main contributors

- Mariano Nuñez Di Croce (webUI and lot's of ideas)
- Juan Pablo Perez Etchegoyen (html output plugin)
- Mike Harbisson (documentation review and testing)
- Bernardo Damele (testing and sqlmap)
- Victor Montero (ideas, moral and political support)

Project information

- Site
 - <http://w3af.sf.net/>
- Mailing list and sourceforge home
 - <http://sourceforge.net/projects/w3af/>
- Project leader contact
 - andres.riancho <at> gmail.com
 - ariancho <at> cybsec.com

Project sponsor



CYBSEC

- 11 years experience
- Clients in LATAM, USA and Europe
- Based in Argentina
- Professional Objectivity

www.cybsec.com

Conclusions

- Frameworks are the future
- Tactical exploitation is going to get more and more important
- w3af has a huge potential
- It is open source, you should contribute!

Questions?

Feature requests ?
ideas? Bug reports?
contributions Rants about
web2.0? i want flash support! Web
Services hacking.