w3af 1.0 (now with stable code) and HTTP Fuzzer analysis

Andrés Riancho
andres@bonsai-sec.com

YSTS - 2009

Introduction

- Questions I'll try to answer today:
  - What's w3af ?
  - How can **you** use it?
  - How does w3af **compare** to commercial scanners?
  - **How hard can it be!** It's HTML + HTTP!

w3af

- **w3af** stands for **W**eb **A**pplication **A**ttack and **A**udit **F**ramework

- **A vulnerability scanner**
- **An exploitation tool**

- An Open Source project (**GPLv2**)

- A set of scripts that evolved into a serious project

Compressed w3af history

Main features

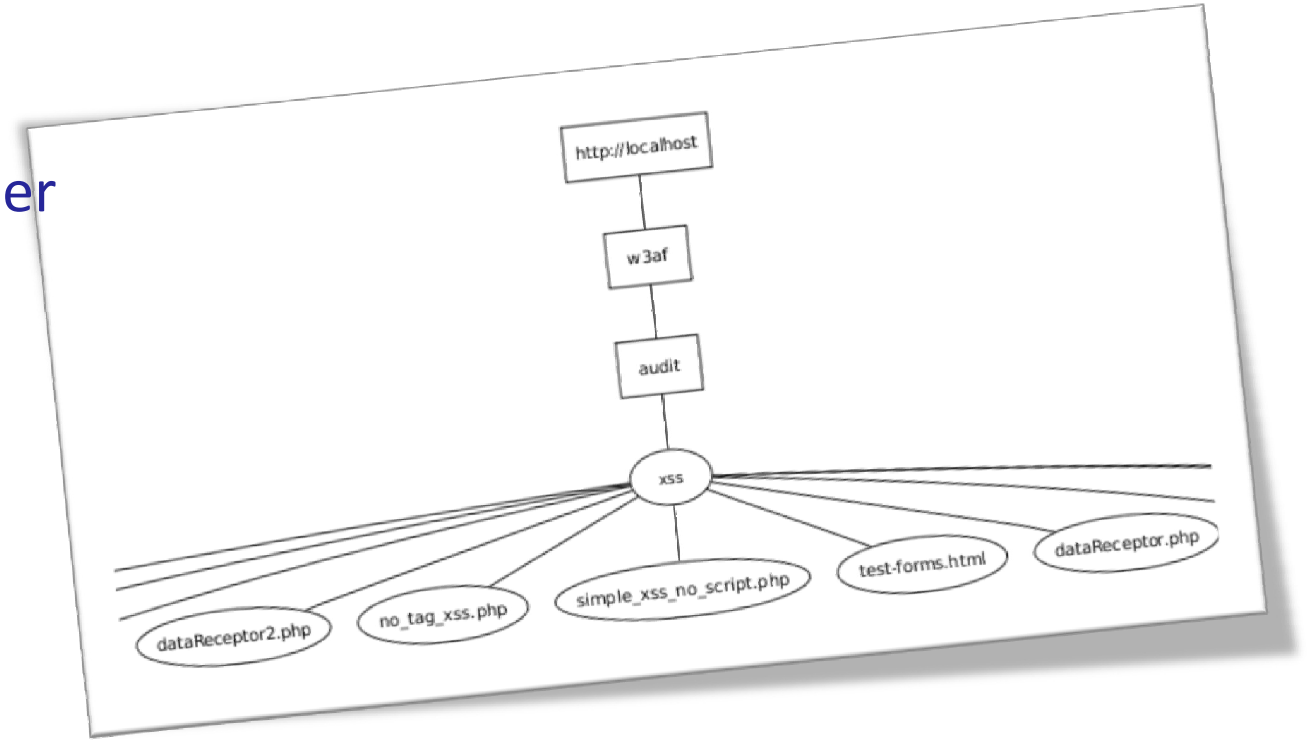
- Extensible using **plugins**
- 136 plugins and growing, the last one was developed by Jon Rose from Trustwave who's in the audience!

- A decent fuzzer, more on this later ;)
- Web Service support
- Broken HTML support

- **Manual and automated** analysis of web applications
  - **MITM proxy**
  - **Manual request editor**
  - **Fuzzy request generator**

Plugins | Discovery

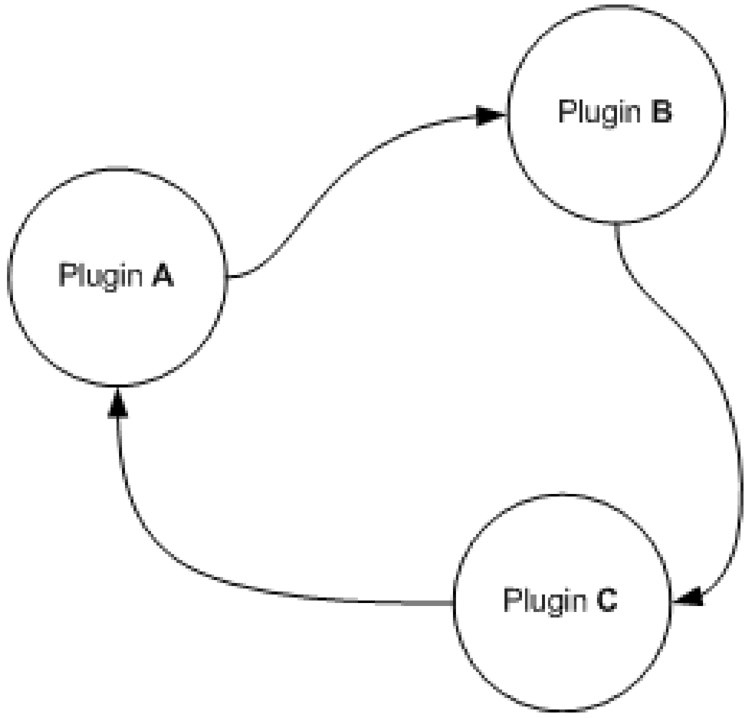
- They **find new URLs , forms**, etc. and create a complete sitemap. The findings are saved in the core as **fuzzable requests.** Examples of discovery plugins are:
  - webSpider
  - urlFuzzer
  - googleSpider
  - pykto



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 resource.

- This feature increases the **code coverage** of each scan, allowing the audit plugins to find more vulnerabilities.



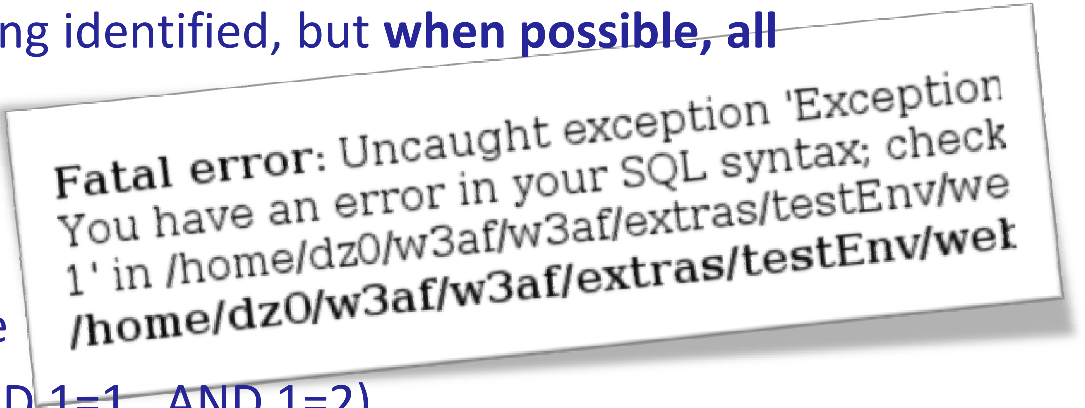
Plugins | Discovery

- Other discovery plugins try to fingerprint remote httpd, verify if the remote site has an HTTP load balancer installed, etc.
  - halberd
  - hmap
  - afd
  - fingerprint_WAF
- I need some **refactoring…**
  - Crawlers
  - Infrastructure

Plugins | Audit

- They take the output of discovery plugins and find vulnerabilities like:
  - [blind] SQL injection
  - XSS
  - Buffer overflows
  - Response splitting.

- Vulnerabilities are identified using **different methods**, that vary on the type of vulnerability being identified, but **when possible, all methods are used**:
  - Error based
  - Time delay
  - Creating a new resource
  - Different responses (AND 1=1 , AND 1=2)

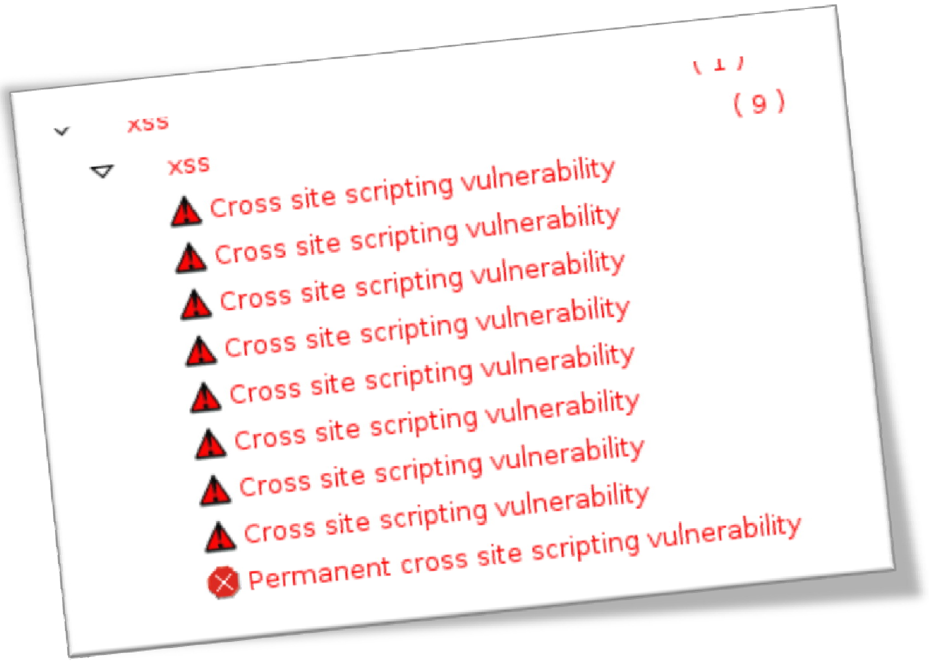


```
Fatal error: Uncaught exception 'Exception
You have an error in your SQL syntax; check
1' in /home/dz0/w3af/w3af/extras/testEnv/we
/home/dz0/w3af/w3af/extras/testEnv/wet
```

Plugins | Audit

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

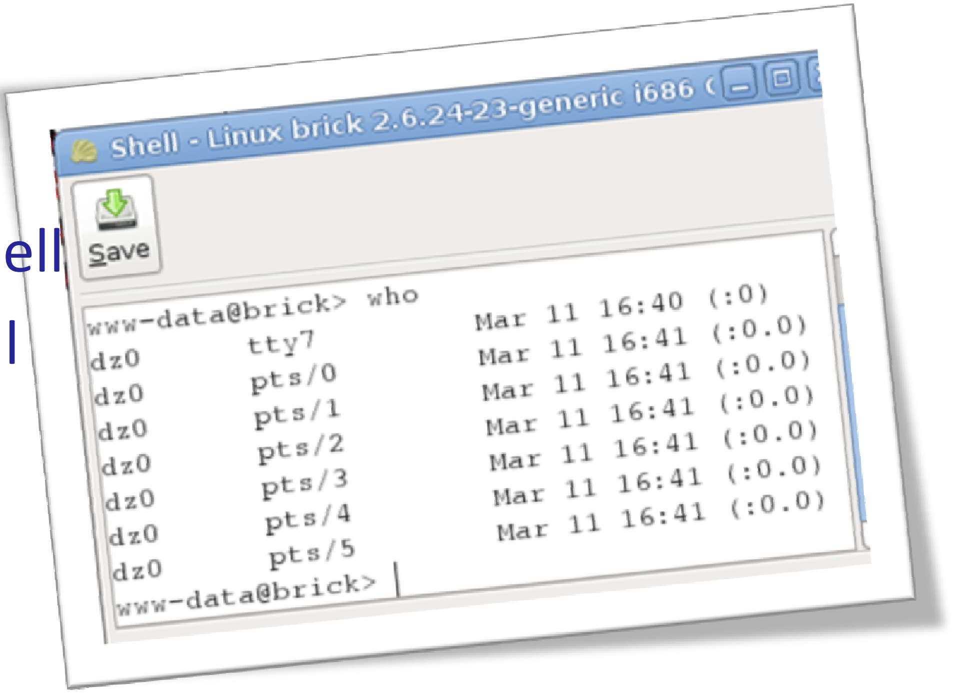
- These vuln objects are then used as the input for attack plugins, that will exploit the vulnerabilities.



```
xss                                                   ( 9 )
  xss
    Cross site scripting vulnerability
    Cross site scripting vulnerability
    Cross site scripting vulnerability
    Cross site scripting vulnerability
    Cross site scripting vulnerability
    Cross site scripting vulnerability
    Cross site scripting vulnerability
    Cross site scripting vulnerability
    Permanent cross site scripting vulnerability
```

Plugins | Attack

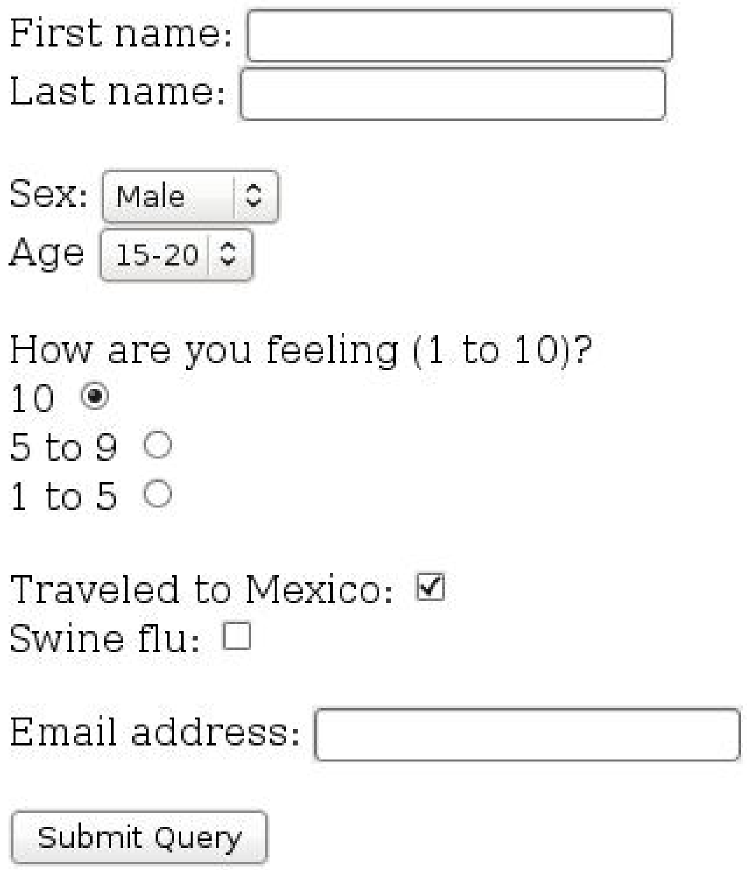
- These plugins read the **vuln objects from the KB** and try to exploit them. Examples of attack plugins are:
  - sql_webshell
  - davShell
  - sqlmap
  - xssBeef
  - remote file include shell
  - OS Commanding shell



```
www-data@brick> who
dz0          tty7          Mar 11 16:40 (:0)
dz0          pts/0         Mar 11 16:41 (:0.0)
dz0          pts/1         Mar 11 16:41 (:0.0)
dz0          pts/2         Mar 11 16:41 (:0.0)
dz0          pts/3         Mar 11 16:41 (:0.0)
dz0          pts/4         Mar 11 16:41 (:0.0)
dz0          pts/5         Mar 11 16:41 (:0.0)
www-data@brick>
```

```
./w3af_gui
```

Fuzzing HTTP is harder than you think…



First name:

Last name:

Sex: Male

Age 15-20

How are you feeling (1 to 10)?

10 ●

5 to 9 ○

1 to 5 ○

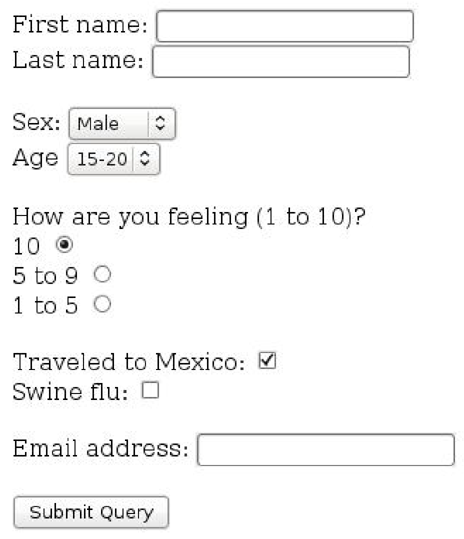
Traveled to Mexico: ☑

Swine flu: ☐

Email address:

```
if ( strcmp('',$firstname) == 0 ||
     strcmp('', $lastname) == 0 ||
     !isValidEmail($email) ){
    echo 'Please fill the form
          properly.';
}
else
{
    // Choose the lovely girl
    if ($sex == 'female' &&
        $age == '21-25') {
        // XSS here
        echo $firstname . ' you’ve been
        randomly selected for manual
        inspection.';
    }
    else
    {echo 'Please go on.';}
}
```

Acunetix


First name:

Last name:

Sex: Male

Age 15-20

How are you feeling (1 to 10)?

[✓ 10] 5 to 9 1 to 5

Traveled to Mexico: ✓

Swine flu: ☐

Email address:

- Send the payload in one parameter, fill the rest with some "meaningful" data:
  - 111-222-1933email@address.tst

- Three **modes for fuzzing**, which define how **select/radio inputs are combined**: Quick, Heuristic, Extensive.

Mode	Number of requests	Found XSS
Quick	23	No
Heuristic	155	No (*)
Extensive	879	Yes

(*) Has a long story behind, bug:

/abc.php?lastname=${varvalues1}&sex=${varvalues2}&some_radio=${varvalues3}&some_check=${varvalues4}

AppScan

- Send the payload in one parameter, **leave the rest of the parameters empty.** In my simple test, this is a killer, because the fuzzer never gets past the first if.

```
if ( strcmp('',$firstname) == 0 ||
     strcmp('', $lastname) == 0 ||
     !isValidEmail($email) ){
    echo 'Please fill the form
          properly.';
}
```

- Only one way of fuzzing. It takes the last value of each select and radio input: age=31-45 & sex=female & some_radio=radio_1. **No permutations are made.** So even if they would somehow go past the first if, they would never get past the second one (girls && 21-25).

Mode	Number of requests	Found XSS
Default	16	No

N-Stalker

- Send the payload in one parameter, **leave the rest of the parameters empty.** In my simple test, this is a killer, because the fuzzer never gets past the first if.

```
if ( strcmp('',$firstname) == 0 ||
     strcmp('', $lastname) == 0 ||
     !isValidEmail($email) ){
    echo 'Please fill the form
          properly.';
}
```

- Only one way of fuzzing. It takes the last value of each select and radio input: age=31-45 & sex=female & some_radio=radio_1. **No permutations are made.** So even if they would somehow go past the first if, they would never get past the second one (girls && 21-25).

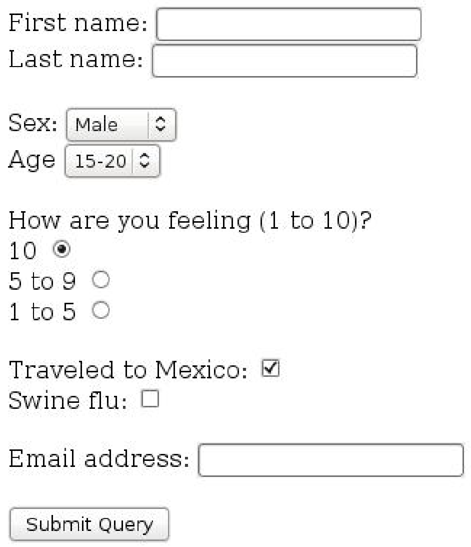
Mode	Number of requests	Found XSS
Default	301	No

w3af

- Send the payload in one parameter, **fill the rest with some meaningful data** using the parameter name to guess a "correct value":
  - If parameter name is “**name**” then fill it with [a-zA-Z] * 6
  - If parameter name is “**pin**” then fill it with [0-9] * 6
  - If parameter name is “**month**” then fill it with [0-9] * 1
  - If parameter name is not in our **db** then fill it with [0-9] * 6
- Five modes of fuzzing: T, B, T-B , **T-M-B(default)**, All.

Mode	Number of requests	Found XSS
T	18	No
B	18	No
TB	234	No
TMB	514	No
All	695	Yes

Parsing HTML is harder than you think...



First name:
Last name:

Sex: Male
Age 15-20

How are you feeling (1 to 10)?
[✓ 10]
5 to 9
1 to 5

Traveled to Mexico: ✓
Swine flu: ☐

Email address:

```
<form action="repeated.php">
    First name: <input type="text" name="p"><br />
    Last name: <input type="text" name="p"><br />
    <br />

    Sex:
    <select name="p">
        <option value="male">Male</option>
        <option value="female">Female</option>
    </select><br />
    Age
    <select name="p">
        <option value="15-20">15-20</option>
        <option value="21-25">21-25</option>
        <option value="26-30">26-30</option>
        <option value="31-45">31-45</option>
    </select><br /><br />
```

Acunetix

- Doesn't know how to handle this specific case:

```
repeated.php?p=%2527
repeated.php?p=%00‘
repeated.php?p=acunetix'“
repeated.php?p=radio_1
repeated.php?p=radio_1
repeated.php?p=radio_1
repeated.php?p=male
repeated.php?p=male
repeated.php?p=male
repeated.php?p=male
repeated.php?p=male
...
```

AppScan

- Doesn't know how to handle this specific case:

```
repeated.php?p=1234WFXSSProbe
repeated.php?p=1234'"WFXSSProbe)/>
repeated.php?p=WF'SQL"Probe;A--B
repeated.php?p=1234'%20exec%20master..xp_cmdshell%20'vol'--
repeated.php?p=1234';
repeated.php?p=1234'%20having%201=1--
repeated.php?p=12341%20having%201=1--
repeated.php?p=1234)%20having%201=1--
repeated.php?p=1234\'%20having%201=1--
repeated.php?p=1234%a5'%20having%201=1--
repeated.php?p=1234%uFF07
repeated.php?p=1234%20and%207659=7659
repeated.php?p=1234'%20and%20'foobar'='foobar
repeated.php?p=1234/**/and/**/7659=7659
...
```

N-Stalker

- Doesn't know how to handle this specific case:

```
repeated.php
repeated.php?p=
repeated.php
repeated.php?nstalkerXSSTest
repeated.php
repeated.php
repeated.php
repeated.php?p=nstalkerXSSTest
...
```

w3af

- Knows about repeated parameters, but at some point it seems to fail:

```
repeated.php?p=SV737&p=sqLVtQk&p=qJhIpNe&p=tgYxMhW&p=nVITovR&
              p=ybzGbTX&p=JLzKfXp&p=xcoUItG&p=zuUeGiF&p=yumQjki
repeated.php?p=gywBndD&p=dsVxY&p=zkvSgnm&p=UuMDrwb&p=tQZjlTz&
              p=KPDwdVo&p=ZHCzmrr&p=lmtTuib&p=aCZJYYf&p=PFrsFre
repeated.php?p=LUWRJRa&p=tTnIzGx&p=L6Aqr&p=kunPewv&p=ROSgPuT&
              p=wGiVgUg&p=osOHNmj&p=YkFrdVy&p=ZoBJKNh&p=ZpWscyc
repeated.php?p=oUdArtv&p=FGPEXxh&p=JecNJdc&p=2qc55&p=EHEuTMz&
              p=CubKGTc&p=FFWzFtS&p=zUgcxDO&p=vGwmygS&p=cKuywsT
repeated.php?p=EEHYRGu&p=eQmfNti&p=tWBjtIl&p=HQwFvSR&p=M616s&
              p=QCKdKHK&p=cbPKenI&p=MhIoSLs&p=qqoumgH&p=lTQHFfX
repeated.php?p=uxpFH
repeated.php?p=jqvPR
repeated.php?p=BQEd1
repeated.php?p=16orb
...
```

Conclusions

- **Do NOT blindly trust** web application scanners.
  - Perform your own tests before buying a scanner
  - Or if it's Open Source, read the code.

- Fast scanners don't cover all the **logical paths** in your web application. **They WILL miss vulnerabilities.**

- Slow scanners may miss vulnerabilities if they aren't smart enough when filling "**the other parameters**".

- w3af is getting closer (in quality) to the commercial scanners, **we need more users, we need more contributors, I want…**

I WANT YOU!

¿Questions? Lets have a beer after this talk :)