WFUZZ
for Penetration Testers

Christian Martorella & Xavier Mendez

SOURCE Conference 2011

Barcelona

Who we are?

- Security Consultants at Verizon Business Threat and Vulnerability Team EMEA
- Members of Edge-security.com

What is this presentation about?

WFUZZ: a Web Application brute forcer / fuzzer

And how this tool can be used in your Penetration test engagements

What is WFUZZ?

It´s a web application brute forcer, that allows you to perform complex brute force attacks in different web application parts as: parameters, authentication, forms, directories/files, headers files, etc.

It has complete set of features, payloads and encodings.

WFUZZ

- Started a few years ago and have been improving until now (and hopefully will continue improving)
- Has been presented at Blackhat Arsenal US 2011
- It's included in the TOP 125 Security tools by Insecure.org

Key features

- Multiple injection points
- Advance Payload management (Iterators)
- Multithreading
- Encodings
- Result filtering
- Proxy and SOCKS support (multiple proxies)

New features

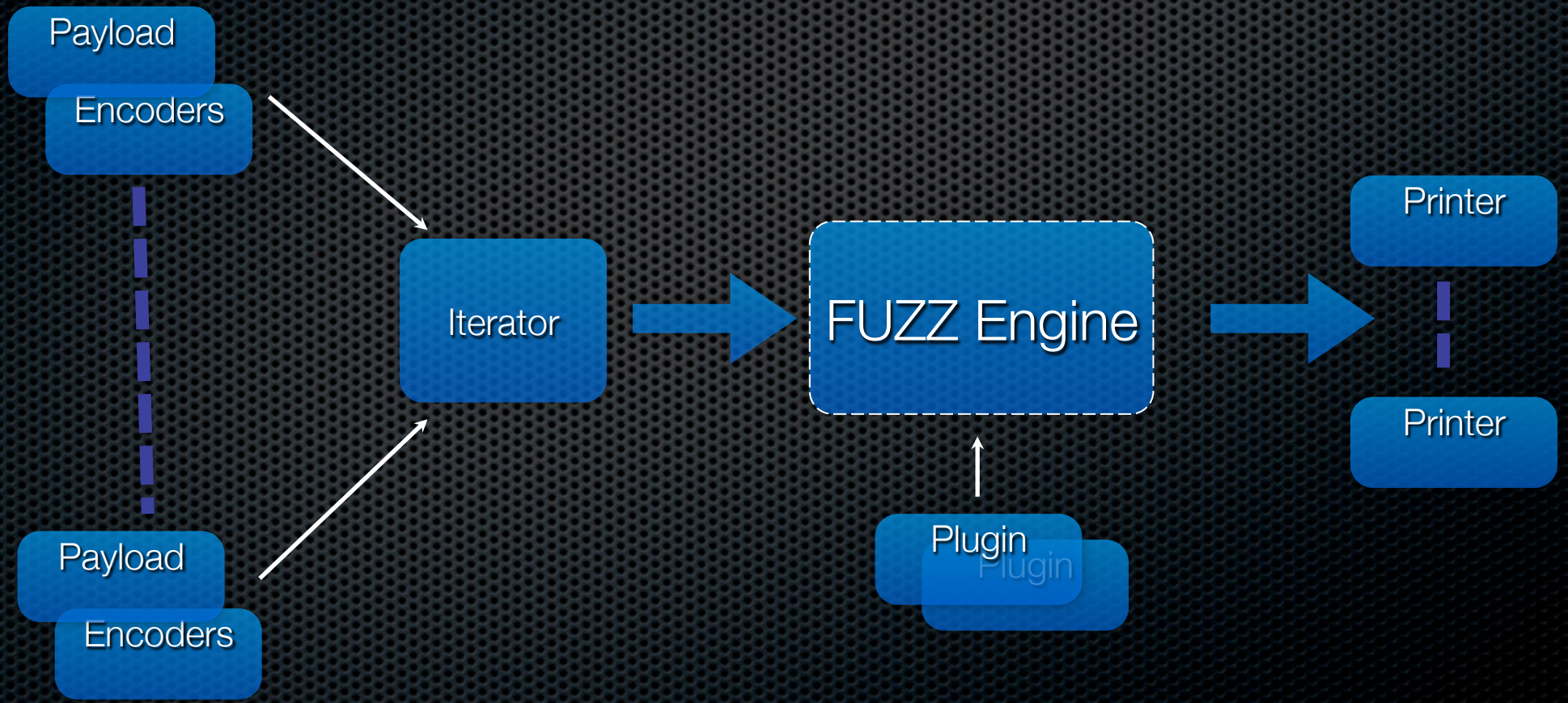
- Added HEAD method scanning
- Fuzzing in HTTP methods
- Added follow HTTP redirects option

New features

- Plugin framework, allowing to execute actions on response contents, or when a condition are met
- Multiple filtering (show, hide, filter expression, regex)
- Attack pause/resume
- Delay between requests

Extensibility

Payloads, encoders , iterators, plugins and printers.



Payloads

A payload is what generates the list of requests to send in the session.

- file: reads from a file
- stdin: reads from the stdin (cwel)
- list: define a list of objects (1-2-3-4-5)
- hexrand: define a hexa random list (
- range: define a numeric range (1-30)
- names: creates potential user names combinations (john.doe,j.doe,etc)
- hexrange: define a random hexa range
- overflow:

Encoders

Converts information from one format to another

- urlencode
- double_urlencode
- first_nibble_hexa
- html_encoder
- uri_hexadecimal
- base64
- mssql_char
- uri_double_hexadecimal
- mysql_char
- utf8
- second_nibble_hexa
- binary_ascii
- double_nibble_hexa
- md5
- none
- sha1
- utf8_binary
- html_encoder_hexa
- uri_unicode
- oracle_char
- random_uppercase
- html_encoder_decimal

Base64 encoder

- Encoders.py

```
class encoder_base64 (IEncoder):
    text="base64"

    def encode(self,string):
        return base64.standard_b64encode(string)

    def decode(self,string):
        res=base64.decodestring(string)
        return res
```

Iterators

An Iterator allows to process every element of a container while isolating from the internal structure of the container.

An Iterator could be created from combining iterables:

Putting it all together

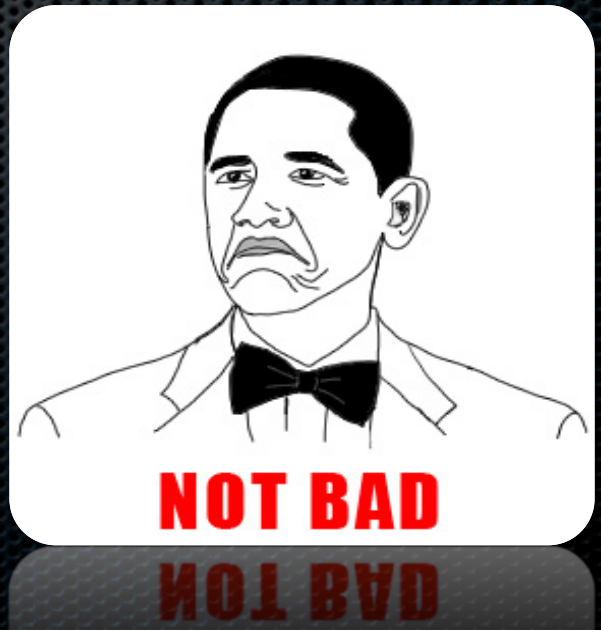
wfuzz.py -z range,0-2,md5 –z list,a-b-c -m product –o magictree http://www.myweb.com/FUZZ

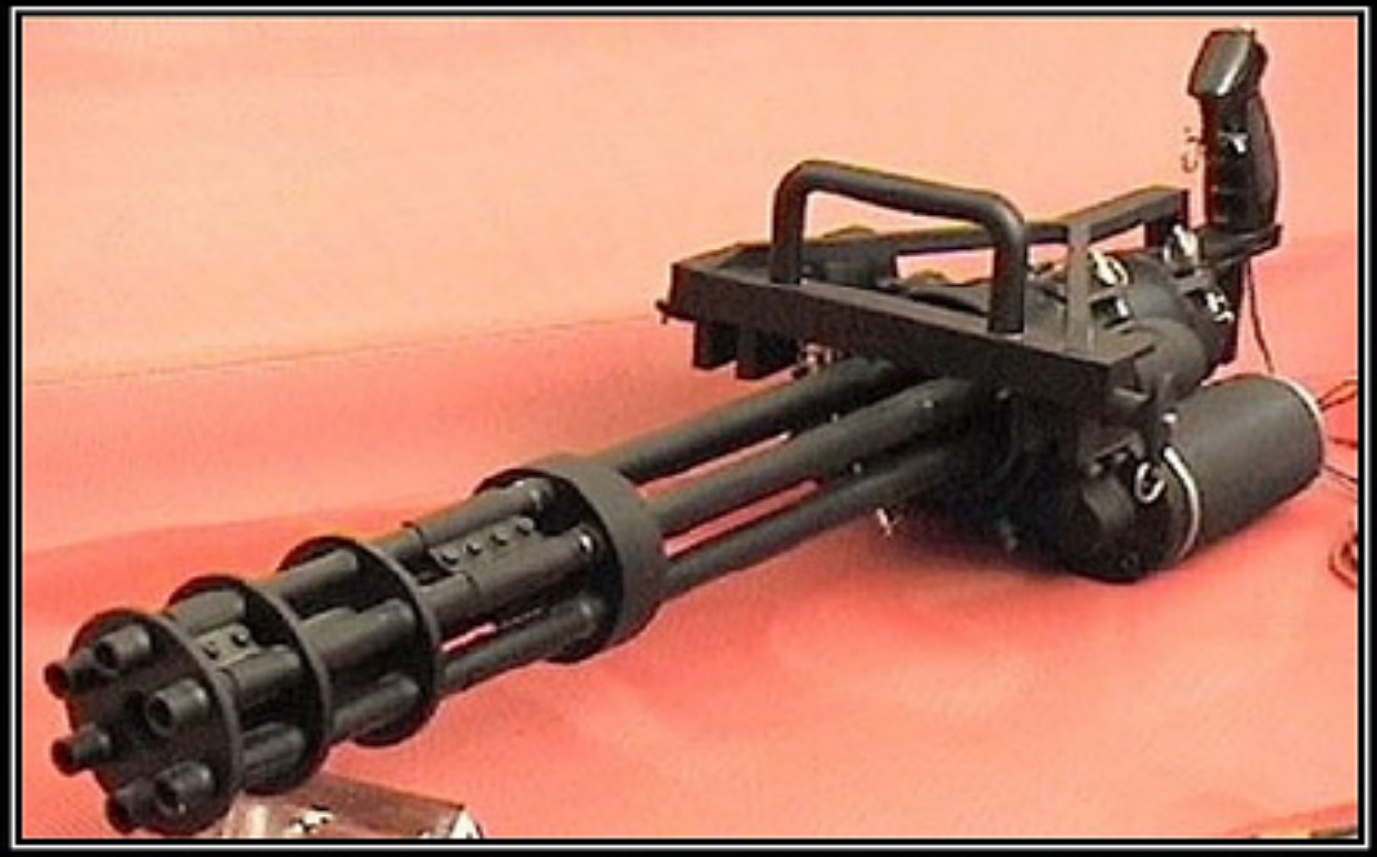
- Payload: range
- Encoder: md5
- Printer: magictree
- Iterator: product

Need for speed

60% faster

Up to 900 request /second





BRUTE FORCE

If it doesn't work, you're just not using enough.

A brute force attack is a method to determine an unknown value by using an automated process to try a large number of possible values.

What can be bruteforced?

- Predictable credentials (HTML Forms and HTTP)
- Predictable sessions identifier (session id's)
- Predictable resource location (directories and files)
- Parameters names, values
- Cookies
- Web Services methods

Where?

- Headers
- Forms (POST)
- URL (GET)
- Authentication


KEEP
CALM
AND
USE THE
FORCE

Basic usage

wfuzz.py -c –z file,wordlist/general/common.txt http://www.target.com/FUZZ

Basic usage - verbose

wfuzz.py -c –z file,wordlist/general/common.txt -v http://www.target.com/FUZZ

Basic filtering

wfuzz.py -c -z file,wordlist/general/test.txt --hc 404 http://target.com/FUZZ

Basic filtering

Don’t underestimate a 404. Use the Baseline!

```
zim | javi@reddwarf: ~
javi@reddwarf:~$ wfuzz.py  -z list,trn -v --hw=BBB https:/[redacted]/FUZZ{notthere}

********************************************************
* Wfuzz  2.0 - The Web Bruteforcer                     *
* Blackhat Arsenal Release                             *
********************************************************

Target: https:/[redacted]/FUZZ{notthere}
Payload type: list,trn

Total requests: 1
=====================================================================
ID      Response   Lines      Word         Chars                 Server      Request
=====================================================================

00001:  C=404     70 L       174 W        2462 Ch     Apache/2.2.20 (Un     " - notthere"
00002:  C=404      9 L        32 W         309 Ch     Oracle-Applicatio     " - trn"
```

Advance filtering

Built-in Expression filter parser

wfuzz.py –filter “c=200 and (w>300 and w<600)”

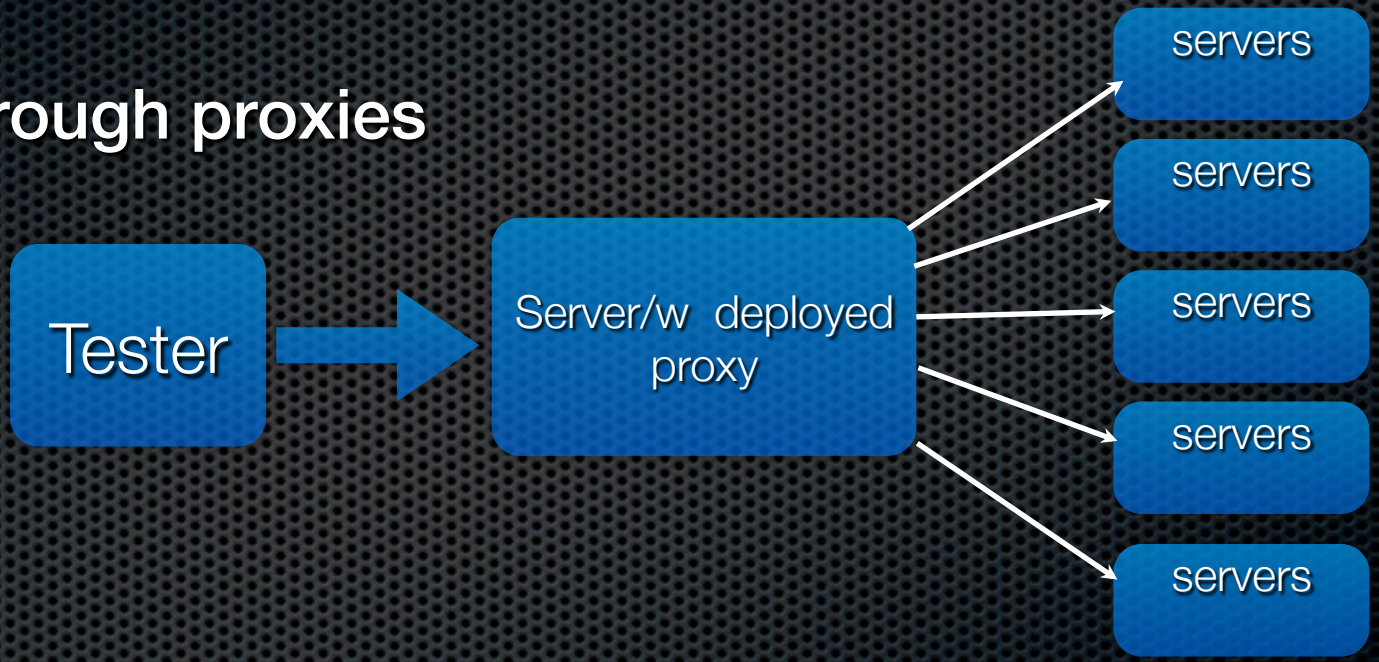
Range sweeping

wfuzz.py -c -z file,hosts.txt -z list,admin-phpMyAdmin-test FUZZ/FUZ2Z

wfuzz.py -c -z range,1-254 -z list,admin- phpMyAdmin-test http://192.168.0.FUZZ/FUZ2Z

Scanning internal networks

Scanning through proxies



wfuzz -x serverip:53 -c -z range -r 1-254 --hc XXX -t 5 http://10.10.1.FUZZ

-x set proxy
--hc is used to hide the XXX error code from the results, as machines w/o webserver will fail the request.

Using multiple encodings per payload

wfuzz.py – z list,..,double_nibble_hexa@second_nibble_hexa

@uri_double http://targetjboss.com/FUZZ/jmx-console

Fuzzing using 3 payloads

wfuzz.py -z list,dir1-dir2 -z file,wordlist/general/common.txt -z list,jsp-php-asp http://target.com/FUZZ/FUZ2Z.FUZ3Z

Username payload

wfuzz.py -c -z username,John-doe -z list,123456- admin-password-love -b "user=FUZZ&pass=FUZ2Z" http://localhost:8888/test/login.php

```
ID           Response   Lines      Word         Chars       Request
=====================================================================

00000:  C=404      7 L       24 W       203 Ch      " - johnd"
00001:  C=404      7 L       24 W       203 Ch      " - j.doe"
00002:  C=404      7 L       24 W       200 Ch      " - jd"
00003:  C=404      7 L       24 W       205 Ch      " - johndoe"
00004:  C=404      7 L       24 W       206 Ch      " - john.doe"
00005:  C=404      7 L       24 W       203 Ch      " - j.doe"
00006:  C=404      7 L       24 W       202 Ch      " - jdoe"
00007:  C=404      7 L       24 W       201 Ch      " - doe"
00008:  C=404      7 L       24 W       204 Ch      " - john.d"
```

User-Agent brute forcing

```
web | javi@reddwarf: ~/herramientas/wfuzz
javi@reddwarf:~/herramientas/wfuzz$ python wfuzz.py -H "User-agent: FUZZ{Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; Trident/4.0; SLCC2; .NET CLR 2.
0; .NET4.0C; AskTbFXTV5/5.11.3.15590; .NET4.0E)}" -z file,/tmp/ua.txt -v  https://[redacted].com
********************************************************
* Wfuzz  2.0 - The Web Bruteforcer                     *
********************************************************

Target: https://[redacted]
Total requests: 3

=====================================================================================================================
ID      C.Time   Response   Lines      Word         Chars                                          Redirect   Request
=====================================================================================================================

00000:  0.625s   C=302      2 L        10 W         175 Ch   https://[redacted]/defaul   " - Mozilla/4.0 (compatible; MSIE 8.0; Wi"
00001:  0.603s   C=302      2 L        10 W         161 Ch         https://[redacted]/enroll/   " - Mozilla/5.0 (iPad; U; CPU OS 3_2 like"
00002:  0.563s   C=302      2 L        10 W         161 Ch         https://[redacted]/enroll/   " - BlackBerry7520/4.0.0 Profile/MIDP-2.0"
00003:  0.572s   C=302      2 L        10 W         140 Ch                                /agentdownload.html   " - Generic Mobile Phone (compatible; Goo"
```

Password cracking

- Vertical scanning (different password for each user)
- Horizontal scanning (different usernames for common passwords)
- Diagonal scanning (different username/password each round)
- Three dimension (Horizontal, Vertical or Diagonal + Distributing source IP)
- Four dimensions (Horizontal, Vertical or Diagonal + Time Delay + Distributing Source IP)

Password cracking

Diagonal

- admin/test
- guest/guest
- user/1234x

Horizontal

admin/test

guest/test

user/test

Password cracking Horizontal

wfuzz –z list,pass1-pass –z list,us1-us2 http://target.com/user=FUZ2Z &pass=FUZZ

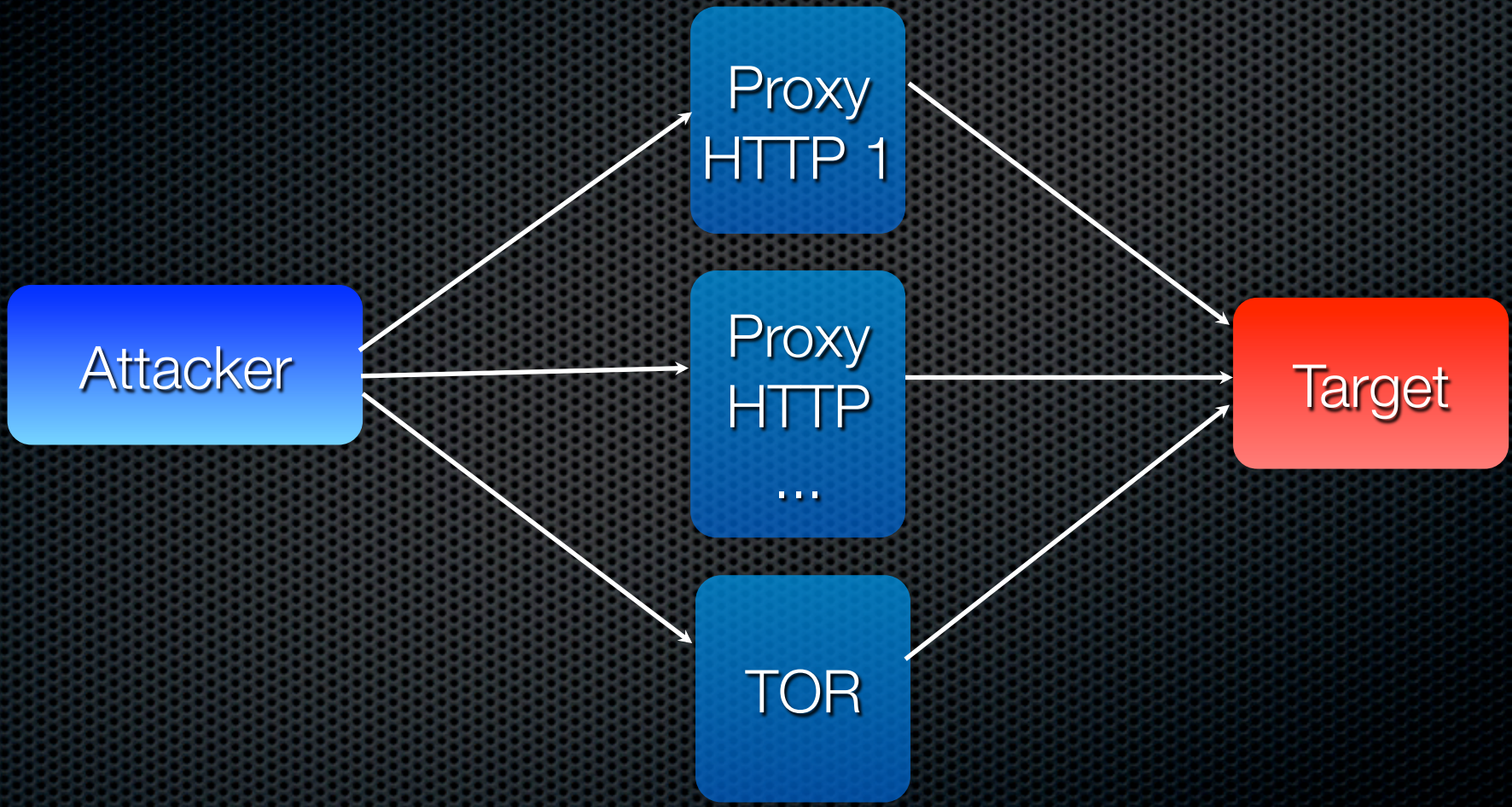
Password cracking Three dimensional

wfuzz –z list,pass1-pass –z list,us1-us2 –s 1 http://target.com/user=FUZ2Z &pass=FUZZ

Password cracking Four dimensional

Wfuzz –z list,pass1-pass –z list,us1-us2 –s 1 –p ip:8080-ip2:8080-ip3:8088 http://target.com/user=FUZ2Z&pass=FUZZ

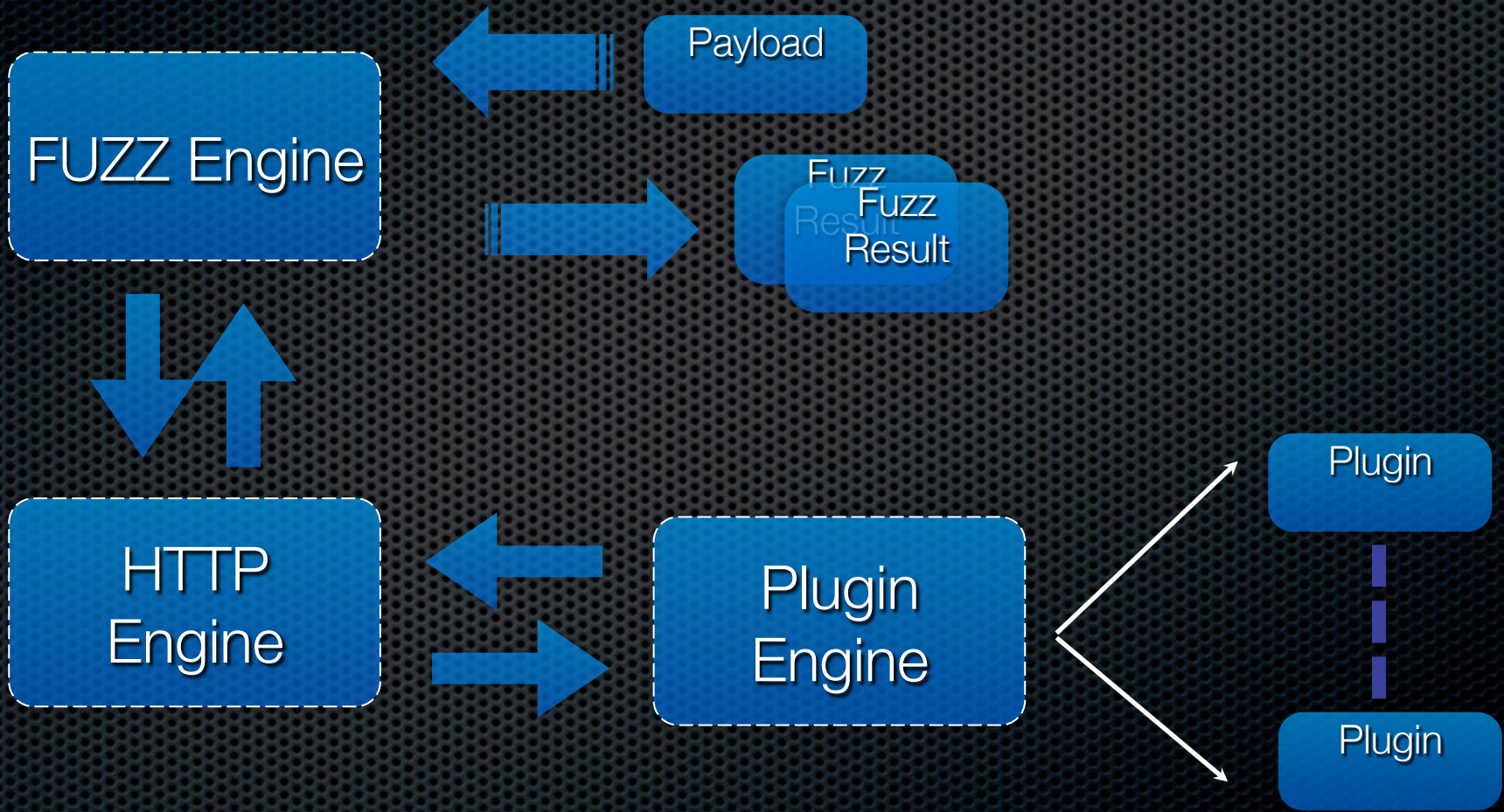
Load balancing



Permutation payload

wfuzz.py -c -z permutation,abcdefghijk-2 -z permutation, 1234567890-2 --hc 404 --hl BBB http://localhost:8888/test/ parameter.php? action=FUZZ{a}FUZ2Z{a}

Scripting engine



“Parsing” HTTP Response

```
 python wfuzz.py -z list,test -H "Accept: foo/bar" --script --follow http://localhost/FUZZ
********************************************************
* Wfuzz  2.0 - The Web Bruteforcer                     *
********************************************************

Target: http://localhost/FUZZ
Total requests: 1


=====================================================================
ID      Response   Lines      Word         Chars          Request
=====================================================================

00000:  C=406     14 L        51 W          490 Ch        " - test"
00001:  C=200      4 L        25 W          177 Ch        "/"
00002:  C=200     37 L        93 W         1210 Ch        "/test.html"
00003:  C=200      4 L         6 W           78 Ch        "/test.js"
00004:  C=200     14 L        57 W          889 Ch        "/uno"
00005:  C=200     14 L        57 W          903 Ch        "/uno/dos/"
00006:  C=200   1002 L      4788 W        72044 Ch        "/icons/"
00007:  C=200     14 L        57 W          889 Ch        "/uno/"
00008:  C=200     14 L        57 W          921 Ch        "/uno/dos/tres/"
00009:  C=200    166 L       644 W         5108 Ch        "/icons/README"
00010:  C=200    815 L      3019 W        36057 Ch        "/icons/README.html"
00011:  C=200     77 L       685 W        13514 Ch        "/icons/small/"
00012:  C=200     14 L        57 W          938 Ch        "/uno/dos/tres/cuatro/"
00013:  C=200     14 L        57 W          955 Ch        "/uno/dos/tres/cuatro/cinco/"
00014:  C=200     14 L        57 W          973 Ch        "/uno/dos/tres/cuatro/cinco/seis/"
00015:  C=200     14 L        57 W          988 Ch        "/uno/dos/tres/cuatro/cinco/seis/siete/"
00016:  C=200     14 L        57 W         1008 Ch        "/uno/dos/tres/cuatro/cinco/seis/siete/ocho/"
00017:  C=200     14 L        57 W         1023 Ch        "/uno/dos/tres/cuatro/cinco/seis/siete/ocho/nuevue/"
00018:  C=200     14 L        57 W         1040 Ch        "/uno/dos/tres/cuatro/cinco/seis/siete/ocho/nuevue/diez/"
00019:  C=200     14 L        57 W         1055 Ch        "/uno/dos/tres/cuatro/cinco/seis/siete/ocho/nuevue/diez/once/"
00020:  C=200     14 L        57 W         1072 Ch        "/uno/dos/tres/cuatro/cinco/seis/siete/ocho/nuevue/diez/once/doce/"
00021:  C=200     13 L        46 W          896 Ch        "/uno/dos/tres/cuatro/cinco/seis/siete/ocho/nuevue/diez/once/doce/trece/
```

“Grep” HTTP responses

```
class parser_extractor(IParser):
    text="extractor"

    def __init__(self):
        dir_indexing_regexes = []

        self.enabled = True
        self.regex = re.compile('name="UserName" type="text" value="(.*?)"',re.MULTILINE|re.DOTALL)

    def process(self, request, control_queue, results_queue):
        l = []
        content = request.response.getContent()

        for i in self.regex.findall(content):
            plres = PluginResult()
            plres.source = "extractor"
            plres.issue = "Pattern match: %s" % i
            plres.details = i
            results_queue.put(plres)

        control_queue.get()
        control_queue.task_done()
```

“Grep” HTTP responses

```
python wfuzz.py -z range,0-999 -H 'Cookie: ASPSESSIONIDASCTTRAB=LJKGI..' --script --filter="c=200 and w>268" https://[redacted]
********************************************************
* Wfuzz  2.0 - The Web Bruteforcer                     *
********************************************************

Target: https://[redacted]/AdminAccounts/AdminUserEdit/FUZZ
Total requests: 1000


=====================================================================
ID      Response   Lines      Word         Chars          Request
=====================================================================

00000:  C=200    516 L      1732 W        33687 Ch        " - 4"
  |_ Pattern match: sysadmin
00052:  C=200    516 L      1722 W        33574 Ch        " - 51"
  |_ Pattern match: [redacted]
00054:  C=200    555 L      1819 W        35347 Ch        " - 59"
  |_ Pattern match: verizonadmin1
00055:  C=200    555 L      1819 W        35348 Ch        " - 60"
  |_ Pattern match: verizonadmin2
00058:  C=200    516 L      1723 W        33584 Ch        " - 61"
  |_ Pattern match: awtest
00065:  C=200    516 L      1723 W        33576 Ch        " - 62"
  |_ Pattern match: test
```

Evidence collection

Imagine an internal assessment 100s or 1000s of webapps and very little time?

```
class parser_scroter(IParser):
    text="Screen shotter"

    def __init__(self):
        self.enabled = False
    def process(self, request, control_queue, results_queue):
        import subprocess
        l = []
        content = request.response.getContent()
        code = request.response.code
        url = request.completeUrl
        if code == 200:
            plres = PluginResult()
            plres.source = "Screen shotter"
            plres.issue = "Scrot"
            plres.details = "Scrot"
            subprocess.call(['python','scrotosx.py','--dir','output',url])
        control_queue.get()
        control_queue.task_done()
```

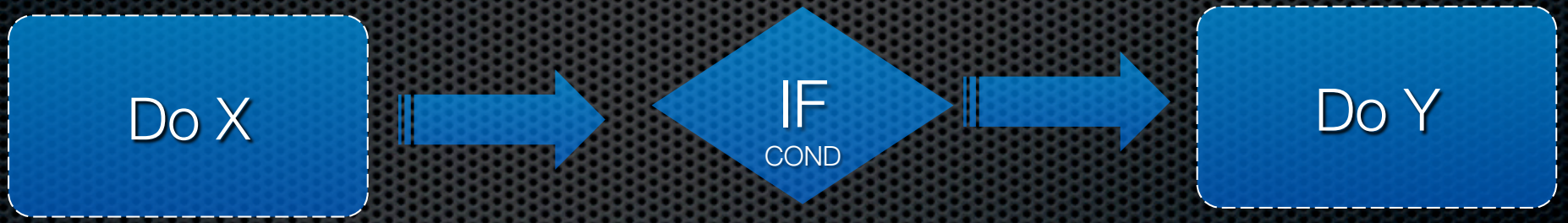
Under development

```
class parser_stopper(IParser):
    text="Show stopper"

    def __init__(self):
        self.enabled = False
    def process(self, request, control_queue, results_queue):
        l = []
        content = request.response.getAll()
        charlen = request.response.charlen
        if charlen == 423:
            plres = PluginResult()
            plres.source = "Show stopper"
            plres.issue = "Show Stopper - Condition met"
            plres.details = "Show Stopper"
            self.mail="Condition met, password cracked, password is " + content
        control_queue.get()
        control_queue.task_done()
```

Under development

- Multi step or sequences



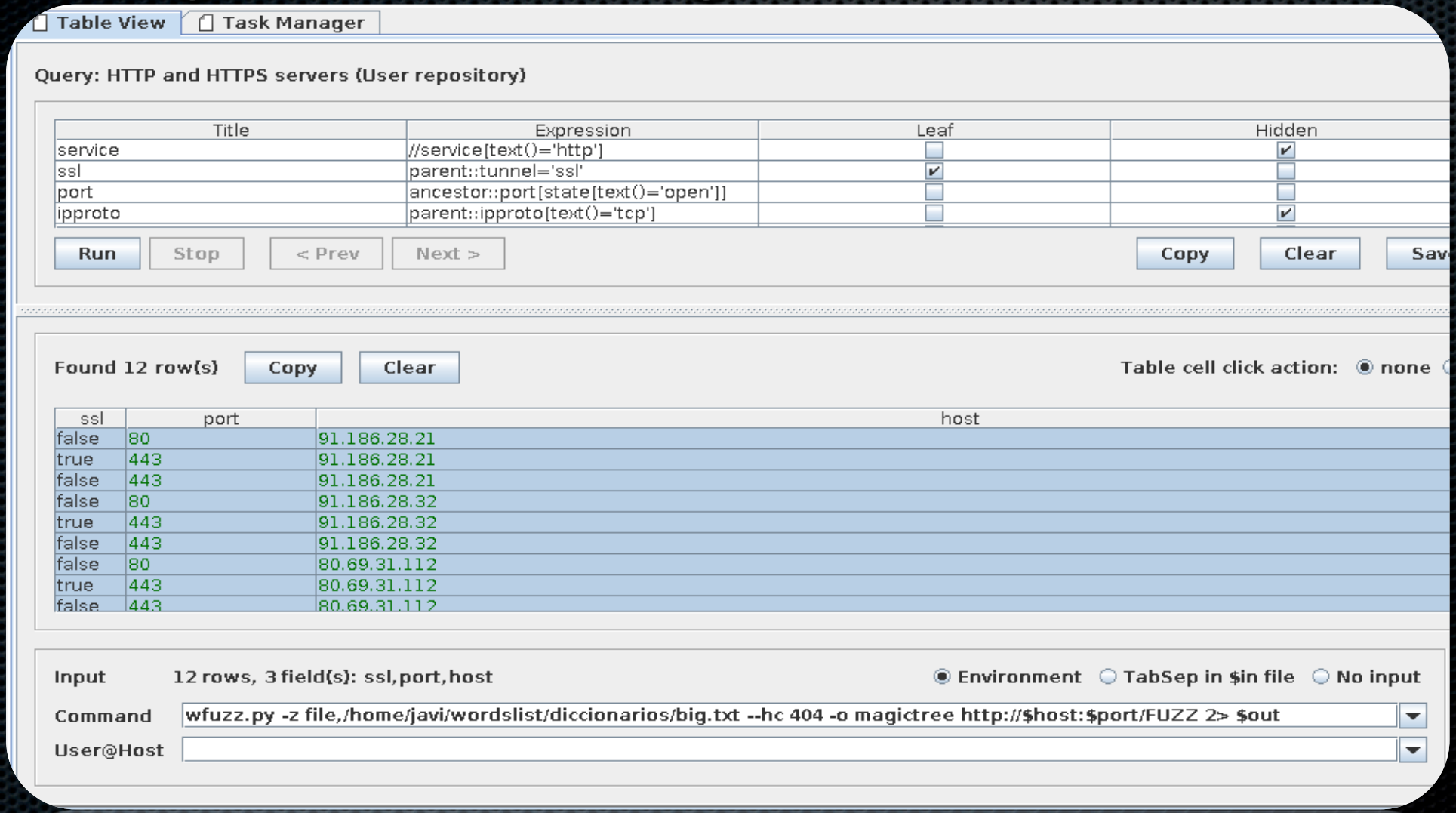
Using external tools

```
$ ../crunch/crunch 1 2 | python wfuzz.py -z stdin, --hc 404 http://www.edge-security.com/FUZZ
********************************************************
* Wfuzz  2.0 - The Web Bruteforcer                     *
********************************************************

Target: http://www.edge-security.com/FUZZ
Total requests: -1


=====================================================================
ID      Response   Lines      Word         Chars          Request
=====================================================================
00238:  C=200     22 L       62 W          415 Ch        " - ie"
00425:  C=404      7 L       24 W          200 Ch        " - pj"
```

Magic tree integration



Latest news and versions

- http://code.google.com/p/wfuzz
- http://edge-security.blogspot.com

References

- http://www.owasp.org/index.php/Testing_for_Brute_Force_(OWASP-AT-004)
- http://projects.webappsec.org/Predictable-Resource-Locatio
- http://projects.webappsec.org/Credential-and-Session-Prediction
- http://projects.webappsec.org/Brute-Force
- http://www.technicalinfo.net/papers/StoppingAutomatedAttackTools.html
- http://gawker.com/5559346
- http://tacticalwebappsec.blogspot.com/2009/09/distributed-brute-force-attacks-against.html
- Detecting Malice, Rsnake