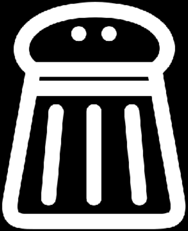
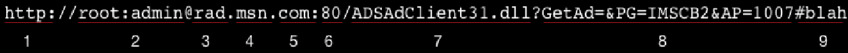
Faup Workshop



Sebastien Tricaud

July 2, 2018

What is a URL?



http://root:admin@rad.msn.com:80/ADSAdClient31.dll?GetAd=&PG=IMSCB2&AP=1007#blah

What is a URL?

```
http://root:admin@rad.msn.com:80/ADSAdClient31.dll?GetAd=&PG=IMSCB2&AP=1007#blah
  1         2      3   4   5  6          7                    8               9
```

Scheme	1
Credential	2
Subdomain	3
Domain	4+5
Domain without TLD	4
Host	3+4+5
TLD	5
Port	6
Resource Path	7
Query String	8
Fragment	9

Ask the Audience

How do you parse a URL?

- Regex
- Perl Script
- Python library
- ...

EXtracting a URL from a TLD

Regex:

```
^(((ht|f)tp(s?))\://)?(www.|[a-zA-Z].)[a-zA-Z0-9\-\.]+\.(com|edu|gov|mil|net|
    ↪ org|biz|info|name|museum|us|ca|uk)(\:[0-9]+)*(/($|[a-zA-Z0
    ↪ -9\.\,\;\?\'\\\+&amp;%\$#\=~_ \-]+))*$
```

EXtracting a URL from a TLD

Regex:

```
^(((ht|f)tp(s?))\://)?(www.|[a-zA-Z].)[a-zA-Z0-9\-\.]+\.(com|edu|gov|mil|net|
     ↪ org|biz|info|name|museum|us|ca|uk)(\:[0-9]+)*(/($|[a-zA-Z0
     ↪ -9\.\,\;\?\'\\\+&amp;%\$#\=~_ \-]+))*$
```

Python urllib

```
>>> from urllib.parse import urlparse
>>> url = urlparse('http://192.168.0.1/index.php3?ref=http://slashdot.org#blah')
ParseResult(scheme='http', netloc='192.168.0.1', path='/index.php3', params='',
     ↪ query='ref=http://slashdot.org', fragment='blah')
```

EXtracting a URL from a TLD

Regex:

```
^(((ht|f)tp(s?))\://)?(www.|[a-zA-Z].)[a-zA-Z0-9\-\.]+\.(com|edu|gov|mil|net|org|biz|info|name|museum|us|ca|uk)(\:[0-9]+)*(/($|[a-zA-Z0-9\.\,\;\?\'\\\+&amp;%\$#\=~_\-]+))*$
```

Python urllib

```
>>> from urllib.parse import urlparse
>>> url = urlparse('http://192.168.0.1/index.php3?ref=http://slashdot.org#blah')
ParseResult(scheme='http', netloc='192.168.0.1', path='/index.php3', params='', query='ref=http://slashdot.org', fragment='blah')
```

```
>>> from urllib.parse import urlparse
>>> url = urlparse('192.168.0.1/index.php3?ref=http://slashdot.org#blah')
ParseResult(scheme='', netloc='', path='192.168.0.1/index.php3', params='', query='ref=http://slashdot.org', fragment='blah')
```

EXtracting a URL from a TLD

QUrl:

```
>>> from PyQt4 import QtCore
>>> url = QtCore.QUrl("192.168.0.1/index.php3?ref=http://slashdot.org#blah")
>>> print(url.host())

>>>
```

EXtracting a URL from a TLD

QUrl:

```
>>> from PyQt4 import QtCore
>>> url = QtCore.QUrl("192.168.0.1/index.php3?ref=http://slashdot.org#blah")
>>> print(url.host())

>>>
```

. . .

Punycode

Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA)

`http://www.ietf.org/rfc/rfc3492.txt`

Punycode	**Unicode**
xn–wgbl6a	رطق

It would be great if. . .

. . . we had a tool that:

- could just parse properly a URL
- is damn fast
- does not allocate to parse URLs
- read character only one time
- is also available as a C library
- with a command line tool
- . . .

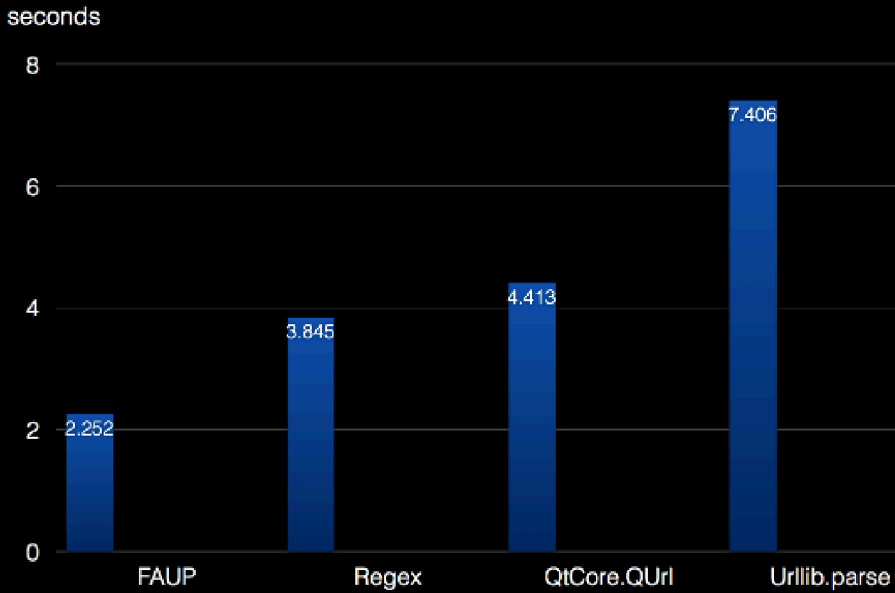
It would be great if. . .

. . . we had a tool that:

- could just parse properly a URL
- is damn fast
- does not allocate to parse URLs
- read character only one time
- is also available as a C library
- with a command line tool
- . . .
- with Python bindings
- a webserver embedded
- and LUA scripting

```
$ faup —o json “http://root:admin@rad.msn.com:80/ADSADClient31.dll?GetAd=&PG=IMSCB2&AP=1007#blah”
{
  “scheme”: “http”,
  “credential”: “root:admin”,
  “subdomain”: “rad”,
  “domain”: “msn.com”,
  “domain_without_tld”: “msn”,
  “host”: “rad.msn.com”,
  “tld”: “com”,
  “port”: “80”,
  “resource_path”: “/ADSADClient31.dll”,
  “query_string”: “?GetAd=&PG=IMSCB2&AP=1007”,
  “fragment”: “#blah”,
  “url_type”: “mozilla_tld”
}
```

Parsing 1 million URLs

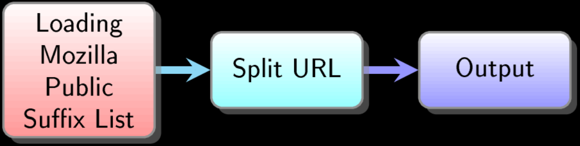


Exercise 1

Run faup with json output on the following URLs

- http://www.google.co.uk
- ftp://ac.bl.uk
- blah.42
- http://192.168.0.42:9843

How Faup works?



Faup Webserver

- To avoid loading the Mozilla public suffix list for every URL, Faup can run as a server

Faup Webserver

- To avoid loading the Mozilla public suffix list for every URL, Faup can run as a server
- HTTP being an universal and popular protocol, Faup listen to HTTP requests

```
$ faup —b —w 0.0.0.0:9876
$ faup –b –w 0.0.0.0:9876
```

```
$ echo ‘‘http://www.slashdot.org’’ |base64
aHR0cDovL3d3dy5zbGFzaGRvdC5vcmc=
$ faup –b –w 0.0.0.0:9876
```

```
$ echo ‘‘http://www.slashdot.org’’ |base64
aHR0cDovL3d3dy5zbGFzaGRvdC5vcmc=
```

```
$ curl http://127.0.0.1:9876/json?url=aHR0cDovL3d3dy5zbGFzaGRvdC5vcmc=
{
  ‘‘scheme’’: ‘‘http’’,
  ‘‘credential’’: ‘‘‘‘,
  ‘‘subdomain’’: ‘‘www’’,
  ‘‘domain’’: ‘‘slashdot.org’’,
  ‘‘domain_without_tld’’: ‘‘slashdot’’,
  ‘‘host’’: ‘‘www.slashdot.org’’,
  ‘‘tld’’: ‘‘org’’,
  ‘‘port’’: ‘‘‘‘,
  ‘‘resource_path’’: ‘‘‘‘,
  ‘‘query_string’’: ‘‘‘‘,
  ‘‘fragment’’: ‘‘‘‘,
  ‘‘url_type’’: ‘‘mozilla_tld’’
}
```

Faup Modules

- Use Lua scripting
- Input scripting
- Output scripting

```
$ faup $
Usage: faup $ shell_command [parameters]

Available shell comands: modules
```

Faup Existing Modules

```
$ faup $ modules list all
Modules enabled:

Modules available:
[0] /usr/local/share/faup/modules_available/redis-url-threatintel.lua
[1] /usr/local/share/faup/modules_available/printcsv.lua
[2] /usr/local/share/faup/modules_available/writeinput.lua
[3] /usr/local/share/faup/modules_available/uppercase.lua
[4] /usr/local/share/faup/modules_available/emulation_ie.lua
[5] /usr/local/share/faup/modules_available/writeall.lua
```

What is a Snapshot?

- A Snapshot is a package of normalized URLs
  - URL Features: *domain*, *tld*, *querystring*, . . .
  - Count
  - First Time Seen
  - Last Time Seen

Create a snapshot

With Alexa top 1 million records

```
$ cat top-1m.csv | cut -d, -f2 | head -n 10 | faup -q -s top10
$ faup $ snapshot get foo tld com
{"value": "com", "count": 9, "first seen": "2018-07-02 21:50:14 -0700", \
  "last seen": "2018-07-02 21:50:14 -0700"}
```

Playing with snapshot

Checking what matches the top-1m

```
$ for line in $(head -n10 ../unique-domains); \
do faup \$ snapshot get top-1m domain_without_tld $line; done

{"value": "0", "count": 1, "first seen": "2018-06-15 23:54:44 -0700", \
  "last seen": "2018-06-15 23:54:44 -0700"}
{"value": "01net", "count": 2, "first seen": "2018-06-15 23:54:44 -0700", \
  "last seen": "2018-06-15 23:54:44 -0700"}
{"value": "0fees", "count": 2, "first seen": "2018-06-15 23:54:44 -0700", \
  "last seen": "2018-06-15 23:54:46 -0700"}
```

Thank you

Questions?
stricaud@splunk.com
sebastien@honeynet.org
@tricaud