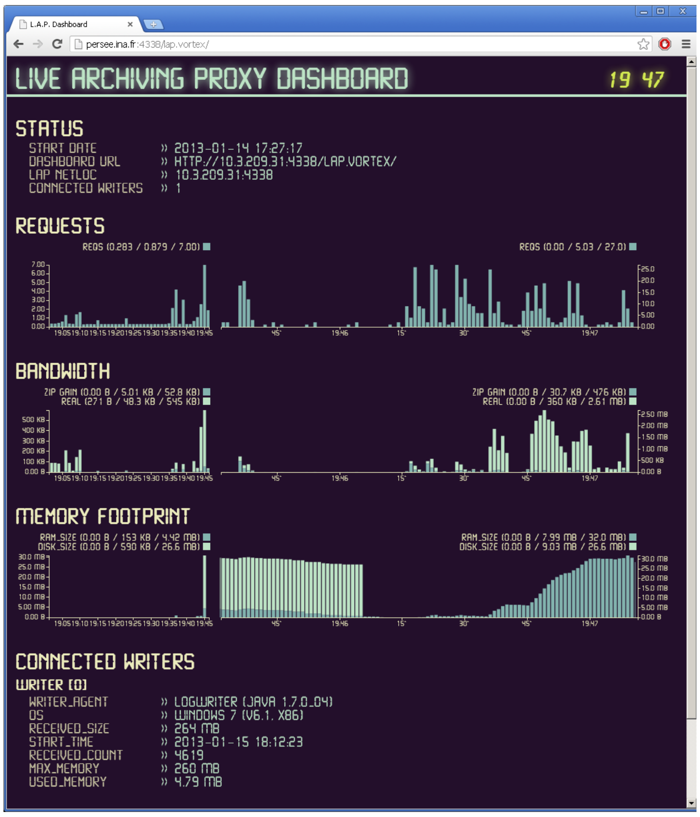
Live Archiving Proxy

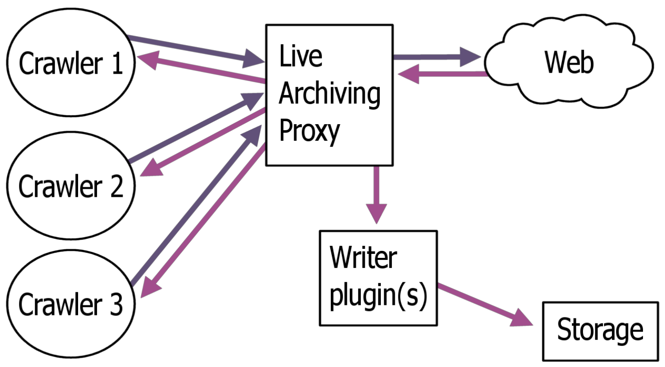
User Guide


L.A.P. Dashboard
persee.ina.fr:4338/lap.vortex/
REQS (0.283 / 0.879 / 7.00)
REQS (0.00 / 5.03 / 27.0)
ZIP GAIN (0.00 B / 5.01 KB / 52.8 KB)
REAL (271 B / 48.3 KB / 545 KB)
ZIP GAIN (0.00 B / 30.7 KB / 476 KB)
REAL (0.00 B / 360 KB / 2.61 MB)
RAM_SIZE (0.00 B / 153 KB / 4.42 MB)
DISK_SIZE (0.00 B / 590 KB / 26.6 MB)
RAM_SIZE (0.00 B / 7.99 MB / 32.0 MB)
DISK_SIZE (0.00 B / 9.03 MB / 26.6 MB)

```
LIVE ARCHIVING PROXY DASHBOARD                    19 47

STATUS
   START DATE          » 2013-01-14 17:27:17
   DASHBOARD URL       » HTTP://10.3.209.31:4338/LAP.VORTEX/
   LAP NETLOC          » 10.3.209.31:4338
   CONNECTED WRITERS   » 1

REQUESTS

BANDWIDTH

MEMORY FOOTPRINT

CONNECTED WRITERS
WRITER [0]
   WRITER_AGENT        » LOGWRITER (JAVA 1.7.0_04)
   OS                  » WINDOWS 7 (V6.1, X86)
   RECEIVED_SIZE       » 264 MB
   START_TIME          » 2013-01-15 18:12:23
   RECEIVED_COUNT      » 4619
   MAX_MEMORY          » 260 MB
   USED_MEMORY         » 4.79 MB
```

General principles

The LAP (Live Archiving Proxy) is a single executable linux 64-bit binary file called "lap".
The LAP is a proxy through which you can access the Web like a normal proxy. All HTTP traffic that goes through the LAP is captured and sent to a dedicated writer. **If no writer is connected to the LAP, no archive will be created** and the used memory by the LAP will increase.



You can use an existing writer plugin or develop your own. As of now, this is the list of existing LAP writer plugins:

- WARC writer : download.
- Print writer (will only print captured URLs to console, not archive anything to disk) : download.

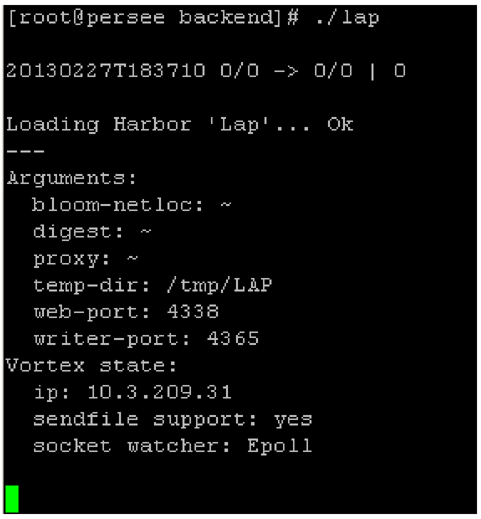
Starting up a LAP with a WARC writer

1) Launch the LAP

The LAP has two essential parameters:

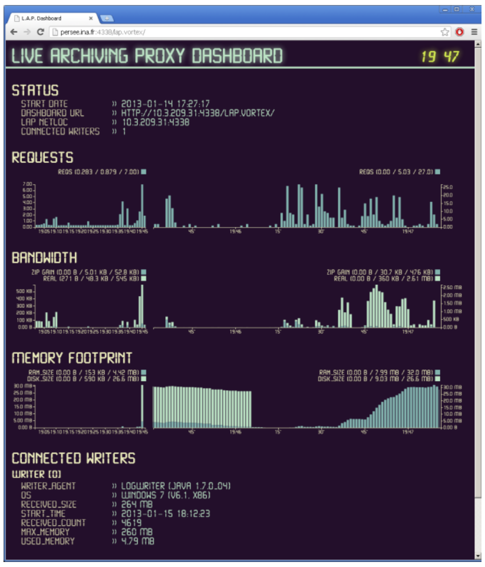
- *WEB PORT* : the port the LAP listens on Web clients (default: 4338)
- *WRITER PORT* : the port the LAP listens on writers (default: 4365)

```
> lap --web-port WEB_PORT --writer-port WRITER_PORT
```



```
[root@persee backend]# ./lap

20130227T183710 0/0 -> 0/0 | 0

Loading Harbor 'Lap'... Ok
---
Arguments:
  bloom-netloc: ~
  digest: ~
  proxy: ~
  temp-dir: /tmp/LAP
  web-port: 4338
  writer-port: 4365
Vortex state:
  ip: 10.3.209.31
  sendfile support: yes
  socket watcher: Epoll
```

Expected output



```
LIVE ARCHIVING PROXY DASHBOARD                19 47

STATUS
  START DATE          » 2013-01-14 17:27:17
  DASHBOARD URL       » HTTP://10.3.209.31:4338/LAP.VORTEX/
  LAP NETLOC          » 10.3.209.31:4338
  CONNECTED WRITERS   » 1

REQUESTS

BANDWIDTH

MEMORY FOOTPRINT

CONNECTED WRITERS
WRITER [0]
  WRITER_AGENT        » LOGWRITER (JAVA 1.7.0_04)
  OS                  » WINDOWS 7 (V6.1. X86)
  RECEIVED_SIZE       » 264 MB
  START_TIME          » 2013-01-15 18:12:23
  RECEIVED_COUNT      » 4619
  MAX_MEMORY          » 260 MB
  USED_MEMORY         » 4.79 MB
```

L.A.P. Dashboard
persee.ina.fr:4338/lap.vortex/
REQS (0.283 / 0.879 / 7.00)
REQS (0.00 / 5.03 / 27.0)
ZIP GAIN (0.00 B / 5.01 KB / 52.8 KB)
REAL (271 B / 48.3 KB / 545 KB)
ZIP GAIN (0.00 B / 30.7 KB / 476 KB)
REAL (0.00 B / 360 KB / 2.61 MB)
RAM_SIZE (0.00 B / 153 KB / 4.42 MB)
DISK_SIZE (0.00 B / 590 KB / 26.6 MB)
RAM_SIZE (0.00 B / 7.99 MB / 32.0 MB)
DISK_SIZE (0.00 B / 9.03 MB / 26.6 MB)

Dashboard interface

Once the LAP is started, the dashboard is accessible with a browser at `http://LAP_HOST:WEB_PORT/`

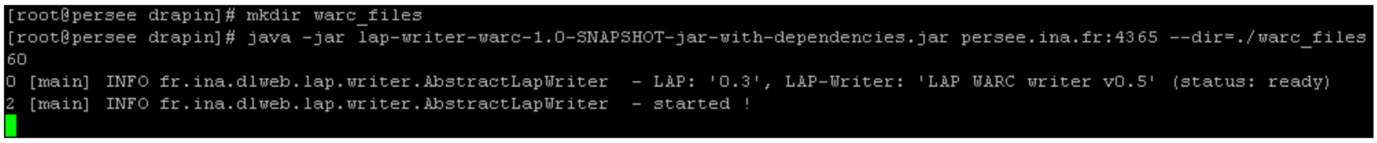
2) Launch the WARC writer

The writer WARC has three essential parameters:

- *LAP HOST* : the IP or hostname of the LAP
- *WRITER PORT* : the port on which the LAP listens for writers
- *WARC DIR* : the directory in which WARC files must be created

```
> java -jar lap-writer-warc-1.0-SNAPSHOT-jar-with-dependencies.jar
LAP_HOST:WRITER_PORT --dir=WARC_DIR
```

The output should be similar to this :



```
[root@persee drapin]# mkdir warc_files
[root@persee drapin]# java -jar lap-writer-warc-1.0-SNAPSHOT-jar-with-dependencies.jar persee.ina.fr:4365 --dir=./warc_files
60
0 [main] INFO fr.ina.dlweb.lap.writer.AbstractLapWriter  - LAP: '0.3', LAP-Writer: 'LAP WARC writer v0.5' (status: ready)
2 [main] INFO fr.ina.dlweb.lap.writer.AbstractLapWriter  - started !
```

3) Archive some stuff

Configure your crawler or any browser to use the proxy LAP_HOST on port WEB_PORT, and watch the LAP in action !

Archiving HTTPS URLs

1) Pseudo-HTTPS mode

To make HTTPS traffic archivable, we make in non-encrypted between the Web client and the LAP. In order to do so, the Web client needs to alter outgoing HTTPS requests to make them HTTP, and append the fake ".HTTPS" TLD to the URL's host.
Here are a few examples of the alteration:

`https://example.com` becomes
`http://example.com.https`

`https://www.example.com/index.html` becomes
`http://www.example.com.https/index.html`

In this mode, the connection between the Web client and the LAP is unsecure and clear, but the connection between the LAP and the target Web server is true HTTPS.
The metadata sent to the writer will not contain the fake URL, but rather the actual HTTPS URL. From the writer's point of view, nothing really makes this request different except for the fact that the URL is HTTPS.

2) Altering URLs in PhantomJS to use pseudo-HTTPS mode

The following code will enable PhantomJS to alter HTTPS URLs on the fly to make them archivable by the LAP.

```
var page = require('webpage').create();

// LAP: alter HTTPS urls with fake .https TLD
var urlRegexp = new RegExp('^(https?)://([^/]+)(/.?)?$')
page.onResourceRequested = function(requestData, networkRequest) {
   var match = urlRegexp.exec(requestData.url);
   if (match.length >= 3 && match[1] === 'https') {
        networkRequest.changeUrl(
        'http://' + match[2] + '.https' + (match[3] === undefined ? '' : match[3])
        );
   }
};

var target = 'https://www.example.org/index.html';
page.open(target, function (status) {
    console.log('got status "' + status + '" for url "' + target + '"');
});
```

LAP advanced configuration

proxy	The network location (host:port) of the HTTP proxy to be used by the LAP to access the Web. Using a caching proxy is useful for external bandwidth reduction.
web-port	The port on which the LAP listens for Web clients (default: 4338).
writer-port	The port on which the LAP listens for writer connections (default: 4365).
digest	The name of a digest (MD5, SHA1, SHA256). This digest will be computed by the LA and sent to the writer. Use `metadata.getInfo("digest")` to read the digest in the writer.
temp-dir	The temporary directory used by the LAP to store big contents (over 5MB) before sending them to a writer (default: /tmp/LAP).
bloom-netloc	The network location (host:port) of a Bloom filter server used for deduplication.

WARC writer advanced configuration

dir	Directory in which WARC file will be created
prefix	Prefix of the WARC files (default: LAP)
deduplication	Use deduplication mechanism (default: false). An embedded database is used to avoid storing the contents multiple times by creating RT_IDX_REVISIT WARC records. The internal database is flushed every time the writer is started.
compress	Use WARC compression (default: false)
max-file-size	The maximum WARC file size in bytes (default: 1073741824 bytes = 1Gb). When the current WARC size reaches this size, it is closed and a new file is created.
timeout	Number of seconds before the connection to the LAP fails if the writer gets no response (default: 10).
ispartof	WARC Info : "ispartof" field (default: empty).
description	WARC Info : "description" field (default: empty).
operator	WARC Info : "operator" field (default: empty).
httpheader	WARC Info : "httpheader" field (default: empty).
verbose	Use verbose output (default: not enabled)