Rewriting History

Steffen Fritz
steffen@fritz.wtf

01.01.2016

Abstract

This article discusses the WARC (Web ARChive) format in general and the potential to misuse it. First published Jan 2016 in *2600 - The Hacker Quarterly*[1]

1 Web Archiving

With the growth of the world wide web and its increasing cultural and political influence the archiving of web published content became an important matter for preserving the cultural heritage. Public institutions like the Library of Congress (LoC) in the United States[2] or the Bibliothèque nationale de France (BnF)[3] and non-profit organizations like the Internet Archive (IA)[4] are doing a great job in this. While the LoC or the BnF don't crawl the whole web - they curate, collect and preserve topic, event or domain specific - the IA takes them all, automatically. At least they try. Other services like http://archive.is or http://webrecorder.io allow users to manually mirror web pages and see the results right away.

Whoever is preserving has three possible archiving methods: transactional archiving, database archiving and remote harvesting. The most common one is the latter and the idea is fairly simple: Copy a web site, search the source code for URLs, copy the referenced resources and repeat recursively until you hit a termination condition, e.g. no new web resource found or

[1] Fritz, Steffen. 2600 - The Hacker Quarterly, 32/4, 2016. New York.
[2] http://www.loc.gov/webarchiving/
[3] http://www.bnf.fr/en/professionals/digital_legal_deposit.html
[4] https://archive.org

when leaving the domain. A program doing this is called a web crawler, popular tools are Heritrix[5] and HTTrack[6]. HTTrack saves files as a web server delivers them, e.g. image.jpg as image.jpg and index.html as index.html. Heritrix creates web archives accordingly the WARC file format, which is the de facto standard for web archives.

2 WARC Format

The WARC file format defines how to store payload content, control information and arbitrary metadata as blocks together in one file. Control information like DNS and HTTP requests and responses make the crawl comprehensible. Hash sums, dates and file sizes describe the digital objects. Each warc record in a warc file is initiated by 'WARC/1.0' and consists of a record header that describes the type and content of the record. It is followed by the content and two newlines.

You can create a warc file with wget >= 1.14, just add the switch –warc-file=FOO, e.g.

```
wget --warc-file=2600 http://2600.com
```

wget creates an warc.gz file. Unzip it and open the warc file with an editor like vim or emacs. The first block in the file describes the warc file itself, the following blocks are related to network traffic and payload. The fields in the blocks have a simple named fields structure, terminated with CRLF. An important field is the WARC-Target-URI. It is identical to the source URI and therefor it also determines the file name of the payload.

Let's have a look at an example, some lines are omitted. All blocks are from the same file. We investigate three blocks.

[5]https://webarchive.jira.com/wiki/display/Heritrix/Heritrix
[6]https://www.httrack.com/

<CODE>

```
WARC/1.0^M
WARC-Type: warcinfo^M
Content-Type: application/warc-fields^M
WARC-Date: 2015-09-15T13:20:42Z^M
WARC-Filename: test.warc.gz^M
WARC-Block-Digest: sha1:XGCP3I5MSJ4DGD7EH5DTLJJXULVOATQK^M
Content-Length: 224^M
^M
software: Wget/1.16.3 (linux-gnu)^M
format: WARC File Format 1.0^M
^M
^M

(...)
```

</CODE>

The above is the first block in our warc file. It is an info block and contains "warc-fields". The content, i.e. the following two lines, has a length of 224 bytes.

The second block is a request block in which the network communication for a single request is logged.

<CODE>

```
WARC/1.0^M
WARC-Type: request^M
WARC-Target-URI: http://test.wtf/^M
Content-Type: application/http;msgtype=request^M
WARC-Date: 2015-09-15T13:20:42Z^M

(...)
```

</CODE>

The third block contains the response from the server. After the warc fields and the metadata you can see the html payload.

```
<CODE>

    WARC/1.0^M
    WARC-Type: response^M
    WARC-Target-URI: http://test.wtf/^M
    WARC-Date: 2015-09-15T13:20:42Z^M
    Content-Type: application/http;msgtype=response^M
    Content-Length: 4896^M
    ^M
    HTTP/1.1 200 OK^M
    Server: nginx/1.8.0^M
    Date: Tue, 15 Sep 2015 13:20:42 GMT^M
    Content-Type: text/html^M

    (...)

    ^M
    <!DOCTYPE html>
    <html dir="ltr" lang="en">
    <head>
      <meta charset="UTF-8" />
      <meta name="viewport" content="width=device-width,
          initial-scale=1">

    (...)

</CODE>
```

For a full description read the specification, the ISO draft is available at the BnF and well readable.[7]

At the time of this writing, the International Internet Preservation Consortium (IIPC) is working on version 1.1 of the specification. Pretty transparently on github, by the way.[8]

What to do with the warc file? Replay it. There are a few tools to render the archived content. One is the (Open) Wayback Machine you may know from the Internet Archive. Another one is Pywb, which I prefer for local

[7] http://bibnum.bnf.fr/WARC/

[8] http://iipc.github.io/warc-specifications/

testing, because it is pretty easy to set up and much lighter.[9] [10]

Whatever you use, you set up a data storage for the warc files and the tool of your choice serves the content, rendered by a browser. Suppose we are using the Wayback Machine on localhost and the above example with WARC-Target-URI http://test.wtf, you would open the URL http://localhost:8080/web/20150915132042/http://test.wtf and you'd see how the website http://test.wtf looked like in September 2015.

Do you see where this is going? Let's assume we could create warc files with arbitrary content. And let us assume further we could manage to inject that file into a trustful archive and that we could share a link with Alice and Bob: Both might be tricked in to believe a website looked like something it never had. Let's call it 'post defacing'.

3 Create a warc file and make Bob trust it

Of course you could create a warc file with a text editor. But the creation of hash sums, length of content etc pp might be a little bit annoying. You could also set up an environment to crawl a fake site. I decided to write a Python script to create minimal, valid warc files.[11]

You call the script

```
python html2warc $URL $SOURCE $TARGET_FILE'.
```

$URL is the root value for the WARC-Target-URI field, $SOURCE must be a directory with the desired content and $TARGET is the name of the warc file.

A proof of concept warc file can be downloaded from github.

You can upload that file to webrecorder.io and watch the result. Fascinating, isn't it? Well, webrecorder.io isn't an archive and the service explicitly states that. But are Alice and Bob aware of that? Checking the trustworthiness of sources isn't a standard procedure in online communication. Sadly.

To upload the file to archive.org and trick Bob, things are a little bit more complicated. You can upload a warc file with an ordinary user account into a collection. But then it is stored as the mediatype 'texts' and can only be downloaded again as a warc file. If you try to change the web memory for a specific site you have to convince a member of the Archive Team to copy your warc into their collection and change the media type from 'texts' to

[9] https://github.com/iipc/openwayback

[10] https://github.com/ikreymer/pywb

[11] https://github.com/ampoffcom/warc_2600

'web'. Obviously, it is possible to steal the archive login from a member and do it by yourself. No doubt, some Mallorys are trying to do this.

Remember: It is not about defacing a web site. It is about changing the political, cultural and social memory.

4 Impact and responsibility

Putting false documents into trusted archives is not a new threat. In 2005 the British National Archives detected faked documents, claiming that Heinrich Himmler was murdered in custody. And in 1967 Gérard de Sède wrote in his book 'Le trésor maudit' that a guy named Pierre Plantard is a descendant of Dagobert II. and therefor the one and only King of France. De Séde refered to documents found in the National Archives in Paris. Placed there by, you guessed it, Pierre Plantard. You may read on this very interesting case by searching for the 'Plantard Dossiers'. I am pretty sure that faked documents have rewritten history and they will in the future. Web archives are just another playground. But an important one.

Who's responsible? Surely, archives have to check their objects and they are responsible for the data they provide - may it be books, birth certificates or web archives. But in my humble opinion users also have to check their sources and should not automatically trust something because of its outer packing. Remember that Trojan horse?