The Archival Acid Test: Evaluating Archive Performance on Advanced HTML and JavaScript

Mat Kelly, Michael L. Nelson, and Michele C. Weigle
Old Dominion University
Department of Computer Science
Norfolk, Virginia 23529 USA
{mkelly,mln,mweigle}@cs.odu.edu

ABSTRACT

When preserving web pages, archival crawlers sometimes produce a result that varies from what an end-user expects. To quantitatively evaluate the degree to which an archival crawler is capable of comprehensively reproducing a web page from the live web into the archives, the crawlers' capabilities must be evaluated. In this paper, we propose a set of metrics to evaluate the capability of archival crawlers and other preservation tools using the Acid Test concept. For a variety of web preservation tools, we examine previous captures within web archives and note the features that produce incomplete or unexpected results. From there, we design the test to produce a quantitative measure of how well each tool performs its task.

Categories and Subject Descriptors

H.3.7 [**Online Information Services**]: Digital Libraries and Archives

General Terms

Experimentation, Standardization, Verification

Keywords

Web Crawler, Web Archiving, Digital Preservation

1. INTRODUCTION

Since much of our cultural discourse occurs on the web, web archiving is necessary for posterity. The goal of web archiving is to capture web pages so they can be "replayed" at a later date. Web archiving tools access these pages on the live web in a manner similar to tools used by search engines (crawlers) and preserve the pages in a format that allows the data and contextual information about the crawl to be re-experienced. These "archival crawlers" take different approaches in digital preservation and thus their capability and scope vary.

Because archival crawlers attempt to duplicate what a user would see if he accessed the page on the live web, variance from what is preserved and what would have been seen compromises the integrity of the archive. The functional difference between archival crawlers and web browsers causes this sort of unavoidable discrepancy in the archives, but it is difficult to evaluate how good of a job the crawler did if the information no longer exists on the live web. By examining what sort of web content is inaccurately represented or missing from the web archives, it would be useful to evaluate the capability of archival crawlers (in respect to that of web browsers that implement the latest technologies) to determine what might be missing from their functional repertoire.

Web browsers exhibited this deviation between each other in the early days of Web Standards. A series of "Acid Tests" that implemented the Web Standards allowed each browser to visually and functionally render a web page and produce an evaluation of how well the browser conformed to the standards. In much the same way, we have created an "Archival Acid Test" to implement features of web browsers in a web page. While all standards-compliant browsers will correctly render the live page, this is not always the case when the archived version of the page is rendered. This difference can be used to highlight the features that archival crawlers are lacking compared to web browsers and thus emphasize the deviations that will occur in web archives compared to what a user would expect from a digitally preserved web page.

2. RELATED WORK

Web archives are generated by a variety of tools in a variety of formats. An ISO standard format utilized by institutional and personal web archivists alike is the Web ARChive (WARC) format [1]. WARC files allow HTTP communication that occurred during a crawl as well as payload, metadata and other archival features to be encoded in a single or an extensibly defined set of WARC files.

Heritrix paved the way for Internet Archive (IA) to utilize their open source Heritrix to create ARC and WARC files from web crawls while capturing all resources necessary to replay a web page [2]. Other tools have since added WARC creation functionality [3, 4, 5]. Multiple software platforms exist that can replay WARCs but IA's Wayback Machine (and its open source counterpart[1]) is the de facto standard.

Multiple services exist that allow users to submit URIs for preservation. IA recently began offering a "Save Page Now" feature co-located with their web archive browsing inter-

[1]`https://github.com/iipc/openwayback`

Table 1: The archiving services and software that we tested.

Tool/Service	URI	Description
Archive.org's "Save Page Now"	`archive.org/web/`	Web-based service, archives replayed via Wayback Machine at Internet Archive
Archive.is	`archive.is/`	Web-based service, Archives replayed within web interface.
Mummify.it	`mummify.it/`	Web-based service, Archives replayed within web interface.
Perma.cc	`perma.cc/`	Web-based service, Archives replayed within web interface.
WebCite	`webcitation.org/`	Web-based service, Archives replayed within web interface.
Heritrix	`github.com/internetarchive/heritrix3`	Locally hosted IA-created archival web crawler. Generates WARC files, replayed in local Wayback.
WARCreate	`warcreate.com/`	Google Chrome Extension, generates WARC files. WARCs replayed in local Wayback.
Wget	`gnu.org/software/wget/`	Command-line executed, generates WARC files. WARCs replayed in local Wayback.

face. Archive.is, Mummify, Perma.cc and WebCitation (see Table 1), all provide URI submission and archive browsing services with obfuscated URIs, though none make accessible a resultant WARC file.

Brunelle et al. [6] highlighted the difficulties that current archival crawlers have with capturing content on the live web that relies on JavaScript, and Kelly et al. [7] showed the ramifications this has had on preserving web pages over time. Brunelle pointed out that sometimes what archives show as preserved actually "reaches out" into the live web on replay, causing false positives [8].

The Acid3 Test[2] is the third in a series of functional evaluations for web browsers' compliance with Web Standards. Evaluation using the Archival Acid Test is done in much of the same way at the Acid3 Test, with correct rendering being the primary, but not sole, basis for correctness.

3. THE ARCHIVAL ACID TEST

Inspired by the Acid Tests administered by the Web Standard Project (WaSP)[3], we built the Archival Acid Test[4] to evaluate how well archival tools perform at preserving web pages. Unlike WaSP's initiatives, evaluation of web archival software is not standardized, so a comprehensive test of what these tools should be able to capture needs to be established. The Archival Acid Test evaluates the archives' ability to render pages employing a variety of standardized and emerging conventions with HTML and JavaScript.

The crux of the tests is to determine how well an archival tool preserves a web page in terms of similarity to what would be expected by a user viewing the page from the live web, i.e., a respectively modern web browser. Web Standards are continuously evolving with the feature set for web browsers temporally lagging the standards in being implemented though frequently containing experimental implementations. Archival crawlers, given a greater need for reliability, lag in implementing newly standardized features than browsers, though they will frequently rely on a common engine utilized by browsers to stay-up-to-date.[5] The deviation from the web page processing engines used by archival tools (whether built-to-purpose or older versions of browser engines) is a source of discrepancy between the content on a live web page and that which is captured by these tools.

We have established a set of tests into three categories to better group web page features that might be problematic for archival tools to capture. Each test is represented by a 10-by-10 pixel blue square. Any deviation from the blue square (e.g., no image present, red square instead of blue) signifies an error in what a user would expect from a preserved web page, and thus the particular test is considered to have been failed by the tool. A reference image (Figure 2(a)) is used as a comparative basis for correctness, much in the same way Web Standards Acid Tests provided a static image to evaluate what was experienced versus what is right.

3.1 Basic Tests (Group 1)

The set of *Basic Tests* is meant to ensure that simple representations of resources on web pages are captured. Each tests' name represents what is presented to be captured by the archival crawler. A sample URI follows each test's name.

1a. Local (same server as test) image, relative URI to test
`./1a.png`
1b. Local image, absolute URI
`http://archiveacidtest/1b.png`
1c. Remote image, absolute URI
`http://anotherserver/1c.png`
1d. Inline content, encoded image
`data:image/png;base64,iVB...`
1e. Remote image, scheme-less URI
`//anotherserver/1e.png`
1f. Recursively included CSS
In style.css: `@import url("1f.css");`

3.2 JavaScript Tests (Group 2)

The second group of tests is meant to evaluate the archival crawler's JavaScript support in terms of how the script would execute were the test accessed on the live web with a browser.

2a. Local script, relative URI, loads local resource
`<script src="local.js" />`
2b. Remote script, absolute URI, loads local resource
`<script src="http://anotherserver/local.js" />`
2c. Inline script, manipulates DOM[6] at runtime
`<script>...(JS code)...</script>`

[2]`http://acid3.acidtests.org/`
[3]`http://www.acidtests.org/`
[4]The source of the test is available at `https://github.com/machawk1/archivalAcidTest` .
[5]For example, the open source V8 and SpiderMonkey rendering engines allow resources that require JavaScript to be present on a web page and be captured by archival tools.
[6]Document Object Model, the structure of the web page that, when manipulated, affects the content

2d. Inline script, Ajax image replacement, loads local resource
`img.src = "incorrect.png";`
`...code to replace incorrect image with local...`

2e. Inline script, Ajax image replacement, Same-origin Policy (SOP)[7] enforcement, replacement (bad) == false positive
`img.src = "correct.png";`
`...code to replace correct image with image from SOP violation...`

2f. Inline script, manipulates DOM after delay
`setTimeout(function(){ ...load image...},2000);`

2g. Inline script, content loaded upon interaction, introducing resources
`window.onscroll = function()`

2h. Inline script, add local CSS at runtime

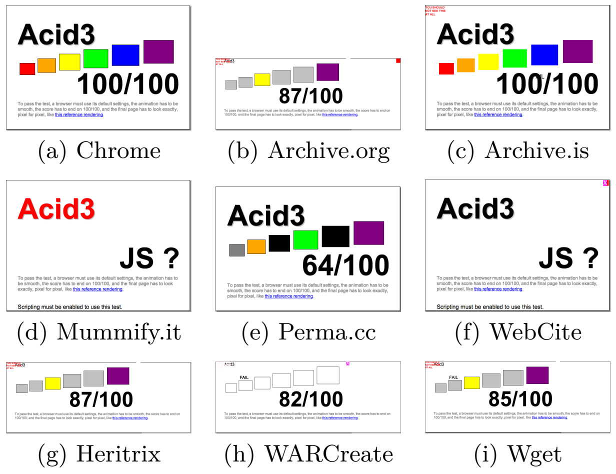
3.3 Advanced Features Tests (Group 3)

The third group of tests evaluates script-related features of HTML beyond simple DOM manipulation.

3a. HTML5 Canvas drawing with runtime-fetched content
3b. Remote image stored then retrieved from HTML5 localStorage
3c. Embedded content using iframe
3d. Runtime binary object

4. EVALUATION

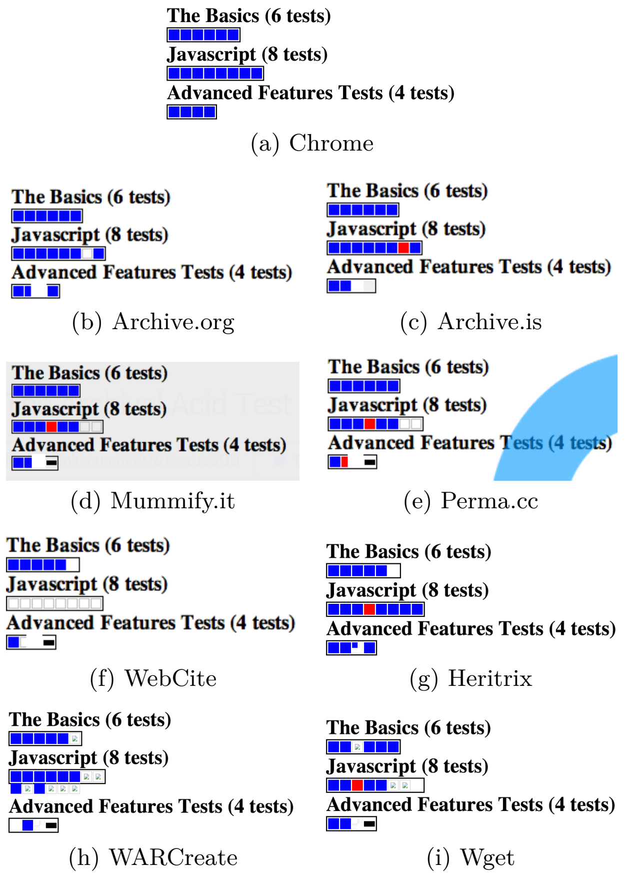
To establish a baseline, we first ran each tool through the Acid3 test. From this we observed preliminary results that were indicative of the archival tools' lack of full support of the features of standards compliant web browsers (Figure 1). Given that we are testing features that have come about since Acid3 was released, the Archival Acid Test further exercises the tested sites' and tools' standards compliance and specifically highlights their failures.



(a) Chrome (b) Archive.org (c) Archive.is
(d) Mummify.it (e) Perma.cc (f) WebCite
(g) Heritrix (h) WARCreate (i) Wget

Figure 1: Preliminary tests show that archival tools exhibit an incomplete feature set compared to modern web browsers.

In Figure 1, we show the results of each tool's attempt at capturing the Acid3 Test web page. Compared to the correct rendering in Chrome (Figure 1(a)), the five service-based tools from Archive.org, Archive.is, Mummify.it, Perma.cc, and WebCite (Figures 1(b), 1(c), 1(d), 1(e), and 1(f), resp.) have more variance in their performance than the three tools of Heritrix, WARCreate, and Wget (Figures 1(g), 1(h), and 1(i), resp.). While Archive.is appears to get the closest with its rendering, subtle stylistic differences are easily observable with error text appearing. This indicates that contrary to the 100/100 rating, neither Archive.is nor any other service or tool tested here fully passes the Acid3 test.



(a) Chrome
(b) Archive.org (c) Archive.is
(d) Mummify.it (e) Perma.cc
(f) WebCite (g) Heritrix
(h) WARCreate (i) Wget

Figure 2: Archiving service and tools' performance on the Archival Acid Test. The reference image (Figure 2(a)) shows what should be displayed if all tests are passed. This image represents what a user sees when viewing the test in a modern web browser.

4.1 Tools' Performance

We evaluated five web archiving services (Archive.org, Archive.is, Mummify.it, Perma.cc, and WebCite) and three WARC-generating archiving tools (Heritrix, WARCreate, and Wget). Each service provided a simple interface where a user can submit a URI, and the web page at that URI is preserved on-command. Heritrix was configured with the test as the lone URI in a crawl. Wget was given a command[8] with the URI as a parameter and WARC as the desired output format. For WARCreate, we navigated to the test's web page and generated a WARC. For each WARC-generating

[7]`https://developer.mozilla.org/en-US/docs/Web/JavaScript/Same_origin_policy_for_JavaScript`

[8]`wget --mirror --page-requisites --warc-file="wget.warc" http://{acid test URI}`

Table 2: By aligning the services' and tools' tests and failures, a theme in capability (and lack thereof) is easily observable between the two classes. Where archiving services exhibit a perfect record in the Group 1 set, the Group 2 set proved troublesome for all but Heritrix. Further, the nearly across-the-board failures of 2g and 3c when modern browsers pass all of the tests emphasizes the functional discrepancy between archiving tools and browsers.

Tool \Test	1a	1b	1c	1d	1e	1f	2a	2b	2c	2d	2e	2f	2g	2h	3a	3b	3c	3d
Archive.org	·	·	·	·	·	·	·	·	·	·	·	·	✗	·	·	·	✗	·
Archive.is	·	·	·	·	·	·	·	·	·	·	·	·	✗	·	·	·	✗	✗
Mummify.it	·	·	·	·	·	·	·	·	·	✗	·	·	✗	✗	·	·	✗	✗
Perma.cc	·	·	·	·	·	·	·	·	·	✗	·	·	✗	✗	·	✗	✗	✗
WebCite	·	·	·	·	·	✗	✗	✗	✗	✗	✗	✗	✗	✗	·	✗	✗	✗
Heritrix	·	·	·	·	·	✗	·	·	·	✗	·	·	·	·	·	·	✗	·
WARCreate	·	·	·	·	·	✗	·	·	·	·	·	·	✗	✗	✗	·	✗	✗
Wget	·	·	✗	·	·	·	·	·	✗	·	·	✗	✗	✗	·	·	✗	✗

· = Test Passed ✗ = Test Failed

archiving tool, we replayed the generated WARC files in a local instance of Wayback[9].

While almost all archiving services and tools tested had difficulty with test 2g, the five service-based archiving websites (Archive.org, Archive.is, Mummify.it, Perma.cc, and WebCite Figures 2(b), 2(c), 2(d), 2(e), and 2(f), resp.) show an interesting common set of features compared to the three archiving tools (Heritrix, WARCreate, and Wget, Figure 2(g), 2(h), and 2(i), resp.). This is better illustrated in Table 2.

4.2 Evaluating the Acid Test's Methods

The features of the Archival Acid Test are not necessarily bleeding edge, yet no service or tool completely passed. More advanced features were considered but as a preliminary test of evaluating the targets, the 18 tests presented here were more than sufficient at pointing out their shortcomings. Of particular interest are tests 2g and 3c, which tested whether the targets were able to capture content loaded after a short delay and content embedded in an iframe. In one of our previous experiments [7], we evaluated content already in the archives that existed in frames, so this discrepancy was unexpected.

There exists a caveat in evaluating archiving tools in that the appearance of how well a page is preserved might vary depending on how well the archive is replayed. Our initial tests of replaying WARCs in the latest non-beta release of Wayback showed that all resources contained in the WARC were not represented in the replayed page, support for uncompressed WARCs (e.g., those generated by WARCreate) was incomplete and a host of other issues that made us question the validity of comparing Wayback-replayed archives to those replayed by archiving web sites. To account for this, we utilized the latest beta release of Wayback (per Section 2) and manually verified that the WARCs did not contain any un-replayed resources to ensure that the tools' performance was accurately represented by the Archival Acid Test.

5. CONCLUSIONS

In this work we created a means of evaluating archival crawlers to pinpoint areas of functionality that result in archives being incomplete or misrepresentative of the live web. By first running a preliminary test of each service and tool on the Web Standards Acid3 test, we were able to show that there is a functional discrepancy in how archiving tools perform versus modern web browsers. By subsequently running our Archival Acid Test with a more modern feature-driven evaluation of each archiving tool, we were able to observe which parts of web pages are problematic for these tools to capture and how good of a job they are currently performing.

Though not comprehensive of every possible shortcoming compared to web browsers, the Archival Acid Test is a first step in establishing a means of evaluating archiving tools. In much of the same way that the Acid Tests for Web Standards provoked browser manufacturers to standardize their implementation, enumerating some of the various shortcomings of archival crawlers' functionality shows that there is room for improvement. A more comprehensive test of browsers' other features on archiving tools and services would likely surface other areas where what a user expects to be captured into the archives is not.

[9]OpenWayback version 2.0.0BETA2, the latest SNAPSHOT, built from source

6. REFERENCES

[1] *Information and documentation - WARC file format*, Std. ISO 28 500, 2009.

[2] G. Mohr *et al.*, "Introduction to Heritrix, an Archival Quality Web Crawler," in *Proc. International Web Archiving Workshops*, September 2004.

[3] Archive Team. (2013) Wget. [Online]. Available: http://www.archiveteam.org/index.php?title=Wget

[4] M. Kelly and M. C. Weigle, "WARCreate - Create Wayback-Consumable WARC Files from Any Webpage," in *Proc. ACM/IEEE Joint Conference on Digital Libraries*, June 2012, pp. 437–438.

[5] Internet Archive. (2013) warcprox - WARC writing MITM HTTP/S proxy. [Online]. Available: https://github.com/internetarchive/warcprox

[6] J. F. Brunelle *et al.*, "The Impact of JavaScript on Archivability," *Submitted for Publication.*

[7] M. Kelly *et al.*, "On the Change in Archivability of Websites Over Time," in *Proc. Int. Conf. on Theory and Practice of Digital Libraries*, September 2013, pp. 35–47.

[8] J. F. Brunelle. (2013) Zombies in the Archives. [Online]. Available: http://ws-dl.blogspot.com/2012/10/2012-10-10-zombies-in-archives.html