Client-side Reconstruction of Composite Mementos Using ServiceWorker

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

ABSTRACT

We use the ServiceWorker (SW) web API to intercept HTTP requests for embedded resources and reconstruct Composite Mementos without the need for conventional URL rewriting typically performed by web archives. URL rewriting is a problem for archival replay systems, especially for URLs constructed by JavaScript; frequently resulting in incorrect URI references. By intercepting requests on the client using SW, we are able to strategically reroute instead of rewrite. Our implementation moves rewriting to clients, saving servers' computing resources and allowing servers to return responses more quickly. Our experiments show that retrieving the original instead of rewritten pages from the archive reduces time overhead by 35.66% and data overhead by 19.68%. Our system prevents Composite Mementos from leaking the live web while being easy to distribute and maintain.

CCS CONCEPTS

Information systems → Digital libraries and archives; World Wide Web;

KEYWORDS

ServiceWorker, Memento, Composite Memento, Web Archive, Archival Replay

ACM Reference format:

DOI: 00.000/000_0

1 INTRODUCTION

ServiceWorker (SW) is a new client-side web API [11] that can be used to intercept all the network requests for embedded resources originating from web pages in its scope. A Composite Memento [2] is an archived HTML page along

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

JCDL ’17, Toronto, Ontario, Canada
© 2017 Copyright held by the owner/author(s). Publication rights licensed to ACM. 000-0000-00-000/00/00 ...$15.00
DOI: 00.000/000_0

Figure 1: Live Ad Zombie Leaks into Archived Page
with all the embedded resources that are necessary to render the page correctly. Web archival replay systems rewrite embedded resource references to point to their archival versions e.g., a reference to external.example.net/logo.png is changed to archive.example.org/<datetime>/external.example.net/logo.png.

We use SW API to reconstruct Composite Mementos from the originally captured data without any such URL rewriting. By intercepting requests on the client-side we are essentially rerouting instead of rewriting. Rerouting is an effective mechanism to block live web leakage, or “zombies” that might happen after executing potential JavaScript (JS), otherwise not discoverable by static analysis. For example, in Figure 1 the page was archived on September 16th, 2008, but when observed on September 28, 2012, it pulled in an ad from the live web which seems to provide a prophetic look at the 2012 presidential candidates [5]. Client-side rerouting also saves bandwidth by allowing necessary rewriting of the content on the client side, such as to include archival banners, hence, there is no need to send extra data with each HTML response. Client-side solutions such as Memento for Chrome1 involve installing a browser add-on, which limits the adoption by users and adds the burden of maintaining the add-on while only available for Google Chrome users. Our exploratory technique works well when SW is supported. However, a server-side fallback is necessary for production usage to avoid the risk of zombies and broken references when SW is not supported.

Our experiments show that retrieving the original instead of rewritten pages from the Internet Archive (IA) reduces time overhead by 35.66% and data overhead by 19.68%. Our system prevents Composite Mementos from zombies while being easy to distribute and maintain. It is a lightweight and portable system that can be used with any Memento server such as a web archive or a Memento aggregator.

1http://bit.ly/memento-for-chrome
2 BACKGROUND

A memento is a timestamped representation of a web resource identified by a URI-M [13]. A web page is often comprised of a base HTML page and various embedded resources such as images, stylesheets, JS, fonts, and other media (each with an independent URI) that are necessary to render the page correctly. A Composite Memento is a memento of a base HTML page and mementos of all corresponding embedded resources around the same time as the base page to render the page the way it looked in the past [2].

References to these embedded resources can be absolute URLs, absolute paths, or relative paths in attributes like href or src. A relative path is relative to the base URL, either explicitly specified using <base> element or implicitly derived from the URL of the current page. In order to dereference a resource over HTTP the client must resolve its reference to an absolute URL. To do so, a client may use various pieces of information to accomplish this such as domain’s root URL, origin’s base URL, and the path of the resource. When the domain name or the root path of the site changes, some of these references may resolve incorrectly.

When an archived page is replayed, both the domain name and the root path are changed. To correctly route all the resource references, web archival replay systems (such as OpenWayback2, PyWB3, Memento Reconstruct4) perform static analysis of HTML pages, CSS, and JS files to rewrite href and src attributes in a way that points to their archival versions. Alternatively, a proxy can be configured or a browser add-on can be installed to reroute requests appropriately.

ServiceWorker is a new client-side web API that acts as a proxy between a web application and the web server. It can intercept all the network requests originating from web pages in its scope. A SW is installed in the form of a JS file loaded from a path on a host like any other resource. Any resource under that path of the host is in the scope of the SW. Subsequent requests originating from any web page in scope are also intercepted by the SW, even if the resource resides on an external domain. This API is often used to make web applications accessible offline using client-side caching, background data synchronization, and push notifications. We use this API to reconstruct Composite Mementos from the originally captured data without any URL-rewriting. Our technique can be utilized by other URI-based services such as web annotations. SW is still in the working draft phase, but already implemented by many major browsers.

3 RELATED WORK

Ainsworth introduced a framework for assessing temporal coherence of a Composite Memento [1]. A temporal violation may occur due to poor archiving or poor playback. The latter is the focus of our work.

Kelly created the state-of-the-art Acid test suite for archival systems [10], both capture and replay. While Kelly’s Archival Acid test was focused on evaluating the capture quality and pixel-perfect rendering, it does not cover all cases of how a network request can be initiated and where the responses come from. Our focus is mainly on the network activity to make sure that each response is coming from the appropriate archive and there are no zombies. We measure rerouting of all the requests originated explicitly, implicitly, or after any interaction with the page.

Measuring the quality of the crawling or capture is beyond the scope of this work. Brunelle has done exhaustive research about the impact of missing resources [6] and capturing the deferred representation [7] (rendered state of a page after some interactivity or JS execution). Our focus is to load those resources properly if present in the archive.

Jones proposed using the Prefer HTTP header with the existing Memento protocol [13] to request the unaltered (raw) archived web content [8]. Current practice is to use a URL based technique (appending id to the datetime digits) which has been a little-known feature of the Wayback Machine.

Sanderson discussed the challenges and solutions discovered for implementing the Memento protocol in a variety of environments, including MementoFox (a Firefox add-on), a plug-in for Internet Explorer, and an Android-based browser [12].

4 METHODOLOGY

Figure 2 illustrates the workflow of our reconstruction method. Suppose a user visits archive.example.org, which installs a SW reconstructive.js in the user’s web browser under the root of the domain. This SW is detached from the page and persists in the browser independently to watch all network activities originated under its scope. The user then loads a copy of www.example.com from archive.example.org that was archived on January 26, 2017. This memento has an embedded image that points to an external domain external.example.net. The browser would have sent the request to the external domain, but due to the presence of the SW, it will be intercepted. The SW reconstructive.js gets access to the request object that contains a referer header (the URL of the originating page that is shown in the address bar of the browser). Based on the available information (e.g., the datetime of the originating page) the SW can create a new request, load the corresponding resource from the archive, make any modification in the response (if needed, such as adding banners in HTML pages), and return the response to the page for rendering. To maintain the same-origin boundary for external resources that might load more resources, such as iframe source or CSS, we first issue a 302 redirect to the corresponding URI-M locally from the SW. All the logic of exclusions and rerouting is present in the reconstructive.js file, which can be updated on the server when needed. When the user visits the home page of the archive, the corresponding SW will be updated automatically. Every request originating from our SW has a custom header X-ServiceWorker: reconstructive.js:v1 so the server can decide whether it needs to return a server-side rewritten response as a fallback for unsupported clients or attempt to install/update the SW.

Our goal in this exploratory work was to effectively reconstruct a Composite Memento with zero rewriting. However,
using client-side rewriting we can add other useful features such as an archival banner with metadata and toolbar, pointing hyperlinks to the archived version, or JS handlers adding custom behavior when hyperlinks are followed.

5 IMPLEMENTATION

The reference implementation of the reconstructive.js is being used by our InterPlanetary Wayback archival replay system [3, 9]. Additionally, we use the reconstructive.js in our Memento aggregator, MemGator [4], to facilitate cross-archive Composite Memento reconstruction. To deal with the Cross-Origin Resource Sharing (CORS) [14] restrictions we added Memento Proxy feature in MemGator. We open-sourced reconstructive.js⁵, IPWB⁶, and MemGator⁷.

6 EVALUATION

To quantify the benefits of client-side rerouting we collected 500 domains⁸ and sequentially fetched their mementos closest to January 26, 2017 (using 20170126000000 as the 14-digit datetime string) in both original and rewritten forms from the Internet Archive (IA). The rewritten responses took a total of 26.97 minutes while the original versions took only 19.88 minutes. We repeated this three times and found that the time variance less than 30 seconds in successive iterations. Hence, there is an overhead of 35.66% in the mean response time with the server-side rewriting and banner inclusion.

Among the 500 requests there were 33 403 Forbidden, five 404 Not Found, and one 504 Gateway Time-out. There were 19 cases where the original archived memento was a 302 Found with a Location header containing absolute path, which if not rewritten by the server, would resolve to an irrelevante or non-existing resource. There were 79 301 Moved Permanently responses for which IA shows a splash page for few seconds then redirects. We eliminated all these 137 problematic responses and evaluated the data sizes for the two cases. Original responses were 44.98 MB total while the rewritten responses were 50.74 MB accumulated, hence, a 12.80% overhead. If we also include the 3.09 MB of 79

5https://github.com/oduwsdl/reconstructive
6https://github.com/oduwsdl/ipwb
7https://github.com/oduwsdl/memgator
8Moz’s list of the top 500 domains on the web ranked by the number of linking root domains on January 26, 2017. https://moz.com/top500.

5 IMPLEMENTATION

The reference implementation of the reconstructive.js is being used by our InterPlanetary Wayback archival replay system [3, 9]. Additionally, we use the reconstructive.js in our Memento aggregator, MemGator [4], to facilitate cross-archive Composite Memento reconstruction. To deal with the Cross-Origin Resource Sharing (CORS) [14] restrictions we added Memento Proxy feature in MemGator. We open-sourced reconstructive.js⁵, IPWB⁶, and MemGator⁷.

6 EVALUATION

To quantify the benefits of client-side rerouting we collected 500 domains⁸ and sequentially fetched their mementos closest to January 26, 2017 (using 20170126000000 as the 14-digit datetime string) in both original and rewritten forms from the Internet Archive (IA). The rewritten responses took a total of 26.97 minutes while the original versions took only 19.88 minutes. We repeated this three times and found that the time variance less than 30 seconds in successive iterations. Hence, there is an overhead of 35.66% in the mean response time with the server-side rewriting and banner inclusion.

Among the 500 requests there were 33 403 Forbidden, five 404 Not Found, and one 504 Gateway Time-out. There were 19 cases where the original archived memento was a 302 Found with a Location header containing absolute path, which if not rewritten by the server, would resolve to an irrelevante or non-existing resource. There were 79 301 Moved Permanently responses for which IA shows a splash page for few seconds then redirects. We eliminated all these 137 problematic responses and evaluated the data sizes for the two cases. Original responses were 44.98 MB total while the rewritten responses were 50.74 MB accumulated, hence, a 12.80% overhead. If we also include the 3.09 MB of 79

5https://github.com/oduwsdl/reconstructive
6https://github.com/oduwsdl/ipwb
7https://github.com/oduwsdl/memgator
8Moz’s list of the top 500 domains on the web ranked by the number of linking root domains on January 26, 2017. https://moz.com/top500.
client side as the server returns the original archived content without any modification. It has many of the same features that are provided by browser add-ons such as MementoFox (deprecated). However, maintaining separate add-ons for each popular browser and keeping them up-to-date is a difficult task. Encouraging users to install add-ons is another barrier to adoption. In contrast, a SW is easier to maintain, update, and distribute as it is a JS file hosted on the web server and updates in clients’ browsers automatically. Additionally, the same code works in many different browsers.

The support for SW was introduced in Chrome 40 and Opera 27 in January 2015. Firefox 44 added support in January 2016. Firefox does not support SW in private browsing mode. For security reasons SW only runs over HTTPS. There is 61.55% support globally as of January 31, 2017.

### 7 FUTURE WORK

We would use the `Prefer` header for content negotiation [8] when it is supported by web archives. We would like to add a customizable archival banner as part of the client-side rewriting using HTML5 Web Components [11] to avoid any style conflicts with the page. Ability to verify the fixity of the archived content would be another valuable addition.

SW is an emerging technology, which has many opportunities to create useful services and tools. We would like to investigate the possibility of a variation to our SW that can be used by webmasters in personal sites, wikis, blogs, or other content management systems to use web archives as a fallback cache when embedded resources are gone missing from the live web. Missing resources are a big issue in social platforms like forums where users post content, including external media, that make the thread less useful when gone. Additionally, we can add the ability to request web archives to capture resources that are loaded by any user of a page from a domain that hosts our SW. These use cases can encourage utilization of web archives by webmasters and cause fewer 404s for users.

### 8 CONCLUSIONS

We utilized the ServiceWorker web API to explore intercepting requests and reconstruct Composite Mementos without the need for conventional URL rewriting of web archives. We developed a prototype implementation and used it as the replay system for an IPFS based standalone web archive and for a Memento aggregator for cross-archive reconstruction. We created a test suite to evaluate different archival replay systems to measure the rerouting quality where our implementation passes all the cases. Our experiments show that retrieving the original instead of rewritten pages from the archive reduces time overhead by 35.66% and data overhead by 19.68%. Our system prevents Composite Mementos from zombies while being easy to distribute and maintain. It is a lightweight and portable system suitable for any Memento server.

### 9 ACKNOWLEDGEMENTS

This work is supported in part by NSF grant III 1526700.

### REFERENCES


