Site	Test	DOM	Img	Net
www.facebook.com	basic	0%	Pass	2%
www.facebook.com	interactive	1%	Pass	4%
docs.google.com	basic	0%	Pass	9%
docs.google.com	interactive	0%	Pass	0%
en.wikipedia.org	basic	0%	Pass	0%
en.wikipedia.org	interactive	0%	Pass	0%
sfbay.craigslist.org	basic	0%	Pass	0%
sfbay.craigslist.org	interactive	N/A	N/A	N/A
www.cnn.com	basic	0%	Pass	8%
www.cnn.com	interactive	5%	Pass	9%
www.amazon.com	basic	1%	Pass	0%
www.amazon.com	interactive	0%	Pass	6%

Table 1: Results from testing the automatic DSI policy using Alhambra. Percentages indicate the measured difference between replay with and without policy, zero percent is the ideal case. For image comparison, a pass indicates that the SIFT matching indicates they are the same page.

pick simple pages at each site and tests to make sure policy does not impede viewing the page. Interaction involves user input, such as typing a message or editing a document and includes navigation from one page to another. We use recorded browsing sessions and use the Alhambra testing framework to revisit each of the pages in the recorded session.

5.1.1 Web application tests

Facebook. As our first case study we choose Facebook, a popular social networking site. Facebook allows users to create profiles, applications, and in general edit the content that Facebook profiles display. To test basic functionality using Facebook we use sessions that login and browse the site and perform basic actions such as accessing profile pages. Our test for interacting with the Facebook application performs a few common actions. First, we change the status message for our Facebook account. Then we post comments to friends' updates and check notifications.

Google. Google offers a number of services beyond the search portal that offer functionality similar to that of desktop applications. We login to Google and test the Google Docs document editor. To test basic functionality we open a document previously saved. Our test for interacting with Google Docs demonstrates the ability to write a text document. In our test, we create a new document and compose a short paragraph, save it and then use the Google provided UI to format text in the paragraph.

Wikipedia. Wikipedia is an online encyclopedia that allows anyone to edit and contribute to articles. Basic use of Wikipedia is simple and involves accessing Wikipedia articles. We access the special random page that directs the browser to a new random page at `en.wikipedia.org` as well as loading the main page. Our test for interacting with Wikipedia tests the ability to edit an article. We choose a random document, edit the contents and preview the modified document.

Craigslist. Craigslist allows users to view and post ads with very simple markup and formatting. Craigslist has different sites based on geographic location and is similar to newspaper classified ads. The test for basic functionality at Craigslist uses the browser to browse advertisements in the San Francisco bay area (`sfbay.craigslist.org`). To test interaction with Craigslist, we search for and then post an advertisement.

CNN. CNN is a popular news source that hosts free online content and provides different types of content such as movies, images and articles. The test for basic functionality at CNN involves accessing news articles by following a link on the main page. CNN is not quite an interactive website. For interactive test, we look up local news.

Amazon. Amazon.com is online store that sells virtually anything. For testing basic functionality at Amazon.com we browse to a product page. Testing interaction at Amazon.com involves searching for a product and initiating the checkout process, though to prevent many unwanted items arriving in two days or less, we do not complete the checkout process.

5.1.2 Policy compatibility

The remainder of this section presents the compatibility results for our policies using the user-generated browsing sessions. The compatibility results for each site tested are presented in Table 1. For each comparison we provide the percent different from an unmodified replay of the identical page. For visual differences, we compare the rendered web pages using the SIFT algorithm and compare the number of matched keypoints against the values for a set of training data generated by the same sites. For DOM differences, we calculate the edit distance between the two DOM trees. For network differences, we divide the number of omitted network requests by the total number of network requests. The comparisons are generated once the page has reached a steady state and after the interactive session has completed.

DOM-based XSS prevention. The taint-tracking policy that prevents DOM-based attacks (Section 4.2) does not cause any compatibility problems on these pages, and we have additionally tested this policy for the top 100 sites ranked by Alexa [2]. Table 2 presents the results of testing this policy. On the top 100 sites we also see no incompatibilities as a result of enforcing our taint-tracking policy; however, our policy is configured to generate warnings when tainted data is passed to the HTML parser and could result in a DOM-based XSS attack. Fourteen sites generate warnings, one of which we constructed an attack to exploit the vulnerability found. This vulnerability is in a popular website[2] and can be used to inject arbitrary JavaScript into the page. After generating a sample attack we confirmed that our policy prevents the attack.

Automatic DSI. The automatic DSI policy (Section 4.3) limits the structure of pages and will modify some pages slightly by removing the elements that violate the automatic DSI policy. For each of the browsing sessions tested in Table 1 the automatic DSI policy does not introduce any rendering incompatibilities and overall document tree differences are less than 5%. The network has slightly higher differences, in the case of `www.cnn.com` 9% of the network requests differ after the automatic DSI policy is introduced. The network connections that differ due to the policy are due to some included files conflicting with the automatic DSI policy and do not cause the page to function differently. The results of the interactive test for `sfbay.craigslist.org` show incompatibilities. The session being revisited is unable to complete since the policy introduces slight differences in the rendering of the page. These small visual differences result in the final user action (clicking to post an advertisement) missing the button on the page.

Effect of page updates. We also use Alhambra to test the effect of page updates on the compatibility of our automatic DSI policies as the document structure of a page can change due to site updates or other server-side configurations. We use `archive.org` and record browsing sessions that visit pages from over one year ago. Each of the browsing sessions is then revisited to determine the impact on page updates. Unlike our previous tests, we do not have in-

[2]We have disclosed this vulnerability to the site administrators but have not been contacted back.