BenchLab: Benchmarking with Real Web Applications and Web Browsers

Emmanuel Cecchet, Veena Udayabhanu, Timothy Wood, Prashant Shenoy
University of Massachusetts Amherst, USA
{cecchet,veena }@cs.umass.edu
{twood,shenoy}@cs.umass.edu

Fabien Mottet, Vivien Quema
INRIA Rhone-Alpes, France
{first.last}@inria.fr

Guillaume Pierre
Vrije University, Netherland
gpierre@cs.vu.nl

Abstract

Popular benchmarks such as TPC-W and RUBiS that are commonly used for evaluation by the systems community are no longer representative of modern Web applications. Many of these benchmarks lack the features such as JavaScript and AJAX that are essential to real Web 2.0 applications. Further, traditional benchmarks rely on browser emulators that mimic the basic network functionality of real web browsers but cannot emulate their more complex interactions. Rather than proposing a new benchmark with a web application and browser emulators that try to approximate real applications, we propose to use *real browsers* with *real applications* and *datasets*. We have rebuilt the Wikipedia software stack with multiple real datasets (Wikibooks, Wikipedia in different languages) and collected real traces from the Wikimedia foundation. We propose BenchLab, an open source framework that allows replaying these real traces using real web browsers (Firefox, IE, Chrome) deployed anywhere on the Internet. We provide virtual machines containing applications, databases and web browsers for researchers to experiment with Internet scale benchmarking of real applications using private or public clouds.

1. Introduction

The research community has relied on open-source benchmarks such as TPC-W [6] and RUBiS [2] for a number of years; however these benchmarks are outdated and do not fully capture the complexities of today's Web 2.0 applications as shown in Table 1 (compare RUBiS to eBay.com or TPC-W to amazon.com). To address this limitation, a number of new benchmarks have been proposed, such as TPC-E, SPECweb2009 or SPECjEnterprise2010. However, the lack of open-source or freely available implementations of these benchmarks has limited their use to commercial vendors. CloudStone [4] is a recently proposed open-source cloud/web benchmark that addresses some of the above issues; it employs a modern Web 2.0 application architecture. However, Cloudstone does not capture or emulate client-side JavaScript or AJAX interactions, an aspect that has implications on the server-side load.

Benchmark	HTML	CSS	JS	Images	Total
RUBiS	1	0	0	1	2
eBay.com	1	3	3	31	38
TPC-W	1	0	0	5	6
amazon.com	6	13	33	91	141
CloudStone	1	2	4	21	28
facebook.com	6	13	22	135	176
wikibooks.org	1	19	23	35	78
wikipedia.org	1	5	10	20	36

Table 1. Browser generated requests per type when accessing the home page of benchmarks or real sites.

In this poster, we propose BenchLab, an open testbed for realistic Web benchmarking that uses real Web applications, datasets, traces and real Web browsers.

2. BenchLab

BenchLab provides Virtual Appliances of the Wikipedia software stack [9][10] along with real database dumps of various Wikipedia web sites. Using modern virtualization technology simplifies the deployment and configuration of these server applications in laboratory clusters and on public cloud servers.

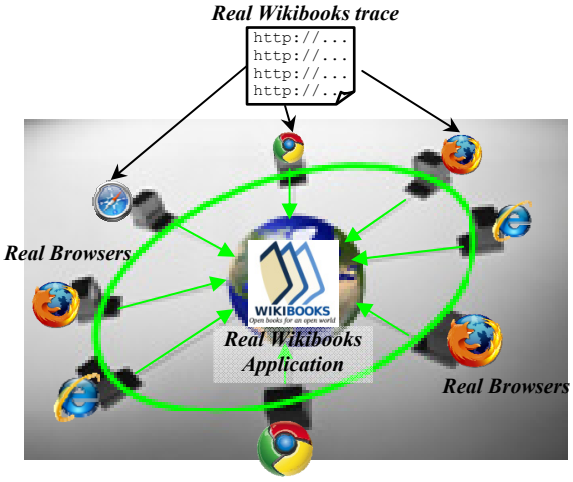


Figure 1. Wikibooks experiment with BenchLab.

We also provide the real traces [7] from the Wikimedia foundation to replay the authentic Wikipedia workload from the date where the database snapshot was taken.

We design BenchLab to use real web browsers, in conjunction with automated tools, to replay existing web traces as depicted in Figure 1.

BenchLab supports web performance benchmarking "at scale" by leveraging modern public clouds---by using a number of cloud-based client instances, possibly in different geographic regions, to perform scalable load injection. Cloud-based load injection is cost-effective, since it does not require a large hardware infrastructure and also captures Internet round-trip times.

3. Implementation

BenchLab is implemented using open source software and is also released as open source software for use by the community. The latest version of the software and documentation can be found on our web site [1].

All components of BenchLab are implemented in Java for portability. The web browser load injection is based on the integration of Webdriver and Selenium [3]. It supports Firefox, Internet Explorer and Chrome on almost all platforms where they are available (Linux, Windows, MacOS). iPhone and Safari support is experimental as well as Webkit based browsers for Android. On Linux machines that do not have an X server environment readily available, we use X virtual frame buffer (Xvfb) to render the browser in a virtual X server. This is especially useful when running clients in the cloud on machines without a display.

We use the HTTP Archive Format (HAR) v1.2 [5] for storing traces and performance results from Web browsers as illustrated on Figure 2.

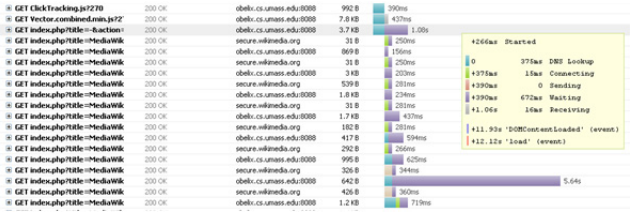


Figure 2. Page loading time details in HAR format

We have built Xen Linux virtual machines with Firefox to use on private clouds. We also built Amazon EC2 AMIs for both Windows and Linux with Firefox, Chrome and Internet Explorer (Windows only for IE). These AMIs are publicly available.

4. Preliminary Results and Challenges

Building BenchLab has exposed many challenges in using real applications and browsers for benchmarks.

- Even the small Wikipedia sites that we have experimented with such as the dawiki (Denmark, 700MB) and nlwiki (Netherland 3.3GB) are larger than existing web benchmarks. The largest Wikipedia database currently exceeds 5TB and special tools must be developed to load such large datasets in a reasonable amount of time.
- Multimedia content is not publicly available due to possible copyright issues, so we had to integrate multimedia content generators that reproduce media (images, audio, video, …) similar to the original.
- The Wikipedia trace [8] includes all sites of the Wikimedia foundation but only 10% of requests are logged. This requires extensive processing to rebuild a realistic trace for a specific site.

Our preliminary results illustrate that existing web client emulators do not authentically generate requests and may not place a realistic load on the server.

- We have compared simple HTTP replays (a la httperf) running in a local cluster to deploying real Web browsers in different Amazon EC2 regions over the globe. While we need to investigate these results further, realistic load injection with real Web browsers seems to have a significant impact on application server resource usage.
- Even small client behavior differences such as the typing speed in search fields can generate a different amount of requests to servers as many applications provide suggestion lists based on the current keystrokes.

We believe that BenchLab offers many new opportunities for realistic benchmarking and system testing. On the server side, many configurations of replication (load balancer, application server, database), caching (e.g. memcached) and system (operating system, virtualization, networking) can be tested. At Internet scale, BenchLab can be used to evaluate Content Delivery Networks (CDN), reproduce flash crowds or Slashdot effects, disaster recovery solutions with wide area network replication and failover mechanisms. Security is also an open challenge at that scale. Finally by offering a repository for applications, datasets, traces, experiment definitions and results, BenchLab can offer a platform for easy experiment reproducibility that has been crucially lacking to the community up to now.

5. References

[1] BenchLab - http://lass.cs.umass.edu/projects/benchlab/

[2] RUBiS web site – http://rubis.ow2.org.

[3] Selenium - http://seleniumhq.org/

[4] Cloudstone – http://radlab.cs.berkeley.edu/wiki/Projects/Cloudstone

[5] HAR v1.2 - http://groups.google.com/group/http-archive-specification/web/har-1-2-spec

[6] TPC-W Benchmark, ObjectWeb implementation, http://jmob.objectweb.org/tpcw.html

[7] G. Urdaneta, G. Pierre and M. van Steen – *Wikipedia Workload Analysis for Decentralized Hosting* – Elsevier Computer Networks, vol.53, July 2009.

[8] WikiBench - http://www.wikibench.eu/

[9] Wikibooks – http://www.wikibooks.org

[10] Wikipedia – http://www.wikipedia.org

BenchLab

http://lass.cs.umass.edu/projects/benchlab/

Benchmarking with Real Web Applications and Web Browsers

Emmanuel Cecchet, Veena Udayabhanu, Timothy Wood, Prashant Shenoy	Fabien Mottet, Vivien Quema	Guillaume Pierre
University of Massachusetts Amherst, USA {cecchet,veena }@cs.umass.edu {twood,shenoy}@cs.umass.edu	INRIA Rhone-Alpes, France {first.last}@inria.fr	Vrije University, Netherland gpierre@cs.vu.nl

Web Applications have changed, not Benchmarks

- ❑ Web interactions too complex to emulate
  - HTML 1.1, CSS, images, flash, HTML 5…
  - Httperf does not execute Javascript, AJAX
  - WAN latencies, caching, Content Delivery Networks…
- ❑ Real Web applications
  - Rich client interactions and multimedia content
  - Replication, caching…
  - Large databases (few GB to multiple TB)

Benchmark	HTML	CSS	JS	Multimedia	Total
RUBiS	1	0	0	1	2
eBay.com	1	3	3	31	38
TPC-W	1	0	0	5	6
amazon.com	6	13	33	91	141
CloudStone	1	2	4	21	28
facebook.com	6	13	22	135	176
wikibooks.org	1	19	23	35	78
wikipedia.org	1	5	10	20	36

Number of interactions to fetch the home page of various web sites and benchmarks

Load injection using real Web browsers

Firefox on Linux, Windows and Mac OS X

Chrome on Linux, Windows and Mac OS X

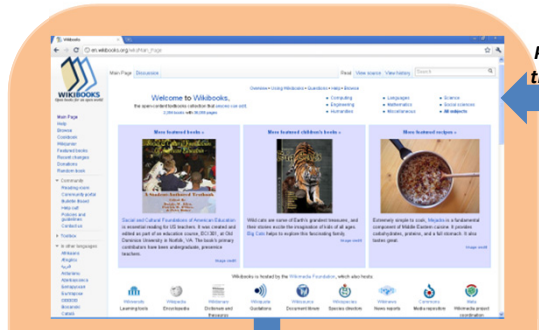
Internet Explorer on Windows

WebKit on Linux, Windows, Mac OS X, iPhone and Android

Standard Benchmarks also available
- RUBiS
- CloudStone
- TPC-W

Real Applications and Workloads

- ❑ Wikimedia foundation Wikis
  - Wikipedia (different languages)
  - Wikibooks
- ❑ Real database dumps (up to 6TB)
- ❑ Multimedia content
  - Images, audio, video
  - Generators (dynamic or static) to avoid copyright issues
- ❑ Real Web traces from Wikimedia
- ❑ Packaged as Virtual Appliances
- ❑ *Test your own applications!*



Web Traces

- HTTP Archive (HAR) format
- Apache httpd recorder for easy capture/replay
- HA Proxy recorder for replicated configurations

- ❑ Upload traces / VMs
- ❑ Define and run experiments
- ❑ Compare results
- ❑ Distribute benchmarks, traces, configs and results

❑ BenchLab Client Runtime (BCR)

- Replay traces in Web browsers
- Multiplatform including headless servers
- Collect detailed response times
- Can record HTML and page snapshots
- Easy deployment in the cloud for Internet scale benchmarks

❑ BenchLab WebApp

- JEE WebApp with embedded database
- Repository of benchmarks and traces
- Schedule and control experiment execution
- Results repository
- ***Can be used to distribute / reproduce experiments and compare results***

❑ A lot to explore…

- Replication: load balancer, application server, database…
- Caching: memcached, CDNs…
- System: operating system, virtualization, networking…
- Multimedia: images, audio, video, flash, HTML 5…
- Workloads: building Internet scale workloads, reproducing flash crowds…
- High availability: WAN replication, failover, disaster recovery…