The Impact of Weights on the Performance of Server Load Balancing (SLB) Systems

Jörg Jung

University of Potsdam
Institute for Computer Science
Operating Systems and Distributed Systems

March 2013

Outline

1 Introduction 3

2 Determine Weights 5

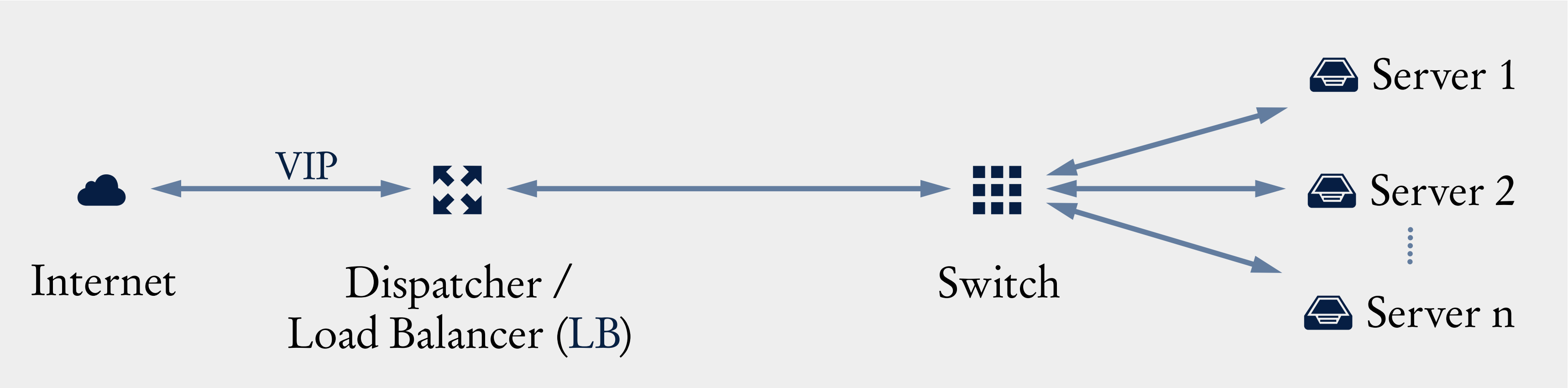
3 Metrics for Benchmarking 7

4 Measurements and Evaluation 13

5 Conclusions 24

1 Introduction

Dispatcher based Server Load Balancing (SLB): scalable, flexible and fault tolerance services



Motivation

Simulations in [Lehmann et al. 2008] confirm impact of incorrectly estimated weights

- Small deviation of 10 % results in significant higher number of dropped requests

Compare algorithms:

- Weighted Round Robin (WRR) and Weighted Least Connection (WLC)

→ Measure the impact of weights on the performance

2

Determine Weights

System administrator *may* run local benchmarks and does an “educated guess”

Factors on Weights

Hardware differences: CPU, Memory, HDD, NIC and PCI bus speed

Software differences: utilized SLB and back end server software

Workload scenarios: which trace characteristics are given

→ Hard to find mappings to set factors into relations

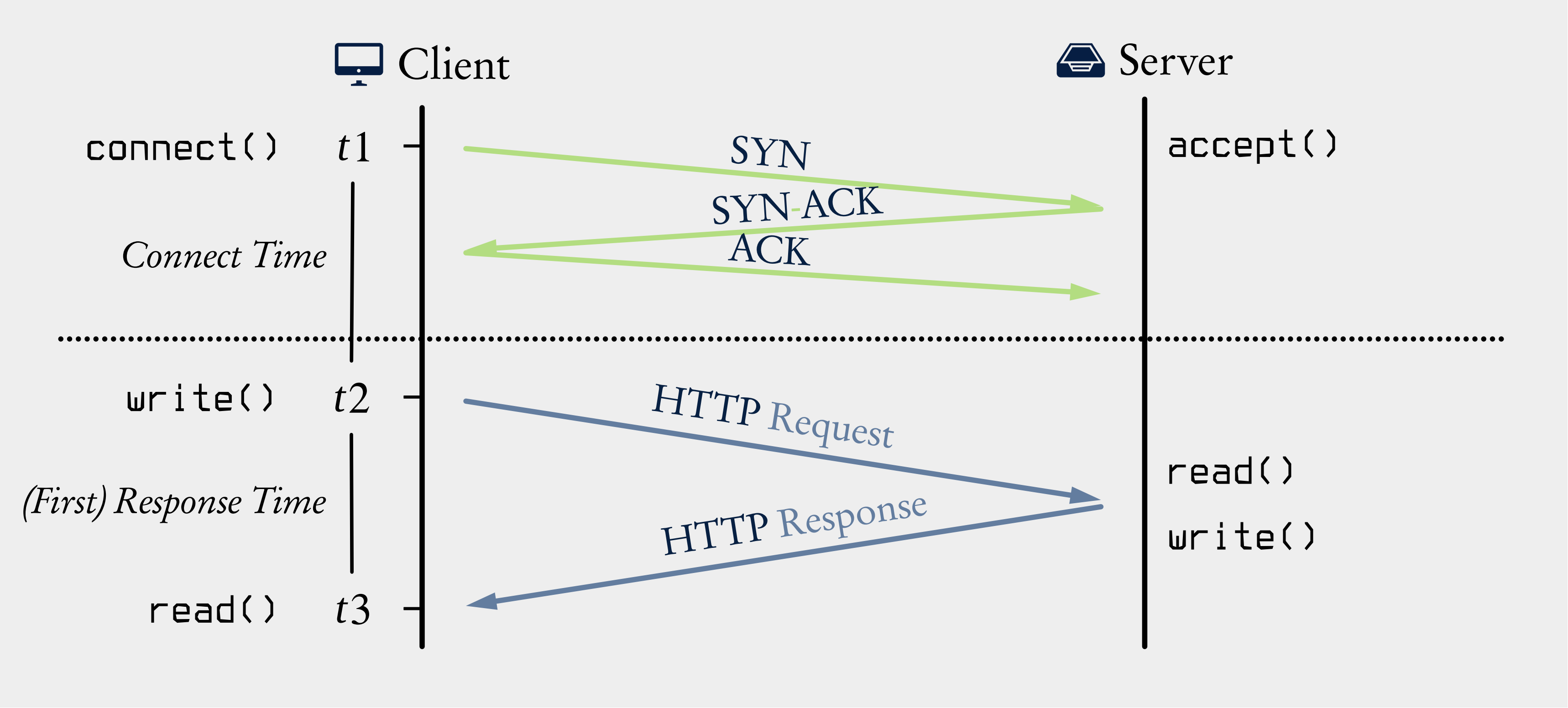
→ Each SLB systems with given setup requires benchmarking

3

Metrics for Benchmarking

SLB algorithm metrics for Internet Service Providers (ISPs) Service Level Agreement (SLA) definitons

Connect Time and *(First) Response Time*



Algorithm Metrics

Connect Time* and *(First) Response Time at client side from start *t*1 until sending start *t*2 and until the receive of the first byte *t*3

Transfer Time the time required to fulfill a request – starts at *t*2 and ends with last byte of the response, usually somewhere past *t*3

Throughput *Connection Throughput*, *Session Throughput* and *Byte Throughput* representing the number of connections, session or bytes per second handled by the application

(Request) Errors* and *Drops on the network layer or service protocol specific due to *Overloaded Servers* or even an *Overloaded Network*

Server Load Balancing Penalty

$$SLB\ Penalty = \left(\frac{response_{mean}}{response_{\max}}\right) \times \left(\frac{request\_error_{mean}}{requests_{total}}\right)$$

mean and max values are calculated from all measurement iterations

errors include network and protocol errors e.g. HTTP 5xx *Server Errors*

→ Created with ISP requirements in mind:

Duration is ignored as not required for SLA definitions

Metrics and Timestamps

Exclude *Connect Time* from *(First) Response Time* as persistent connections are re-used with HTTP/1.1 (keep-alive)

Several time related functions and instructions should be avoided for benchmarking:

`time()` and `gettimeofday()`: both return the so called *Best Guess* of the *Wall Time* which can jump (e.g. influenced by NTP)

RDTSC instruction: With SMP TSC might not be synchronized between cores, might stop or change its frequency when the CPU enters lower power modes, hence probably jump [Brunner 2005]

→ *httperf* [Mosberger et al. 2013] and *http_load* [Poskanzer 2006] use the wrong function: `gettimeofday()`

servload

The web server benchmark *servload*

Load, optionally increase and replay workloads

Use correct timestamp functions and provide metrics

Support for HTTP and DNS

4 Measurements and Evaluation

Measurements in a real SLB environment: *Wikipedia* instance based on a dump from 2008

Dispatcher based SLB scenario: two armed, NAT based and using route path

Comparing WRR and WLC algorithms with different weights

Outcomes and Metrics

Service of the SLB cluster is to answer HTTP requests

Requests can be successfully completed or fail

- Failures on the network connection may result in aborted or incomplete requests and responses
- Fail due to *Overloaded Servers* may result in aborted requests and wrong, incomplete or aborted responses

SLB Penalty is used for comparison

Workload: Wikipedia

Wikipedia instance access traces from 2008 are used as available from [Pierre 2010]

Input workload is from 12. November 2007:

Reduced to the first ten minutes of the log

Filtered and reduced to common upload content (e.g. images) and English requests

Converted to *Common Log Format* as input for *servload*

→ Remaining 1,584,996 requests are reduced to three final traces

Workload: Reduced Traces

Number of requests from the first ten minutes of the *Wikipedia* trace

Factor	Number of Requests
1/32	49,532 requests
1/16	99,063 requests
1/8	198,125 requests

Environment: Hardware

Hostname	***Client LB* and *Web Server 1***
CPU	Dual 1.8 GHz AMD Opteron 244 with 1,024 KByte Cache
GE NIC	Broadcom BCM95704A7
Hostname	***Web Server 2***
CPU	2.8 GHz Intel Pentium 4 with 1,024 KByte Cache
GE NIC	Broadcom BCM5721
Hostname	***Web Server 3***
CPU	1.86 GHz Dual Core Intel Xeon 3040 with 2,048 KByte Cache
GE NIC	Broadcom BCM95754

All machines have 4 GByte memory and GBit links

Environment: Software

3 *Apache* HTTP server 2.2.3 configured to handle 96 Clients at maximum each

LVS LB with ipvsadm 1.24

Client with *servload 0.5* configured to 1,021 concurrent sessions at maximum

OS LB and Servers: *CentOS Linux 5.7* with kernel 2.6.18-274.12.1.el5

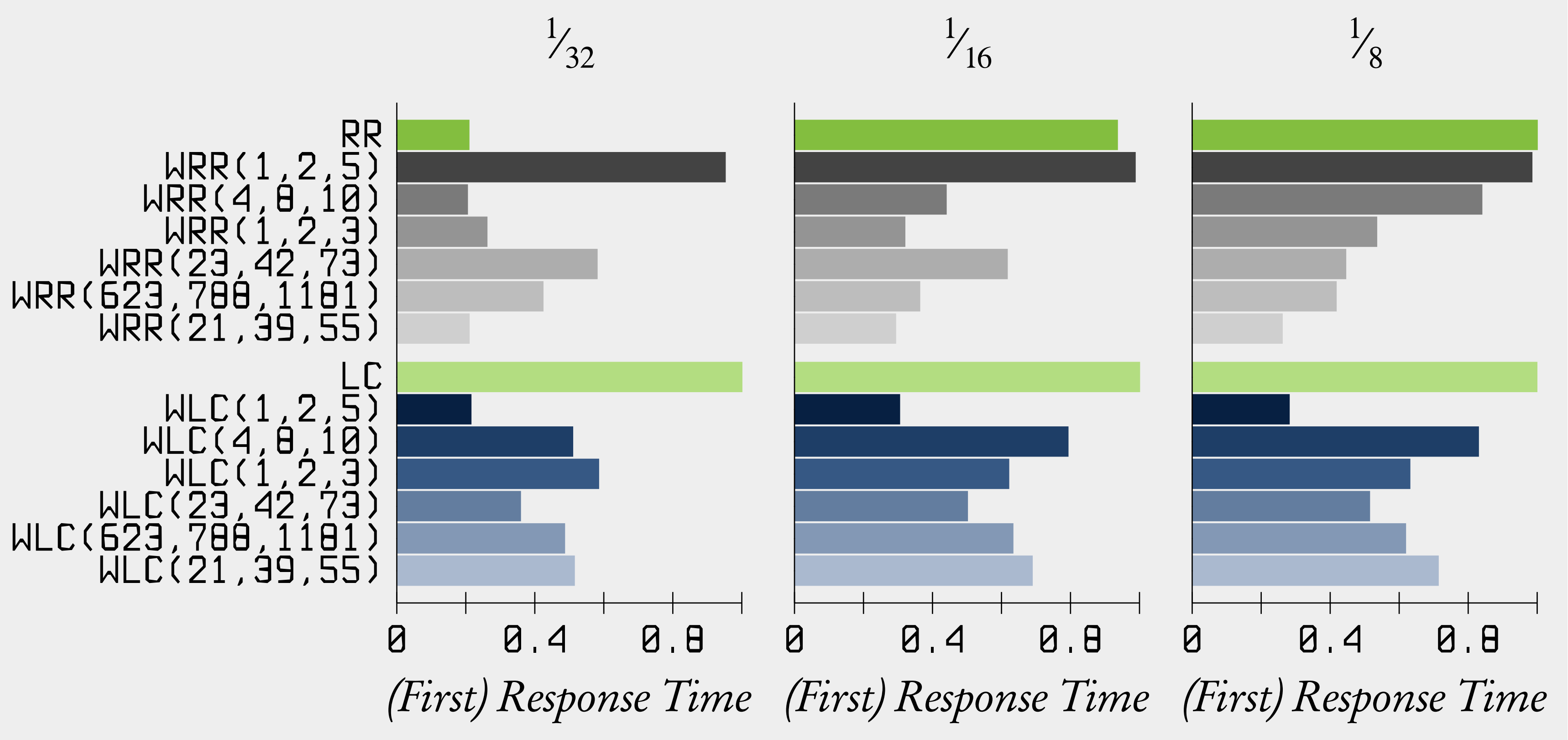
OS Client: *Debian Linux 5.0.10* with kernel 2.626-2-amd64

Monitoring: SNMPv1 requests once a minute from LB to localhost, client and web servers

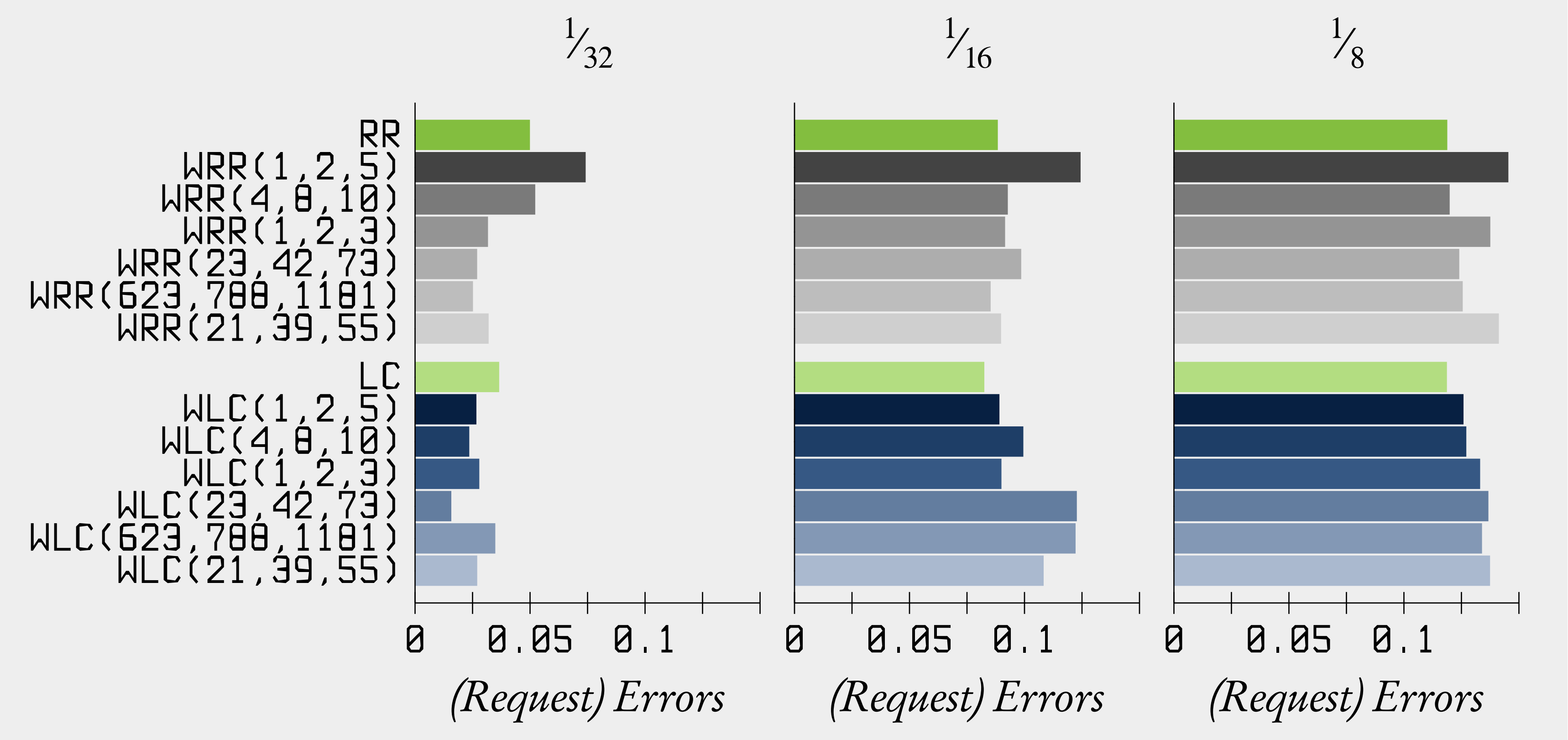
Weights and Scenarios

Web Server 1	*Web Server 2*	*Web Server 3*	**Remark**
1	1	1	RR/LC
2	1	5	
8	4	10	
2	1	3	
42	23	73	
788	623	1181	*Byte-Unixbench*
39	21	55	
Each pass for WRR/WLC: 11 times with $^{1}/_{32}$, $^{1}/_{16}$ and $^{1}/_{8}$			

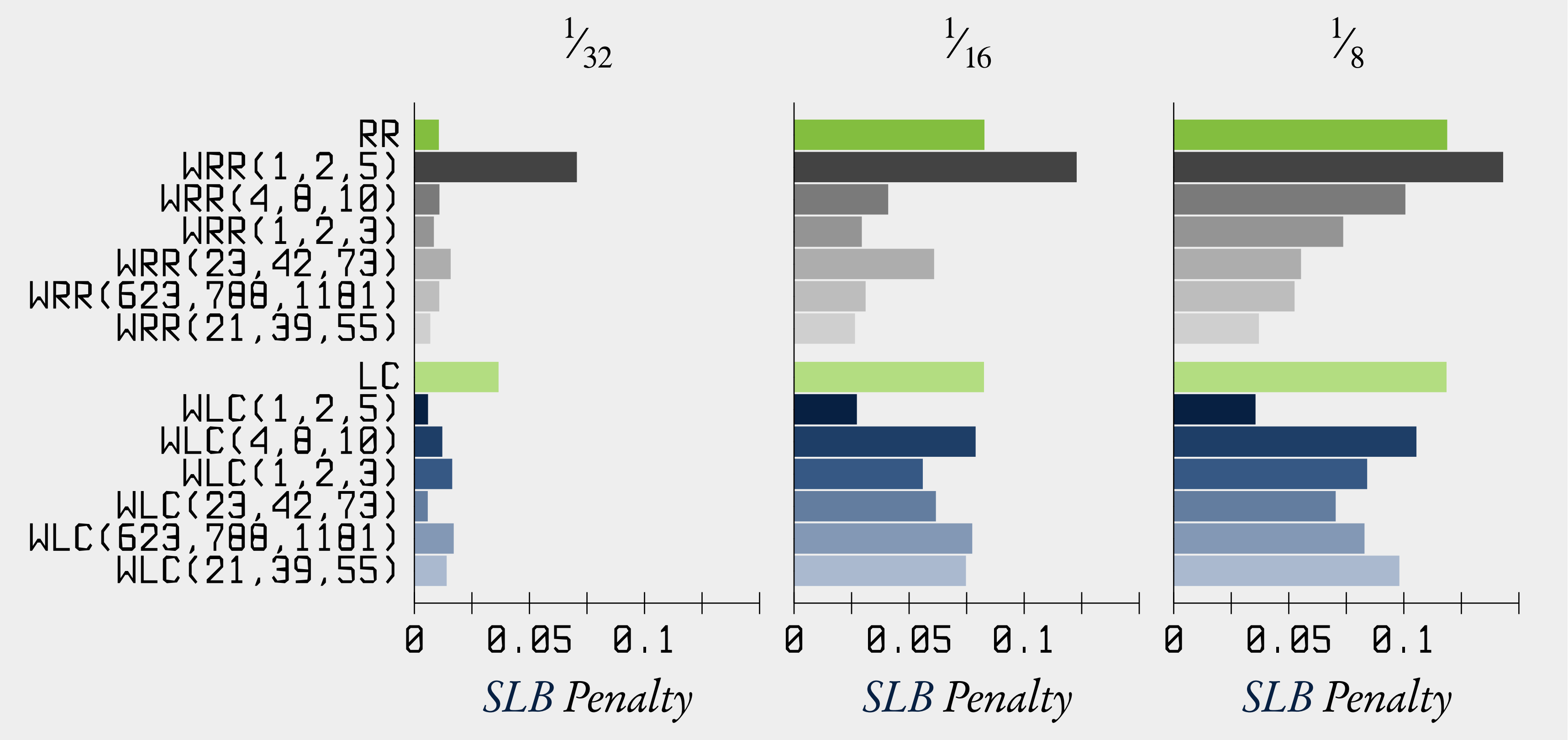
Results: (First) Response Time



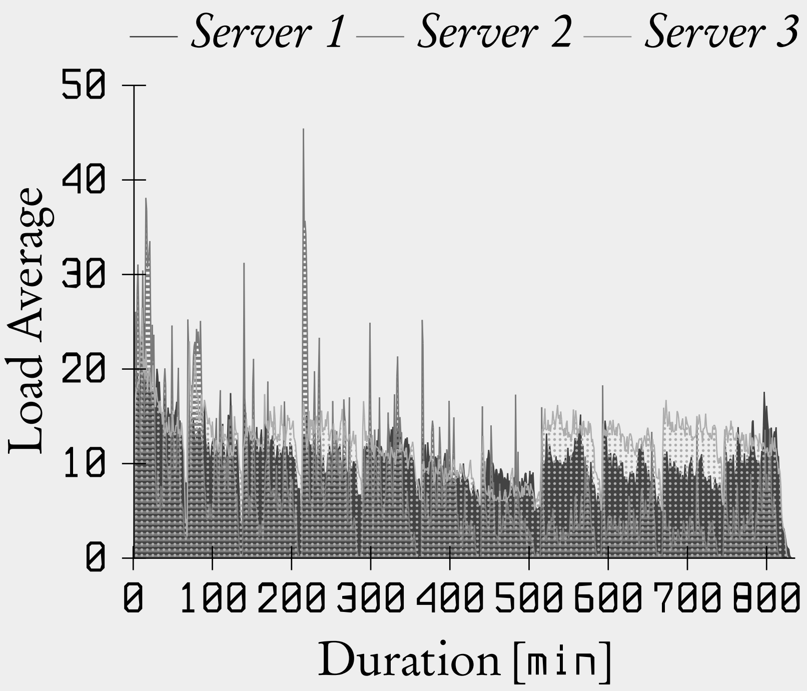
Results: (Request) Errors



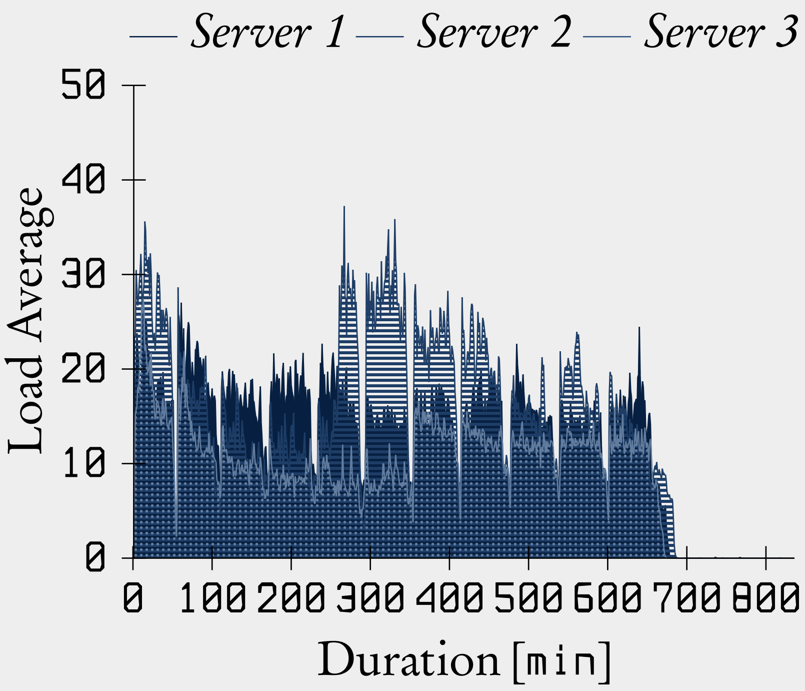
Results: SLB Penalty



Load Averages on Web Servers for ⅛ workload



WRR (21, 39, 55)

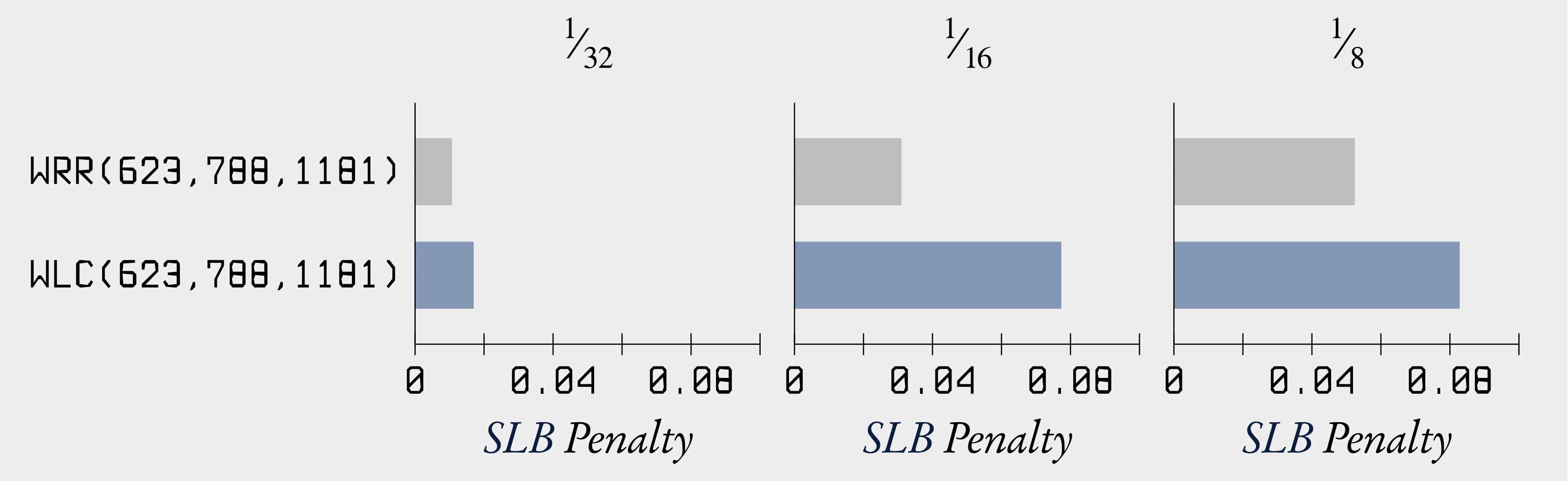


WLC (21, 39, 55)

5

Conclusions

SLB Penalty of WRR and WLC with triple (623, 788, 1181)



Conclusions and Future Work

SLB Penalty introduced

Previous simulations are confirmed

Badly chosen weights may lead to unpredictable substantive worse results

Byte-Unixbench is a good option to determine weights

WRR may be better choice in ISP scenarios and under peak load

Next step: *SALBNET* and self-adapting weights

References

[Brunner 2005] Brunner, Richard. TSC and Power Management Events on AMD Processors, November 2005. URL *http://lkml.org/lkml/2005/11/4/173*. Accessed November 2012

[Lehmann et al. 2008] Lehmann, Janette and Schneidenbach, Lars and Schnor, Bettina and Zinke, Jörg. Self-Adapting Credit Based Server Load Balancing. In Helmar Burkhart, *Proceedings of the IASTED international Conference on Parallel and Distributed Computing and Networks (PDCN)*, pages 55–62. PDCN. ACTA Press. IASTED, Innsbruck, Austria, February 2008. ISBN: 9780889867130, ISBN CD: 9780889867147

[Mosberger et al. 2013] Mosberger, David and Arlitt, Martin and Bullock, Ted and Jin, Tai and Eranian, Stephane and Carter, Richard and Hately, Andrew and Chadd, Adrian. httperf - The httperf HTTP load generator - Google Project Hosting, June 2013. URL *http://code.google.com/p/httperf/*. Accessed February 2013

[Pierre 2010] Pierre, Guillaume. Wikipedia access tracesWikibench, October 2010. URL *http://www.wikibench.eu/?page_id=60*. Accessed May 2012

[Poskanzer 2006] Poskanzer, Jef. http_load, March 2006. URL *http://www.acme.com/software/http_load/*. Accessed June 2012