Evaluation of Hypothesis: Towards Better Architecture for "Web of Things"

Shrinivasan R.P. Patnaikuni
Walchand Institute of Technology
Solapur. India.
Solapur University, Solapur.

Raj B. Kulkarni
Walchand Institute of Technology
Solapur. India.
Solapur University, Solapur.

ABSTRACT
In this paper we study Internet of Things and Web of Things to layout a clear distinction between them and we argue that there is a need for gateway like server in the basic architecture defined for Web of Things. We state some issues which emphasize on need for intermediation. We propose solution by introducing a default gateway in architecture. We test our approach by using Apache bench a web application benchmarking tool to load test the architecture proposed. We also conclude from the results obtained that our approach is a promisable one.

Keywords
"Web of Things" (WoT), Ubiquitous Computing, "Internet of Things" (IoT), Gateway Web server.

1. INTRODUCTION
Today, the Web is a global platform for information-based applications, but that is about to change. What is driving this is the rapidly changing incremental cost of networking for all kinds of devices. This is a happy side effect of Moore's law which describes the ongoing exponential improvements in integrated circuitry which by now has been happening for more than half a century. It is now easy to integrate radio-frequency components alongside digital circuitry for microcontrollers. We are in the midst of a proliferation of devices that are largely invisible as they are embedded within everyday objects from toasters to cameras and cars. Microcontrollers are the fastest growing segment of the computer industry, with hundreds in every home. These devices are programmed to serve a single purpose, and today are mostly isolated. Networking them will allow many new kinds of applications that add values in the way that the original manufacturer may not have envisaged.

In this scenario there is strict need for good scalable and reliable architecture for existing "Web of Things". Making the smart things interconnectable such that bits can be transferred between devices is only the first step, more works are expected to make smart things interoperable such that they are understandable with each other. Interoperability in context of IPv4 and IPv6 is particularly essential, and a must, to build system with various devices. In this paper we will discuss some of the possible loop holes in existing architecture and propose solutions towards better architecture for "Web of Things".

2. RELATED WORK
During the early stages of "Web of Things" [1] two architectures where proposed these architectures rely on concept that sensors act as a RESTful resources [2]. Here sensors can be of any type. Main architecture is REST based architecture [3]. This allows the end devices to be accessible through HTTP protocol using RESTful APIs [4]. The two architectures are Web oriented architectures. Creating resource oriented architecture has been the main achievement of "Web of Things". The first architecture earlier proposed [5] is for direct access to the API on de- vices which have capability to run embedded web servers on them hence the capability to interact using REST principles. Second architecture has been for access to API on smart gateways which act like an intermediately in between end devices and web server [6]. Even earlier similar architecture was proposed [7] but they are not a promising one. Ostermaier et al. [8] presented a prototype using programmable low power WiFi modules for connecting things directly to the web. No suitable methods have been proposed till date in context of heterogeneous IPv4/IPv6 enabled devices in "Web of Things". Detailed description about REST principles and Resource Oriented Architecture can be found at [4].

3. "WEB OF THINGS" & "INTERNET OF THINGS"
While there has been various definitions of Internet of Things , the word "Internet of Things" has been derived from two "Internet" and "Thing", where "Internet" can be defined as "The world-wide network of interconnected computer networks, based on a standard communication protocol, the Internet suite (TCP/IP)", while "Thing" is "an object not precisely identifiable" Therefore, semantically, "Internet of Things" means "a world-wide network of interconnected objects uniquely addressable, based on standard communication protocols". The main motive of Internet of Things is to envision a scenario where Things which are uniquely addressable can connect to and communicate under user, social or various application domain specific contexts. IoT can be seen as a part of internet services similar to the World Wide Web (WWW), email, file sharing, video, online chat, file transfer, telephony, shopping, or rating. The attributes of the IoT almost completely exclude humans from direct intervention.
Web of Things as said by Dominique Guinard & Vlad Trifa is

- "It's about taking the Web as we know it and extending it so that anyone can plug devices to it."
- "It's basically about giving eyes, ears, and all kinds of sensory appendixes located worldwide to it."
- "It's about seamlessly connecting the physical world with the virtual."

WoT focuses on software standards and frameworks such as REST, HTTP and URIs to create applications and services that combine and interact with a variety of network devices. So, you could think of the Web of Things as everyday objects

being able to access Web services. The key and most important aspect here in Web of Things is that this doesn't involve new standards for of communication but existing standards are used.
In summary Internet of things focuses on interconnecting the things, creating a network of devices. IoT basically envisions on making every smart physical thing a tiny computing resource attached to Internet, making it a part of Internet. And Web of Things envisions on making the attached smart things to internet a part of World Wide Web using popular application layer protocols like HTTP. We see Web of things as an extension to Internet of Things where Internet of Things serves as foundation for Web of Things.

4. NEED FOR REFINEMENTS AND SOLUTIONS IN WOT

The prerequisite for WoT is for the "things" to have embedded computer systems that enable communication with the Web. Such smart devices would then be able to communicate with each other using existing Web standards. We reintroduce the issues and solutions stated out in our own previous work [15]

Issues: We see following issues in architecture proposed earlier.

Embedded devices are designed to be more power efficient and hence in most cases they have computationally less powerful resources. Though they are equally capable to function like a normal device running web server like utility but they cannot function as robustly as computationally powerful system for example a Web Server running on multicore processor with high clock speed.

These most low powered devices often for need to save power run customized execution runtime which makes it not suitable for running computationally intensive algorithms to detect intrusions, spams, and denial of service attacks. Due to this they are vulnerable to attacks easily.

Solutions: As a solution to issues stated, we propose introduction of Gateway webserver which takes care of detection of intrusions, spams, and denial of service attacks. This gate way web server is computationally powerful system which is capable to run computationally intensive algorithms. Apart from security aspects this gate way server would also focus on caching and gatewaying of messages in between IPv4/IPv6 enabled devices using SOCKS protocol [9]. The figure 1 shows architectural overview and the extended functions of gateway webserver in web of things architecture derived from the previous RESTful architecture [1] and considers scenario where embedded devices are IPv4 /IPv6 enabled and interact directly with help of protocols of TCP/IP suite. We can see that this architecture can be easily extended to a scenario where devices in web of things context communicate in proprietary protocols like ZigBee [10].

5. EXPERIMENT SETUP

Experimental setup contains a Gate way server which is computationally powerful desktop machine and a low power device called Netduino Plus [11] which runs .NET micro runtime and is provided as Open Source. The low powered device is based on ARM architecture with configurations, as Atmel 32-bit microcontroller incorporating ARM7TDMI ARM Thumb Processor, Speed of 48MHz, ARM7Code Storage of 64 KB and without networking of 128 KB and RAM of 28 KB with networking and without networking of 60 KB.

The low powered device runs a small program which acts like a web server when requested using GET method and sends a JSON response which contains the current room temperature. The current room temperature is obtained by using a Temperature sensor LM35 [12] attached to the low powered device.

The gateway server in our experiment is Apache Tomcat Server and all the functionalities have been implemented as servlets running on the Apache Tomcat [13]. The caching feature is embedded into the server using Java caching system [14] with cache expiration time set of 10 seconds.

In another experiment setup, the actual proposed architecture has been tested with an Android based smart phone acting as a device in "Web of Things". The devices' computational specifications are as follows

- Processor: 830 MHz ARMv6.
- Ram: 180 MB internal, 290 MB user available RAM.
- OS: Android 2.3.57
- Linux Kernel Version: 2.6.35.7

The Android Smartphone had a jetty based web server on it which can host a Java based web application. We developed a web application using Java servlets technology. The Java servlet's task was to compute a GCD (Greatest Common Divisor) value for two numbers provided in GET request and return a JSON response containing the GCD value as HTTP response. The algorithm for GCD computation is Euclid's Algorithm (Iterative version), the complexity of this algorithm is bounded by quadratic function [17].

Figure 1: An architectural overview of proposed architecture for Scenario in which embedded devices are IP enabled.

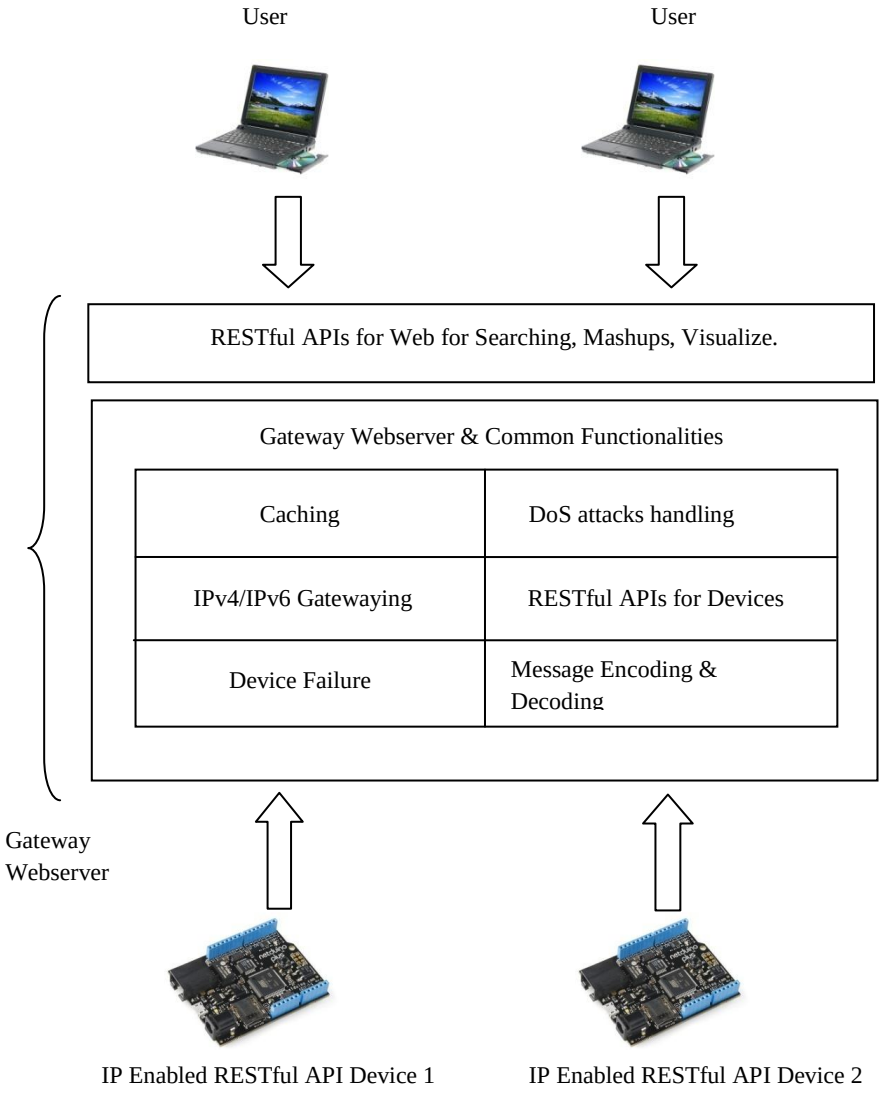


Figure 2: A gnuplot obtained for case 7 of Table 1 (Direct request to Embedded Device).

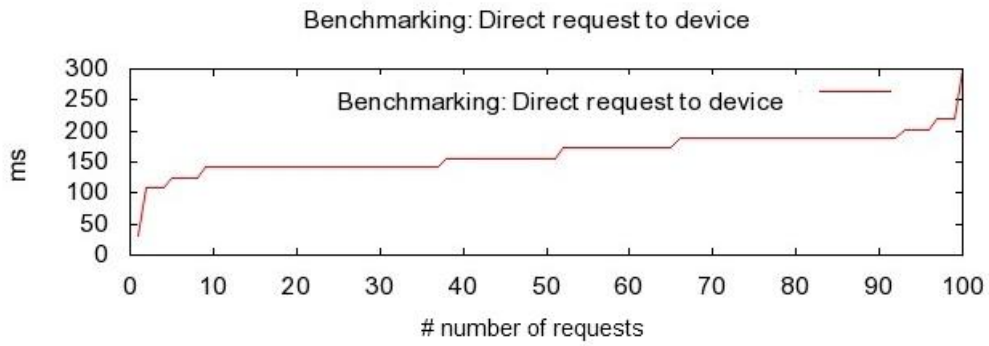


Figure 3: A gnuplot obtained for case 7 of Table 2 (Indirect request to embedded device through Gateway Server).

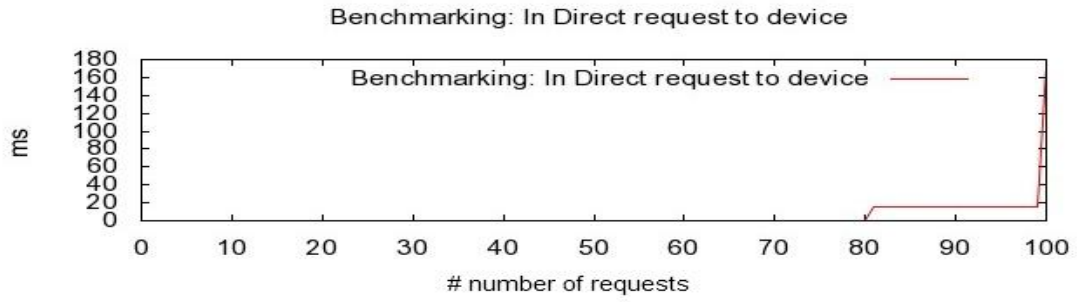
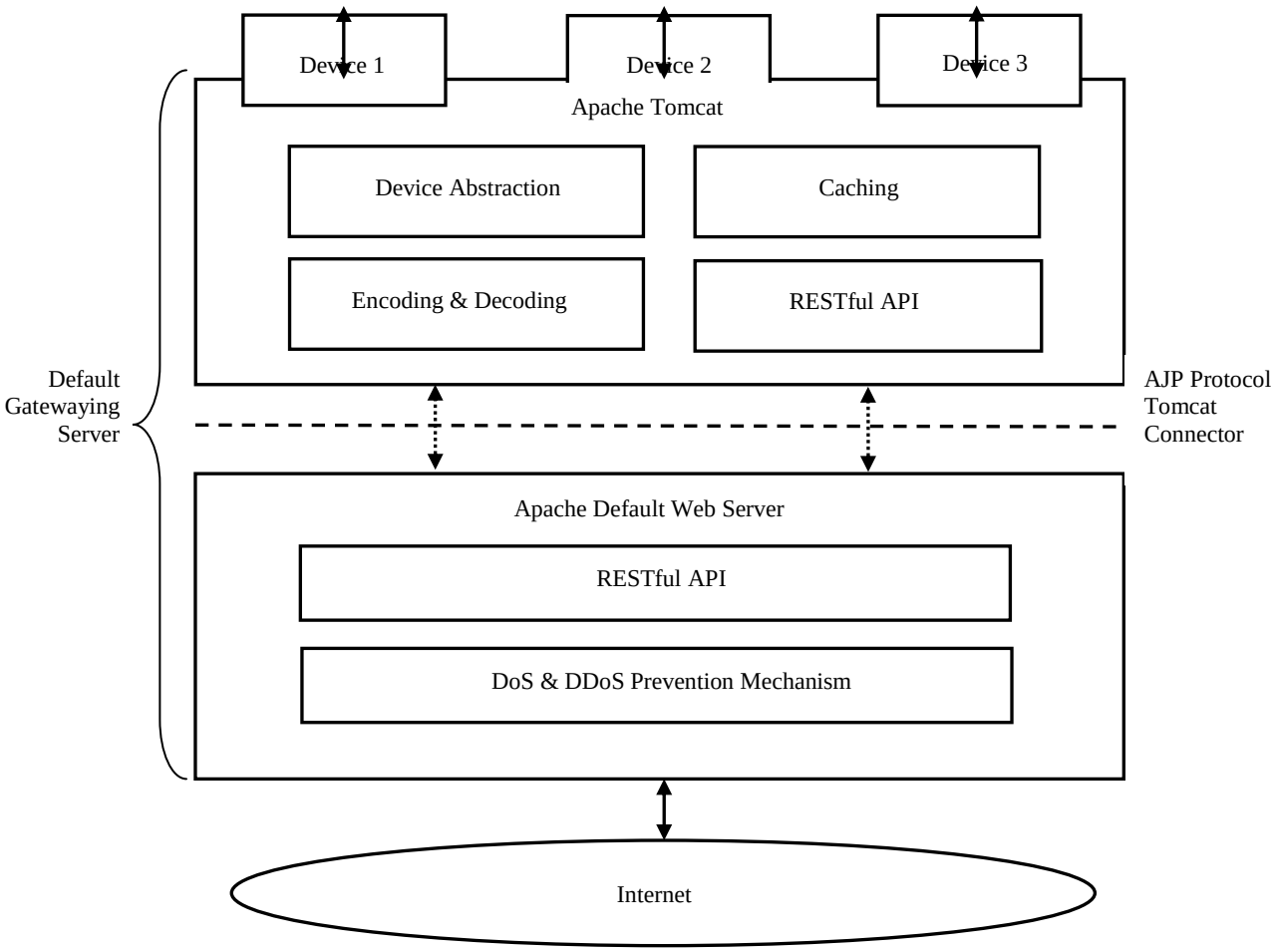


Figure 4: An overview of experiment setup.



6. RESULTS AND SUMMARY

The experimental setup was evaluated using Website benchmarking tool, Apache Bench [16]. The experimental setup was evaluated under various test cases using various options of Apache Bench. The results of evaluation are tabularized below.

Results in Table 3 have been obtained by using Apache jMeter [18] as load testing tool. This Euclid's GCD algorithm being a computationally intensive, our tests indicate viability of proposed architecture in computational aspect.

Table 1: Results: Direct Requests to Device

Serial No	Number of Requests	Concurrency	Time for completion of load test (seconds)	Requests/sec (mean)	Time/Request (mean)(msecs)	Transfer rate (mean)(Kbytes/sec)	Longest response time (msecs)
1	50	25	27.844	1.80	13921.875	0.23	24359
2	50	10	3.9535	2.53	395.313	0.33	3953
3	10	1	3.0000 (Forced)	6.00	166.667	0.78	281
4	10	1	30.00(Forced)	6.06	165.093	0.80	313
5	10	1	1.641 (Keeplive)	6.10	164.063	0.80	188
6	10	1	1.688 (Keeplive)	5.93	168.750	0.78	172
7	100	1	16.484	6.07	164.844	0.79	297
8	10000	10	2.00 (Forced)	6.12	1634.615	0.80	828
9	100000	10	(Failed to complete) Failed after 258 requests.				
10	10000	10	(Failed to complete) Failed after 634 requests.				
11	500	10	83.297	6.00	1665.938	0.79	30844

Table 2: Results: Indirect Requests to Device through Gateway Web Server

Serial No	Number of Requests	Concurrency	Time for completion of load test (seconds)	Requests/sec (mean)	Time/Request (mean) (msecs)	Transfer rate (mean) (Kbytes/sec)	Longest response time (msecs)
1	50	25	25.172	1.99	12585.938	0.45	25156
2	50	10	4.125	2.42	4125.00	0.57	4109
3	10	1	3.000 (Forced)	317.33	3.151	66.64	172
4	10	1	30.000 (Forced)	444.93	2.248	95.14	172
5	10	1	0.219	45.71	21.875	9.84	188
6	10	1	0.031	320.00	3.125	68.75	16
7	100	1	0.484	206.45	4.844	44.61	172
8	10000	10	2.00 (Forced)	518.50	19.286	111.15	844
9	100000	10	176.297	567.23	17.630	120.22	3953

Table 3: Results: Load test in case of Android Smartphone as Device

Parameters	Direct Request	Request through Gateway	Description
# Samples	10795	10789	The number of samples(# Requests)
Average	405	239	The average elapsed time of a set of results (seconds)
Min	6	6	The lowest elapsed time for the samples
Max	104222	1758	The longest elapsed time for the samples
Standard Deviation	3276.41129707272	277.462072513851	The Standard Deviation of the samples' elapsed time
Error %	0.00194534506716072	0.00194642691630364	Percent of requests with errors
Throughput	60.0319206321842	96.9553730297093	Throughput (Requests/sec)
Bandwidth	9.94210258048281	20.4435028431046	Throughput (Kb/sec)
Avg. Bytes	169.588327929597	215.915284085642	Average size of the samples' response in bytes

Summary

We find some of the observations very impressive showing our proposal of intermediation in between Web of Things and the Web. From Table 1 we see that if the direct requests are made to embedded device the average number of requests per second across all the test cases having concurrency parameters value as 10 is 6 requests /sec and from Table 2, under same concurrency value of 10 the average number of requests / second is 281 requests/sec when the HTTP request to embedded device is made through an intermediate gateway web server. We also see a great difference in average rate of transfer, in case of direct request to embedded device this rate is far low while in case of indirect request to embedded device through a gateway webserver its really very high as compared to former case. It's also evident from the results obtained that the response time is far less if the request to embedded device is intermediated and cached by intermediating web server. We see here a graceful type of failure handling by gate way web server in case of device failure. Test case 9 and 10 of Table 1 clearly points out the lack ability to handle large bursts of requests by low powered embedded device if it is directly made open to Web while test case 8 and 9 from Table 2 show that our approach of intermediation is capable to handle the burst of requests. From the experimentation and results obtained we find our approach of introduction of intermediation through introducing a gate way like web server with added functionality taking some of non vital functions of devices to gateway webserver is promisable and feasible for towards better architecture for "Web of Things ".

Results in Table 2 and 3 when analyzed with Table 1 clearly indict that, the gatewaying approach outweighs in terms of throughput and even bandwidth in case of lower powered IP enabled devices in context of "Web of Things".

The Standard deviation calculated in Table 3 for direct and indirect method of requesting for a device clearly shows a consistent behavior by the system in gatewaying approach as compared to approach in which direct request are made to IP enabled embedded device.

We see considerable enhancements in architecture of "Web of Things" in terms of security message encryption, use of SSL and TLS and further improvement in failure handling, power consumption reducing techniques, more resilience of architecture as future a scope.

7. REFERENCES

[1] S. Duquennoy, G. Grimaud and J. Vandewalle, "The Web of Things: Interconnecting Devices with High Usability and Performance," Proceedings of the 6th IEEE International Conference on Embedded Software and Systems (ICESS'09), Hangzhou, 25-27 May 2009, pp. 323-330.

[2] T. Luckenbach, P. Gober, S. Arbanowski, A. Kotsopoulos and K. Kim, "TinyREST—A Protocol for Integrating Sensor Networks into the Internet," Proceedings of REALWSN, Stockholm, 20-21 June 2005, pp. 89-93.

[3] V. Stirbu, "Towards a Restful Plug and Play Experience in the Web of Things," In IEEE International Conference on Semantic Computing, Santa Clara, 4-7 August 2008, pp. 512-517.

[4] L. Richardson and S. Ruby, "RESTful Web Services," Information, UC Berkeley, November 2007. O'Reilly Media, Inc., Sebastopol, 2007.

[5] V. Trifa, S. Wieland and D. Guinard, "Towards the Web of Things: Web Mashups for Embedded Devices," Second Workshop on Mashups, Enterprise Mashups and

Lightweight Composition on the Web, Madrid, 20-24 April 2009.

[6] V. Trifa, S. Wieland and D. Guinard, "Design and Implementation of a Gateway for Web Based Interaction and Management of Embedded Devices," SAP Research, Zurich, 2009.

[7] P. Schramm, E. Naroska, P. Resch, J. Platte, H. Linde, G. Stromberg and T. Sturm, "A Service Gateway for Networked Sensor Systems," IEEE Pervasive Computing, Vol. 3, No. 1, 2004, pp. 66-74.

[8] B. Ostermaier, M. Kovatsch and S. Santini, "Connecting Things to the Web Using Programmable Low-Power Wifi Modules," Proceedings of the 2nd International Workshop on the Web of Things (WoT 2011), New York, June 2011.

[9] P. Patnaikuni and R. Kulkarni, "An Architecture for "Web of Things" Using SOCKS Protocol Based IPv6/IPv4 Gateweying for Heterogeneous Communication," Advances in Internet of Things, Vol. 2 No. 1, 2012, pp. 8-12.

[10] Zigbee. http://www.zigbee.org

[11] Netduino Plus. http://www.netduino.com/netduinoplus/specs.htm (Accessed, 2 April 2012).

[12] LM35: http://www.national.com/mpf/LM/LM35.html, Precision Centigrade Temperature Sensor (Accessed, 2 April 2012).

[13] Apache Tomcat: http://tomcat.apache.org (Accessed, 2 April 2012).

[14] Java Caching System http://commons.apache.org/jcs/ (Accessed, 2 April 2012).

[15] P. Shrinivasan , R. Patnaikuni and Raj. B. Kulkarni, Towards Better Architecture for Web of Things, ASME Press, New York, NY ISBN 9780791859735, doi: 10.1115/1.859735.paper14 http://dx.doi.org/10.1115/1.859735.paper14

[16] Apache Bench, http://httpd.apache.org/docs/2.0/programs/ab.html (Accessed, 2 April 2012).

[17] Phong Q. Nguyen , Damien Stehlé, An LLL Algorithm with Quadratic Complexity, SIAM Journal on Computing, v.39 n.3, p.874-903, September 2009 [doi>10.1137/070705702]

[18] E. Halili, Apache JMeter. Packt Publishing, 2008.