Methodology for SIP Infrastructure Performance Testing

Miroslav Voznak, Jan Rozhon
Department of Telecommunications
VSB – Technical University of Ostrava
17. listopadu 15, Ostrava
Czech Republic
miroslav.voznak@vsb.cz, jan.rozhon@vsb.cz

Abstract: - This paper deals with a testing method suitable for SIP infrastructure. The performance testing is an issue of research and no standardized methodology has been adopted yet. We present the main ideas of the methodology that allows for testing the keystone of SIP based infrastructure – the SIP Server – in both SIP Proxy and B2BUA (Back to Back User Agent) configurations. Our methodology has its foundations in the work of the IT Company Transnexus and these foundations have been enhanced with the ideas reflecting the nature of the SIP protocol. In addition, the entirely new methodology for benchmarking the SIP server in the B2BUA configuration has been introduced. This method utilizes one of the attributes of the B2BUA – the flow of media passing through the B2BUA – and measures the effectiveness of codec translation, which relates to the performance measured in cases without codec translation. Our approach offers the complex method for testing SIP infrastructure, which has been verified experimentally. The out-coming results are the part of this paper together with appropriate comments and conclusions.

Key-Words: - Asterisk; B2BUA; codec translation; Opensips; Performance testing; SIP Proxy

1 Introduction
The whole topic of SIP infrastructure performance testing is under development and there are no unified recommendations as how to perform the tests and what to pay attention to. Moreover, the proprietary solutions offer huge comprehensibility of testing scenarios but they do not use generally recognized means and ways to perform the testing, so the results may not be compatible. Many issues in this area have been solved by Transnexus. Their white papers [1] and general approach to the testing has significantly inspired our research because it is based on open source solutions and allows us to integrate basic thoughts mentioned in the IETF draft [2]. This RFC draft focuses on methodology for benchmarking SIP environment. Considering this information, it is obvious that there is a big gap in the area of SIP infrastructure performance testing and benchmarking. This gap and its elimination is the main motivation for our research. Simple SIP infrastructure performance testing configured in both B2BUA and SIP Proxy modes and the examples of the output results are the main contribution of this paper.

In this paper the current state of development of the SIP infrastructure performance testing will be described. In addition, new methodology for SIP benchmarking will be presented and verified experimentally. The results obtained in the experiment will then be analyzed and commented. From this output of our paper the optimal and maximal load of the SIP server can be determined, which is useful mainly in small and medium business VoIP installations.

2 State of the art
As mentioned in the introduction, there are some proprietary solutions for SIP testing, the main advantage of which is a huge comprehensibility of testing scenarios. However, in the real world, there are also disadvantages, such as high price and possible incompatibility of the results, as each company focuses on a different main area of interest. On the other hand, the IETF has published several drafts which have the methodology and the metrics of SIP infrastructure testing as their main topic of concern, see [2], [3] and [4]. These drafts try to define the basic terms for SIP benchmarking as well as the times, the measuring of which is important to gain the relevant results. Given the early stage of development of these drafts, there are no software or hardware means for SIP benchmarking that would utilize these drafts yet. Halfway to creating a suitable and generally applicable testing method is the Transnexus' SIP benchmarking model which can serve as an inspiration [1], [5]. This company created a useful SIP infrastructure testing method using an open source traffic generator SIPp. In order to develop a method which would reflect the main thoughts of the IETF drafts it is useful to modify the Transnexus' procedure to better reflect the nature of SIP protocol from the SIP transactions and dialogs point of view and the results will be sufficient to determine the effectiveness of a system, the highest load which it can handle as well as the dynamically changing characteristics of a system, which is crucial for assessing whether the SIP server can be operated in given environments.

3 Methodology

Although Transnexus' benchmarking model served as an inspiration in the early phase of the development of our methodology it lacks the effort for standardization. They measure times between transmission and reception of some key messages (e.g. Invite, 100 Trying, 180 Ringing), however their approach does not look at these messages as the part of the SIP transaction. This results in outputs from which the user is unable to read more complex attributes of the system. To be more specific, you can learn how quickly the SIP server is able to respond to your message, but you cannot learn how quickly it can process it and resend it to the destination. Our approach on the other hand makes this possible, so it is not the issue to recognize the "real world" parameters of the SIP server such as Call Setup Length (later described as SRD).

From the practical point of view Transnexus' model is rather too complex. As the commercial subject, Transnexus has focused on creating the model that would utilize some of their commercial products, which led them to use their management and billing platform, which required two more separate computers. Moreover, the testing scenarios they created utilize several different end locations for the simulation of call rejection, no route issue, no device problem and so on. This again increases the complexity of the test platform due to the need of more physical machines. From mentioned it is clear that this model is unsuitable for practice. From our point of view it is beneficial to create the testing platform that would be as simple as possible, which would make it easier to deploy in any practical environment. This is why we decided not to use any other special hardware and to simulate the end location for calls just by the listening UASs, which is made possible by the fact that we want to evaluate the ability of the SIP server to successfully connect calling and called party.

In order to perform SIP testing, we simulate both ends of the SIP dialogue to test the main part of the SIP infrastructure, the SIP server. The SIP server represents a set of servers always involving SIP Registrar and SIP Proxy or B2BUA (Back to Back User Agent). The latter is the most used solution in enterprise environment, for both SMEs (Small and Medium sized Enterprise) and LEs (Large Enterprise). Fig. 1 depicts test hardware configuration for testing the SIP Proxy and B2BUA.

This is a general configuration which does not reflect all the aspects of test platform used for our measurements. Firstly, we used both physical and virtual computers to simulate SIP traffic. The results with both configurations were almost identical allowing future user of this methodology to decide for topology that would be best for him according to available hardware.

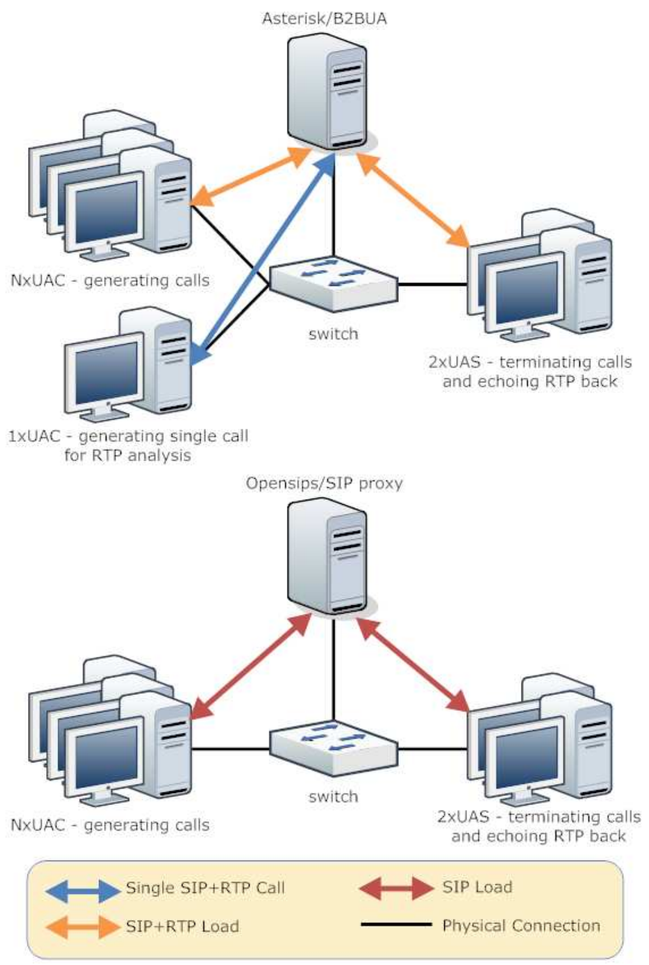


Fig. 1. Test Bed Diagram for B2BUA and SIP Proxy Configuration.

The only condition required for testing SIP server successfully and comparably is the interconnecting device (or system). Basically, this can be any device or network capable of routing of SIP messages among SIP traffic generators, SIP server and SIP traffic recipients, but to make the results of measurements comparable with those taken in different network, we would be required to use the exact same topology, which may be the issue. This is why it is advantageous to use as simple topology as possible to reduce additional costs and work caused by the need of some special topology. So, the most flexible variant is to use the single switch, which is undoubtedly a commonplace in all modern SIP installations.

Secondly, the number of devices used for the testing may vary due to the performance of the SIP server. The more the SIP server is efficient the more devices are needed to test its performance especially on the UAC side. Due to the software limitations of the SIP traffic generator (SIPp) one computer in UAC mode is capable of creating 200 simultaneous calls with media (for testing B2BUA) and about 220 calls per second without media (for testing SIP Proxy) no matter what the hardware configuration of the PC running SIPp instance is. Therefore we need to estimate the SIP server performance to determine the number of computers (physical or virtual) needed for test, which makes the virtualization the more viable option. Number of UASs is not affected by the SIP server's performance that much, however it is necessary to force the SIP server to decide between different paths to UAS, therefore there have to be at least two computers in UAS mode in the test topology.

As well as the topology the test scenario should be as simple as possible mainly to reduce the complexity of the test and except of that also because it is not possible to test the SIP Proxy (and B2BUA as well) in all the possible configurations. Thus it is useful to focus on basic default configuration and perform the tests with it. The output results then carry the information about the "best case scenario" according to which we can decide about the SIP server's performance and compare it with its rivals.

A. *Measured parameters*

As mentioned in the Introduction we use the parameters defined in IETF draft for all our measurements. But except of them we use the hardware utilization parameters as well. Let's now take a look at the locations, where these groups of parameters are measured.

First group is measured at UAC and includes the call statistics such as number of (un)successful calls and durations of the message exchanges. RTP samples for analysis are captured here as well.

Second group – the hardware utilization parameters – is measured directly on the SIP server. At this place CPU and memory utilization and network traffic is measured. The complete list of all measured parameters includes:

- CPU utilization.
- Memory utilization.
- Number of (un)successful calls.
- Registration Request Delay – time between first Register method and its related 200 OK response [2].
- Session Request Delay (SRD), the time between first Invite method and related 180 Ringing message [2].
- Mean Jitter a Maximum RTP Packet Delay.

Fig. 2 shows the meaning of the RRD and SRD delays in more detail.

B. *Limit definition in results analysis*

The previously defined parameters do not suffice to assess the SIP server's performance. To be able to determine the SIP server's performance from the collected data we need to define the limit values for each category of the measured parameters. This definition must come out from the features of the SIP protocol and generally recognized convention from IP and classic telephony.

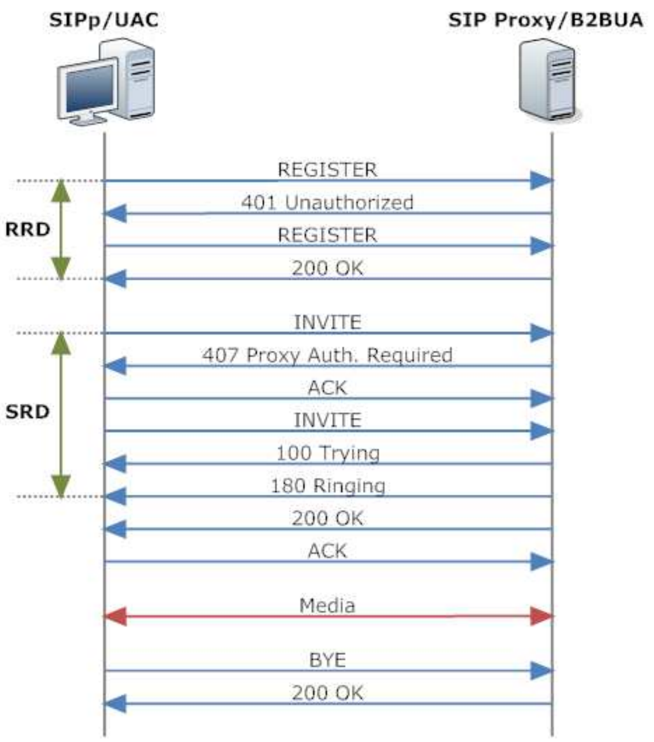


Fig. 2. Registration Request Delay and Session Request Delay in SIP Dialog.

From the hardware utilization characteristics the CPU utilization plays the main role in performance analysis of the SIP server. This conclusion is logical because of the importance of CPU in the computer architecture and the CPU oriented operations of the general SIP server architecture.

In general, the CPU utilization characteristic is limited by the maximal CPU performance, which is 100%, but this boundary can be reached rarely. To be more specific, due to the time intervals between particular measurements of the CPU utilization can cause that short peak in CPU utilization characteristic is not registered. However, during this peak delays and call quality impairments can occur. To reflect this imperfection of our methodology, performance boundary under 100% should be anticipated. Actual value of the CPU performance boundary may vary, though. Therefore we search the CPU utilization characteristic for the first point where maximum CPU utilization is reached. This point is then the maximum number of calls, which the SIP server can handle from the hardware performance point of view.

The limit definition for the SIP delay characteristics RRD and SRD comes from the nature of the SIP protocol. When the call is set up the delays between messages should not exceed several hundreds of milliseconds and although these limitations are tied up with the travel of the SIP message from one end of call to another, it can be used for our purposes as well, because of the similarities that come from the need to set up a call quickly enough not to bother the user with noticeable delays.

From this, we can estimate that the quality boundary for RRD and SRD is somewhere around 300 milliseconds. However, this value may vary in accordance to the need of each one particular user. Generally, we can say that limit from the SIP transactions point of view is reached, when SRD and RRD characteristics start increasing rapidly. This boundary will give us a slight space as the potential reserve.

The quality of speech is vulnerable to great delays between consecutive RTP packets. It is affected by the jitter as well, but the jitter issue can be eliminated by the sufficient Jitter buffer on the receiving side, therefore maximum packet delay is the key characteristic in RTP stream analysis. From the theory of IP telephony the delays between packets should be in the tens of milliseconds, therefore and because of the similar reasons mentioned with SRD and RRD, we decided to set this boundary to approximately 90 milliseconds.

All the delay characteristics use similar analogy with the theoretical values for end-to-end delays, that is why their definition could not be exact and these parameters may vary in different environments. To eliminate different interpretation of the same results and to simplify the delays analysis, we use as the quality boundary for all the delay characteristics the point, where the particular characteristic change its "almost constant" trend to rapid increase. This approach gives us correct results, which was tested experimentally, and the methodology of the analysis is much simplier.

C. SIP Proxy testing

In basic configuration of the SIP Proxy we are able to measure just the SIP and utilization parameters. RTP stream does not flow through SIP Proxy and thus it does not represent the load for it. This is why we do not have to think about the call length because no matter how long the call is the hardware utilization is the same, so the only appropriate metric for measuring SIP Proxy is the number of calls generated per second (or any other time interval).

Each measurement on SIP Proxy consists of several steps. Every single step takes about 16 minutes, this means that for 15 minutes, 10-second long calls are to be generated at a user-defined call rate. Then there is a 10-second period when the unfinished calls are terminated. This repeats for every single step of the call rate. Every call consists of a standard SIP dialogue and pause instead of media. Because the load is not constant but increases slowly at the beginning of the test (first 10 seconds) and decreases at the end of it (last 10 seconds), the results taken after this starting period and before the ending one are the only ones which are going to be considered valid. To allow additional changes in time interval setting in the scenario and to strengthen the consistency of the method we decided to use the data collected during the middle 10 minutes of each step. All the parameters named in the previous subsection are measured except those related with RTP stream.

The 10 second long time interval that was mentioned several times came from the compromise between reasonable call length and the need for generating as much of the calls per second as possible. It allows for decent performance and does not require huge database of subscribers. This interval can be changed but cannot exceed 2.5 minutes that allow for collecting the valid data.

SRD is measured although this scenario cannot be considered as end-to-end (this condition is defined in draft [2]). We decided to measure it because the load on the UASs is minimal even for high call rates, which makes the delays created by the UASs both minimal and almost constant. Therefore we can use this parameter to decide about the SIP Proxy's performance, because the delays created by it are the only variable making the collected data useful. This is the only deviation of our method from the draft [2].

D. B2BUA testing

Unlike SIP Proxy for this type of SIP server the RTP stream presents the highest load on the SIP server therefore the number of simultaneous calls must be used as a metric. This is the main difference between the B2BUA and SIP Proxy testing scenarios. Second not so important difference (from the methodology point of view) is that in this configuration we are to measure effectiveness of codec translation because in this scenario performance of the B2BUA is not affected only by its setting but also by UAC and UAS configurations. The test routine will then be repeated for each case of different codec setting.

The method of the test is however almost the same, the only issue we face is the new metric together with the need for revising the time interval for a single call. The new metric is an issue when the SIP traffic generator cannot be ordered to create certain number of simultaneous calls. In this case it is necessary to calculate the number of calls generated per second. This can be done by this equation:

$$C_R = C_S \cdot T \qquad (1)$$

C_R is the desired Call Rate, C_S is the number of simultaneous Calls we want to generate and T is Time interval defining how long the call (media) should be. Time interval used for B2BUA in our measurements was set to 60 seconds because most calls have this length, but again this parameter can be changed. To perform the testing of RTP streams we use a special computer, which allows us to use more sophisticated tools for capturing the network traffic without the RTP and SIP parts of the tests influencing each other. Because we focus on testing effectiveness and speed of codec translation we were, at this point, able to determine the maximum load which the SIP server can handle from the SIP or RTP point of view. However, these results would only be valid for a single machine/platform and that is why we add one more step to the data analysis. The same procedure of testing as mentioned above is performed on a machine configured to allow media to only pass through the SIP server. The results taken during this test serve as a basis to which we relate all the other results. The relation is expressed in (2) as a performance ratio. The performance rating factor P_{RF} is a ratio of any previously mentioned parameter measured in codec translation case (P_{CT}) with a certain number of simultaneous calls to the value of the same parameter (P) taken in case without codec translation and the same load.

$$P_{RF} = \frac{P_{CT}}{P} \cdot 100 \qquad (2)$$

This step allows us to compare the results from hardware and platform independently.

4 Experiment

To simulate both UACs and UASs, we are going to use the SIP performance testing tool called SIPp [6]. This open source utility can simulate many concurrent SIP calls. Moreover it allows measuring important times such as those defined in the IETF draft [2]. SIPp performs the calls which follow user-defined scenarios in xml language. This xml scenarios are distributed on every computer and SIPp is invoked by using bash script and SSH. One of the computers works as a SSH client and controls the whole test by sending orders to other computers (SSH servers) via SSH. The message call flows exchanged between related UAC and UAS SIPp instances are depicted in the Fig. 3. As a B2BUA we use Asterisk PBX and as SIP Proxy we deploy Opensips.

The key values of hardware utilization on the SIP server are measured by System Activity Reporter (SAR) every 10 seconds and 60 times, i.e. during the middle 10 minutes of the test when the generated load is constant.

The media for B2BUA testing consist of a 60-second long music song recorded in G711u pcap file, which is used by UAC. UASs are configured to use G711u-law, G711A-law, G726-32 and GSM codecs. The Asterisk PBX performs the codec translation. RTP streams can be captured and analyzed with Wireshark. Wireshark offers very complex means for RTP analysis [7]. However, the generation of RTP streams on the client side consumes a lot of CPU power, this means that we have to limit the number of calls generated by a single machine, which leads us to multiply the number of PCs running the UAC scheme. The total number of the computers can be decided according to an estimated maximum load on a SIP server. Since in our case the SIP server is a PC with merely a dual-core processor, the total number of simultaneous calls will not exceed one thousand [8]. Each PC with our hardware configuration can generate around 200 simultaneous calls. This is why the number of clients should equal or exceed four. In our case, four is just enough to perform the test of B2BUA. UASs can handle much higher load, and this is why there will be just two of them.

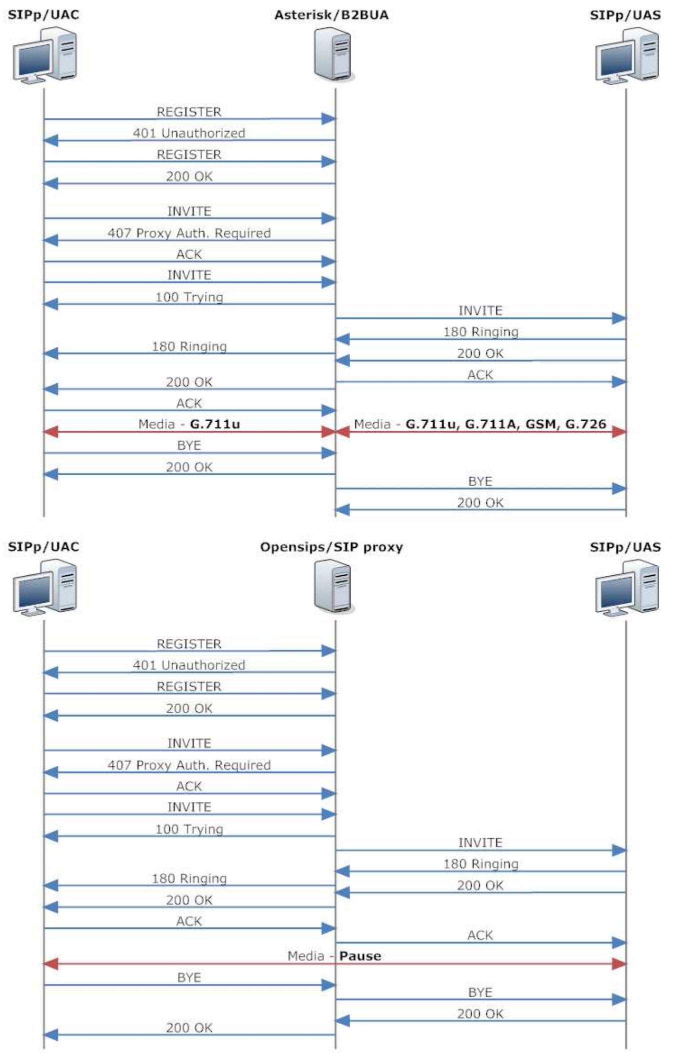


Fig. 3. Flow of messages in tests on B2BUA and SIP Proxy.

Since the media are not required for testing the SIP Proxy, the scenario places a 10 second long pause instead of them. During this time period no SIP messages and no media is transmitted. Due to the much higher performance of the SIP Proxy in comparison with B2BUA we can estimate that our machine is able to handle around 2 500 calls generated per second, which forces us to use at least 12 computers as UAC. Two UASs are sufficient though. The entire process of performance testing needs multiple computers to generate SIP traffic. To be able to successfully perform a test, the whole process must be automated. Therefore all the computers are being given orders by a Main UAC via SSH. On the Main UAC the bash script is invoked to deal with this task. In the first step, main UAC counts the number of calls that each computer should generate per one second period. Then it orders the UASs to register and starts listening on UDP port 5060. Secondly, SIPp on all UACs is invoked to generate traffic. As the last step, SAR is invoked. This is done after 2,5 minutes to ensure the stable load has been reached already. The results contain CPU, memory and network statistics, and are stored in a file *data_callrate.sar* in binary format.
As mentioned in the methodology the hardware configuration of the computers running SIPp is not too important, even five years old hardware is up to the task, but the configuration of the SIP server is crucial. For our measurements we used SIP server with these attributes:

- CPU – AMD Athlon 64 X2 5200+
- RAM – 4GB DDR2 (3.5 GB used due to x86 system)
- Debian 5.0 x86
- Asterisk PbX v.1.6.2
- Opensips v.1.6.0

All the devices are connected to a gigabit switch when SIP Proxy is tested and to 100 megabit switch when B2BUA is tested. The switches can differ because there is no actual need or reason to compare the results of SIP Proxy and B2BUA, since their operation is completely different.

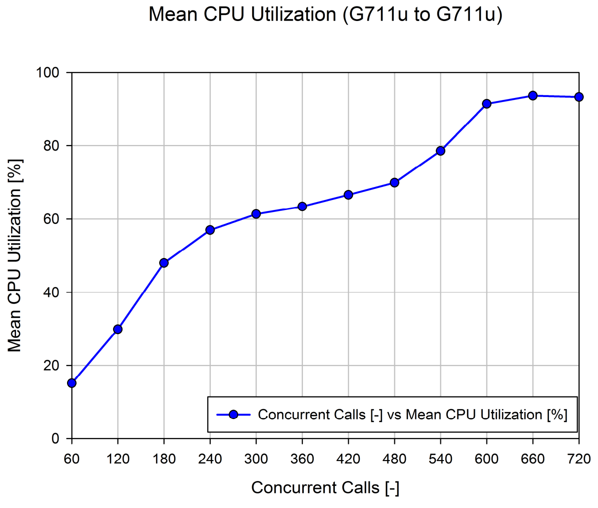
5 Results

The data collected during the whole test of SIP Proxy or B2BUA are in text format (binary data can be converted), so the data analysis can easily be done by any spreadsheet application, but for the correct interpretation of the data we have to perform a series of the same measurements to ensure that the effect of random events such as data packet scheduling techniques is marginal. The actual data then can be determined as the average of the collected data or the multitude of measurements can just serve to reveal the flawed data, which then can be replaced by the interpolated values.

A. B2BUA

For each category, there are two different charts. The first one shows the results for the case without codec translation and is colored in blue. The second shows the normalized values (acquired by inserting the collected data to equation (2)) of the cases with a codec translation and is colored in three different colors.

1) Mean CPU Utilization



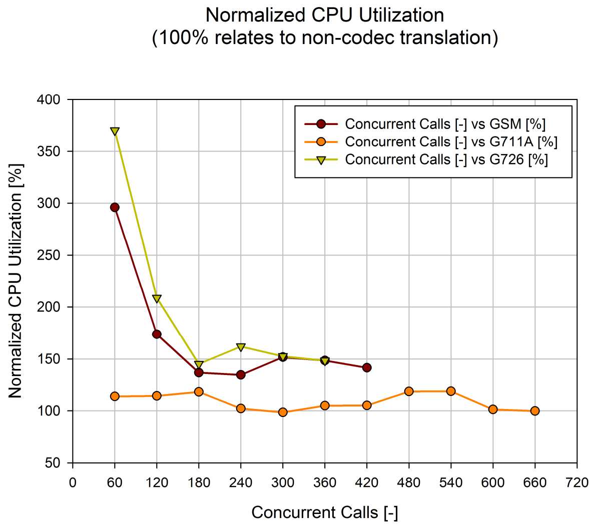


Fig. 4. Mean CPU utilization for case without codec translation (G.711u-G.711u) and its related Normalized Values for cases with codec translation.

First chart shows a simple relation between the number of concurrent calls passing through the B2BUA and its CPU utilization. The second chart shows that (as expected) codec translation from G711u to G711A consumes about 20% more CPU power than a simple G711u case without translation. On the other hand, the most demanding is the G726-32bit codec. The lowest load returns the most interesting information. With the load of 60 calls, the differences in CPU power consumption for GSM and G726 are the highest compared to the one without codec translation. With higher loads it starts decreasing rapidly.

2) RRD and SRD Delays

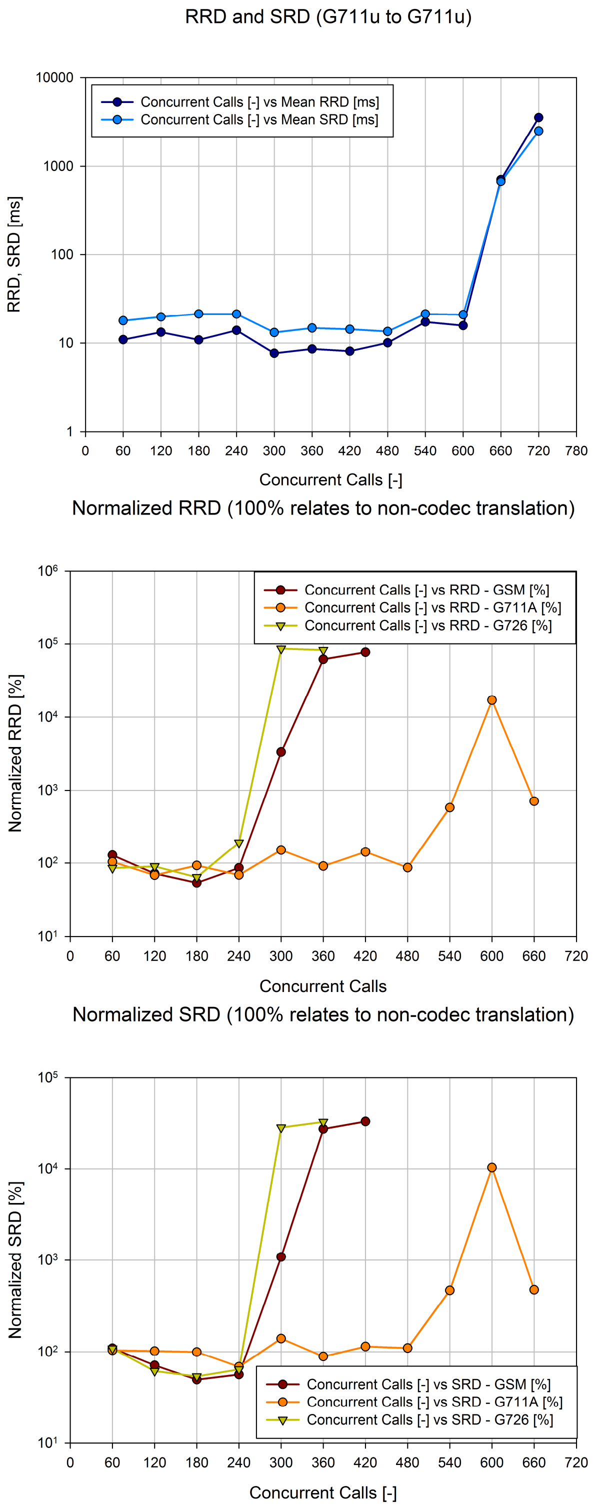


Fig. 5. Mean CPU utilization for case without codec translation (G.711u-G.711u) and its related Normalized Values for cases with codec translation.

Charts on Fig. 5 clearly illustrate that the call is set up even quicker when there is a codec translation in use and the load is under 240 simultaneous calls. Then, as the CPU utilization increases, the delays get very long. The last G711A value for both charts is so low due to a rapid increase of delays for G711u to G711u case between 600-660 simultaneous calls. The fluctuations in charts with normalized values are caused by the random events during the measurements with and without codec translation. Because we relate these values in a single equation, the variances get more distinctive, however this does not affect the final decision about the B2BUA performance from the SIP point of view.

3) Mean Jitter and Maximum RTP Packet Delay

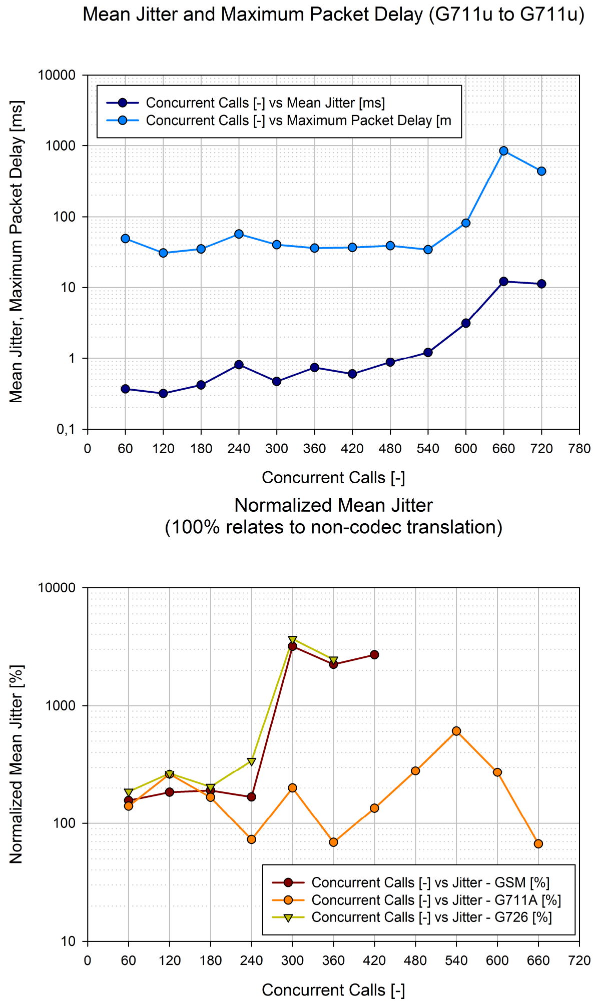


Fig. 6. Mean Jitter and Maximum Packet Delay and related Normalized Jitter.

Normalized values of mean jitter and maximum packet delay confirmed expected outcome as the values related to a small load are very similar to the main values from the case without codec translation. Peaks in the area of the medium load are caused by the volatile nature of the parameters and have no effect on final decision about the B2BUA's performance. A very rapid decrease of both normalized values for G711A is caused by the increase of the main values from non-translation case and by the significant number of unsuccessful calls in this scenario.

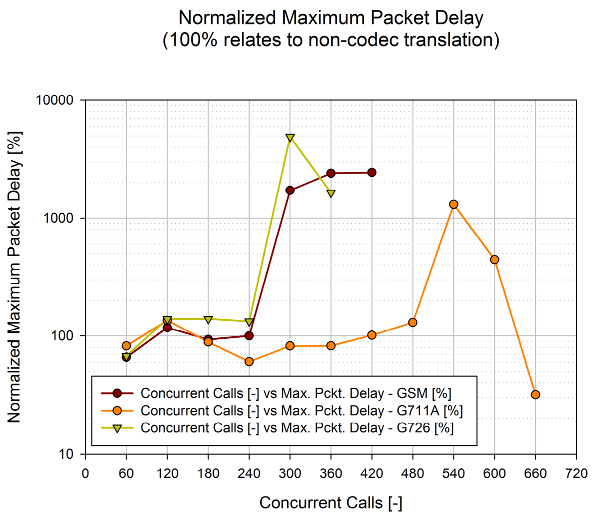


Fig. 7. Normalized Maximum Packet Delay.

B. *SIP Proxy*

The situation is much simpler than with the B2BUA. The only output from our measurements is raw data describing the ability of the SIP Proxy to handle increasing number of calls generated per second, but all the representatives of the SER architecture allow user to set the number of listening subprocesses and this number should (in theory) affect the performance of the SIP Proxy as well. Therefore all the measurements were performed with the number of UDP listeners to the values of 4 and 16.

1) *Mean CPU Utilization*

The results are not surprising except of the peak that appeared when 600 calls per second were generated. This data is not flawed, because the similar peak in the same area was measured many times, therefore it is much likely caused by the call handling mechanism of the SIP Proxy. From both charts it is obvious, that increased number of UDP listeners does not have the positive effect on the SIP Proxy's performance. On the contrary, the CPU utilization with 16 listeners is comparable with the performance of the SIP Proxy subprocessed to 4 listeners with the load of about 150 calls higher. This may be caused by the insufficient performance of the CPU, which cannot handle increased number of processes in real time and causes delays. The limiting factor for this measurement was the CPU utilization, however increased performance can be reached if other than MySQL database is used, because database itself consumed 17% of the CPU power. Unfortunately, when we performed the measurements, no working database module was released for Opensips except of MySQL due to the transition between two major releases.

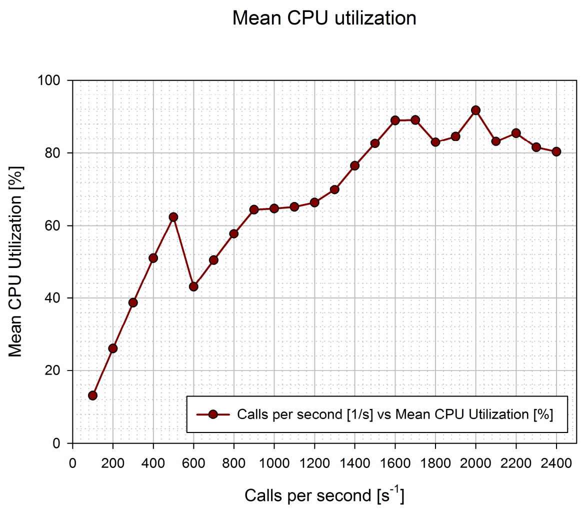


Fig. 8. Mean CPU Utilization (SIP Proxy – 4 UDP listeners).

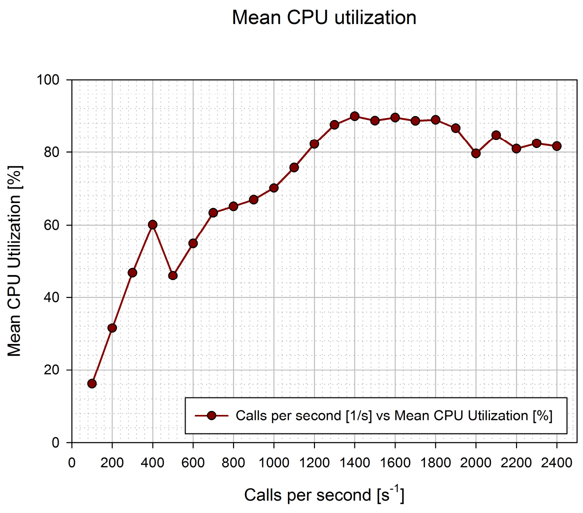


Fig. 9. Mean CPU Utilization (SIP Proxy – 16 UDP listeners).

2) *RRD and SRD*

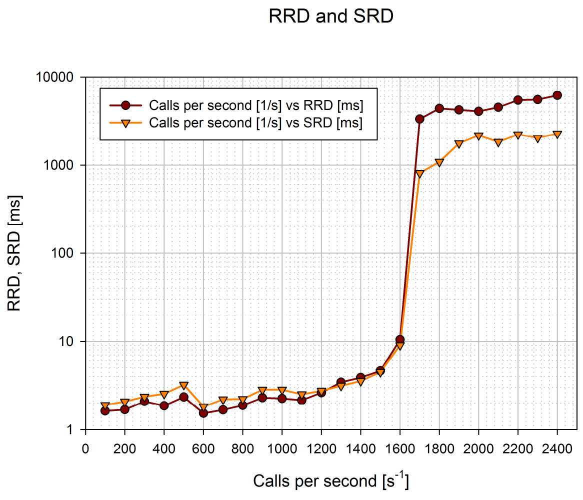


Fig. 10. RRD and SRD (SIP Proxy – 4 UDP listeners).

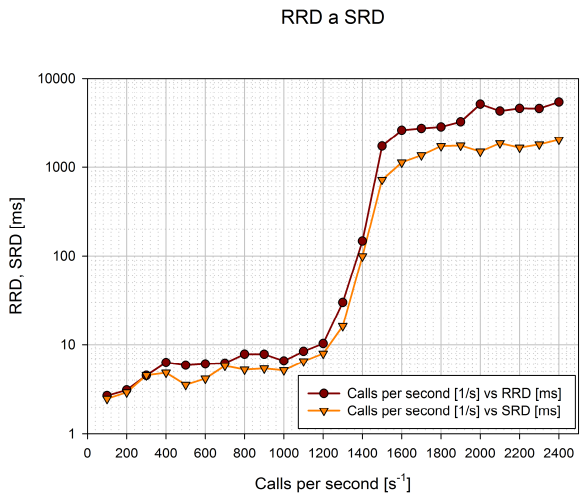


Fig. 11. RRD and SRD (SIP Proxy – 16 UDP listeners).

From the SIP perspective, the situation is similar to one that came out from CPU utilization statistics. Again, the number of subprocesses has negative influence on the overall performance of the SIP Proxy. RRD and SRD delays are about 2-3 times higher when the number of subprocesses is set to 16. Moreover, the huge leap in both characteristics (RRD and SRD) appears about 200 calls earlier. From the perspective of the two presented parameters (CPU utilization and delays) it is obvious that low cost processor should operate small number of processes to achieve best performance.

C. *Successful and unsuccessful calls*

In the Methodology section we discussed the number of (un)successful calls as a parameter that would help us determining the SIP server's performance, however no chart containing this parameter has appeared. In this subsection we are going to explain this and to do so, we will present charts of this parameter from SIP Proxy measurements, on which we will show the role of the parameter in the process of determining the SIP server's performance. Mentioned charts are depicted on Fig. 12 and Fig. 13.

The parameter IRA describes number of unsuccessful registrations in percents while ISA describes number of unsuccessful calls in percent.

To ensure that these two parameters don't influence each other, more precisely that IRA doesn't affect ISA, ISA is computed only from total number of successful registrations not from total number of created calls.

From both figures we can see the limits for optimal operation of the SIP Proxy. Since in telecommunications the number of unsuccessful calls is fairly limited by the regulations to values around 1% we cannot operate the system that exceeds this limitation. Therefore the maximum number of calls per second that measured SIP Proxy can handle successfully is 1600 and 1400 respectively. This value is however highly correlated with the values that came from the CPU utilization measurements. In other words, the IRA and ISA parameters are highly related to CPU utilization characteristic and therefore they do not provide new information about the limits of the SIP server, which makes them redundant in SIP server performance analysis. On the other hand, in some special cases these parameters might be useful and therefore we included measurements of these parameters to our methodology.

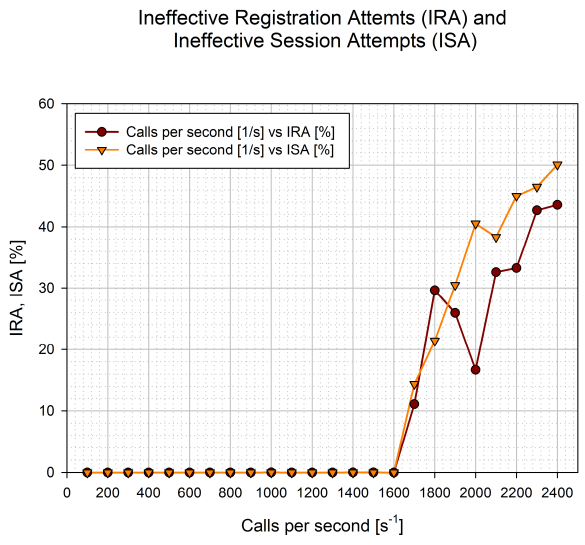


Fig. 12. IRA and ISA (SIP Proxy – 4 UDP listeners).

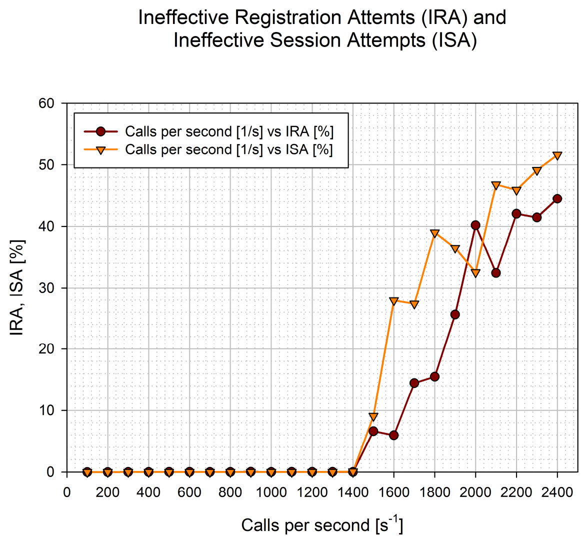


Fig. 13. IRA and ISA (SIP Proxy – 16 UDP listeners).

6 Conclusion

The method of SIP infrastructure testing and benchmarking we presented in this paper was designed for benchmarking SIP based VoIP infrastructure. It allows determining the maximum load of the system, shows the dynamically changing characteristics of the system such as response times and packet delay. It is

useful to decide which system should be installed in a particular environment.
From the presented data this can be learned:

- Maximum number of simultaneous calls that B2BUA can handle in any mentioned configuration.
- Maximum number of calls per second that SIP Proxy can handle.
- B2BUA's effectiveness of codec translation.

Table 1 summarizes some of the knowledge collected about the SIP server and presents an example of output of the measurements with our methodology.

Table 1. Example of the measurements output.

B2BUA (G.711u – G.711u)	
Criterion	*Max. simultaneous Calls [-]*
CPU utilization	660
RRD and SRD	600
Jitter and MPD	600
Total	**600**
SIP Proxy (4 listeners)	
Criterion	*Max. Calls per Second [s^{-1}]*
CPU utilization	1600
RRD and SRD	1600
Total	**1600**

This information can be acquired by examining the presented charts and looking for optimum, which can be simply described as:

- First point where maximum CPU utilization is reached.
- Last point before significant leap in any delay characteristic.

Our designed method could be used in the INDECT project where a set of SIP servers will be operated. This benchmarking test is able to ascertain the

Acknowledgement
This research has been supported by the “Optical Network of National Research and Its New Applications” (MSM 6383917201) research intent of the Ministry of Education of the Czech Republic.

References:
[1] Transnexus, *Performance Test of Asterisk V1.4 as a Back to Back User Agent (B2BUA)*, http://www.transnexus.com/.
[2] Malas, D., Morton, A. *SIP End-to-End Performance Metrics*, IETF, Internet-Draft, September 2009
[3] Poretsky, S., Gurbani, V., Davids, C., *Terminology for Benchmarking Session Initiation Protocol (SIP) Networking Devices*, IETF, Internet-Draft, February 2010
[4] Poretsky, S., Gurbani, V., Davids, C., *Methodology for Benchmarking SIP Networking Devices*, IETF, Internet-Draft, February 2010
[5] Transnexus, *Performance Benchmark Test for OpenSER and SIP Express Router*, http://www.transnexus.com/.
[6] Narayan S., Kolahi S. S., Waiariki R., Reid M., *Performance Analysis of Network Operating Systems*. Proceedings of the 2nd WSEAS International Conference on COMPUTER ENGINEERING and APPLICATIONS, Acapulco, Mexico, 2008.
[7] Farsi H., Mozaffarian M. A., Rahmani H., *Improving Voice Activity Detection Used in ITU-T G.729.B*. Proceedings of the 3rd WSEAS International Conference on CIRCUITS, SYSTEMS, SIGNAL and TELE-COMMUNICATIONS, Ningbo, China, 2009.
[8] Simian C., Georgiev V., *On Some Aspects Regarding Computer Networks' Performance Analysis*. Proceedings of the 13th WSEAS International Conference on COMPUTERS, Rhodos, Greece, 2009.
[9] Voznak, M., *Voice over IP*. VSB-Technical University of Ostrava:, 2008.
[10] Voznak, M., Rozhon, J., *SIP Infrastructure Performance Testing*, In Proceedings of the 9th WSEAS International Conference on TELECOMMUNICATIONS and INFORMATICS , Catania, Italy, May 29-31, 2010, ISBN 978-954-92600-2-1.
[11] Nagi-ki F., Fan Y., Kai-hau Y., *Balancing Throughput and Delay Performance by Effective Shortest Path Routing*. Proceedings of the 9th WSEAS International Conference on APPLIED INFORMATICS AND COMMUNICATIONS, Moscow, Russia, 2009.