FSM BASED SIMULATION FOR PERFORMANCE EVALUATION OF SIP SERVER LOSS-BASED OVERLOAD CONTROL

Pavel Abaev
Konstantin Samouylov
Ivan Sinitsyn
Department of Applied Informatics and Probability Theory
Peoples' Friendship University of Russia
Mikluho-Maklaya str., 6
Moscow 117198, Russia
E-mail: {pabaev, ksam, isinitsyn}@sci.pfu.edu.ru

Sergey Shorgin
Federal Research Center "Computer Science and Control", Russian Academy of Sciences
Vavilova, 44-1,
119333, Moscow, Russia
Emil: sshorgin@ipiran.ru

KEYWORDS
SIP, overload control, loss-based scheme, FSM, simulation.

ABSTRACT

The rapid development of services provided on SIP networks not only define the necessity of creating new equipment and standards but also requires the development of new methods and programming software tools for modeling and analyzing the effectiveness of the overload control mechanisms in SIP-server networks. The modeling process utilizes mathematical and simulation models as well as simulators. Only simulators or simulation tools make it possible to solve problems related to the analysis and optimization of the control parameters. The most appropriate modeler is the simulators reflecting the protocols and functions, which are fully or partially built into the original system.

At present, there are no simulators for modeling the work of SIP servers in overload conditions with an application of mechanisms, which are currently under development by IETF SIP Overload Control workgroup. Simulation tool that supports the main SIP RFCs such as RFC 3261 are proposed in the paper. The architecture of SIP node is developed, and numerical example is presented.

INTRODUCTION

SIP is an application-layer signaling protocol for creating, modifying, and terminating sessions with one or more participants. In November 2000, SIP was accepted as a 3GPP signaling protocol and main protocol of the IMS architecture. In 2002, recommendation (RFC 3261 2002), which determines the current protocol form, was accepted.

The rapid development of the market for services based on the SIP protocol and the growing user needs have revealed a number of shortcomings in the protocol, specifically, in the basic overload control mechanism (mechanism 503). In 2009, Rosenberg, one of the protocol designers, demonstrated in (RFC 5390) the protocols' main shortcomings concerning overload prevention and formulated the main requirements toward the future of overload control mechanism. In mid-2010, the SOC working group was created within the IETF Committee. Its work aims at creating overload prevention mechanisms. The first result of their work was the document (RFC 6357 2011), which was permanently accepted in August 2011. The document provides a discussion of the available types of overload control mechanisms – local, hop-by-hop, and end-to-end, a classification of SIP networks, and presents the overall architecture of overload-control systems. The SOC group's work focuses on developing two hop-by-hop schemes for overload control as this type of mechanisms has a number of indisputable advantages over the other two types (RFC 7339 2014). At present, two overload control schemes have been proposed – one with flow sifting on the sender side (LBOC, Loss-based overload control) documented in (RFC 7339 2014) and one with restricting the flow rate of signaling messages (RBOC, Rate-based overload control) described in (RFC 7415 2015). However, only the basic principles were described in SOCs' documents and methods for calculation of the control parameters were not specified. The control parameters can be determined based on analysis of mathematical models or as the results of simulation modeling. Some approaches for estimating overload control mechanisms and its control parameters are proposed in (Abaev and etc. 2012, Abdelal and etc. 2010, Azhari 2012). As the processes going on in the SIP networks are difficult to describe mathematically and depend on a large number of different factors, the task needs to be solved through the creation of a simulator.

Network dimensioning and scalability is one of the key objectives for Telecommunication Carriers that offer VoIP to their customers. SIP network simulator

Proceedings 29th European Conference on Modelling and Simulation ©ECMS Valeri M. Mladenov, Petia Georgieva, Grisha Spasov, Galidiya Petrova (Editors)
ISBN: 978-0-9932440-0-1 / ISBN: 978-0-9932440-1-8 (CD)

allows sketching out network environment and making up your own SIP solution to see how it works. Rich statistics that are collected will provide us with the capability to estimate and analyze the real metrics of SIP network solution efficiency. Different overload control algorithms can be compared for various SIP traffic profiles.

There are many commercial general-purpose simulation tools such as Anylogic or Mathlab Simulink with a graphical editor, customizable block libraries, and solvers for modeling and simulating dynamic systems. SIP protocol modeling with such tools in accordance with main RFCs and with implementation of overload control schemas is not a trivial task. Furthermore, the only way to estimate delays that are contributed by network layer is a SIP simulator, which uses TCP/IP stack.

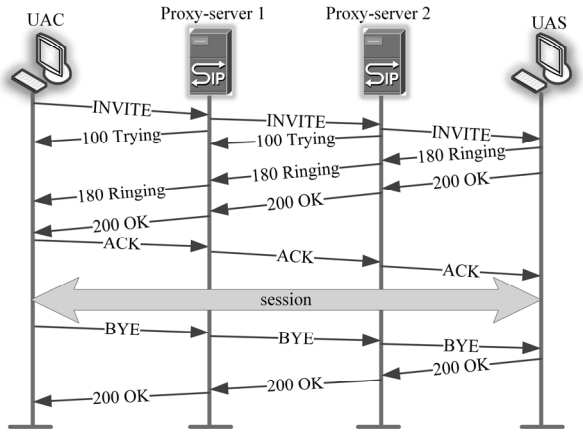
This paper is organized as follows. We analyze IETF experience for SIP-signaling overload control problem solutions. Then we investigate overload control techniques, which are implemented on the server and the client side. And finally, we introduce the architecture of the SIP simulation tool for modeling different overload control techniques.

SIP CALL FLOW AND OVERLOAD CONTROL PROBLEM OVERVIEW

SIP Call Flow Overview

SIP is a request/response-based protocol. User agents (UAs), which take the role of a user agent client (UAC) or user agent server (UAS) for a request/response pair represent end users. A UAC creates a SIP request and sends it to a UAS. On its way, a SIP request typically traverses one or more SIP proxy servers. The main purpose of a SIP server is to route a request one hop closer to its destination. Responses trace back the path the request has taken.

The UAC sends an INVITE request to the UAS to initiate a SIP session as shown in Fig. 1. Each server on the path confirms the reception of this request by returning a 100 Trying response to the previous hop. Instead of forwarding a request, a SIP server can reject it if it is unwilling or unable to forward the request. Once the request is received by the UAS, it typically responds with a 180 Ringing response to indicate that the called user is being alerted and a 200 OK response when the user has accepted the session. After the 200 OK is received by the UAC, it sends an ACK request to complete the three way handshake of an INVITE transaction. The INVITE request is the only SIP request that uses a three way handshake. Sessions can be terminated at any time by sending a BYE request, which is confirmed with a 200 OK response.



Figures 1: SIP basic call flow

Basic 503 mechanism

The SIP protocol provides a basic overload control mechanism through the 503 (Service Unavailable) response code. SIP servers that are unable to forward a request due to temporary overload can reject the request with a 503 response. The overloaded server can insert a Retry-After header into the 503 response, which defines the number of seconds during which this server is not available for receiving any further requests from the upstream neighbor. A server that receives a 503 response from a downstream neighbor stops forwarding requests to this neighbor for the specified amount of time and starts again after this time is over. Without a Retry-After header, a 503 response only affects the current request and all other requests can still be forwarded to this downstream neighbor. A server that has received a 503 response can try to resend the request to an alternate server, if one is available. A server does not forward 503 responses toward the UA and converts them to 500 Server Internal Error responses instead.

Note that RFC 3261 provides, in the case of overload the recipient to discard incoming messages without notifying the sender.

A SIP server overload occurs if a SIP server does not have sufficient resources to process all incoming SIP messages. Several reasons, including Poor Capacity Planning, Component Failures, Avalanche Restart or Flash Crowds and the list of the problems that arise as a result of the 503 mechanism are presented in (RFC5390 2008).

Load Amplification. The supplementary result of the 503 mechanism is the tendency to amplify the load during periods of overload significantly, thus causing further aggravation of the problem and bringing the collapse of the network closer.

Underutilization. RFC 3261 does not cover how the 503 message recipient should react. In fact, there are some network configurations where it is not possible to clearly identify the sender of the message if the sender knows only a domain name of the cluster of receivers. Therefore, the sender may slow down the load to the entire cluster of servers with no overloaded servers, but not to the actual server in overload.

The Off/On Retry-After Problem. When the sender is balancing requests between a small number of the receivers, the 503 mechanism with Retry-After becomes noneffective because of its all-or-nothing technique. The 503 mechanism with Retry-After tends to cause highly oscillatory behavior under even mild overload.

Ambiguous Usages. The standards do not clearly determine when the server must send the message with the code 503, and because of various implementations, the message 503 is used to indicate different states.

Rosenberg formulated 23 requirements to overload control mechanisms in (RFC5390 2008); mechanisms matching them will be able to predict and to avoid or quickly to cope with an overload on the server.

Explicit Overload Control Scheme

The problem domain of SIP overload control can be split into overload control between a user agent and a SIP server and overload control between SIP servers. The first document (RFC 6357 2011) developed by SOC, contains the overload control mechanism classification with local, hop-by-hop and end-to-end schemes, and the following network topologies for "server-server" interoperation – load balancer, multiple sources and mesh.

Current work of the group is focused on the development of two hop-by-hop overload control schemes – Loss-based overload control and Rate-based overload control. The choice in favor of hop-by-hop mechanism was made because of the advantages over the other two mechanisms: the solutions implemented hop-by-hop scheme have better scalability and the scheme requires a SIP entity to aggregate overload status values of SIP servers only that each server communicates with.

The basic idea of LBOC scheme is that the sending entity (SE) reduces the number of messages on receiving entity's (RE) request that will be send to the RE by specified in the request amount of the total number of messages. RBOC scheme operates in the following way: RE informs SE about the maximum message rate which RE would like to receive from SE within a specified period of time. RE sends the control information to SE periodically depending on RE load changes.

Both of these schemes are based on the idea of feedback control loop between all neighboring SIP servers that directly exchange traffic. Each loop controls only two entities. The Actuator is located on the sending entity and throttles the traffic if necessary. The receiving entity has the Monitor, which measures the current server load.

The four Via header parameters ('oc', 'oc-algo', 'oc-validity' and 'oc-seq') are introduced in (RFC 7339 2014) to transfer the control information between two adjacent entities.

The integer parameter 'oc' consisting of 10 digits and its value defines what percentage of the total number of SIP requests are subject to reduction at the SE when the loss-based scheme is used. Analogously, when the rate-based scheme is used it indicates that the client should send SIP requests at a rate of 'oc'-value SIP requests or fewer per second. 'oc-algo' parameter defines the scope of algorithms supported by SE, e.g. 'oc-algo'="loss", "rate". `Parameter 'oc-validity' contains a value that indicates an interval of time (measured in milliseconds) that the load reduction specified in the value of the oc' parameter should be in effect, its default value is 500 ms. Parameter 'oc-sequence' is the sequence number associated with the 'oc' parameter, timestamps usually use as its value.

The message format with the control information, the procedures of choice overload control algorithm, and the behavior of client and server using overload control mechanism are described in detail in (RFC 7339 2014). However, these documents remains open the following questions that need further research:

- Criteria determining the choice of moments for sending messages with control information from SE to RE;
- Rule for choosing the value of 'oc' parameter;
- Rule for choosing the value of 'oc-validity' parameter.

To address these problems we have developed SIP simulation tool that provide with the capability to collect rich statistics of server to server SIP messaging.

OVERLOAD CONTROL TECHNIQUES OVERVIEW

Threshold Overload Control on the Server Side

As a criterion determining the choice of moments for sending messages with control information from SE to RE we propose to use hysteretic control technique (Abaev and ec. 2012). The system during operation changes its state depending on the total number of messages n present in it.

Choose arbitrary numbers L and H such that $0 < L < H < B$, where B is the buffer capacity. When the system starts to work it is empty, $n = 0$, and as long as the total number of messages in the system remains below $H-1$, system is considered to be in normal state, $s = 0$.

When total number of messages exceeds $H-1$ for the first time, the system changes its state to overload, $s = 1$, and RE informs SE that traffic load should be reduced: it stays in it as long as the number of messages remains between L and $B-1$. Being in overload state, RE's system waits till the number of messages drops down below L after which it changes its state back to normal and informs SE about changes, or exceeds $B-1$ after which it changes its state to blocking, $s = 2$, and ask SE for temporary suspension of sending SIP requests. When the total number of messages drops down below $H+1$, system's state changes back to overload, and RE informs SE that the process of sending of messages can be resumed with the current limitations.

Default Algorithm on the Client Side for LBOC case

In the case of LBOC scheme, the default algorithm for throttling incoming to the server traffic is used on the client side. The idea of the algorithm presented in (RFC 7339 2014) is to sift the client's outgoing flow. Let us consider the example of the implementation of the algorithm.

The client maintains two types of requests – the priority and non-priority. Prioritization of messages is done in accordance with local policies applicable to each SIP-server. In situations where the client has to sift the outgoing flow, it first reduces non-priority messages, and then if the buffer contains only priority messages and further reduction is still needed, the client reduces the priority messages.

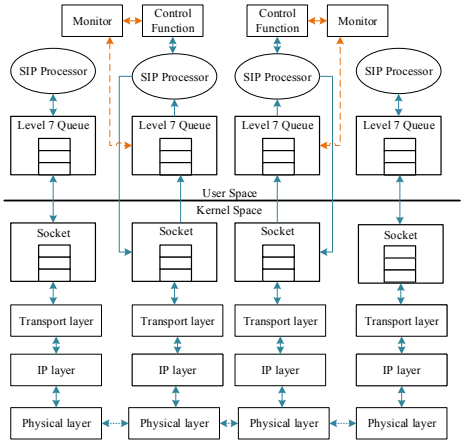
Under overload condition, the client converts the value of the 'oc'$=q$ parameter to a value that it applies to non-priority requests. Let N_1 denote the number of priority messages and N_2 denote the number of the non-priority messages in the client's buffer. The client should reduce the non-priority messages with probability $q_2 = \min\left\{1, q\frac{N_1+N_2}{N_2}\right\}$ and the priority messages with probability $q_1 = \frac{q(N_1+N_2)-q_2N_2}{N_1}$ if necessary to get an overall reduction of the 'oc' value.

To affect the reduction rate with probability q_2 from the non-priority messages, the client draws a random number between 1 and 100 for the request picked from the first category. If the random number is less than or equal to converted value of the 'oc' parameter, the request is not forwarded; otherwise the request is forwarded. Recalculation of probabilities is performed periodically every 5-10 seconds by getting the value of the counters N_1 and N_2.

SIP NETWORK SIMULATOR

Below we will describe the simulation approach and the design of SIP node.

Figure 2 illustrates the steps a packet takes as it moves from the device driver through the Linux kernel to the SIP layer and back to the device driver. Based on this figure, there are three distinct entities involved in processing a SIP packet. The first one of them is Kernel Network Stack, which provides the procedure of receiving of packets.



Figures 2: SIP message path over OS layers

As soon as a packet is received from the physical device, it arrives at the device driver and is transferred to a ring buffer in kernel space. The packet then undergoes processing within the Linux kernel stack before it is handed to the application – SIP layer.

Application Layer provides SIP Packet Processing procedures. The SIP layer has a blocking loop waiting for packets to arrive on the socket. Processing at the SIP

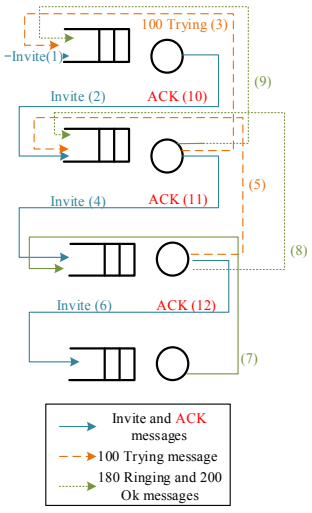
layer consists of two parts: one that is common to all messages and one that is message dependent.

During the common part, the message is parsed, it is classified as a SIP request or response, and then certain operations are performed. Since the proxy operates in stateful mode, a lookup is performed to determine if the SIP transaction is already present. If not, a new transaction is created, otherwise the existing transaction is returned and the SIP message-specific processing phase begins. Once the forwarding, reply or relay decision has been made, a send buffer is created with the updated data for the SIP header, and the packet is sent to the transport layer.

The next layer Kernel Network Stack provides Packet Sending procedures. Once a packet completes processing at the SIP layer, it is passed on to the kernel for forwarding to the UAC/UAS.

First, for parsing, the text based SIP messages are syntactically analyzed, broken down into parts, and converted into internal representations. Second, the newly created internal message representations have to be analyzed to infer on their later processing. For a SIP proxy server this means that it has to determine the messages destination and forward them towards there. For a UAS it means extracting the content of the message, probably displaying something to the user and creating and sending responses. Each of the two parts of processing a SIP message in a proxy server uses a specific amount of processing time. We assumed that parsing of a message always takes approximately the same amount of processing time, which independent of its type and content.

SIP session state chart according SIP call flow described above in SIP overview paragraph is shown in Fig. 3.



Figures 3: Session state chart

The main objective of SIP network simulator is to provide means for the simulation of end-to-end SIP connectivity over TCP/IP transport. To this aim, SIP node model supports SIP INVITE and non-INVITE transaction model, which is implemented in Statechart and State Java classes. Class Transition provide the functionality for changing states of SIP transactions. The UML Java class diagram of SIP node is shown Fig. 4.

Numerical example

The network topology that we modeled consists of UAC, UAS and two SIP proxies. SIP nodes functioned as a single server model with finite capacity buffer. We did not implement any special local policy at the client side to prioritize of dropping of messages. We set the value of thresholds $L = 44$ and $H = 55$, and dropping probability $q = 0.3$ (Abaev and etc. 2012). The dependence of number of successful sessions on input load is shown in Fig. 5.

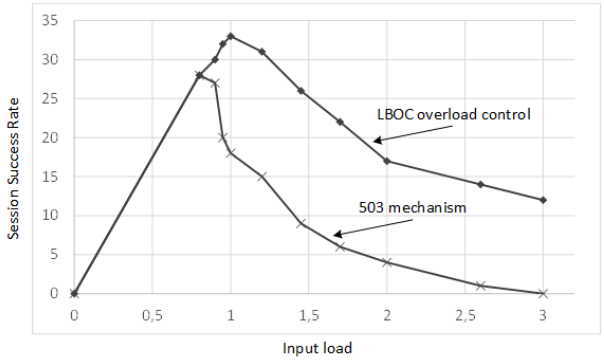


Figure 5: Session success rate

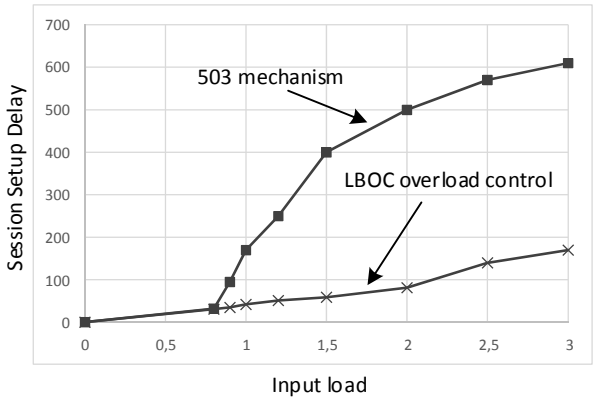
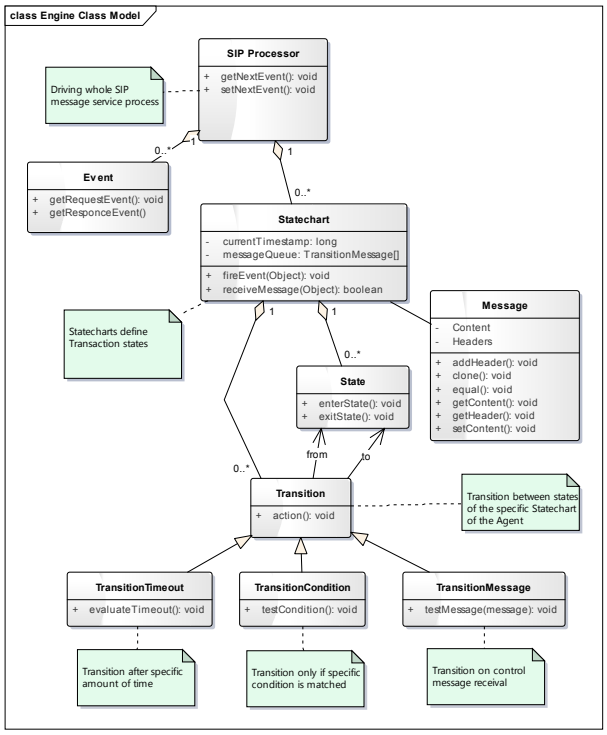


Figure 6: Session Setup Delay

With the increase of input load, session success rate gradually decreases in the case of LBOC overload control and decreases dramatically in the case of 503 mechanism. For sessions that successfully established we calculated mean session setup delay, which

increases up to 610 ms for 503 mechanisms and up to 200 ms for LBOC overload control.



Figures 4: SIP node class model

SUMMARY AND FUTHER STUDY

In this paper, we give an overview of the SIP protocol and basic 503 mechanisms shortcomings. The architecture of SIP network node is proposed and a testbed with implemented at server side hysteretic overload control was constructed. The numeric example illustrates some benefits of LBOC overload control mechanism against basic 503 mechanisms.

For further study RBOC schema is to be implemented in the simulator and a comparative analysis of both schemas will be done for different traffic profiles.

Notes and Comments. This work was supported in part by the Russian Foundation for Basic Research (grant 15-07-03608).

REFERENCES

Abaev, P., Gaidamaka, Yu., Samouylov, K. 2012. "Modeling of Hysteretic Signaling Load Control in Next Generation Net¬works. Lecture Notes". In Computer Science. Germany, Hei¬delberg, Springer-Verlag. -Vol. 7469. 371-378.

Abaev, P., Gaidamaka, Yu., Samouylov, K. 2012. "Queuing Model for Loss-Based Overload Control in a SIP Server Us¬ing a Hysteretic Technique. Lecture Notes". In Computer Sci¬ence. Germany, Heidelberg, Springer-Verlag. -Vol. 7469. 440-452.

Abaev, P., Gaidamaka, Yu., Pechinkin, V., Razumchik, R., Shorgin, S. 2012. "Simulation of overload control in SIP server networks". In Proceedings of the 26th European Conference on Modelling and Simulation, ECMS 2012. Germany, Koblenz. 533-539.

Abdelal, A. and Matragi, W. 2010. "Signal-Based Overload Control for SIP Servers". In Consumer Communications and Networking Conference (CCNC), 2010 7th IEEE. IEEE, Las Vegas, NV, 1-7.

Azhari, S.V. and Nemati, H. 2012. "Stability analysis of tandem SIP proxies". In Communications (ICC), 2012 IEEE International Conference on. IEEE, Ottawa, ON, 1244-1248.

Baghadi, M. and Azhari, S.V. 2013. "Signaling during overload using capabilities of connection-oriented transport protocol". In Electrical Engineering (ICEE), 2013 21st Iranian Conference on. IEEE, Mashhad, 1-6.

Gurbani, V., Hilt, V., Schulzrinne, H. 2013. Session Initiation Protocol (SIP) Overload Control. RFC 7339.

Hilt, V., Noel, E., Shen, C., Abdelal, A. 2011. Design Considerations for Session Initiation Protocol (SIP) Overload Control. RFC 6357.

Hilt V. and Widjaja I. 2008. "Controlling overload in networks of SIP servers". In Network Protocols, 2008. ICNP 2008. IEEE International Conference on. IEEE, Orlando, FL, 83-93.

Montazerolghaem, A. and Yaghmaee M.H. 2013. "SIP overload control testbed: Design, building and Evaluation". In International Journal of Ambient Systems and Applications, Vol. 1, No.2 (June), 17-26.

Noel, E., Williams, P. 2015 Session Initiation Protocol (SIP) Rate Control. RFC 7415.

Ohta, M. 2008. "Performance Comparisons of Transport Protocols for Session Initiation Protocol Signaling." In Telecommunication Networking Workshop on QoS in Multiservice IP Networks, 2008. IT-NEWS 2008. 4th International (Venice, Italy, Feb. 13-15). IEEE, 148-153.

Rosenberg, J., Schulzrinne, H., Camarillo, G. et al. 2002. SIP: Session Initiation Protocol. RFC 3261.

Rosenberg, J. 2008. Requirements for Management of Overload in the Session Initiation Protocol. RFC 5390.

AUTHOR BIOGRAPHIES

PAVEL ABAEV received his Ph.D. in Computer Science from the Peoples' Friendship University of Russia in 2011. He is an Assistant Professor in the Department of Applied Probability and Informatics at Peoples' Friendship University of Russia since 2013. His current research focus is on signaling networks congestion control, performance analysis of wireless 4G/5G networks and M2M communications, applied probability and queuing theory, and mathematical modeling of communication networks. His email address is: pabaev@sci.pfu.edu.ru.

KONSTANTIN SAMOUYLOV received his Ph.D. from the Moscow State University and a Doctor of Sciences degree from the Moscow Technical University of Communications and Informatics. During 1985-1996 he held several positions at the Faculty of Science of the Peoples' Friendship University of Russia where he became a head of Telecommunication System

Department in 1996. Since 2014 he is a head of the Department of Applied Informatics and Probability Theory. His current research interests are probability theory and theory of queuing systems, performance analysis of 4G/5G networks, teletraffic of triple play networks, and signaling networks planning. He is the author of more than 100 scientific and technical papers and three books. His email address is: ksam@sci.pfu.edu.ru.

IVAN SINITSYN received a BSc. degree in Applied Mathematics and Informatics in 2012 from People's Friendship University, Moscow, Russia. Currently he enrolled in a MSc program of the Department of Applied Probability and Informatics at the same university. His present research focuses on performance analysis of NGN, congestion control of signaling networks, mathematical modeling and performance analysis of computer and communication systems. His email address is: isinitsyn@sci.pfu.edu.ru.

SERGEY SHORGIN received a Doctor of Sciences degree in Physics and Mathematics in 1997. Since 2003, he is a professor, and since 2015 he is a Deputy Director of Federal Research Center "Computer Science and Control", Russian Academy of Sciences. He is the author of more than 100 scientific and conference papers and coauthor of three monographs. His research interests include probability theory, modeling complex systems, actuarial and financial mathematics. His email address is: sshorgin@ipiran.ru.