Software modelling of network traffic filtering process in information system regulating the access to Internet content via HTTP

K I Budnikov and A V Kurochkin

Institute of Automation and Electrometry of Siberian Branch of the Russian Academy of Sciences, 1, Academician Koptyug Avenue, Novosibirsk, 630090, Russia

E-mail: budnikov@iae.nsk.su

Abstract. One of the most popular ways to restrict the access to information hosted on the network, which is necessary to comply with the requirements of security, copyright, labor regime, etc., is filtering of requests to an Internet resource by its URL. It allows using a selective approach to information resources located on one and the same IP-address and prohibiting access when necessary. It can be considered as the most balanced in terms of the advantages and disadvantages. The authors carried out simulation computer modeling of the filtering process for the information web-system, in which a filtering device is built in using specially developed software. Filter processes requested flows to a server by a group of users connected to it and countered information from a server. All network interfaces were simulated by software, and all network data flows proceeded in the memory of the simulating computer. This approach allows eliminating the influence of the network infrastructure on the operation of the model. Two filtering methods: the standard method of preliminary analysis of requests and the proposed post-analysis method were compared in the process of modeling. The decrease of the average waiting time for a response from the web-server when a user request to a web-resource passed through the simulated filtering device, which worked in the post-analysis mode, up to 14% compared to the device that worked in standard mode were shown during computer simulation. Filter throughput increased by up to 54%.

1. Introduction

The appearance in the last years of the large number of information Internet resources, access to which needs to be restricted for a number of reasons, including age and moral and ethical criteria, requirements for compliance with security, copyright, labor regime, etc., requires improvement of methods and tools of ensuring a selective ban on access to information on the network. Currently, these methods include: restricting access by IP address, by URL, changing requests to DNS servers, using proxy servers, packet filtering. These approaches combine both advantages and disadvantages [1].

The most popular method of filtering of requests to a resource is filtering by its URL. It can be recognized as the most balanced in terms of its strengths and weaknesses. It allows making a selective analysis of requests to information resources located on the same IP-address and prohibiting access to a resource when necessary. Technically it is possible to use a specially installed program to perform the filtering process for a separate Internet access device (computer, smartphone, tablet) [2, 3], and a filtering device with Internet access for a group of devices connected to it [4-7]. To study the process of filtering in the framework of this paper the authors consider a model of an information web-system...
in which the filtering device processes information flows from a group of connected subscribers.

The standard filtering algorithm by the URL presupposes a preliminary check of the request at the input of the device, and only by its results the request either passes further or is blocked. It is described, for example, in patents Cisco Technology, Inc. [5] and Ironport Systems, Inc. [6]. The verification itself takes the time necessary for intercepting the request, extracting the URL from it and searching for it in the lists of forbidden addresses. At this time, the request is delayed by the filter.

Improving filtration method, we came to a filtering algorithm with post-analysis of requests [7]. It was expected that it would provide a reduction of the waiting time for a response from the web-server and an increase of the throughput of the filtering device and as a result of the system as a whole. A computer simulation to confirm this assumption was carried out. It will be described and discussed below.

2. Model of the system and algorithms of filtration

For experiments the authors created web-system model (Figure 1) which consisted of a web-server, its clients, sending requests to web-resources located on it and a filtering device that regulates client access to resources. The task of its simulation is focused on the problem of the fastest passing requests through the device and its filtering.

![Figure 1. Simulated web-system](image)

Each channel contains the following modules shown on Figure 1: network packet reading (MPR), packet sorting (MPS) and packet transfer (MPT). The central module of the model is a packet analyzer (MAP) common to both channels. Interaction with communication lines occurs through the modules of network interfaces MNI1 and MNI2. The operation of the filtering device in the simulated system is described in more detail in [8].

![Figure 2. Time diagrams for filtering of requests to the web-server using the standard algorithm (A) and the algorithm for post-analysis of requests (B).](image)

The following sequence of actions is used in the standard filtering algorithm mentioned above. A device that filters by an URL intercepts a user's request passing through it, extracts from it the address of the resource being accessed. Further, depending on the built-in algorithm, this address is searched in
the lists of resources that are prohibited or allowed. If the URL that is being accessed is allowed, the request is passed to the Internet, reaches the server with the required resource, and the server returns a response with the requested information. If access to a resource of user interest is denied, the request is blocked by the filter. The timing diagram of events that occur during filtering by this algorithm is shown in Figure 2 (A).

Using post-analysis of requests to improve this algorithm gives a gain in the time of a user request passing through a filtering device compared to the standard packet processing sequence. It includes the time spent on determining the TCP session for each packet, forming a user request from the packets, extracting the identifier of the requested resource URL and checking the request for access to the requested resource using internal lists of forbidden URLs. The timing diagram of events that occur during filtering by this algorithm is shown on Figure 2 (B). Both algorithms are discussed in detail in [7].

Inside the simulation process the authors compared 2 filtering algorithms mentioned above because the main focus of the research is on the problem of the fastest passing requests through the filter. The authors decided to simulate the system, putting requests, responses and filtering device itself into a computer memory. This approach allows getting the result of the comparison, which would not depend on the communication lines that connect the components of the system, but contained only a comparison of the processing times of requests to the web-server and its responses.

3. Structure of software simulator and algorithm of working

The simulation process takes place using preliminary prepared files containing network packets recorded in advance. They are exchanged between the web-client and the web-server during an HTTP session. The authors replaced the network interface modules (MNI) with network simulation modules (MNS) in the above model. Other modules remained unchanged. Both MNS have the same architecture and operation algorithm, and the role of each module as a web-client or server is set when the simulator of the system is configured. In the process of simulation, the MNS uses the above-mentioned file containing network packets that the web-client exchanged with the server during the HTTP session. Each packet in such a file has a number of attributes, the main one being the identifier of the sender of the packet. Each MNS performs the main work on simulation.

![Figure 3. Structure of the network simulation module and its interaction with MPR and MPT](image_url)

Each of them includes four main components (Figure 3):
- Simulation thread.
- Session descriptor table.
- Buffer of packet transfer.
- Ready-to-read descriptor list.
Simulation thread provides capability of the module to simulate the operations of receiving and transmitting packets. Session descriptor table contains all the information about the process of emulation of HTTP sessions during the test. Each descriptor contains copy of the file with session packets, a pointer to the current packet being processed, a repetition counter for this session, and a number of other auxiliary parameters. The size of the table, the number of repetitions and the file with packages are set in the parameters of the test. Buffer of packet transfer accumulates packets devoted for transmission. Discipline of packet service is consistent with the principle of FIFO. Packages come from MPT and are added to the end of the list. The internal simulation thread reads packets from the list, starting with the first one. Ready-to-read descriptor list consists of descriptors of sessions which are ready for reading. This list contains descriptor numbers in which the current packet corresponds to the incoming packet for the current session state. Numbers are added by the simulation thread to the end of the list. When requesting a packet, the MPR selects a descriptor number randomly and reads the corresponding current packet.

Figure 4 shows the algorithm of the simulation thread work. It consists of the following actions. When a simulation experiment runs, both MNS are loaded into memory and initiated. As a part of this process, they get their roles: the client or the server. Their descriptor tables are being formed. Packages from the specified session file are recorded into each descriptor and modified so that, first, the session is unique with respect to other descriptors, and second, the task of determining the package to the descriptor is most simplified. The current packet pointer is set to the first packet. The repeat counter is reset. The packet transmission buffer is created empty. At startup, the simulation stream is set to the waiting state of the packet for transmission.

![Figure 4. Workflow chart for algorithm of the simulation thread](image)

The list of descriptors of session which are ready for each MNS is formed differently. In the MNS, which emulates the exchange with a web-server, the list is created empty, and when the MNS of web-client is loaded, the numbers of all descriptors are entered in this list. Therefore, after the launch of the corresponding read thread in the MPR, the processing of HTTP session packets begins. They are fed to the MPR input. The main cycle of the simulation process starts.
When the MPT module adds a packet to the buffer of packet transfer, the simulation thread is activated. The packet is read from the buffer and determined to belong to one of the session descriptors. The simulation thread goes to the next packet in the session (increments the current packet pointer in the descriptor). The sender ID of this packet is analyzed. If the packet needs to be sent to the MPR, then the descriptor number is added to the list of ready-to-read descriptors. Next, the thread checks for data in the packet transfer buffer. If unprocessed packets are present, they are processed in the same way. If the buffer is empty, then the simulation thread pauses. In case of reaching the end of the session after the increment of the current packet pointer, this pointer is redirected to the first packet in the session and the repetition counter in the descriptor increases. If the repetition counter reaches the value specified in the test parameters, this session descriptor is disabled and no longer participates in the simulation process. The experiment is considered complete when all descriptors are disabled.

Reading of packets from the session descriptor table follows a similar algorithm. The packet reading thread loads the number of one of the descriptors that is ready for reading and copies the packet into the specified memory area. It then moves to the next packet in the session and analyzes the sender ID of this packet. If the packet also needs to be sent to the MPR, the number of the corresponding descriptor is returned to the list of ready-to-read descriptors. Packet that is read goes to the MPR, and the reading thread continues processing of descriptors. When the end of the session is reached, the current packet pointer is set to the first packet and the session repeat counter increases. The corresponding descriptor is no longer serviced when the repeat counter equals the value specified in the experiment parameters.

4. Simulation experiments and discussion
Using the above presented web-system simulator, the authors compared the filtering algorithms mentioned earlier. They are a standard filtering algorithm by URL and the improved algorithm based on post-analysis of requests to a web-resource. The work of the system in the process of testing was simulated with different intensities of request streams from clients as well as sizes of responses from the web-server ranging from 1Kbytes to 100Kbytes.

![Figure 5](image5.png)  
**Figure 5.** Dependencies of response time from the web-server (µs) on the intensity of virtual network streams in the emulated system (Mbit/s). Size of the response from web-server equals to 1Kbyte

![Figure 6](image6.png)  
**Figure 6.** Dependencies of response time from the web-server (µs) on the intensity of virtual network streams in the emulated system (Mbit/s). Size of the response from web-server equals to 100Kbyte

The authors made simulation using an office computer with an Intel Core i7 870 4 × 2.93Ghz processor and 4Gb of memory. In the process of testing the following system characteristics were investigated, such as: the number of processed HTTP sessions per 1 second, the intensity of virtual network streams, and the client's waiting time for a response from the web-server. The virtual exchange rate reached 7.7Gbit/s (8770 requests per second) with the size of server responses of 100Kbytes. The
number of requests per second reached 135600 (1.9Gbit/s) with the size of server responses of 1Kbytes. Figures 5 and 6 show graphs of the response time from the web-server (µs) as a function of the intensity of virtual network flows in the emulated system (Mbit/s) for standard filtering of requests (A) and filtering with post-analysis (B) for the size of the response from web-server of 1Kbyte and 100Kbyte.

As it seen from Figures 5 and 6, simulation showed the decrease in the average waiting time for a response from the web-server when a user request to a web-resource passed through the emulated filtering device, which worked in the post-analysis mode, compared to the device that worked in the preliminary query analysis mode. At the end of the initial segment of the graph (its linear part) for the standard (A) algorithm, it reached 14% for server responses of 1Kbyte and 11% for server responses of 100Kbytes. Filter throughput increased by up to 54% with 1K server responses and up to 22% with 100K server responses. The obtained values refer to a specific modelling computer and the model of the system that is implemented on it. In other cases, results can be obtained that differ from the shown above.

5. Conclusion and future work
The solutions for problems that cannot always be solved or their solution is difficult to find using other modelling methods can be found using computer simulation. A software simulation on one computer of the information web-system with a built-in filtering device was presented in this paper. Specially created for this aim software emulates a filtering device, network interfaces, communication lines, clients, and a web-server. This approach allows excluding the network component of the system by placing a filtering device in the centre of the model and building infrastructure of modelled system around it. Functionality of the filter model is not being changed during simulation when for example the algorithm of working is being changed. It allows comparing filtering algorithms by conducting experiments on the same computer with the same set of data being processed. In this research work the authors compared the standard filtering algorithm and the improved algorithm using the post-analysis of requests in the process of modelling. The improved algorithm showed its advantage and confirmed perspective of use in filtering devices. The authors considered relatively small gain in waiting time for a response from a web server during simulation for the improved filtering algorithm as the need for hardware support for it. The perspective of further development of the filtering device is seen as improving its algorithm in conjunction with its hardware support.

6. Acknowledgments
The reported study was funded by the Ministry of Science and Higher Education of the Russian Federation. Project # AAAA-A17-11706061006-6

References
[1] Apetyan S, Kovalev A and Veybach A 2013 Filtering content in Internet. Analysis of international practice (Foundation of Civil Society Development) Available at: http://civilfund.ru/Filtraciya_Kontenta_V_Internete_Analiz_Mirovoy_Praktiki.pdf
[2] Osipov G S, Tihomirov I A and Sochenkov I V 2012 Method and system for web content filtration RU Patent No. 2446460 C1. IPC G06F 21/20 (2006.01) (27 March 2012) Official Bulletin of Rospatent "Inventions. Useful Models"
[3] Bellinson C, Evans C, Fravert H and Taylor W 2004 Content filtering for web browsing U.S. Patent No. US20040006621 A1, IPC G06F17/30, G06F13/00, G06F15/00, G05B1/00, G06F17/00 (08 January 2004)
[4] Kumar B, Sekhar Ch and Kodukula N 2014 Pre-emptive url filtering technique U.S. Patent No. US 8,838,741 B1 Int. Cl. G06F 15/16 (16 September 2014)
[5] Balasubrahmanian J, Daftary K, Yarlagadda V R and Kumar K 2006 System and method for url filtering in a firewall U.S. Patent No. 2006006469A1, Int. Cl.G06F 15/16 (2006.01) (23 March 2006)
[6] Bloch E et al 2010 Apparatus for monitoring network traffic U.S. Patent No. 7849502 B1, Int Cl
G06F 15/16 (2006.01), G06F 11/00 (2006.01) (7 December 2010)

[7] Budnikov K I, Kuruchkin A V, Lubkov A A and Yakovlev A V 2016 Regulation of access to web-resource based on post-analysis of http-requests Proc. Int. Conf. Information Technology and Nanotechnology (ITNT 2016) CEUR Workshop Proc. 1638 284–9 Available at: http://ceur-ws.org/Vol-1638/

[8] Budnikov K I, Kuruchkin A V, Lubkov A A and Yakovlev A V 2018 Experimental Study of Symmetric Computer Model of Http-Filter Proc. of the 2018 3rd Russian-Pacific Conf. on Computer Technology and Applications (RPC) 18–25 August 2018 (Vladivostok, Russia) Available at: https://ieeexplore.ieee.org/document/8482147 DOI: 10.1109/RPC.2018.8482147