Internet Traffic Filtering System based on Data Mining Approach

Valentina V. Glazkova, Vladimir A. Maslyakov, Igor V. Mashechkin, Mikhail I. Petrovskiy

Computer Science Department of Lomonosov Moscow State University

***Abstract*—Today it is hard to imagine education in schools, universities and commercial organizations without usage of Internet resources. At the same time monitoring, control and filtering of Internet traffic are still hardly ever used. One of the main causes of low usage of internet traffic filtering systems is the dynamic and rapidly growing nature of modern Internet, which makes traditional tools, methods and approaches not applicable to this problem. We propose applying new Data mining multi-class multi-label classification methods to the problem of Internet traffic filtering and a scalable, distributed system architecture, based on these methods.**

I. Introduction

Recent researches in European Union, USA and Russia showed that problem of Internet filtering (traffic filtering) is of big interest at this time.

According to a survey, carried out in USA by analytical company Salary.com [1] each year personal usage of Internet resources during working time costs American employers about 750 billions of dollars.

A survey, recently carried out in European Union [2], during which about 600 hundred different companies and more than 2000 employees were questioned, showed that only 30% of these companies used Internet Traffic filtering software and 90% would like to use it.

In Russia the problem is of big interest because of national project "Education", during which it is planned to connect to Internet more than 50.000 schools, where it will be necessary to control incoming traffic in order to prevent access to unwanted (antisocial, offensive, erotic, etc) content.

Usually the main goals of traffic filtering are:

1. to prevent access to unwanted content. A common scenario for schools and libraries;
2. to detect harmful content (viruses, spam, trojans, etc) or links to such content [3];
3. to detect possible intrusion threats or suspicious traffic[4];
4. to reduce number of leakages of confidential information;
5. to prevent unwanted usage of Internet during working time.

However, modern Internet consists of lots of dynamic and constantly changing resources and the usage of Internet in organizations is very heavy. It leads to serious requirements, which modern Internet traffic filtering software should satisfy:

1. the ability to process and filter information online, which means that end users will not experience significant delays because of traffic filtering;
2. high precision of filtering (low rates of false-positive and false-negative errors);
3. the ability to process dynamic content;
4. the ability to adapt to new types of resources and filter resources using its content as well as its metadata;
5. scalability, the ability to be installed in organizations of different scale.

To meet these requirements a new type of Internet traffic filtering system should be developed. We think that it should be based on Data mining methods to meet the third and fourth requirements, it should use comparably fast and precise methods, such as for example multi-class multi-label classification [5] to meet the first and second requirement and it should have a distributed , scalable architecture to meet the fifth requirement.

II. Related work

Currently there are many traffic filtering systems, developed and used worldwide. There are commercial, freeware and open-source implementations.

To name a few:

1. Poesia, an open-source system developed in European Union and one of the first to be based on Data Mining

Manuscript received April 10, 2007. This work was supported by 06-07-08035-ofi, Leading Scientific Scool Support Grant No. 02.445.11.7427.

V. V. Glazkova is a post-graduate student with Computer Science Department of Lomonosov Moscow State University, Building 2, MSU, Vorobjovy Gory, Moscow, 119992, Russia Phone: +7 495 9391789, Fax: +7 495 9391988 (e-mail: glazv@mail.ru).

V. A. Maslyakov is a post-graduate student with Computer Science Department of Lomonosov Moscow State University, Building 2, MSU, Vorobjovy Gory, Moscow, 119992, Russia Phone: +7 495 9391789, Fax: +7 495 9391988 (e-mail: maslyakov@gmail.com).

I. V. Mashechkin is a professor with Computer Science Department of Lomonosov Moscow State University, Building 2, MSU, Vorobjovy Gory, Moscow, 119992, Russia Phone: +7 495 9391789, Fax: +7 495 9391988 (e-mail: mash@cs.msu.su).

M. I. Petrovskiy is an assistant professor with Computer Science Department of Lomonosov Moscow State University, Building 2, MSU, Vorobjovy Gory, Moscow, 119992, Russia Phone: +7 495 9391789, Fax: +7 495 9391988 (e-mail: michael@cs.msu.su).

methods but with several limitations (no personalization of traffic, no support for languages other then European, etc) [6].
2. CyberPatrol, a commercial system, giving an ability to analyze http, p2p, instant messaging and others types of traffic [7].
3. SurfControl, a commercial system with an ability to personalize traffic (to distinguish who is requesting particular type of resource) [8].
4. NetNanny, a commercial parental control system [9].

Analyzing these and many other filtering solutions we suggested a classification of existing systems.

Modern traffic filtering system differ in:

Scale. There are complex solutions of country scale, such for example are being developed in China or Saudi Arabia; solutions of internet service provider scale, AOL for example offers its customers a parental control service; of local area network scale, such are being deployed in separate local network for purposes of internal traffic analysis .

Type of analysis shows what type of resource data is analyzed, its metadata (url of resource, type of content, etc), content or both.

Time of analysis shows when the analysis is performed, online while user requests some resource or offline, in background with no incoming requests.

Type of caching determines where results of analysis are stored: in a centralized repository, usually available in Internet, or in a local repository, not available outside of local area network.

Type of blocking determines whether blocking decision is based on static lists of allowed and forbidden resources, some decision trees or resource categorization.

If we categorize resources, then there are several different approaches for **categorization:** tree hierarchy, similar to Google Directory[10], static(immutable) list of categories, or dynamic lists, similar to tagging systems[11].

Each type of system requires different approach and each has its own subset of requirements. Our subject of interest are **systems of LAN scale, which analyze http traffic**.

Along with classification we suggested to use following parameters to measure performance and effectiveness of existing Internet filtering systems:

1. speed of filtering, usually measured in Kbytes/sec or just seconds to point out possible delays associated with traffic filtering;
2. false-positive errors, cases when system forbids access to legal resource (legal in sense of organization, where the system is deployed);
3. false-negative errors, cases when system allows access to forbidden resource;
4. precision of filtering, ratio of correctly allowed and correctly forbidden resources to the total number of analyzed resources.

Today, most of the local network filtering systems uses signature methods to Internet filtering problem, which impacts on speed and precision of analysis. The common working scenario of Internet filtering system, based on signature methods is following:

1. Work of a system starts with bringing signature database of analyzed resources to up-to-date state. This work is commonly done by human experts, who update database with new signatures.
2. During signature database update system can mark some resource A as “positive” or “legal”, because at the moment of update its content is absolutely harmless.
3. After update of signature database system starts to process requests from users in real-time.
4. If some user requests Internet resource A, this request is redirected to Internet Filtering software, which allows access to resource, because during update phase this resource was marked as harmless.
5. But after update the content of resource might have changed to harmful. As a result user gets access to undesired content.

This and other factors lead to the following drawbacks of modern Internet Filtering software:

1. inability to analyze traffic in real-time, which is essential when content of one and the same resource can change from time to time, what is a common feature for modern Internet resources
2. speed and precision of analysis that significantly depends on quality of centralized knowledge base, that itself depends on speed and accuracy of updates by third-party organization and its human experts. This is again a hard task because of Internet at state-of-the-art.
3. analysis of Internet resources, based solely on resource metadata without using its content.
4. inability to personalize traffic to distinguish who precisely and what type of resource is requesting.
5. inability to add new types and categories of resources in order to adapt to each organization needs.

All these drawbacks make existing solutions hard to customize and implement in real organizations with existing Internet environment.

Sometimes signature database doesn’t contain analyzed resource, in this cases content analysis is being applied along with signature methods. This content analysis is based on searching for forbidden keywords, set by an administrator, in an analyzed document. Though such approach is very effective in speed terms, it has serious limitations in precision and high rates of false-positive errors.

Absolute majority of modern systems use similar architecture: a cache proxy server (Squid[12], Shweby, Jigsaw or other) redirects traffic to Internet Filter Kernel, which perform some analysis. Protocol between cache proxy server in most cases is ICAP (Internet Content Adaptation Protocol) [13] or sometimes some original protocol, based on HTTP, XML-RPC[14] or SOAP[15].

The main difference lies in organization of Internet Filter

Kernel and additional modules. Usually the main logic is concentrated in the kernel itself. During analysis stage, kernel analyzes requested URL, url's content retrieved from Internet, user's rights and makes some decision on whether to allow or to block access to requested resource.

However, some systems, such as Poesia[6], suggest a slightly different approach. Kernel main function is to control incoming requests and responses from cache proxy server and store results in a database while the main logic of decision making is implemented in separate modules, thus allowing us to switch between different decision making algorithms without modifying kernel. Decision making algorithms can vary significantly from simple ones, based for example on associative rules to complex ones, based on natural language processing techniques and semantic extraction[16]. Simple ones show good speed results but usually poor precision, while complex ones lack in speed but has good precision.

The main problem today is absence of standardized test benchmark and set of web resources for Internet Filtering Software. That's why each Internet Filter System uses its own test data, and the results of testing can be hardly compared to each other.

Though average precision shown by modern systems is about 80%, developers of Poesia System[6] performed a vast analysis on different test data and using different filters and decision making algorithms[17]. They tested two main types of filters: light filters, which perform lightweight text analysis, usually tokenization and/or stemming, and heavyweight filters, which involve complex text analysis techniques like lemmatization, natural language processing and others.

The results of these tests revealed, that for example for English filters: lightweight filter showed average precision of 93% with 3% of false-positive errors (so-called over-blocking), while heavyweight filters showed average precision of about 95% with rate of false-positive errors increased to 3.5%. On other hand difference of time of analysis (speed of filtering) between lightweight and heavyweight filter is quite significant and could differ by one order of magnitude, depending on the size of Internet resource and number of unique words in document. On average this numbers are 0.1-0.2 seconds for lightweight filters and about 1-3 seconds for heavyweight ones. These tests showed that using heavy text processing algorithms don't dramatically increases accuracy of traffic filtering, but significantly decreases the speed of analysis. This results in delays that are quite significant for end users and are not applicable in real-time environment.

III. Our Approach

We suggest a scalable distributed architecture, based on data mining multi-class multi-label classification methods[5] instead of signature methods.

By applying these methods the working scenario of Internet Filtering system will become as following:

1. Work of a system starts with a fully automatic process of learning. During this process system is trained on test data.
2. If user again requests resource A with some undesired content, request is redirected to Internet Filtering system.
3. Unlike with signature method approach system performs full analysis of content in real-time and assigns some labels to analyzed resource.
4. Based on classification labels, resource metadata and user rights, system forbids access to undesired content.

This approach will allow us to get:

1. necessary speed of analysis (decimal fractions of a second);
2. necessary precision of analysis (more than 90% with 1% of false-positive errors);
3. adaptation and self-learning, which will allow to adapt to specific organization needs;
4. scalability of result system, which will allow to deploy system in organizations of different scale;
5. autonomy from external knowledge bases and human experts.

The only significant disadvantage is necessity of availability of test training data. For this purpose one of standard test dataset can be used or it can be once compiled by organization, where the system will be deployed.

The design of suggested architecture is shown in fig. 2.

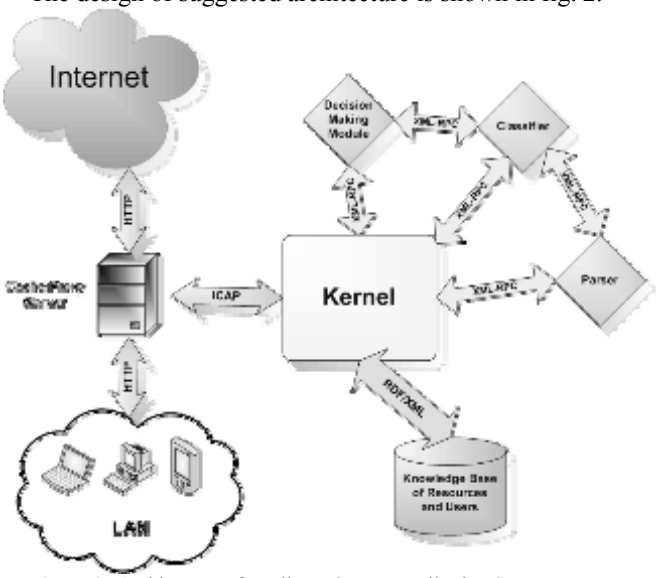

Figure 2: Architecture of Intellectual Internet Filtering System

It consists of:

1. Cache proxy server. It was decided to use thoroughly tested and stable version of Squid with ICAP support [12].
2. Kernel, responsible for accumulating requests from cache proxy server and storing information about users, categories and analyzed resource in a knowledge base.

3. Decision Making Module, responsible for making decisions what resources to allow each user or group of users.
4. Classification Engine, used to classify resources and assign each new resource a set of appropriate categories.
5. Parser, responsible for parsing html documents and extracting links.

Cache Proxy Server is used in more than 90% of systems of this scale. Today there are plenty of different available cache proxy servers, to name a few: Squid, Jigsaw (a W3C Web Server Platform)[18], Shweby, Microsoft ISA Server and many others. First of all it was decided to use existing protocol for filtering HTTP requests and responses, rather then invent our own. ICAP[13] was chosen because it fulfilled all our requirements , it is a standard solution, proposed by IETF, and it has many implementations. Another advantage of using ICAP is that Squid has a module that supports it. The main idea of ICAP is that a cache proxy server has a built-in ICAP client, which redirects all new requests and responses to ICAP Server, in our case a part of Kernel. The protocol itself is very similar to HTTP with three main commands: OPTIONS(used to get options from cache proxy), REQMOD (used for filtering incoming requests from users) and RESPMOD (used to filter responses from Internet).

The Kernel is the central part of the system. It has a built-in ICAP server, which gets requests from Cache Proxy Server's ICAP client. Kernel functions are the following:

1. to control the process of filtering, i.e. understand who is requesting information, store each request in the knowledge base, ask decision making module whether to allow current user access to information or not, store results of decision making and classification in a knowledge base.
2. to provide API for other modules, for example an API for storing links, extracted by parser from resource content, an API for decision making module, that can request additional information about resource or user, etc.
3. to provide a web interface to its knowledge base, which will allow users and administrators of the system to watch some statistics and customize the system.
4. to identify who is requesting information. Currently identification is performed by IP address. But technically it is possible to add LDAP or other types of identification.
5. to store white lists of allowed domains and ip addresses, black lists of forbidden domains and ip addresses, users of system and their rights for different categories of resources.

User hierarchy is organized in Unix-like form. Each user can belong to one or more groups. Each user or group is assigned a white and black list of allowed and forbidden domains and ip addresses and a list of allowed and forbidden categories of resources.

To identify resource we use its url[19]. Therefore each request is associated with time of request, user who requested it and a resource, identified by its url.

For interoperability with other components it was decided to use XML-RPC[14] protocol due to its simplicity, a great number of libraries supporting different languages, its stability and speed of execution. Using XML-RPC allowed us to write components in different languages and deploy them on different physical machines. Another advantage of XML-RPC that modern XML-RPC libraries are rather lightweight and efficient and can easily cope with hundreds of simultaneous requests.

One of the main parts of the system is Decision Making Module (DMM). The sole purpose of this module is to analyze data coming from the kernel and make decision whether to allow or block certain user access to a certain page.

Decision making module has two functions. One analyzes and filters requests, second filters responses with content, received from Internet.

The first one is called by kernel during the stage of processing user's request. On this stage kernel passes following parameters to DMM: user information, as mentioned before we use ip address to identify user but are working on other types of identification, url resource to identify what is being requested, meta information about resource, i.e. all headers, retrieved from HTTP request. Using this information DMM tries to make a decision. The decision can be made if requested domain is in white or black lists for current user or if resource was classified before and hasn't changed since then.

If this information is insufficient DMM asks Kernel for resource content. Kernel redirects DMM's request to Cache Proxy Server, that downloads content from Internet and returns it to Kernel. Kernel invokes second function that filters responses, passing again user information, url resource, additional metadata, such as resource content-type, date of last modification and all other metadata, retrieved from HTTP response, and resource content.

DMM can now make decision using for example resource content type (some organizations might forbid access to flash or video content). If DMM needs some information about content it requests this information from the classification engine.

The main intellectual part of our system is the classification engine or classifier[5]. It allows to specify for each organization its own set of categories. Basing on this set each new resource is classified and assigned appropriate categories. Classification Engine has its own ranking algorithm, which can return a ranked set of categories or ranked set with probabilities. Ranking algorithm determines which categories are more appropriate to the resource and which are less. Threshold function decides which of categories are essential and which are not.

For example if category set equals {sports, news, politics, entertainment, terrorism} then a new incoming resource can be assigned to these categories with probabilities: {terrorism 0.9, news 0.7, politics 0.5, entertainment 0.1, sport 0.1}, meaning that resource contain most probably information about terrorism, less probably it is a news information and less probably it has to do with politics. A threshold function allows us to cut off unnecessary categories, in our example they are entertainment and sports. Moreover we can trust ranking algorithm and for a specified resource just get essential categories without any probabilities, e.g. {terrorism, news, politics}.

Classifier during analysis uses parser to modify HTML content to document vector appropriate for classifier. Another important function of parser is to extract links and store them in the knowledge base. Moreover, classifier may request information about hyperlink categories if they were classified before. Link structure is used by classifier to increase the precision of classification[20].

Among other benefits are independence from language of content, which is achieved by using n-gram approach for text analysis and an ability to dynamically set list of categories. N-gram approach suggest dividing the document in equal chunks of size n, each of which is concerned as a word, before all these symbols are converted to Unicode format to prevent disambiguates. As a result we don't make any difference between languages. Moreover, pages with text in different languages can be analyzed as all others. Dynamic set of categories means that each organization that deploys system can change or set its own set of categories of resources. The only requirement, that data in training set should contain examples of resources of new category.

The typical scenario of traffic analysis in our system is the following:

1. User from local area network requests some information from Internet.
2. Cache Proxy Server redirects request to Kernel
3. Kernel performs primary analysis of request. Primary analysis includes checking whether resource's domain or ip is in black (forbidden) or white (allowed) lists and whether resource was classified before.
4. If kernel needs content of resource and resource wasn't classified before it requests content for Cache Proxy Server.
5. Cache Proxy Server gets response for user's request from Internet and redirects it to Kernel.
6. Kernel redirects request and response to Decision Making Module.
7. Decision Making Module extracts content from response and redirects it to classifier.
8. Classifier first asks parser to parse document and extract necessary links (actually as many as it will be possible).
9. The result of parsing is returned to classifier in form of some document vector.
10. Classifier performs classification, assigns given content some classes and returns result to Decision Making Module.
11. Decision Making Module based on user rights and categories of requested resource decides whether to allow current resource to user or not.
12. The result is returned to Kernel, which returns Internet resource if DMM allowed access to resource or some error page if access was forbidden.
13. The result page is redirected to Cache Proxy Server, which returns it to user in local area network.

The current research is mostly dedicated to organization of knowledge base. Today we use a common relational database with a self-developed database schema. But such organization has several disadvantages:

1. serialization of a graph structure of Internet resources to a relational structure of a database;
2. inability to process hard queries with knowledge base.

The advantages are:

1. a big number of available, stable and reliable sql databases;
2. efficiency of available sql databases.

Switching to ontology representation will allow:

1. to store resource and user hierarchy more efficiently;
2. to process harder knowledge base queries;
3. to visualize data more efficiently, what is relevant due to the potential size of knowledge base;
4. to use complex decision making algorithms to improve filtering precision.

The suggested approach is to use one of existing ontologies to organize users and resources. For user description it is suggested using a FOAF ontology[21] due to its description power.

For resource description suggested ontologies are:

1. TAP[22]
2. Dublin Core[23]

For example, TAP ontology is used in SemTag project[24]. Each resource in this project is associated with concepts in this ontology.

Dublin Core is widely used for organizing library structure and other taxonomies.

An open question is what databases and frameworks to use for efficient storing, accessing and processing data in a ontology-based knowledge base.

IV. Results

At the moment along with research work, major components of the system (kernel, decision making module, classifier and parser) were developed.

Cache Proxy Server, Kernel and Decision Making Module are tested as a whole system.

Classifier and Parser are developed and tested separately.

Concerning the average size of Internet resource(40 kbytes), average speed of modern Internet connection (200 kbytes/sec) and average ping and delays of web server response, we can assume that it takes about 1-3 seconds to get access to an average Internet page.

Tests on real data showed that work of kernel and decision making module takes in average less than 0.05 seconds.

Results shown by classifier on test set of Reuters text data showed delays less than 0.1-0.2 seconds.

Summing up these results we expect that average speed of analysis will be less than 0.3 seconds, which is lower than time of access to these resource by one order of magnitude and shouldn't cause delays, observable by end users.

V. Conclusion

We propose an architecture of Internet Traffic Filtering System, based on applying a new method of multi-class multi-label classification[5] to the problem of Internet traffic filtering. These methods will allow the system to be customized according to each organization's requirements and at the same time to adapt to complex and dynamic nature of modern Internet resources. Such system can be easily deployed in commercial, public and educational organizations. Its main advantages are speed of analysis, which doesn't cause significant delays to end users, good accuracy with filtering precision about 90% and rate of false-positive errors not more than 2-3% and adaptability.

References

[1] Dan Malachowski (2005, July 11) Wasting Time at Work Costing Companies Billions [online]. Available: http://www.sfgate.com/cgi-bin/article.cgi?file=/g/a/2005/07/11/wastingtime.TMP&type=printable

[2] Karl Donert, Sara Carro Martinez (2002, December 23) End-user Requirements: Final Report Deliverable 2.1 [online]. Available: http://www.poesia-filter.org/pdf/Deliverable_2_1.pdf

[3] Harvard Internet Security Policy [online]. Available: http://www.security.harvard.edu/network_security/content/cont_traffic.html

[4] University of Western Australia External Traffic Filtering [online]. Available: http://www.ucs.uwa.edu.au/web/tech/security/external_traffic_filtering

[5] Mikhail Petrovskiy, Valentina Glazkova. Linear Methods for Reduction from Ranking to Multilabel Classification // Springer-Verlag, Lecture Notes in Artificial Intelligence, 2006, vol. 4304, pp. 1152-1156.

[6] Open-Source Filtering Software. http://www.poesia-filter.org/

[7] CyberPatrol Internet Security Software. http://www.cyberpatrol.com/

[8] SurfControl url and keyword-based Internet filtering and blocking software. http://www.surfcontrol.com/

[9] NetNanny Parental Control. http://www.netnanny.com/

[10] Google Directory. The web organized by topic into categories. http://directory.google.com/

[11] Wikipedia. Tag Metadata [online]. http://en.wikipedia.org/wiki/Tags

[12] Squid Web Cache Proxy Server. http://www.squid-cache.org/

[13] Internet Content Adaptation Protocol. RFC 3507 (2003 April) [online]. Available: http://www.ietf.org/rfc/rfc3507.txt

[14] XML-RPC Specification (1999, June 15). http://www.xmlrpc.com/spec

[15] SOAP Version 1.2 (2003, June 24). http://www.w3.org/TR/soap/

[16] Stefan Guerra, DM Filtering Components Software (2003, January 27) [online]. Available: http://www.poesia-filter.org/pdf/POESIA_IAP-2117-27572_D8.1.pdf

[17] Present and Future of Open-Source Content-based Web Filtering (2004, January 21) [online]. Available: http://www.ilc.cnr.it/poesia_prg/POESIA_FinalWorkshop_Program.htm

[18] Jigsaw – W3C's Server. http://www.w3.org/Jigsaw/Overview.html

[19] URIs, URLs, and URNs: Clarifications and Recommendations 1.0. http://www.w3.org/TR/uri-clarification

[20] Chakrabarti S., Dom B.E., Indyk P. Enhanced hypertext categorization using hyperlinks", Proceedings of the ACM International Conference on Management of Data, SIGMOD 1998, pages 307-318.

[21] FOAF Vocabulary Specification. http://xmlns.com/foaf/0.1/

[22] Semantic Annotation of the Semantic Web. http://www.deg.byu.edu/ding/research/SemanticAnnotation.html

[23] Dublin Core Metadata Initiative. http://dublincore.org/documents/abstract-model/

[24] Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien SemTag and Seeker: Bootstrapping the Semantic Web via Automated Semantic Annotation [online]. Available: http://www2003.org/cdrom/papers/refereed/p831/p831-dill.html