Denial of Service Detection and Analysis Using Idiotypic Networks Paradigm

Marek Ostaszewski, Pascal Bouvry
Faculty of Sciences, Technology
and Communication
University of Luxembourg, Luxembourg
{marek.ostaszewski, pascal.bouvry}@uni.lu

Franciszek Seredynski
(1) Institute of Computer Sciences
Polish Academy of Sciences, Poland
(2) Polish-Japanese Institute
of Information Technology, Poland
sered@ipipan.waw.pl

ABSTRACT

In this paper we present a novel intrusion detection architecture based on Idiotypic Network Theory (INIDS), that aims at dealing with large scale network attacks featuring variable properties, like Denial of Service (DoS). The proposed architecture performs dynamic and adaptive clustering of the network traffic for taking fast and effective countermeasures against such high-volume attacks. INIDS is evaluated on the MIT'99 dataset and outperforms previous approaches for DoS detection applied to this set.

Categories and Subject Descriptors: C.2.3 [Computer-Communication Networks, Network Operations]: Network Monitoring

General Terms: Security.

Keywords: Idiotypic Networks, Denial of Service.

1. INTRODUCTION

Intrusion Detection (ID) plays a vital role in the process of securing network-based computer systems by analyzing communications and reporting on malicious or abnormal activity. On one hand the process of ID has to deal with large computational cost due to the volume of data produced by high-speed networks. On the other hand ID is getting more difficult due to the constant evolution and development of intrusion methods and tools making intrusions more stealth and effective. The methods of defence against intrusions are also getting more and more sophisticated in this multi-objective "weapon race" between attackers and defenders. Nature-inspired algorithms offer robustness, speed and adaptability features that seem appealing from the point of view of Intrusion Detection Systems (IDS) [1]. Moreover, some of them are abstractions of natural defence systems, what makes them especially interesting from the point of view of computer security, especially when the nature of the attack makes classical methods inefficient, like Denial of Service (DoS) attacks.

GECCO'08 July 12–16, 2008 Atlanta, Georgia, USA.

1.1 Detection of Denial of Service attack

DoS are a special case in the process of ID, for which the goal of the attacker is to make some, or all, network services of the target unavailable. The most general classification of DoS attacks divides them into vulnerability and flooding attacks [3]. The former focuses on exploiting certain flaws of networking mechanisms to disable them with several carefully crafted packets. The latter is performed by flooding the victim with large amounts of packets impossible to process. The packets in the flood usually are not crafted - the volume of the unwanted traffic alone forces the target to deplete its resources while processing it, and makes network services unavailable to legitimate users. With the growth of the bandwidth and interconnectivity of computer networks flooding DoS attacks become one of the greatest threats in cybercrime [9] and one of the most devastating attacks possible to throw across the network.

The properties of vulnerability attacks can be analyzed, as they exploit a certain flaw in the construction of networking mechanisms, and a proper pattern can be designed to be used by signature-based IDS. It can be safely assumed, that such packets are not present in regular, legitimate communication. Their presence can be considered as a deviation from a regular model of network traffic, making them susceptible to anomaly detecting IDS. However, the traffic generated by flooding usually displays no distinctive differences from the regular one and exposes no clear patterns or strategies. For a webserver it is hard to tell the difference between two queries from two users, legitimate and malicious one. Blocking off the attacked service will give the same result - denying it to users. Additionally, data flood features vary from one attack to another between classes and even instances of an attack. DoS conducted with two different configurations (groups of bots, protocols in use) will have similar effect, yet the properties of the flood will be different. Second, looking up for high amount of packets may pose a problem to signature-based IDS, causing packet losses and high consumption of system resources. While, anomaly-based IDS generate false alarms caused by similarities between a DoS attack and a legitimate, yet abrupt and intense activity called flash events [3].

To counter a DoS the flood needs to be stopped and the attacker - identified. In both cases precise information about incoming traffic is required, what in case of more complex DoS scenarios is not a trivial problem. Taking into account that pattern-based IDS work only in the case of vulnerability attacks (scenario possible to analyze), makes anomaly-

detecting systems a more appealing solution to DoS flooding issues. In a recent survey [3] the authors present different approaches to differentiate DoS flooding from flash events. At the same time, they emphasize the need for efficient and dynamic clustering algorithms to group similar network traffic and detect DoS by providing information about cluster activity.

1.2 Artificial Immune Systems

Artificial Immune System (AIS) is one of the youngest of nature-based approaches that imitates the Human Immune System by using some abstraction of its mechanisms. The Human Immune System is capable of performing real-time operations on numerous and advanced data structures (proteins) and it can process great amount of information with high speed to keep human organism in balance and protect it against outside threats [4]. These features are appealing for solving many problems of the real world and dealing with DoS attacks can be one of them. Albeit AIS paradigms have been already applied for ID problem [8], some of them, like Negative Selection have been proven inefficient [12]. In other work [2], the author points out that Negative Selection and Danger Theory are insufficiently abstracting the mechanisms of the Human Immune System, and, as paradigms, require revision. AIS paradigms, although investigated, prove to be of little use in the field of ID, however exploiting the properties of the Human Immune System to reinforce defense mechanisms of computer networks is still an interesting perspective.

We believe that AIS properties and mechanisms can be applied to reinforce the ID process, especially when handling attacks that are difficult to define and are coupled with large amounts of data. The Human Immune System deals with high level of uncertainty and enormous amounts of molecules from surrounding environment, yet it is able to provide reliable level of security to the organism. This inspiration from nature may be an aid in DoS detection and analysis. In this paper we propose a new architecture for IDS, aiming at large scale attacks and outperforming previous results achieved in AIS-improved IDS field [11].

The remainder of this paper is organized as follows: the next Section presents investigation of DoS attacks in more detail and their analysis from the point of view of ID. Section 2 introduces the concept of Artificial Immune Systems and the paradigm of Idiotypic Networks (IN) as an aid in DoS countermeasure strategy. In Section 3 a model of IN is proposed that analyses incoming traffic and provides information needed for efficient reaction for DoS attack. Section 4 contains the results of experiments evaluating the proposed approach and the last Section presents conclusions along with plans for future experiments.

2. IDIOTYPIC NETWORK PARADIGM

The theory of idiotypic networks, or immune networks (IN), originates from the hypothesis, that antibodies, while recognizing different antigen and interacting with them, are presenting their own internal image as patterns of antigenic nature [7]. Therefore, antibodies are able to recognize not only foreign structures, but also themselves, creating a network of suppression-stimulation interactions. This statement draws a new picture of the Human Immune System, as a system maintaining dynamic equilibrium (homeostasis) while reacting to structures both friendly and malicious.

Recently, an application of IN paradigm has been proposed for ID purpose by [2] and [13]. Both papers emphasize the ability of reducing false alarms by proper analysis of input signals. Albeit results presented in both cases seem promising, the case of DoS analysis remains an open issue. Research in the domain of ubiquitous computing introduced an interesting application of IN paradigm and presented promising results in the domain of the network traffic analysis [10]. The authors proposed an approach using IN metaphors to solve the problems of context-sensitive systems, an aspect of ubiquitous computing. A model of idiotypic network which represents antibodies in the form of Artificial Recognition Ball (ARB) was used to represent the data, which are both heterogeneous and of high dimensionality. An ARB is a structure describing a region of antigen space covered by a group of similar antibodies [10]. It is constructed with a data structure as its centre, and two values defining the threshold of suppression and stimulation. First, an ARB can be stimulated by a recognition of a proper data sample. This happens when the sample falls in area of suppression (affinity exceeds suppression threshold), meaning that the ARB represents the sample in sufficient manner. Second, they can stimulate other ARBs similar to them to some degree (affinity exceeds the stimulation threshold).

Additionally, the temporal changes have been introduced to the idiotypic network model by the so called decay function [10]. All ARBs are described with an abstract parameter called a current resource level - the cumulative reward that an ARB obtains for responding to a data sample. Resources are needed for any ARB to remain in the network, as the decay function reduces the amount of resources of every ARB in the network, what imitates the behavior of antibodies in the Human Immune System. This process gradually eliminates ARBs that are insufficiently stimulated, leaving only those, that are relevant to presented data, or are similar to the responsive ones. This way an IN can adapt to incoming data and react according to the current context (situation) of ARBs.

3. A PROPOSAL OF IDS ARCHITECTURE

Typically IDS map multidimensional information about network traffic into a two state space, alert and no alert. If an alarm is raised, a security specialist will be informed about the situation that caused the alert by examining IDS logs. However, this provides little information about the overall situation that was considered malicious, making investigation difficult. The IDS aimed at attacks without any clear pattern or packet sequence should focus on gathering information rather than making decisions - in case of DoS the security analyst is the most important part of the IDS, as he has to combine many sources of information in a short amount of time to take effective countermeasures. The target of such an DoS-oriented IDS would be to look for the most repetitive, dominant traffic, and information about correlations between various traffic parameters, instead of well-defined symptoms of specific attacks. The system would follow some general rules meant as directions for gathering information about the most interesting activities on the link. Such a system would be stronger against flooding attacks (big repetitiveness, a lot of data to analyze), however at the same time it would be weaker against attacks that characterize themselves with crafted, diversified, low volume traffic.

The architecture of INIDS is presented in Fig. 1. The first two elements, the Idiotypic Network-based clustering, and the Repository of ARBs, are the parts responsible for clustering incoming network traffic using ARB-based Idiotypic Network model. The incoming traffic is processed using a repository of already created clusters represented in the form of ARBs that interact with incoming data. The clustering process consists in assigning the data sample to one ARB (called suppressive ARB) and stimulating its neighbors according to the similarity between them. There may be two possible results of presenting a data sample to the ARB network: the data sample fits one of existing clusters, or is completely new from the observed traffic. The Detection Engine decides if a change that occurred is something that requires reporting to the administrator, or the security specialist. If it is not the case, repertoire of ARBs is modified.

In both cases, the ARBs remaining in stimulation area of the considered ARB, either new, or suppressive, are stimulated. The last part of the architecture proposed in this paper is the mechanism responsible for the decay process described in the previous section. Stimulation of all ARBs remaining in the repository is gradually reduced and any ARB with insufficient stimulation is removed. However, if an ARB was active and was reacting to intrusive data samples, it may be moved to Immune Memory compartment and stored in a compressed form, to be used for the purpose of detection engine. The goal of the component is to store interesting information with a little impact on the performance of the IN itself. It provides feedback when the clustering process of presented data is finished. Then, changes in IN state can be compared to Immune Memory. Compressed ARBs can in some cases be restored, to perform their role again, if similarity between the traffic and the memorized ARB will be sufficient. Detection process reinforced this way may alert on DoS attacks at their very beginning. This part of the INIDS architecture is a subject of ongoing research and will not be evaluated in the experimental part.

The IN model proposed for IDS purposes consists of set of heterogeneous ARBs reflecting different aspects of the network traffic. Any cluster of data is described as ARB that covers some data space, and is stimulated whenever a matching sample appears. An ARB as a structure gathers a sum of its stimulations (resources), that is used to prolong its existence in the IN. We will refer to the sum of stimulation acquired by a given ARB as the lifetime of this ARB. During network traffic analysis parameters cannot be taken separately from each other, so idiotypic network should be

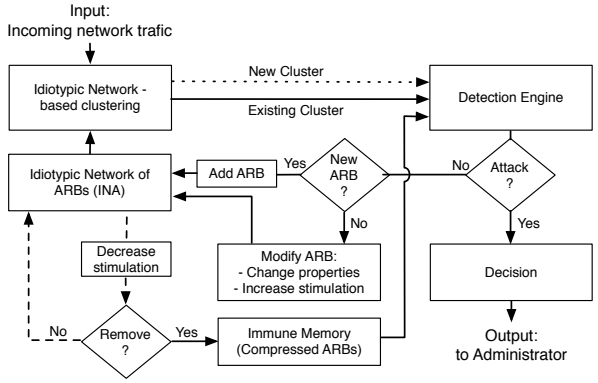


Figure 1: Architecture of IN-based IDS (INIDS)

constructed using different classes of ARBs, being sets of parameters reflecting any aspect of the traffic to be monitored. Different classes of ARBs will also define what classes of data samples can be presented to them. For that reason metrics that define the affinity of an ARB to a data sample should be specific to any pair of data sample and ARB classes.

Algorithm 1: Idiotypic Network Algorithm

```
   Input: Network traffic (Data) Idiotypic Network (IN)
1  for each sample in Data do
2      for each arb in IN do
3          arb.present(sample) //see below
4          if sample.isSuppressed() == 0 then
5              IN.addARB(new ARB(sample))
6          if arb.getDecayInterval() ≥ Δt then
7              arb.decreaseLifetime()
```

Procedure `arb.present(`*sample*`)`

```
   Input: sample: A sample of network traffic
1  aff = arb.calculateAffinity(sample)
2  if aff ≥ arb.getSuppressionThreshold() then
3      arb.increaseLifetime(aff)
4      sample.setSuppressed(1)
5  else if aff ≥ arb.getStimulationThreshold() then
6      arb.increaseLifetime(aff)
```

Network data clustering is performed by presenting incoming data sample to the repertoire of ARBs already stored in the system. The affinity measure, specific to every ARB class, is calculated and a response of IN is obtained. If the data sample falls into suppression area, defined by suppression threshold, it means that this ARB is a proper representation of the data sample, thus its lifetime is increased. If it is not the case, a new ARB is created. In both cases any ARB that is inside of stimulation threshold of suppressive, or new ARB, is also stimulated, thus its lifetime is increased. Here the stimulation of neighbors of a given ARB takes place. At the same time a decay process is applied regularly to all ARBs in the repertoire, reducing their lifetime. This way ARBs irrelevant to incoming data are removed, preserving system resources and keeping the repertoire (IN) relevant to current situation of the network traffic.The process of ARB construction and the IN growth requires no initialization phase [10] which makes this approach applicable to online monitoring. The algorithm describing the IN behavior is presented below (see Algorithm 1).

The IN on the input of the algorithm is empty, however it is assumed that a set of ARB classes is pre-defined. Every ARB has following variables with class-specific values: DecayRate, DecayInterval, SuppressionThreshold and StimulationThreshold. The algorithm proceeds as follows: whenever a data on the monitored network appears, it is presented to already existing ARBs, calculating their stimulation. If no ARB suppresses the sample, thus represents it in satisfying degree, a separate ARB is created. It is possible then, to create an ARB having its center in Stimulation Area of another ARB, but not in Suppression Area, as suppressed samples do not create ARBs. Lines 6 and 7 describe decay function - every time interval a Lifetime value of a given ARB is decreased by DecayRate by function *decreaseLifetime*. Below a procedure of presenting a sample to an ARB is described. A function of calculation of affinity is specific to the ARBs of a given class (line 1 in Procedure `arb.present()`). Therefore, to execute the Procedure `arb.present()` a class of ARB and sample must match. The same, in line 5 of the Algorithm

1 an ARB is added that has a class according to the data sample that creates it.

4. EXPERIMENTS

A series of experiments has been performed to analyze abilities of INIDS for performing its goals. The data set used in these experiments is MIT'99 [6], containing recorded network traffic from small LAN used for simulation of different intrusions. This set of data is well documented and contains variety of attacks, that can be used to expose the weaknesses and advantages of tested IDS. The scope of experiments was narrowed to two weeks of MIT'99 data. The 1st week contains regular traffic and was used to evaluate the performance of the system under regular conditions. The 4th week contains several DoS attacks thrown against three of four webservers in MIT network. Because of the scope of attack scenario, the experiments have been narrowed to four webservers (`hume`, `marx`, `pascal` and `zeno`) and the outside traffic was taken into consideration.

DoS attacks in the experimental dataset were performed using TCP and ICMP and for that reason we focused the scope of the INIDS on them. Clustering process is performed by the means of ARBs, which are stimulated by data samples falling into their suppression, or stimulation area. An ARB represents cluster of network traffic and its stimulation and lifetime describe, how often represented data appears on the network, and how it is similar to other traffic data.

ARBs of a given class are constructed on the basis of the structure of data they analyze, and for ICMP, TCP and STCP ARBs the structures are ICMP Header, TCP Header and TCP ARB respectively. ICMP ARB is constructed using Source IP, Destination IP, Type and code fields of the header. TCP ARB is constructed using Source IP, Destination IP, Source Port, Destination Port of the header and a value of TCP Stack Code (see below). STCP ARB is constructed using Source IP, Destination IP, Source Port, Destination Port of the TCP ARB and a value of Termination Code (see below). The affinity of a given ARB to a respective data sample is calculated on the basis of the distances between mentioned parameters. Below we discuss respective parameters and their influence on affinity.

IP address - the distance between two IP addresses is calculated using discrete metric - if they are equal, then the distance between them is 0, else it is 1. Affinity value for match of IP addresses in local network is equal to 0.3, and in outside network is 0.7.

Port value - ports are used by transport protocols (TCP, UDP) and distance between two port values is an absolute value of their difference. The distances for different groups of ports are defined on the basis of IANA port assignments [5] and are as follows: the well known port distance is 10, the registered port distance is 3 and the dynamic port distance is 1. Affinity value for a match of ports in local network is equal to 0.3, and in outside network is 0.7. **ICMP Type and Code** - the distance for ICMP types and codes is an absolute value of their difference. The distance value for ICMP Type and Code was set to 0 (match required) and affinity values for matching these parameters are 0.5 and 0.3 respectively.

TCP Stack Code - TCP packets are arranged in streams, and a pair of IP addresses and ports (source and destination) called socket is an unique identifier of the stream. Because of that a match on all socket fields is required for a packet to be assigned to a proper ARB. Because TCP protocol requires stream reassembly to assess correctness of every packet in its stream, an additional feedback from the TCP stack has been introduced - every TCP packet has so called TCP Stack Code assigned, informing about the role of the packet in the stream. These codes are derived from TCP protocol definition in RFC 793. The transition from one state to another (i.e. from SYN to SYN/ACK) introduced by stack code of packet belonging to a stream defines the distance. Affinity values assigned for a given transition can be assigned from the interval from 0.3 to 1.3, depending on the impact of the TCP Stack Code on the current state of the stream (small stimulation in case of regular behavior).

Stream Termination Code - Stream TCP ARB was constructed to gather the information about terminated TCP streams. Whenever a packet comes, that ends a given TCP stream, the ARB mapping that stream is presented to the Idiotypic Network as a data sample. The affinity is calculated on the basis of the TCP socket fields and an absolute value of difference between last TCP Stack Code of the TCP stream (called Stream Termination Code) ARB and data sample. The distance value for Termination Code was set to 0 (match required) and affinity value for matching these parameters is 0.4.

Distances between respective parameters are combined by summarizing affinity measures specific to the given parameters. Because of the differences in construction, a suppression and stimulation thresholds are specific to every class of ARB, respectively presented as follows:1.8 and 1.08 for ICMP, for TCP suppression a match of IP port an pairs is required and for stimulation threshold is 1.6, finally for STCP the values are 2.4 and 2.04. All ARB classes has the same value of DecayRate, equal to 0.1 and DecayInterval set to 1 sec. Both values of affinity and thresholds were chosen after observation of IN behavior, however we plan to further investigate this area.

4.1 Experiment #1: Observations of the regular traffic

Regular traffic of MIT'99 data (week 1) has been presented to the system to verify its performance. Four webservers of the MIT network were under observation, and Fig. 2 presents the activity for STCP ARBs of the webservers. One can notice five groups of lines, depicting the activity of the webservers during five days of the week 1. It may be difficult, to decipher precisely the activities of respective hosts, however one clearly see that the value of average lifetime is at most around 50, except two clear peaks caused by host `hume`. Analysis of the peaks has shown, that they are represented by a single ARB with high lifetime, describing an apparent network malfunction: numerous RST/ACK TCP packets sent to outside hosts (209.3.209.166 and 206.132.135.201). This actually shows the sensitiveness of the system for unusual events. The rest of results of observations during the week were as follows: the STCP ARB number was less than 50, the average lifetime of ICMP ARBs was less than 12, and the values of ICMP ARB number was less than 5.

4.2 Experiment #2: IN-based clustering in DoS detection

The 4th week of MIT'99 data contains several DoS attacks, as listed in Table 1. Every day of traffic was presented packet by packet to INIDS, and the activity of ARBs was

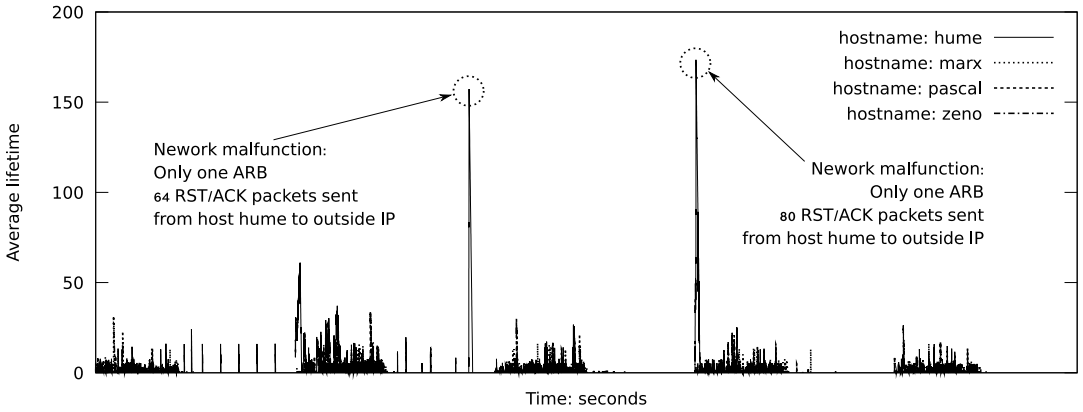


Figure 2: Behavior of STCP ARBs of `hume`, `marx`, `pascal` and `zeno` hosts during week 1

Table 1: DoS attacks in 4th week of MIT'99 data

(Day) Time	Time (sec)	Victim	Attack name
(1) 21:34:16	48856	`pascal`	`smurf`
(2) 15:51:16	28276	`marx`	`mailbomb`
(2) 17:49:15	35355	`marx`	`process table`
(3) 16:54:17	32057	`pascal`	`mailbomb`
(4) 18:32:17	37937	`marx`	`mailbomb`
(5) 12:32:17	16337	`zeno`	`mailbomb`

monitored, both taking into account the average lifetime of ARB of a given class, and the numbers of ARBs of a given class. Every case of DoS attack was separately investigated, and the study of the three separate attack classes (`smurf`, `mailbomb` and `process table`) is presented below. Information about the average lifetime and number of ARBs affected by the attack has been depicted in following figures.

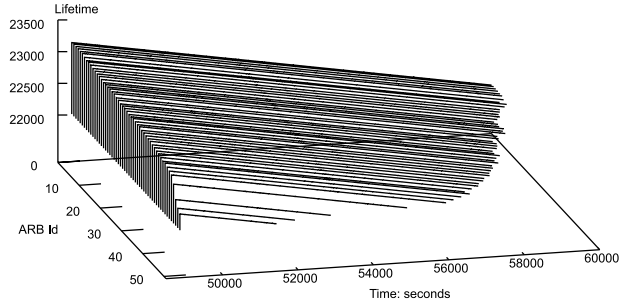


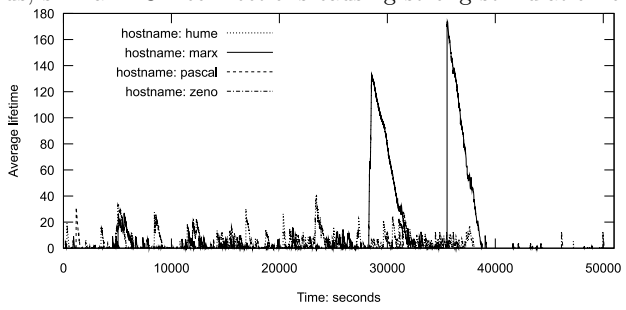
Figure 3: Behavior of 50 strongest ICMP ARBs of host `pascal` during Day 1

Smurf attack is performed using ICMP protocol, when the attacker sends a number of ICMP Echo Request (ping) packets to various hosts in the network, spoofing his source address to point at the victim. Hosts receiving ping packet, reply to the victim, flooding it with unwanted ping replies. The characteristics of the attack suggests, that activity of ICMP ARBs should be prevailing, indicating sources of the attack.

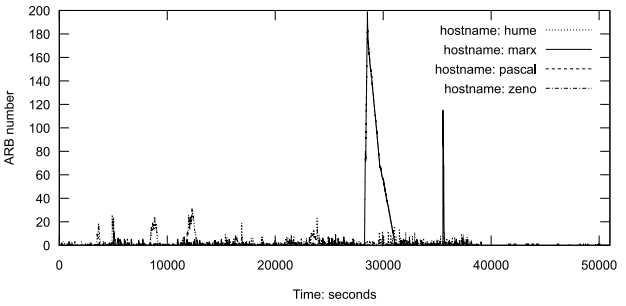
Such attack was placed during the Day 1 of the 4th week. On the time of attack (48856 sec) there was an immediate growth of ARB number (to almost 200) and their average lifetime exceeded 20000, while the value of the average lifetime during regular activities never reached value of 1. Fig. 3 presents the behavior of the Idiotypic Network and in particular the behavior of 50 strongest ICMP ARBs. One can notice the moment of attack by a sudden increase of the lifetime. Other ARBs were behaving in the same way, but for the sake of clarity of the Figure they have not been depicted. The reason why Fig. 3 depicts only last hour of the Day 1 is because the regular activity is invisible in this scale. The attack lasted for 11 seconds, however the stimulated lifetime of ARBs was keeping them in the IN until the end of measurement.

Mailbomb attack is performed by sending multiple email messages to the server, causing an overflow of the mail queue on that server, what may cause a system failure. Because mail protocol is used, TCP activity should be more intense during such an attack.

Fig. 4 and Fig. 5 present the activity of ARBs for webservers of MIT'99 dataset, namely `hume`, `marx`, `pascal` and `zeno`. In particular the activity of SFTP class of ARBs in the Day 2 and Day 3 has been depicted. Figures illustrate the average lifetime (a) and the number (b) of the ARBs. In Fig. 4(a) for host `marx` one can see two clear peaks, one starting as `mailbomb` takes place (28276 sec) due to numerous, similar TCP connections causing strong stimulation of



(a)



(b)

Figure 4: Behavior of STCP ARBs of `hume`, `marx`, `pascal` and `zeno` hosts during Day 2

neighboring ARBs. The average lifetime (around 190) is much lower than the response of ICMP ARBs in previous case. This can be explained by the nature of the attacks - in case of `smurf` one ICMP packet similar to others was enough to stimulate the net, in case of STCP ARBs it has to be terminated TCP connection. Two clear peaks can be denoted as well in Fig. 4(b), the first one starts at the same time as peak of Fig. 4(a). It is caused by `mailbomb` attack and can be explained with the diversity of TCP connections used for launching the attack. The presence of the second peaks for the Figures 4 (a) and (b), also appearing at the same time, will be discussed below.

In Fig. 5(a) for host `pascal` one can see a clear peak, starting as `mailbomb` takes place (32057 sec). The peak for host `zeno` is also very clear, and it has to be mentioned, that it is caused by `guessftp` attack, however its analysis is out of scope of this paper. In Fig. 5(b) three peaks, less clear, can be observed. Two latter reflect mentioned attacks, however the peak on the activity of host `hume` is the effect of a regular, but high activity. The results obtained for `mailbomb` during Days 4 and 5 were similar to the presented ones, and for the sake of brevity, have not been included.

Subfigures of Fig. 6 present the lifetime of all ARBs of host `pascal` (a), average lifetime of ARBs of hosts `hume`, `marx` and `zeno` (b), and the number of ARBs for all four mentioned hosts (c). Although host `pascal` was not a victim of any outside attack during the Day 4, extremely high lifetime of two ARBs can be noticed. They have been created due to apparent network malfunction, as ARB with the Id 129 describes 2031 trials of TCP connection (repetitive SYN packets) from host `pascal` to port 25 of IP 195.73.151.50, and ARB with Id 151 describes 2015 of connections to the same port and the same IP. This behavior is not reported in MIT documentation, nevertheless it can serve as an example of a compromised host taking part in Distributed DoS as one of the bots. Another reason for presentation of behavior of ARBs for host `pascal` separately from Fig. 6(b) was the sake of scale - the lifetime of the two mentioned ARBs was an order of magnitude greater than the ARBs of the other hosts and it would obscure several interesting details.

In Fig. 6(b) and Fig. 6(c) one can notice a clear peak formed by the activity of ARBs of host `marx`, which is formed by malicious DoS traffic (37937 sec). Average lifetime of ARBs for host `zeno` can be seen in Fig. 7(c). Number of ARBs during the attack was greater than for any other host and was more than 200 for the moment of the attack. Respective figure is omitted for the sake of brevity.

Process table attack exploits network services of UNIX systems by connecting to them and initializing many processes to fill up the table of processes of the machine, thus denying other users to use the service, as no more processes for the service can be created [6]. This attack may be considered as a combination of vulnerability and flooding DoS, because it uses a vulnerability of unlimited creation of processes by network services, and a flood of requests for such services. Flooding part makes it possible to discover due to repetitiveness of produced traffic.

Such an attack was placed during the Day 2 of the analyzed data. As it can be noticed, it is clearly visible in Fig. 4 as second peaks on both subfigures. It is worth noticing, that the average lifetime, and therefore the stimulation, is greater and at the same time the number of ARBs was

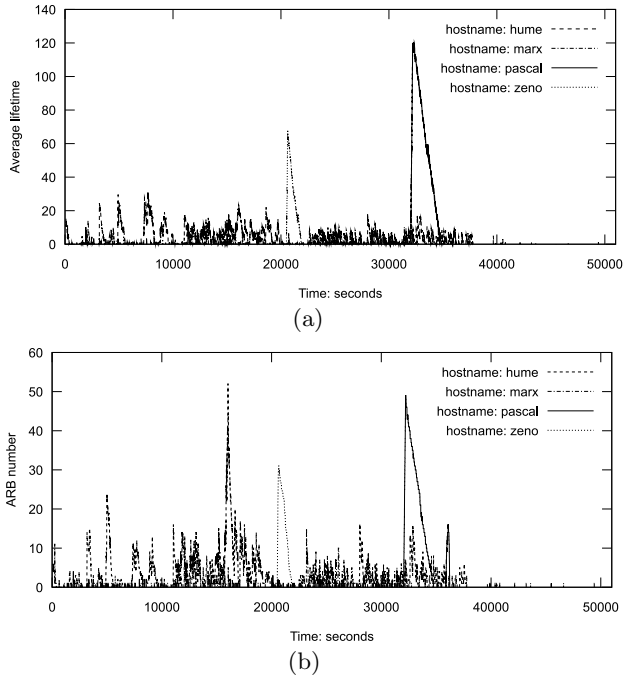


Figure 5: Behavior of STCP ARBs of `hume`, `marx`, `pascal` and `zeno` hosts during Day 3

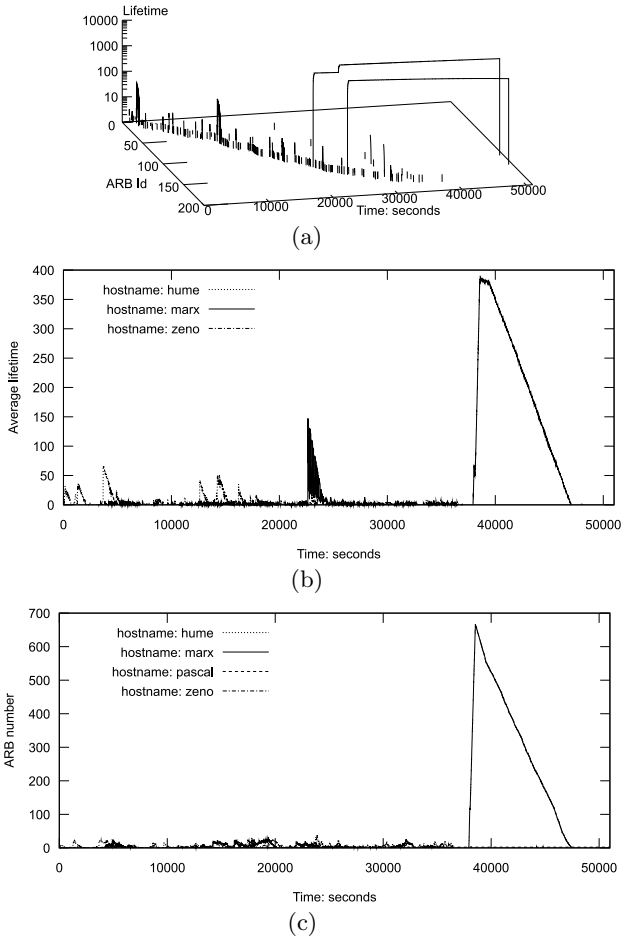


Figure 6: Behavior of STCP ARBs of `pascal`, `hume`, `marx` and `zeno` hosts during Day 4

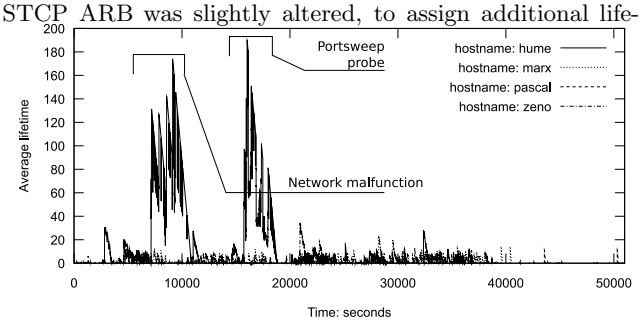
Table 2: Portsweep attacks (TCP based) in 4th week of MIT'99 data

(Day) Time	Time (sec)	Victim
(1) 12:22:15	15736	`hume`
(3) 16:43:15	31395	`hume`
(5) 18:10:15	36615	`zeno`

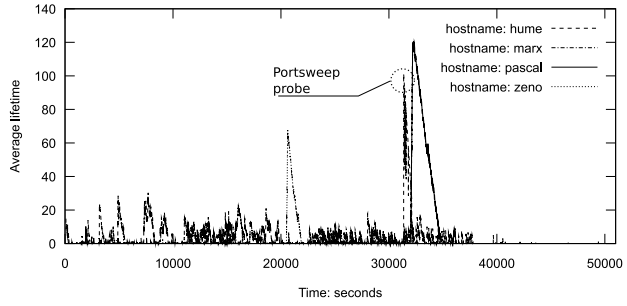
lower. It can be explained by the construction of the `process table` attack - it produces more repetitive traffic, than `mailbomb` attack.

4.3 Experiment #3: Improvement of INIDS accuracy of detection

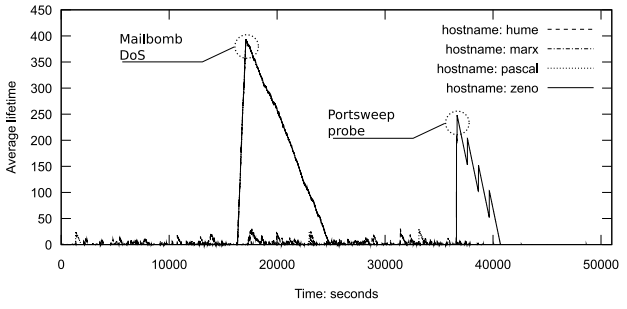
ARBs are keeping more detailed information than only their lifetime. Basic statistics (average lifetime and number) were used in previous experiments. However, more sophisticated process of detection can be performed if the information within ARBs are used. A detection of probe attacks was considered in this experiment. Table 2 presents TCP based instances of `portsweep` attack aimed at mentioned webservers. Previously, these attacks were unnoticed, because they require relatively low number of packets to be executed. In case of Experiment #3 a construction of STCP ARB was slightly altered, to assign additional lifetime of value 100 to any ARB holding a specific combination of flags gathered during the TCP stream activity. Usually `portsweep` characterizes itself with connection initialization (SYN and SYN/ACK packets), and immediately after the connection is ended with RST flag coming from client. Such short and anomalous connection allows for checking, if targeted port is used by any network service.



(a)



(b)



(c)

Figure 7: Behavior of STCP ARBs of hosts `hume` Day 1(a), `hume` Day 3 (b) and `zeno` Day 5(c)

Fig. 7 presents Average lifetime of STCP ARBs during three days, when TCP-based `portsweep` version was used. The peaks formed by the IN response to the attack are indicated. There are two clear peaks in Fig.7(a), and the first one is caused by network malfunction. Similarly to situation in Fig. 2, the first peak is formed in main part by two ARBs created by sequences of 74 and 27 RST flags sent by the host. Fig.7(b) and 7(c) depict a probe attacks that although performed with several packets, are visible on the level of other DoS attacks. In Fig. 7(c) one can see that the `portsweep` peek consists of five small peaks, created as attack sequence was repeated five times, for ports 17, 19, 23, 37 and 514 [6]. The experiment proves, that it is possible to optimize the process of traffic analysis, making scope of INIDS more focused or general, by applying customized rules of ARB stimulation.

4.4 Evaluation of the approach

The performance of presented approach was evaluated concerning accuracy and efficiency. As it was mentioned in Section 3, the goal of the proposed system is to provide comprehensive information about the network behavior. INIDS is a system oriented on gathering information about current network traffic, emphasizing repetitiveness and unusual behavior, however not focused on any particular attack. Because of that, while comparing accuracy of INIDS with other IDS we take into consideration amount of information provided by the system during unusual event.

Accuracy: In a regular IDS the main feedback after raising an alert is to indicate which signature (in case of pattern detection) or parameters (in case of anomaly detection) caused an alarm, however this usually provides little information about a situation on the network. Such a feedback leaves security analyst with a little clue about countermeasures to take. For instance, approach of [11] involved monitoring three parameters and alert was raised whenever a combination of them was falling outside of the model of regular behavior. Although this approach was able to indicate properly all attacks in 2nd week of MIT'99 data for host `marx`, no information beside an alert itself was provided. In case of performed experiments it is possible to detect all presented DoS attacks by applying a following rule to IN parameters:

if Average Lifetime of ARB class > 120
and Number of ARB in class > 40 **then** Alert.

However, ARBs store more parameters than their lifetime, and if this information is gathered, a more comprehensive picture of the indicated anomaly is visible. In case of `smurf` attack all ICMP ARBs had extraordinary high stimulation, indicating their relevance to the attack. Analyzing the group of ARBs, it has been noticed that all 199 of them come from outside, and have ICMP type and code equal to 0, what describes Echo Replay message, while no Echo Request message have been sent (what is normally the case and would be represented by an ARB). Moreover, there were strong correlations between the source addresses as all are originating

from only four class C networks, 40 IPs from 6.238.105.1 (the network of actual attacker with IP 6.238.105.108), 48 IPs from 116.204.35.1, 48 IPs from 21.15.139.1 and 57 IPs from 242.99.186.1 network.

In case of `mailbomb` attacks, taking example of host `marx` during Day2, when ARBs were gathered, an interesting case could be noticed. First part of the ARBs with strongest stimulations were ARBs describing from outside, carrying the IP of the attacker (194.27.251.21 [6]), and 39 source ports used for attack were from 3057 to 18259 interval, the target port was mail protocol port 25. At the same time second dominant part of ARBs was traffic from the victim machine to the attacker IP to port 113, which is used for determining the remote user of a given client network connection. This information was obtained by examining the ARBs with strongest stimulation, allowing reconstruction of the attack scenario. Similarly, examining of ARBs that cause the IN to exceed previously defined thresholds exposes that the ones with the strongest stimulation are reflecting scenarios for presented DoS attacks.

Efficiency: Time consumption during experiments was measured to analyze usefulness of the approach for processing large amounts of data, especially during DoS attacks. The total amount of data in the traffic of the whole 4th week was 11439 Mbits in 6585499 packets, the IN model was implemented in Java 1.5.0, and the testing machine was Intel Core 2 Duo processor 2,16 GHz with 3GB of RAM memory, however the model was verified using only one thread, thus only one core. The analysis of the 4th week traffic was 108 seconds long, and average processing speed was 105,7 Mbit and around 60000 packets per second.

5. CONCLUSIONS

This paper introduces a new approach based on immune networks to analyze the network traffic, which focuses on the intrusion detection process for DoS flooding attacks. The idea behind the proposed approach is to dynamically cluster the network traffic and monitor activity of the clusters to look for dominating features of the traffic [3]. Such approach allows in the first place to gather information about incoming, or proceeding attack, to take the most efficient countermeasures against the threat.

Experiments have shown, that DoS attacks of test data are clearly visible when the activity of IN is monitored. If the traffic presents many similarities, it is clustered by a low number of ARBs, which are then strongly stimulated and kept for longer in the system. Different kinds of attacks have their own particular impacts on the system, however, in our experiments, it has always been possible to discover the attacks by examining statistics of IN activity. Moreover, helpful information can be extracted from ARBs raising the alarm, giving a broader view on an anomalous situation. It should be emphasized, that the system was additionally able to indicate situations that were generally anomalous. In cases of week 1 (Fig. 2) and Day 4 of week 4 (Fig.6) clear peaks are present, however no attack was simulated that time, according to MIT documentation. Indication of anomalies of such a character is desirable and cannot be considered false positives of the system. Especially in second case, which can be percieved as participation in DoS attack by sending huge amounts of abnormal traffic to outside IP address.

The results of the ongoing research presented in this paper show different interesting properties of the proposed approach, like recognition of repetitiveness of the traffic, temporary memorization of passing events and response to the current network context. These features combined with a low resource consumption during experiments are encouraging. Future research concerns improvement of efficiency of the system through distributed computations on ARB sets and further improvement of the model of Immune Network.

6. REFERENCES

[1] J. P. Anderson. Computer security threat monitoring and surveillance. Technical report, James P. Anderson Co., April 1980.

[2] H. Bersini. Why the first glass of wine is better than the seventh. In *16th International Workshop on Database and Expert Systems Applications DEXA'05*, pages 100–111. Springer, 2005.

[3] G. Carl, G. Kesidis, R. R. Brooks, and S. Rai. Denial-of-service attack-detection techniques. *IEEE Internet Computing*, 10(1):82–89, 2006.

[4] L. N. de Castro and J. Timmis. *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer-Verlag, London, UK, 2002.

[5] http://www.iana.org/assignments/port–numbers. Internet Assigned Numbers Authority (IANA).

[6] http://www.ll.mit.edu/IST/ideval/index.html. MIT Lincoln Laboratories data set, 1999.

[7] N. K. Jerne. Towards a network theory of the immune system. *Ann. Immunol. (Inst. Pasteur, Paris)*, 125C(1-2):373–389, 1974.

[8] J. Kim, P. J. Bentley, U. Aickelin, J. Greensmith, G. Tedesco, and J. Twycross. Immune system approaches to intrusion detection — a review. *Natural Computing: an international journal*, 6(4):413–466, 2007.

[9] J. Mirkovic, S. Dietrich, D. Dittrich, and P. Reiher. *Internet Denial of Service: Attack and Defense Mechanisms*. Prentice Hall PTR, 2004.

[10] P. H. Mohr, N. Ryan, and J. Timmis. Exploiting immunological properties for ubiqitous computing systems. In *Proceedings of 3rd International Conference on Artificial Immune Systems, ICARIS'04*, pages 277–289. Springer, 2004.

[11] M. Ostaszewski, F. Seredynski, and P. Bouvry. Coevolutionary-based mechanisms for network anomaly detection. *Journal of Mathematical Modelling and Algorithms*, 6(3):411–431, 2007.

[12] T. Stibor. *On the Appropriateness of Negative Selection for Anomaly Detection and Network Intrusion Detection*. PhD thesis, Darmstadt University of Technology, 2006.

[13] A. O. Tarakanov, L. B. Goncharova, and O. A. Tarakanov. A cytokine formal immune network. In *Advances in Artificial Life, 8th European Conference, ECAL'05*, pages 510–519. Springer, 2005.