N3: A Geometrical Approach for Network Intrusion Detection at the Application Layer

Juan M. Estévez-Tapiador, Pedro García-Teodoro, and Jesús E. Díaz-Verdejo

Research Group on Signals, Telematics and Communications
Department of Electronics and Computer Technology
University of Granada
{tapiador, pgteodor, jedv}@ugr.es

Abstract. In this work, a novel approach for the purpose of anomaly-based network intrusion detection at the application layer is presented. The problem of identifying anomalous payloads is addressed by using a technique based on the modelling of short sequences of adjoining bytes in the requests destined to a given service. Upon this theoretical framework, we propose an algorithm that assigns an anomaly score to each service request on the basis of its similarity with a previously established model of normality. The introduced approach has been evaluated by considering datasets composed of HTTP and DNS traffic. Thus, a large amount of attacks related with such services has been gathered, and detailed experimental results concerning the detection capability of the proposed system are shown. The experiments demonstrate that our approach yields a very high detection rate with a low level of false alarms.

1 Introduction

As Carl Landwehr brilliantly indicates in his introductory article to the field of computer security [1], several paradigms have configured the research scene in this area from the early days of modern computing to date. What can be designated as the first generation of security technologies defines a broad spectrum of defense techniques oriented to *prevent* the occurrence of successful intrusions or violations of the security policy. Due to various reasons that fall out of the scope of this discussion, the security research community started to develop procedures and mechanisms intended to *detect* and *limit* attacks that are very difficult to prevent because of the nature of our current technologies. Firewalls and Intrusion Detection Systems (henceforth referred to as IDS) are surely the maximum exponent of this paradigm. Nowadays, and although many relevant problems still remain unsolved within the two previous paradigms, *intrusion tolerant* schemes constitute a new and challenging research topic. Projects like OASIS (Organically Assured and Survivable Information System) [2] and MAFTIA (Malicious-and Accidental-Fault Tolerance for Internet Applications) [3], among others, have developed concepts, architectures, and prototypes within the intrusion tolerance approach.

A. Laganà et al. (Eds.): ICCSA 2004, LNCS 3043, pp. 841–850, 2004.

Even though a number of open problems still remain unsolved, research in IDS constitutes a relatively mature field. Interested readers can find good introductions to IDS in [4], [5], [6], and [7], among others. The two key methodologies that have traditionally been used to detect intrusions in network systems are related to how the collected data from the sensors are analyzed. In the so-called *misuse-based* detection, each known attack is described through an specific pattern, commonly referred to as its signature, which identifies it without ambiguity. The core of the detection engine is basically a pattern-matching algorithm, in such a way that incoming activities that match a pattern in the library of signatures cause an alarm raise. On the other hand, the basic principle supporting *anomaly-based* detection systems is the hypothesis that every anomalous event is suspicious from a security point of view. An event can be catalogued as anomalous because its degree of deviation with respect to the profile of characteristic system behavior. Although current intrusion detection technology mainly relies on misuse detection mechanisms, anomaly detection has been typically conceived as a more powerful mechanism due to its theoretical potential for addressing novel or unforeseen attacks.

In this work, we deal with the problem of anomaly detection at the application layer for network-based IDS. A theoretical framework is introduced and applied to HTTP and DNS protocols. Section 2 serves to the purpose of presenting the core of the approach proposed in this work, which relies on the modelling of short sequences of adjoining bytes in specific service requests. A detection algorithm based on this principle of operation is then introduced. The experimental results presented in Section 3 confirm that short sequences are excellent features for distinguishing between normal requests and those containing several forms of attack. Finally, a performance analysis of the developed system is introduced in Section 4, and Section 5 concludes the paper summarizing the benefits of the introduced work as well as future research objectives.

2 N3: A Geometrical Method for Intrusion Detection at the Application Layer

As the vast majority of the proposals developed in the field of anomaly-based network IDS, the technique introduced in this article tries to model the network traffic with the aim of obtaining a representation of normal behavior ([8], [9], [10], [11], [12]). In this case, the objects we intend to model within the proposed approach are application-level payloads from a number of traffic sources (clients), specifically those containing requests destined to a given service. In the case of HTTP, this application-layer protocol is defined by RFC 2068, albeit certain elements like URIs (Uniform Resource Identifiers) are defined in other standard documents (RFC 2396 for the mentioned identifiers). In its basic form, HTTP payloads are human-readable strings enclosing elements like the version of the protocol, the identifier of the requested object, several parameters related to the request, etc. As just exposed, some works in this field have established that, due precisely to the nature of the contents transported by the protocol, HTTP requests destined to a server share a common structure, in such a way that it is possible to measure certain degree of similarity among the normal payloads received by a given server.

The analysis for the DNS system is similar. DNS queries and responses are carried in a standard message format (see Internet Standard 13 or, equivalently, RFC 1034). Each message is composed of a header with a number of fixed fields and four sections (*Question*, *Answer*, *Authority*, and *Additional*) containing query parameters and resource registers. Although the contents carried over this fixed format can vary according to the header options, they always enclose a number of readable strings with values for the different fields. Interested readers can find in the above mentioned standard various examples that illustrate this fact in detail.

In what follows, we introduce a formal background intended to manipulate and model the type of short sequences previously referred.

2.1 Sequence Analysis

Let $\Sigma = \{s_1, s_2, \ldots, s_n\}$ be a finite set of size n, namely *alphabet* and composed by elements which we refer to as *symbols.* In this discussion, we shall assume that Σ is the ASCII code, in such a way that $|\Sigma| = 256$. Therefore, each complete payload (i.e., a transaction destined to a server) can be represented as an element $p = s_{i_1} s_{i_2} s_{i_3} \cdots s_{i_{|p|}} \in \Sigma^*$, where Σ^* is the set of all possible sequences generated by concatenating a finite number of symbols inΣ.

Since Σ is a finite set, it is possible to compute and enumerate all the different sequences of a fixed length, k, that can be generated from it. To be precise, given that $|\Sigma| = n$, there are n^k different sequences of length k, which can be ordered from 1 to n^k. Once fixed a given order, we shall denote by σ_i the i-th sequence.

As stated, a payload p is easily conceived as a sequence of symbols. Let us consider the following transformation λ_k, according to which each complete payload p is mapped to a n^k-dimensional vector space, Θ_k, as follows:

$$\begin{aligned} \lambda_k \ &: \ \Sigma^* \to \Theta_k \\ \lambda_k(p) &= (t_1, t_2, \ldots, t_{n^k}) \end{aligned} \tag{1}$$

where each component of the image vector, t_i, is the number of times that the sequence σ_i, of length k, appears in the complete sequence p.

Our objective is to measure similarities between payloads once transformed into the new space of representation. Since Θ_k is a Hilbert space, for every pair of points $x_a = (t_1^a, \ldots, t_{n^k}^a), x_b = (t_1^b, \ldots, t_{n^k}^b) \in \Theta_k$, their dot product, denoted $\langle x_a, x_b \rangle$, is defined. Given a dot product, we can also define the distance between two points $x_1, x_2 \in \Theta_k$ as:

$$d(x_1, x_2) = \|x_1 - x_2\| = \Big(\langle x_1, x_1 \rangle - 2\langle x_1, x_2 \rangle + \langle x_2, x_2 \rangle \Big)^{\frac{1}{2}} \tag{2}$$

Likewise, given two payloads $p_1, p_2 \in \Sigma^*$, and a fixed k, we can define the distance between them as $d(\lambda_k(p_1), \lambda_k(p_2))$. In many cases, it is computationally very expensive to explicitly obtain the representation exposed above for each payload. For instance, the ASCII code contains 256 different symbols, and assuming that we take short sequences of length $k = 5$, each payload is mapped to a vector of $256^5 = 1099511627776$ components. We are interested, however, in the distances between payloads and, therefore, in establishing a procedure for computing them without an explicit mapping of the payloads to their images.

Fortunately, the field of algorithms on sequences is a relatively well-known and studied discipline. There exists a dynamic programming-based algorithm that computes in $O(n^2)$ operations the number of subsequences of length k shared by two input sequences (where n is the length of the input sequences). The precise description of that algorithm is out of the scope of this paper, although interested readers can found a deep description of it in [13]. In any case, note that, by the very definition of $\lambda_k(p)$, it is easy to compute the dot products involved in the distance calculation given by the expression (2) by using this algorithm.

2.2 The N3 Anomaly Detector

The analysis of sequences presented allows us to develop a novel anomaly-based intrusion detection approach. For that, let us make the following initial definitions.

Definition 1. *A mathematical* ***model*** *of an application-layer protocol L, denoted $\mathcal{M}_L$, is defined as a representative set of the normal payloads of such a protocol; that is, $\mathcal{M}_L = \{p_1, p_2, \ldots, p_N\}$*

Definition 2. *The* ***distance*** *from a payload p to a model of the corresponding service, denoted $D(p, \mathcal{M}_L)$, is defined as the distance from p to its nearest neighbor element in the model, using for this purpose the similarity function d defined in expression (2). This distance will be termed the* ***anomaly score*** *of the payload p, denoted $A_s(p)$.*

Considering these definitions, it is possible to construct an anomaly detector based on the distance within the context of the introduced framework. Thus, assuming that the model of normal behavior is given by the set $\mathcal{M}_L$ of normal payloads observed for the selected service, deciding whether a captured payload, p, is labelled as anomalous or not is performed by calculating its anomaly score:

$$A_s(p) = D(p, \mathcal{M}_L) = \min_{\forall m \in \mathcal{M}_L} d(p, m) \tag{3}$$

Once computed the previous score, the detection rule is straightforward:

> **Detection Rule:** A payload p is designated as anomalous if $A_s(p) \geq \theta$, where θ is a threshold which acts as a tuning parameter.

Since the model of application-layer traffic is exclusively composed of normal payloads, we will refer to this detection algorithm as *Nearest Normal Neighbor* (N3 for shorthand). The essence of the detection procedure allows us to designate this technique as a geometrical method.

Table 1. Test-bed of normal and attack traffic used for the evaluation.

Dataset Information		**Size (No. of service requests)**			
Protocol	Class	Total	Distinct	Training	Test
HTTP	Normal, host `hume`	12154	2271	1590	681
HTTP	Normal, host `marx`	16539	2388	1672	716
HTTP	Attack	1500	119	–	119
DNS	Normal	193083	66783	46849	19934
DNS	Attack	6	6	–	6

3 Experimental Results

In order to evaluate the detection capabilities of the proposed method N3, several experiments have been carried out. The evaluation framework considered and the results obtained are discussed in this section.

3.1 Test-Bed of HTTP and DNS Traffic

An important aspect of any evaluation process is the dataset to use. The DARPA 1999 IDS Evaluation Program [14] has been considered in this work for this purpose. Although it is not free of drawbacks (see [15] for an excellent critique), it is undeniable that this has been the only remarkable effort to provide a public and common facility for the evaluation of IDSs. The framework is basically composed of several off-line test sets, each one consisting of traffic captured during 5 weeks on a network with hundreds of hosts and a connection to Internet. The training data consists of the first 3 weeks, while the remaining 2 weeks constitute test traffic.

In our approach of tackling the problem of anomaly detection at the application layer, complete data sets of both normal traffic and anomalous connections are required. We have collected normal traffic from the DARPA'99 IDS Evaluation data sets, specifically from weeks 1 and 3, which are attack-free. Since our purpose is studying HTTP and DNS traffic, we have extracted packets destined to two different servers: `hume` (NT Server with IP address `172.16.112.100`) and `marx` (Linux Server with IP address `172.16.114.50`). The total amount of requests extracted and reassembled, if needed, has been 12154 for `hume` and 16539 for `marx`. Please note that there is a large amount of redundancy within the data, i.e. the same request originated from distinct clients. Table 1 summarizes the most important information concerning these datasets. In the case of DNS traffic, the gathering task has been similar to that just described for HTTP. A total amount of 193083 requests have been extracted from the traffic files. After processing them with the aim of removing duplicate elements, the useful dataset is composed of 66783 different service payloads.

Additionally, we have collected several well-known vulnerabilities in the HTTP and DNS services. The attack datasets used include several variants of 86 HTTP exploits based on vulnerabilities listed in arachNIDS database [16].

Table 2. Attacks against the DNS service used in the experimentation.

ID	**Attack name**	**Category**	**ArachNIDS Ref.**
$A1_{DNS}$	PROBE-IQUERY	Information Gathering	Ref. IDS277
$A2_{DNS}$	EXPLOIT-TSIG-LSD	System Integrity	Ref. IDS489
$A3_{DNS}$	EXPLOIT-TSIG-LUCYSOFT	System Integrity	Ref. IDS490
$A4_{DNS}$	EXPLOIT-TSIG-LUCYSOFT2	System Integrity	Ref. IDS490
$A5_{DNS}$	EXPOIT-TSIG-TSIG0WN	System Integrity	Ref. IDS491
$A6_{DNS}$	EXPLOIT-INFOLEAK-LSD	System Integrity	Ref. IDS482

Attack payloads are generated by means of programs that implement the corresponding exploit for each attack. For evaluation purposes, a total amount of 1500 malicious payloads were generated, captured and recorded in the same way that was done for normal traffic. The number of known attacks against the DNS system is more reduced. Table 2 lists the 6 attacks against a name server that have been used in this work.

3.2 Evaluation and Detection Results

With the aim of evaluating the introduced N3 approach, we have performed the following experiments. Given a specific protocol L (HTTP or DNS), the total amount of normal traffic available is divided into two subsets. The first of them, denoted $\mathcal{M}_L$ as stated in Section 2.1, is composed of 70% of randomly chosen payloads and constitutes the model of normality. The remaining 30% is devoted to a different subset, namely $\mathcal{N}_L$, for evaluation purposes. Our experimental scenario is thus composed of three sets of payloads for each protocol:

- The models, $\mathcal{M}_{HTTP}$ and $\mathcal{M}_{DNS}$, of normal traffic for each protocol.
- $\mathcal{N}_{HTTP}$ and $\mathcal{N}_{DNS}$, containing payloads of normal traffic for each protocol. These will be used for the evaluation of the detection performance.
- $\mathcal{A}_{HTTP}$ and $\mathcal{A}_{DNS}$, containing the datasets of attack traffic described in Section 3.1. These will be used for the evaluation, together with the datasets of normal traffic.

For each payload p in the datasets used for the evaluation, the anomaly score $A_s(p)$ is computed by using the N3 algorithm. The key results of this experiment for the HTTP protocol are graphically shown in Fig. 1 for different values of the parameter k involved in the distance computation (short-sequence length). In order to distinguish between normal and anomalous traffic, some parameters related to the frontiers of each region in the decision surface are of the utmost importance. With the aim of illustrating this fact, the mentioned figure shows the range (minimum and maximum) and the average distance among all the evaluated payloads. For instance, in the case of HTTP payloads and using a length $k = 8$, normal payloads present an anomaly score within the interval $[3.464, 11.533]$, with an average value of 6.496. On the other hand, attack

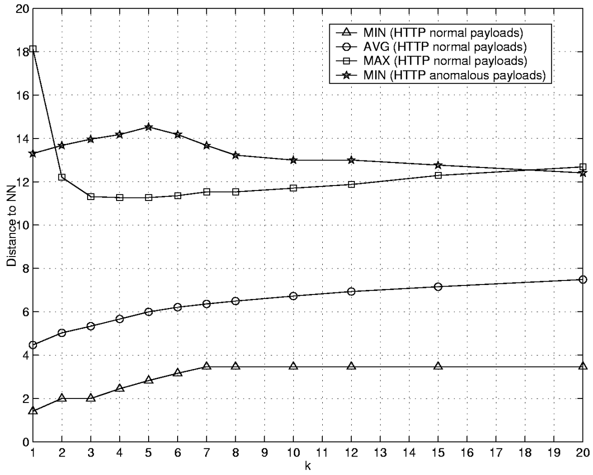

Fig. 1. Ranges and average value corresponding to the distance from each evaluated HTTP payload within the test data set to its nearest neighbor in the model. For each value of the window length k, the minimum, maximum and average distances are represented for normal traffic. In the case of attacks, however, only the minimum is shown due to limit problems (average values are around 270 and maximums around 1500).

payloads obtain scores within the interval $[13.229, 1501.232]$, with an average value of 263.472.

It is easily observed a clear separation between both types of traffic for values in the range $2 \leq k \leq 17$, reaching the maximum difference at $k = 5$. For these values of k, it would be possible to distinguish payloads carrying an attack from normal traffic with potentially no false alarms. Considering these results, a threshold $\theta \simeq 12$ for $k = 5$ seems adequate for an accurate detection. To be precise, for values of $k \in [2, 17]$, all the attacks considered during the evaluation were successfully detected without false alarms.

This is not, however, the case for DNS traffic, in which some overlapping exists between both regions. For instance, the anomaly score for normal DNS payloads ranges between 1.414 and 16.912 for a value of $k = 5$, with an average value of 2.034. The following table shows the anomaly score (A_s) obtained by each of the six attacks used in the evaluation with the same value of the parameter k:

Attack	$A1_{DNS}$	$A2_{DNS}$	$A3_{DNS}$	$A4_{DNS}$	$A5_{DNS}$	$A6_{DNS}$
A_s	4.472	35.972	41.061	28.775	254.790	4.000

According to this, it is clearly observed that attacks $A1_{DNS}$ and $A6_{DNS}$ are not correctly classified as anomalous, whilst the remainder four attacks yield anomaly scores easily recognizable as unusual. A brief analysis of the nature of these two attacks reveals the cause for such an unsuccessful detection. In the case of $A1_{DNS}$ (usually known as PROBE-IQUERY), it is basically a request attempting to determine if a name server supports inverse queries. Even though this action constitutes a usual pre-attack probe and, hence, it should be restricted by the site security policy, it does not involve elements that can be designated as out of the ordinary. A similar rationale can be provided for the $A6_{DNS}$ attack (EXPLOIT-INFOLEAK-LSD).

4 Performance Considerations and Improvements

The detection process is performed through a nearest neighbor (NN) search and a subsequent comparison with an established threshold. Since the computation of each of the N required distances (one per pattern in the model) involves $O(n^2)$ operations, the time required to decide whether a given payload is anomalous or not can be approximated by the following expression:

$$t_{Detection} \approx \mathcal{C} \cdot \tau(n^2) \cdot N \qquad (4)$$

where $\mathcal{C}$ is a constant factor that includes further operations involved in the decision as well as implementation-dependant details.

According to Table 1, the model sizes have been, respectively, $N = 3262$ payloads for HTTP traffic, and $N = 46849$ for DNS. Concerning the factor corresponding to the distance algorithm, the time required to carry out the computation has been estimated during the experiments in a computer with a Pentium 4 processor at 2.4 GHz with 1 GB of RAM. The obtained times, which obviously depend on the length of the input payloads, range between 0.000298 ms for shorter requests and 0.0579 ms for the largest, with an average value of 0.00483 ms.

Taking into consideration the average value, it is possible to derive an approximation for the number of service requests per second that is possible to process under these conditions. In the case of HTTP with the current model, the average time required for reaching a decision about the anomalous nature of a request is $t_{Detection} = 0.00483 \cdot 3262 = 15.76$ ms. This value implies that, in a computer similar to that used in our laboratory, the system will be able to process around 63 HTTP requests per second. In the case of DNS, this value is around 5 requests per second. Considering the typical load of a server, these results could sternly limit the application in a real-time environment. Despite this undeniable fact, it is necessary to remember that most of the currently proposed anomaly methods involve detection mechanisms much more complex and inefficient than that exposed here.

As established in expression (4), the two major factors involved in the system operation and capable of being improved are the distance computation algorithm and the model size (i.e., the number of prototypes that define the notion of normal traffic). The second of them could be easily reduced by means of a clustering algorithm. In this sense, we have obtained some preliminary results through the application of a $k - means$-like procedure to the HTTP and DNS models used in Section 3. Although a complete discussion is not shown here due to space reasons, in both cases a reduction in the number of prototypes up to 98% has been obtained without affecting the detection capabilities. Thus, considering a reduced model composed of 25 HTTP prototypes and using the same time values for distance processing provided below, the detection time is reduced from the original $t_{Detection} = 15.76$ ms to $t_{Detection} = 0.00483 \cdot 25 = 0.121$ ms. With this value, the detector can handle around 8264 requests per second, which clearly outperforms the 63 requests per second of the original model. In the case of DNS, this improvement is still higher: With a model of $N_p = 107$ prototypes, the system can handle around 1937 requests per second, in contrast with the limited 5 requests per second corresponding to the original complete model.

5 Conclusions and Future Work

In this article, a new approach for detecting anomalies in network traffic at the application layer has been presented, together with some experimental results that confirm its efficacy. The proposed method has been applied to HTTP and DNS traffic, and it models each service request, p, through its structural components provided by mapping it to a defined feature space Θ_k. Due to the very nature of the attacks at the application layer, the introduced distance function seems to separate accurately between normal and anomalous payloads, allowing thus to detect suspicious traffic. Despite the undoubted benefits of the introduced work, there exist a number of features that can be improved in order to construct more efficient and effective detection devices based on the proposed technique.

An important objective is that of reducing the computational complexity of the algorithms involved in the distance calculation. As stated in the previous discussion, this is currently a limiting factor in the detection performance. On the other hand, there exist more efficient algorithms for performing the NN search than that used during our experiments, which can be used in a real application. Likewise, the use of alternative, and perhaps more sophisticated distances can yield better results than those obtained in our experimentation and exposed in this article. By using the proposed framework as a basis, one of the most promising research lines is that of developing less coarse models that make use of the knowledge provided by the well-known message formats. We firmly believe that the inclusion of protocol-dependant information of a semantic nature, coupled with the use of more complex algorithms on sequences, will provide enhanced detection mechanisms.

Acknowledgements. This work has been partially supported by the Spanish Government through MECD (National Program PNFPU, reference AP2001-3805) and MCYT (project SERVIRA, TIC2002-02798, FEDER funds 70%).

References

1. Landwehr, C.E. “Computer Security”, in *International Journal on Information Security*, Vol. 1, No. 1, 2001, pp. 3–13.
2. Project OASIS: Organically Assured and Survivable Information System. Available online at: `http://www.tolerantsystems.org/`
3. Project MAFTIA: Malicious and Accidental Fault Tolerance for Internet Applications. Available online at: `http://www.newcastle.research.ec.org/maftia/index.html`
4. McHugh, J., “Intrusion and Intrusion Detection”, in *International Journal on Information Security*, Vol. 1, No. 1, , 2001, pp. 14–35.
5. Kemmerer, R.A. and Vigna, G., “Intrusion Detection: A Brief History and Overview”, in *IEEE Computer*, Vol. 35, Issue 4, April 2002, pp. 27–30.
6. Allen, J., Christie, A., Fithen, W., McHugh, J., Pickel, J., and Stoner, E. “State of the Practice of Intrusion Detection Technologies”. Technical Report CMU/SEI-99-TR-028, Software Engineering Institute, Carnegie Mellon, January 2000.
7. Axelsson, S. “Intrusion Detection Systems: A Survey and Taxonomy”. Technical Report 99-15, Department of Computer Engineering, Chalmers University of Technology, Goteborg.
8. Krügel, C., Toth, T., and Kirda, E. “Service Specific Anomaly Detection for Network Intrusion Detection”, in *Proceedings of the 17th ACM Symposium on Applied Computing (SAC)*, pp. 201–208, Madrid (Spain), 2002.
9. Mahoney, M.V. and Chan, P.K., “Learning Nonstationary Models of Normal Network Traffic for Detecting Novel Attacks”, in *Proceedings of the 8th International Conference on Knowledge Discovery and Data Mining*, 2002, pp. 376–385.
10. Mahoney, M.V. and Chan, P.K., “An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection”. Florida Institute of Technology Technical Report CS-2003-02, 2003.
11. Mahoney, M.V., “Network Traffic Anomaly Detection Based on Packet Bytes”, in *Proceedings of the 18th ACM Symposium on Applied Computing (SAC)*, Melbourne, FL (USA), 2003, pp. 346–350.
12. Estevez-Tapiador, J.M., Garcia-Teodoro, P., Diaz-Verdejo, J.E., “Stochastic Protocol Modeling for Anomaly-Based Network Intrusion Detection”, in *Proceedings of the 1st IEEE International Workshop on Information Assurance (IWIA’03)*, Darmstadt (Germany), March 2003, pp. 3–12.
13. Gusfield, D., *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*, ISBN: 0521585198, Cambridge University Press, 1997.
14. Lippmann, R. Haines, J. W., Fried, D. J., Corba, J., and Das, K.: “The 1999 DARPA Off-line Intrusion Detection Evaluation”, in *Computer Networks*, Vol. 34, No. 4, 2000, pp. 579–595.
15. McHugh, J., “Testing Intrusion Detection Systems: A Critique to the 1998 and 1999 DARPA Intrusion Detection Evaluations as Performed by Lincoln Laboratory”, in *ACM Transactions on Information and Systems Security*, Vol. 3. No. 4, November 2000, pp. 262–294.
16. arachNIDS: Advanced Reference Archive of Current Heuristics for Network Intrusion Detection Systems. Available online at: `http://www.whitehats.com/ids`