Improving Accuracy of Immune-inspired Malware Detectors by using Intelligent Features

M. Zubair Shafiq
nexGIN RC
NUCES-FAST
Islamabad, Pakistan
zubair.shafiq@nexginrc.org

Syed Ali Khayam
WisNeT
SEECS-NUST
Rawalpindi, Pakistan
khayam@niit.edu.pk

Muddassar Farooq
nexGIN RC
NUCES-FAST
Islamabad, Pakistan
muddassar.farooq@nexginrc.org

ABSTRACT

In this paper, we show that a Bio-inspired classifier's accuracy can be dramatically improved if it operates on *intelligent features*. We propose a novel set of intelligent features for the well-known problem of malware portscan detection. We compare the performance of three well-known Bio-inspired classifiers operating on the proposed intelligent features: (1) Real Valued Negative Selection (RVNS) based on the adaptive immune system; (2) Dendritic Cell Algorithm (DCA) based on the innate immune system; and (3) Adaptive Neuro Fuzzy Inference System (ANFIS). To empirically evaluate the improvements provided by the intelligent features, we use a network traffic dataset collected on diverse endpoints for a period of 12 months. The endpoints' traffic is infected with well-known malware. For unbiased performance comparison, we also include a machine learning algorithm, Support Vector Machine (SVM), and two state-of-the-art statistical malware detectors, Rate-Limiting (RL) and Maximum-Entropy (ME). To the best of our knowledge, this is the first study in which RVNS and DCA are not only compared with each other but also with several other classifiers on a comprehensive real-world dataset. The experimental results indicate that our proposed features significantly improve the *TP rate* and *FP rate* of both RVNS and DCA.

Categories and Subject Descriptors

D.4.6 [**Software**]: Security and Protection—*Invasive software*; C.2.0 [**Computer Systems Organization**]: Computer Communication Networks—*Security and protection*

General Terms

Algorithms, Experimentation, Security

Keywords

Artificial Immune System, Dendritic Cell Algorithm, Adaptive Neuro Fuzzy Inference System, Negative Selection, Network Endpoints, Support Vector Machines

1. INTRODUCTION

Over the last few years, there has been a dramatic increase in the volume and sophistication of network attacks. These attacks disrupt e-business and e-commerce activities, thus resulting in significant loss of revenue and credibility. In order to effectively contain rapidly self-propagating malware[1], real-time defense mechanisms must be designed to detect *zero-day* (i.e., previously unknown) attacks. Therefore, a significant amount of security research effort in the recent past has been focused on network-based malware and anomaly detection. These detectors learn the benign behavior of a network entity and then flag anomalous behavior by measuring deviations from the learned behavior.

In the domain of Bio-inspired security, Artificial Immune Systems (AIS) have served as a natural source of inspiration for network-based anomaly detection. Two popular AIS paradigms exist: (1) Real Valued Negative Selection (RVNS) based on the adaptive immune system [4],[5]; and (2) Dendritic Cell Algorithm (DCA) based on the innate immune system [1],[2]. Our observation is that both paradigms, though inspired from totally different biological processes, use naive traffic features as an input to the classifier. In other words, both paradigms pay no attention to intelligent features that can accurately discriminate malicious and benign network activity. *The thesis of this paper is that naive features degrade classification accuracy and, therefore, intelligent traffic features should be used to fully exploit the potential of Bio-inspired classifiers.*

This thesis can be intuitively argued for the RVNS classifier, which uses hyper-spheres as detectors to cover the non-self space. The average volume of hyper-spheres converges to zero at higher dimensions resulting in inadequate coverage [8]. Similarly, the classification accuracy (defined with two parameters: true positive rate, TP rate $= \frac{TP}{TP+FN}$, and false positive rate, FP rate $= \frac{FP}{FP+TN}$, [23]) of DCA is sensitive to the selection of signals. Naive signals, consequently, have a direct adverse impact on the DCA's classification accuracy. The results of our pilot studies on a real world traffic dataset clearly indicate that, due to these shortcomings, both RVNS and DCA have poor classification accuracy.

This insight into existing Bio-inspired classifiers' shortcomings motivated us to identify novel intelligent features that: (1) have the potential to discriminate benign and malicious traffic patterns; (2) can reduce the dimensionality of the feature space without compromising the classification ac-

[1]Due to our network-based focus, throughout this paper the term malware implicitly refers to 'self-propagating malware'.

curacy; and (3) have computational complexity and memory requirements that do not explode with respect to the volume of the observed traffic. In this paper, we propose and evaluate novel information-theoretic and statistical features that satisfy these design requirements. We input these features to both RVNS and DCA for classification. The results of our experiments show that due to the use of the proposed intelligent features, classification accuracies of RVNS and DCA improve dramatically. This improvement substantiates our thesis that the root cause of bio-inspired classifiers' low accuracy is a poor choice of training features, not the classification algorithm. In addition to comparing RVNS with DCA, we also compare the enhanced algorithms with another Bio-inspired algorithm, Adaptive Neuro Fuzzy Inference System (ANFIS) [20], a machine learning algorithm, Support Vector Machine (SVM) [22], and two state-of-the-art statistical malware detectors, Rate Limiting (RL) [11] and Maximum Entropy (ME) [13] detectors.

In order to provide unbiased, comprehensive and realistic performance comparison of all schemes, we have spent 12 months in collecting the traffic statistics of a diverse set of endpoints in home, office, and university environments. For malicious network activity, we use real and simulated worms which vary in both their propagation rates as well as in their scanning techniques. To the best of the authors' knowledge, this work is the first one in which so many anomaly detectors are compared on such a comprehensive dataset. We believe that our comprehensive real world dataset[2] will significantly help researchers, working in the anomaly detection domain, in validating the classification accuracy of their algorithms. We summarize the most important contributions of the work presented in this paper as follows:

- Identification of an intelligent feature set for detecting self-propagating malware, which can act as an input to any classifier;
- The first ever one-to-one comparison, according to the best of our knowledge, of the classification accuracy of RVNS and DCA algorithms;
- A proof-of-concept that intelligent training features are significantly more important than the Bio-inspired classifier, all three intelligent Bio-inspired classifiers of this paper provide comparable performance;
- A comprehensive real world network dataset collected from the endpoints deployed in diverse environments;
- An unbiased comparison of three Bio-inspired anomaly detectors, RVNS, DCA, and AFNIS, with SVM, RL and ME detectors.

The rest of the paper is organized as follows. In the next section, we present the related work. We describe the traffic test bed used in this study in Section 3. In Section 4 we report the results for AIS based anomaly detectors, RVNS and DCA, which use classical traffic features. In Section 5, we present an overview of our proposed intelligent information theoretic features. In Section 6, we present an overview of non-AIS classification schemes, namely ANFIS, RL and ME, used in this study. In Section 7, we present the comparative results of all classifiers utilizing our proposed intelligent features. We then conclude the paper in Section 8 with an outlook to our future work.

[2]available online at http://www.nexginrc.org

Table 1: Statistics of Benign Profile Collected for This Study

End-point ID	**Endpoint Type**	**Total Profile Time** months	**Total Sessions**	**Mean Session Rate** /sec	**Var in Session Rate** /sec
1	Office	8	33, 487	0.25	0.26
2	Office	10	21, 066	0.22	0.43
3	Home	3	373, 009	1.92	11.98
4	Home	2	444, 345	5.28	25.93
5	Home/Univ	3	27, 873	0.44	2.0
6	Univ	9	60, 979	0.19	0.35
7	Univ	11	171, 601	0.28	0.6
8	Univ	13	41, 809	0.52	0.71
9	Univ	13	235, 133	0.41	0.81
10	Univ	13	152, 048	0.21	0.37
11	Univ	13	207, 187	0.31	0.96
12	Home/Univ	13	100, 702	0.33	0.73
13	Univ	3	11, 996	0.23	0.66

2. RELATED WORK

In [1],[2], the authors have used AIS utilizing DCA for SYN scan detection. They have used a number of classical features as the input signals to DCA. This system achieves 100% detection rates at appropriate thresholds on the dataset used by the authors in their experiments. However, the authors have provided no information about the scalability of their dataset in terms of the scanning rates. Moreover, the authors did not correlate the relevance of the scanning techniques used in their experiments to those utilized by the real worms.

Stibor et al. in [9] have compared negative selection, positive selection, one-class SVMs and Parzen window based detector using KDD Cup 1999 dataset. Their results show that the positive selection is the best classification algorithm but has large computational complexity. SVMs also provide good classification accuracy. In [3], the authors have carried out a comparative study of fuzzy inference system, neural network and ANFIS for portscan detection. Their results show that ANFIS successfully combines the benefits of fuzzy representation with the back-propagation learning algorithm, and as a result, gives better classification accuracy as compared to both techniques.

The most commonly used endpoint-based & network-level malware detection technique is RL. This technique proposed by Twycross and Williamson limits the rate of an endpoint's network traffic to curb and detect malware propagation [11]. Lakhina et al. proposed a subspace method to detect and characterize network-wide volumetric traffic anomalies. The authors then extend their work and use entropy to detect anomalies (see [12] and references therein). A recent study by Gu et al. [13] uses maximum entropy estimation to quantify a baseline distribution at a network gateway or a router, which is then used to classify an anomalous activity using the KL divergence measure.

3. TRAFFIC DATASET COLLECTION

In this section, we explain the two main datasets collected for our study. The first dataset comprises of benign traffic profiles which are collected from several hosts with regular human users. The second dataset comprises of real and simulated worm traffic.

3.1 Benign Traffic Profiles

Our first step towards doing an unbiased comparison of different types of anomaly detectors was to collect the pertinent network traffic data. We invested 12 months in monitoring the network profiles of a diverse set of 13 endpoints. The users of these endpoints included home users, research students, and technical/administrative staff with Windows 2000/XP laptop and desktop computers. The laptop endpoints were used by their users both at their homes and at offices. Some endpoints, in particular home computers, were shared among multiple users. The endpoints used in this study were running different types of applications, including peer-to-peer file sharing software, online multimedia applications, network games, SQL/SAS clients etc.

Data was collected by a multi-threaded windows application called `argus`, which runs as a background process storing network activity in a log file. The log file is periodically and securely uploaded to a secure copy (SCP) server. **argus** only logs session-level information, where a *session* corresponds to a bidirectional communication between two IP addresses. The communication between the same IP address on different ports is considered part of the same network session. This session-level granularity reduces the complexity of the worm detector, while providing the complete information about the sessions originating from or terminating at an endpoint. Each session is logged using the information contained in the *first* packet of the session. A session expires if it does not send/receive a packet for more than τ seconds. In the collected data, τ is set to 10 minutes.

Each entry of the log file has the following 6 fields:

```
<session id, direction, src port, dst port,
          protocol, timestamp>,
```

Some pertinent statistics of the collected benign data are listed in Table 1[3]. Diversity of the endpoints used in this study is evident from Table 1, which shows that the endpoints operate in different environments and hence run different types of applications. Also, the total size of the dataset (i.e., total number of sessions) varies from 11,996 for endpoint 13 to 444,345 for endpoint 4. In general, we observed that the home computers generate significantly higher traffic volumes than the office and the university computers because: (1) they are generally shared between multiple users, and (2) they run peer-to-peer and multimedia applications. The large traffic volumes of home computers are also evident from the high mean and the variance of the number of sessions per second. In addition to benign profiles, we have also collected malicious data generated by the real and simulated worms. The following section explains collection of the malicious traffic data.

3.2 Worm Classification

To generate traffic patterns for each worm, we infected a vulnerable machine with a worm and observed the traffic generated by the worm using the `argus` data utility described in the previous section. The vulnerable machines used here are different from the operational endpoints used for benign profile collection. This section details the worms collected and simulated in this study.

[3]It should be noted that the mean and the variance of the session rates in Table 1 are computed using time-windows containing one or more new sessions. As can be inferred intuitively, time-windows without new network sessions are fairly common on endpoints.

Table 2: Information of Worms Used in This Study

Worm	Release Date	Avg. Scan Rate	Port(s) Used
`Blaster`	Aug 2003	10.5 sps	TCP135,4444, UDP 69
`Dloader-NY`	Jul 2005	46.84 sps	TCP 135,139
`Forbot-FU`	Sep 2005	32.53 sps	TCP 445
`MyDoom-A`	Jan 2006	0.14 sps	TCP 3127 − 3198
`RBOT.CCC`	Aug 2005	9.7 sps	TCP 139,445
`Rbot-AQJ`	Oct 2005	0.68 sps	TCP 139,769
`Sdbot-AFR`	Jan 2006	28.26 sps	TCP 445
`SoBig.E`	Jun 2003	21.57 sps	TCP 135,UDP 53
`Zotob.G`	Jun 2003	39.34 sps	TCP 135,445,UDP 137
`Witty`	Mar 2004	357.0 sps	UDP 4000
`CodeRed II`	Jul 2004	4.95 sps	TCP 80
`Sim Src Port`	Simulated	3.57 sps	TCP 1500

3.2.1 *Real Worms*

A critical aim of our study is to use real and diverse worm data to compare the efficacy of different techniques. To this end, we installed original and unpatched releases of Windows 2000 and Windows XP on a computer using Microsoft Virtual PC. The advantage of using virtual machines (VMs) was that once a virtual host was infected, we could reinstall it by overriding just a few key files. We assigned static IP addresses to both virtual machines and connected them to the Internet. These hosts were then compromised by the following malware: `Zotob.G`, `Forbot-FU`, `Sdbot-AFR`, `Dloader-NY`, `SoBig.E@mm`, `MyDoom.A@mm`, `Blaster`, `Rbot-AQJ`, and `RBOT.CCC`. An interested reader can find the details about the worms used in our experiments in [16].

Table 2 shows the diversity of the worms used in this paper. These worms have different (and sometimes multiple) attack ports and transport protocols. Also, these worms include both high- and low-rate worms; `Dloader-NY` has the highest scan rate of 46.84 scans per second (sps), while `MyDoom-A` and `Rbot-AQJ` have very low scan rates of 0.14 and 0.68 sps respectively. In addition to this, we also simulated three additional worms that were somewhat different from the worms described above.

3.2.2 *Simulated Worms*

We first simulated the source port `Witty` worm [17],[16]. We tested the worst-case scenario with 20,000 scan packets at the average scan rate of 357 sps. We also simulated the HTTP-based `CodeRed II` worm using an average scan rate of 4.95 sps [16]. We acknowledge that it is unlikely that an endpoint will be running a service that can be infected by an HTTP worm. Nevertheless, we simulated an HTTP worm because its scan packets use destination port 80, which is a very common port in the benign profile of an endpoint. Finally, we also simulated a source-port worm that sends scan packets with a fixed TCP source port of 1500 at an average scan rate of 3.57 sps. Note that this scan rate is exactly 100 times less than the `Witty`'s average scan rate, which makes this simulated worm very challenging to detect.

3.3 Inserting Worm Data in the Benign Traffic Profile

We implemented the propagation modules of the simulated worms. A vulnerable VM was then infected with each of the 12 worms. We then used `argus` to log malicious traffic traces from the VM in the same format as of the benign data. Armed with this information, we inserted T minutes of malicious traffic data of each worm in the benign profile of each endpoint at a random time instance. Specifically, for a given endpoint's benign profile, we first generated a random infection time t_I (with millisecond accuracy) between the endpoint's first and last session times. Given n worm sessions starting at times $t_1, \dots, t_n$, where $t_n \leq T$, we created a special *infected* profile of each host with these sessions appearing at times $t_I + t_1, \dots, t_I + t_n$. Thus in most cases once a worm's traffic is completely inserted into a benign profile, the resultant profile contains the interleaved benign and worm sessions starting at t_I and ending at $t_I + t_n$. For all worms except `Witty`, we used $T = 15$ minutes and to simulate the worst-case behavior of `Witty`, we inserted only $20,000$ scan packets (i.e., approximately $T = 1$ minute of malicious traffic) in each `Witty`-infected profile.

4. PILOT STUDIES

In the following subsections, we will now introduce two well known Bio-inspired anomaly detectors, RVNS and DCA, which use classical naive traffic features for classification of the malicious traffic. We also report the results of our pilot studies.

4.1 Artificial Immune System utilizing Negative Selection

Negative Selection was proposed by Forrest et al. [4] and derives inspiration from the adaptive immune system. Lymphocytes (detectors) mature in thymus and undergo the negative selection. Only those lymphocytes survive the negative selection phase which do no match any self cells presented in thymus. The matured lymphocytes have the ability to distinguish between self and non-self cells (antigens). Negative selection algorithm in AIS has been improved gradually [5],[14]. In this paper, we consider real-valued negative selection algorithm with fixed sized detectors. The variable sized detectors have been also proposed by the authors in [14], however, our pilot studies show that the variable sized detectors give similar classification accuracy as compared to the fixed sized ones but at slightly less memory requirements. Moreover, they also have relatively higher computational complexity. The antigen representation is <`direction, dst ip, src port, dst port, protocol`>. In this study, we use the Euclidean matching rule because of the real-valued nature of the input feature space [14]. We use a fixed population size of 1000 detectors. For this study we have used a custom implementation of RVNS in C++. See [5] for details of this implementation.

4.2 Artificial Immune System utilizing Dendritic Cell Algorithm

Dendritic Cells (DC) are the biological cells whose functionality is the part of the innate immune system. DCs can stimulate immune system response with the help of *signals* and location markers which are called *antigens*. DCA is designed to detect 'danger' instead of non-self as is the case with the negative selection based AIS. The signals include danger signals, safe signals, PAMPs (Pathogenic Associated Molecular Patterns), and inflammatory cytokines. An immature DC converts to either a mature DC or a semi-mature DC depending upon the relative concentration of these signals. Mature DCs represent the danger and in turn activate the immune response whereas semi-mature DCs represent safe and in turn suppress the immune response. In [2], the authors have used 4 signals for the SYN scan detection problem: (1) the number of error messages generated per second by a failed network connection (PAMP); (2) the number of transmitted network packets per second (danger signal); (3) the inverse rate of change of number of network packets per second (safe signal); (4) high system activity (inflammatory signal). However, the PAMP and inflammatory signals are host based features which are undesirable because network-based malware detectors are generally more reliable, robust, and, most importantly, generic and scalable than host-/OS-based detectors. Therefore, we use the variance of destination IP addresses as a PAMP and the inverse of average inter-arrival session time as an inflammatory signal (see [12] and references therein for this decision).

The classification accuracy of DCA is sensitive to the weights which are used to derive the combined context of all signals. The founders of the DCA have not given any systematic method to derive these weights. In this paper, we have randomly tuned these weights in the initial training phase using a labeled training traffic dataset. In [2], the authors have used 3 danger signals with positive weights and one safe signal with a negative weight. We observed that, in case of a safe signal, we can simply use the rate of change instead of the inverse of rate of change but with a positive weight in order to achieve the same classification accuracy. All other parameters values are the same as in [2]. In this study, we have used a custom implementation of DCA in C++. More implementation details can be found in [2].

4.3 Discussion on Results

The classification accuracy for RVNS and DCA, utilizing classical input feature space, are tabulated in Table 3[4].

The work by Stibor et al. has shown that the negative selection is not appropriate for higher dimensional datasets [8], [10]. The expected volume of the hyperspheres (detectors) is given by:

$V(n,r) = r^n \frac{\pi^{n/2}}{\Gamma(\frac{n}{2}+1)}$,

where n is the number of dimensions and r is the radius of sphere. Therefore $V(n,r)$ converges to zero as $n \to \infty$. The dataset used in this study is a time-series data consisting of 8 dimensions: The session id represents a unique set of source and destination IP addresses. Since an IP address is divided into 4 sub-integers, the dataset consists of: 4 (dst IP) + 1 (protocol) + 1 (direction) + 1 (src port) + 1 (dst port) = 8 dimensions. We observed that providing these classical features to a negative selection based AIS leads to poor classification accuracy. This poor performance is possibly because of the fact that the definition of *self* is not stable, i.e. the *self* region changes gradually. The 'self' destination IP addresses and ports tend to vary over time due to the pseudo-random user behavior.

DCA with these classical input features produces relatively high false positive rates for the endpoints 3 and 4. These endpoints are home endpoints where multimedia and peer-to-peer applications resulted in high and unstable traffic rates. This clearly shows that the used danger signals tend to oversimplify the challenging real world scenarios and hence can lead to a large number of false positives. We also

[4]The values reported in Table 3 are with an overall 95% confidence level using *t distribution.*

Table 3: Accuracy of RVNS and DCA using Classical Input Features

Endpoint ID	1	2	3	4	5	6	7	8	9	10	11	12	13
RVNS													
TP Rate	48.3%	36.5%	56.2%	53.7%	54.8%	57.3%	59.6%	56.5%	57.9%	54.2%	53.7%	52.9%	54.5%
FP Rate	6.0%	4.5%	18.2%	20.8%	5.7%	7.5%	6.4%	4.1%	6.6%	6.2%	5.1%	5.8%	6.5%
DCA													
TP Rate	62.1%	63.4%	61.5%	61.1%	63.1%	63.5%	61.3%	61.1%	61.1%	61.3%	60.0%	59.0%	62.4%
FP Rate	2.3%	3.2%	22.5%	25.6%	1.3%	2.2%	2.3%	2.3%	2.2%	2.3%	2.2%	2.3%	2.3%

performed similar experiments as reported in [2] by swapping the order of the signals. In contrast to the results reported by the authors in [2], the results of all possible permutations were exactly the same in our case. We believe that this is due to the reason that the weights were not set to a pre-defined value [2]; rather they were tuned in the learning phase. Therefore, the order of the signals has no impact on the classification accuracy of the system.

This leads us to our thesis that *poor classification accuracy of RVNS and DCA algorithms is attributable to the naive classical features; otherwise AIS is not a good paradigm for self-propagating malware detection.* As a first step, we propose some novel intelligent features to substantiate our thesis in an incremental manner. This strategy is also motivated by Ji and Dasgupta [15] in the concluding remarks as the future work.

5. INTELLIGENT TRAFFIC FEATURES

In this section we present an overview of our intelligent, information theoretic features. We calculated each feature in a fixed time-window of 30 seconds. But qualitatively similar results were obtained even for other window sizes.

5.1 Burstiness of Session Arrivals

Intuitively, automated malicious traffic should have different burstiness behavior than benign traffic. For instance, a human user does not typically initiate sessions as fast as a worm. This is because the traffic due to a worm is usually a constant artificial burst that is different from the bursts observed during high-rate benign network activity (e.g., peer-to-peer applications). Therefore, monitoring session burstiness can potentially highlight anomalous traffic activity.

Given the session arrivals or initiations in a time-window, we divide the observed session arrival data into equal-sized discrete bins of length τ seconds each. Then, if one or more sessions are initiated in a particular bin, we set its value to 1, where a 1 represents a session arrival event. In our pilot studies, we have determined that using $\tau = 0.001, 1, 2, 3, 4$ seconds gives us appropriate information for both low- and high-rate attack detection. However, more resolutions can be incorporated to obtain higher levels of accuracy if complexity is not an issue. To capture the burstiness of session arrival events we model the discrete arrivals as a Gilbert Markov chain. The Gilbert chain is a first-order discrete-time Markov Chain with two states (say 0 and 1) [19]. An information-theoretic measure to quantify the memory of a Gilbert model, where the memory μ was defined in [19] as: $\mu = 1 - P_{0|1} - P_{1|0}$, where $-1 \leq \mu \leq 1$. Alternatively, the above expression can be written as $P_{0|1} = \pi_0(1-\mu)$ and $P_{1|0} = \pi_1(1-\mu)$, where π_0 and π_1 represent the steady-state probabilities of staying in states 0 and 1, respectively. Based on this definition, $\mu = 0 \Rightarrow \pi_0 = P_{0|1}$ and $\pi_1 = P_{1|0}$.

Increasing values of μ represent high levels of burstiness/memory. As mentioned earlier, we develop five aggregate

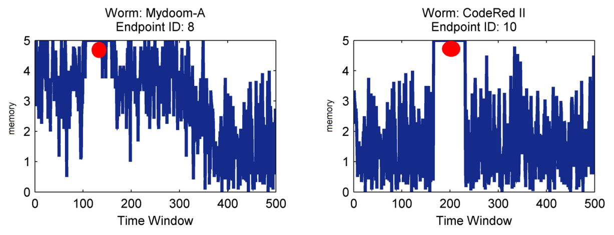

Figure 1: Gilbert memory results for low and high scan rate worms. Red circle highlights the center of the infection period.

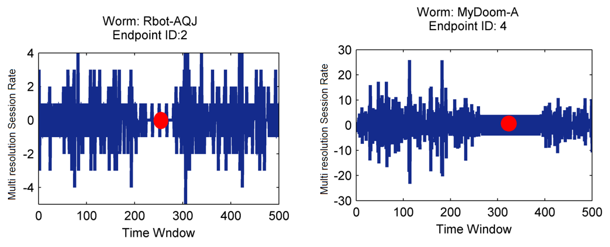

Figure 2: Multi-resolution session rate for low scan rate worms.

session arrival processes using bin lengths of $\tau = 0.001, 1, 2, 3, 4$ seconds. For each of these processes, we calculate the Gilbert model's parameters and the consequent memories, μ_τ, in a time window of 30 seconds. We then use the sum of these memories as a feature: $\mu_\sum = \mu_{0.001} + \mu_1 + \mu_2 + \mu_3 + \mu_4$, where $-5 \leq \mu_\sum \leq 5$.

Fig. 1 shows the results of the proposed memory feature for a low- and a high-rate worm on two distinct endpoints. We observe that the memory of normal user activity is usually low, i.e., user's traffic behavior changes continuously, even if the change is somewhat small. On the other hand, the worm traffic in the infected profiles shows a large amount of memory.

5.2 Multi-resolution Session Rates

Network traffic from a compromised machine generally has considerably higher volumes than from a benign host. We used the aggregate processes defined in the last section to compute session rates at different time resolutions; for each resolution, we count the number of arrivals in non-overlapping time-windows of 30 seconds. Moreover, to avoid data replication, we make the aggregate processes mutually exclusive by accounting for each session in only one of the aggregate processes.

Let λ_τ represent the session rate at resolution τ. Then as in the last section, we use the following aggregated feature: $\lambda_\sum = \lambda_{0.001} + \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4$. We also observed that $\lambda_\sum$ values do not vary considerably during an attack. Therefore, at any time instance n, rather than looking at the aggregate session rate, say $\lambda_{\sum}^{n}$, we consider its first-order derivative: $\lambda_{\sum}^{'} = |\lambda_{\sum}^{n} - \lambda_{\sum}^{n-1}|$.

Fig. 2 shows multiresolution session rates for low-rate worms. We should mention that this multiresolution session rate feature can be defeated if a worm is sophisticated enough to adapt its scan rates in accordance with the changing session initiation rates of benign traffic. However, such adaptability will place increased computational and program memory requirements on the worm, thereby compromising its propagation speed and stealthiness.

5.3 Entropy of Destination IP Addresses

A worm propagates by scanning for vulnerable hosts on the Internet. To this end, a compromised host is used to probe different IP addresses either randomly or using a hitlist. Similarly, in DoS attacks, multiple source IPs typically target the same destination IP subnet. Under both attack scenarios, the number of unique IP addresses per time-window increases significantly. This effect can be captured using the entropy.

To calculate the entropy of destination IP addresses, let Δ_n denote the set of destination IPs observed in the window n. Define $X_n = \{p_i^n, i \in \Delta_n\}$ as the destination IP histograms derived from the time window n, where p_i^n is the number of packets having a destination IP i in the time-window n. Also define, $p_n = \sum_{i \epsilon \Delta_n} p_i^n$ as the aggregate frequency of destination IPs observed in the window n. We use an entropy measure called *Tsallis entropy* [19]. This measure is a generalization of the Boltzmann-Gibbs entropy and is computed as:

$S(X_n) = \frac{1}{b-1}\left(1 - \sum_{i \epsilon \Delta_n}\left(\frac{p_i^n}{p_n}\right)^b\right)$,

where b is a real-valued parameter. We tuned this parameter experimentally and got the best results for $b = 0.5$. Entropy captures the increased variance in the IP usage due to the scan traffic.

Fig. 3 gives the Tsallis entropy results for two infected profiles. It can be observed that while high-rate attacks are easily highlighted using the IP-entropy, both entropy measures fail to detect low-rate attacks. After some investigation, we determined that anomalous activity goes undetected for low-rate attacks because entropy does not take the actual values of ports into consideration.

5.4 Divergence in Destination Port Distributions

Most worms target specific destination port(s) for portscans (see Table 2). This means that the destination port distribution will significantly change in case of the worm traffic. In the paper by Lakhina et al. [12], entropy was used as a measure to detect changes in the port distributions. However, as shown by Fig. 3, entropy cannot clearly highlight IP perturbations for low-rate attacks. Therefore, we need a measure that can differentiate between the port distributions on a port-by-port basis. An appropriate information-theoretic measure that quantifies the difference between the two probability distributions is the Resistor-Average (RA) divergence [19].

RA divergence is an information theoretic measure of the similarity or dissimilarity between two probability distributions. Let us denote the benign destination port histogram derived from an endpoint's benign profile as $X = \{p_i, i \in$

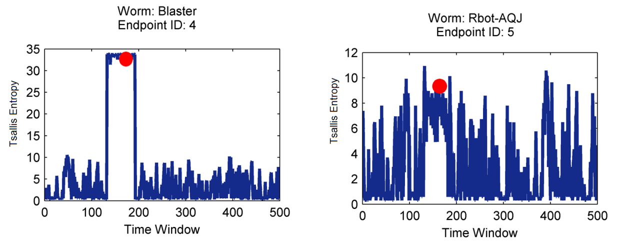

Figure 3: Tsallis entropy for high and low scan rate worms.

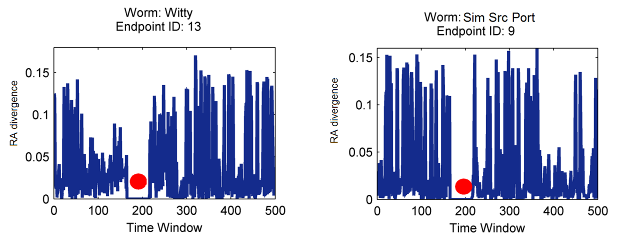

Figure 4: RA divergence for high rate and low rate worms.

$\Delta\}$, where Δ denotes the set of destination ports observed in the benign profile. Then RA divergence between the benign and currently observed port histograms can be expressed as: $\frac{1}{R(X_n||X)} \equiv \frac{1}{D(X_n||X)} + \frac{1}{D(X||X_n)}$, where $D(.||.)$ is the Kullback-Leibler (KL) divergence.

KL divergence is defined as:

$D(X_n||X) = \sum_{i \in \Delta_n} \frac{p_i^n}{p_n} \log_2 \frac{p_i^n/p_n}{p_i/p}$, where $p = \sum_{i \in \Delta} p_i$ represents the aggregate destination port frequency observed in the benign profile.

The results of RA divergence are shown in Fig. 4. We note that similar to the multiresolution session rate feature, RA divergence also gets perturbed at infection time and remains constant at the perturbed value for the infection interval. Therefore, in a time-window n, we use $R^{'} = |R_n - R_{n-1}|$ as the final feature output.

6. OVERVIEW OF NON-AIS CLASSIFIERS

In this section we present the review, parametrization and implementation details of non-AIS classification schemes used in our comparative analysis.

6.1 Adaptive Neuro Fuzzy Inference System

Adaptive Neuro Fuzzy Inference System (ANFIS) is a fuzzy rule-based classifier in which the rules are learnt from the examples using the standard back propagation algorithm. In this study we have used the implementation of ANFIS toolbox available in MATLAB [21]. We use a Sugeno type fuzzy inference system because of its low computational complexity and guaranteed continuity of output space. The subtractive clustering was used to divide the rule space. Five triangular membership functions were chosen for the inputs and two triangular membership functions were chosen for the output. We use a hybrid of least-squares method and back-propagation gradient descent method for the learning phase. Error tolerance was chosen to be 5% with 500 epochs. Please refer to [20] for details.

6.2 Support Vector Machine

Support Vector Machines (SVMs) are well-known machine learning classifiers. SVMs are inherently designed for binary decision tasks, such as anomaly detection. SVMs are trained in the learning phase using a labeled training dataset for minimum error tolerance with a maximum generalization. Given training vectors $x_i \in R^n$, $i = 1, 2, ..., l$ in two classes, and a vector $y \in R^l$ such that each $y_i \in \{+1, -1\}$, a C-SVM for non-separable data considers the following optimization problem [22]:

$$\min \quad \frac{1}{2}w^T w + C\sum_{i=1}^{l} \alpha_i y_i \big(w^T K(s_i, x) + b\big),$$
$$\text{subject to} \quad \alpha_i \geq 0, i = 1, 2, ..., l$$

In the objective function w is a perpendicular to the hyperplane that separates the positive and negative points, C is a parameter that is used to cost the α_i's, $K(s_i, x)$ is a non-linear kernel that maps the input data to another (possibly infinite dimensional) Euclidean space, and $s_i's$ are the points called the support vectors that maximize the separation between the positive and negative examples. We use a degree-3 radial basis kernel function to train the C-SVM. We have used `LIBSVM` implementation in this study [18]. Default settings were chosen for all other parameters.

6.3 Rate Limiting (RL) Detector

RL [11] is a technique used to limit the propagation of a self-propagating malware. RL is the only well-known worm detection technique that has been designed specifically for the endpoints. This approach is based on the observation that during malware propagation, an infected host tries to connect to as many different machines as possible, as fast as possible. A benign host, on the other hand, mostly attempts connections to the hosts that are locally correlated. Thus a large number of connections to new non-local hosts is treated as a malicious behavior by the RL algorithm. To curb such behavior, RL limits the rate of connections to new hosts. See [11] for more details.

6.4 Maximum Entropy (ME) Detector

Maximum entropy framework estimates the packet distribution of the benign traffic to detect anomalies. According to [13], the packets in the network traffic are divided into a set of two-dimensional packet classes. The first dimension deals with the anomalies concerning TCP and UDP packets. The TCP packets are further divided into two classes based on whether the packet is SYN or RST. The second dimension deals with dividing the packet into 587 classes according to their destination port numbers. The ME technique estimates the distribution of different packets in the benign traffic according to this classification, and then uses it as the baseline distribution to detect anomalies in the network traffic. See [13] for more details.

7. RESULTS

We have now developed a set of intelligent, effective and low-complexity traffic features which can be used for malware detection. We input these features to three Bio-inspired classifiers, namely RVNS, DCA and ANFIS. A labeled feature set was used to train these classifiers, followed by online classification and performance evaluation on unlabeled data. As mentioned earlier, in addition to the Bio-inspired classifiers, we also evaluate the performance of an SVM operating on these intelligent features and two existing standalone (RL and ME) detectors.

Table 4: Accuracy Comparison of All Classifiers

	***i*RVNS**	***i*DCA**	***i*ANFIS**	***i*SVM**	**RL**	**ME**
Endpoint ID - 1						
TP rate	95.1%	94.8%	96.2%	99.4%	91.0%	83.0%
FP rate	0.3%	0.1%	0.4%	0.0%	0.0%	0.0%
Endpoint ID - 2						
TP rate	95.6%	94.5%	96.5%	98.6%	92.0%	83.0%
FP rate	0.4%	0.1%	0.5%	0.1%	0.0%	0.0%
Endpoint ID - 3						
TP rate	95.9%	95.0%	96.9%	99.5%	85.0%	84.0
FP rate	0.4%	0.1%	0.5%	0.1%	11.0%	22.0%
Endpoint ID - 4						
TP rate	94.8%	94.6%	96.2%	98.7%	83.0%	83.0%
FP rate	1.3%	0.3%	1.7%	0.3%	7.0%	33.0%
Endpoint ID - 5						
TP rate	95.0%	94.8%	96.3%	99.5%	83.0%	83.0%
FP rate	0.1%	0.0%	0.2%	0.0%	0.0%	0.0%
Endpoint ID - 6						
TP rate	95.2%	94.8%	96.4%	99.4%	83.0%	83.0%
FP rate	0.1%	0.0%	0.2%	0.0%	0.0%	0.0%
Endpoint ID - 7						
TP rate	94.6%	94.6%	96.2%	99.4%	83.0%	83.0%
FP rate	0.2%	0.0%	0.3%	0.0%	0.0%	0.0%
Endpoint ID - 8						
TP rate	94.6%	94.6%	96.2%	99.4%	83.0%	83.0%
FP rate	0.1%	0.0%	0.2%	0.0%	0.0%	0.0%
Endpoint ID - 9						
TP rate	94.6%	94.6%	96.2%	99.4%	83.0%	83.0%
FP rate	0.1%	0.0%	0.2%	0.0%	0.0%	0.0%
Endpoint ID - 10						
TP rate	93.9%	94.0%	96.0%	99.0%	83.0%	83.0%
FP rate	0.1%	0.0%	0.2%	0.0%	0.0%	0.0%
Endpoint ID - 11						
TP rate	94.6%	94.6%	96.2%	99.3%	83.0%	83.0%
FP rate	0.1%	0.0%	0.2%	0.0%	0.0%	0.0%
Endpoint ID - 12						
TP rate	94.7%	94.7%	96.2%	99.3%	83.0%	83.0%
FP rate	0.1%	0.0%	0.2%	0.0%	0.0%	0.0%
Endpoint ID - 13						
TP rate	94.7%	94.7%	96.2%	99.4%	83.0%	83.0%
FP rate	0.1%	0.0%	0.2%	0.0%	0.0%	0.0%

Table 4 shows the comparative classification accuracy of all classifiers[5]. The term i in Table 4 represents the use of our proposed *intelligent* features as an input. The results clearly indicate a dramatic improvement in the classification accuracy of iRVNS and iDCA as compared to their respective classical counterparts, RVNS and DCA, reported in Section 4. Specifically, iRVNS and iDCA provide an average (over all endpoints) TP rate improvements of 41.32% and 33.27% respectively over classical RVNS and DCA classifiers. Similarly, average FP rate improvements of 7.72% and 5.37% are achieved over RVNS and DCA respectively.

The results of Table 4 prove our thesis that *the reason for poor classification accuracy of RVNS and DCA is* not *the classification algorithm, rather it is the use of naive input features.* In case of RVNS, the improvement can be attributed to two reasons: (1) dimensionality reduction; (2) stable definition of self. For DCA, this performance improvement can be attributed to an intelligent signaling model. It is interesting to note that the classification accuracy of both algorithms is approximately the same (see Table 4), provided they are given the same input features. We intuitively argue that the signal mapping in DCA is probably an equivalent of the self/non-self mapping in RVNS, since the weights in

[5]The values reported in Table 4 are with an overall 95% confidence level using *t distribution.*

DCA are tuned during the learning phase in a similar fashion as the detectors mature in the negative selection. Consequently, the antigens in DCA only determine the sampling instant of the signals. It appears that the only difference between the two approaches is the aggregate sampling by multiple DCs which is absent in RVNS [2]. But, based on our results, this difference does not significantly alter the behavior of the algorithm. In a future work, we will investigate the behavioral similarities of different components of RVNS and DCA. The TP rate of ANFIS is slightly better than both the AIS-based algorithms (albeit slightly higher FP rates), which is likely because of a more sophisticated learning algorithm which is a hybrid of least-squares and back-propagation algorithms.

It is important to note that the best classification results are obtained with the SVM classifier. However, SVMs have high algorithmic complexity and extensive memory requirements [22]. Therefore, they are not suitable for deployment in real-world networks. Nevertheless, it is an ideal candidate to act as a benchmark in our comparative study because of its high classification accuracy. This high classification accuracy is due to the fact that the learning algorithm in SVM not only minimizes the training error but also maximizes the generalization [22]. Generalization is a fundamental concept of machine learning which is not catered for in Bio-inspired (RVNS, DCA or ANFIS) classifiers. This statement is congruent with the conclusion of Stibor et al. [10], although an earlier work by Ji and Dasgupta [14] shows that generalization is dependent on the stopping criterion in a negative selection based learning algorithm.

The experimental results for statistical malware detectors used in our experiments show that the features alone do not have the ability to achieve high classification accuracy. Note that the classification accuracy of both RL and ME detectors is significantly lower than the Bio-inspired or SVM detectors. Therefore, we conclude that it is very important to give intelligent input features to *any* classifier in order to achieve a high classification accuracy. Interestingly, however, the statistical detectors manage to achieve considerably better accuracy than the classical RVNS and DCA detectors (compare with Table 3).

8. CONCLUSIONS

In this paper, we reported the results of an unbiased evaluation of Bio-inspired classification algorithms for the well-known problem of portscan detection. The important conclusion of the study is: *the use of intelligent features as an input to RVNS and DCA is the correct design strategy to achieve high classification accuracy in real-world networks.* We also conclude that AIS-based classifiers operating on intelligent traffic features can be readily deployed in real-world networks because of their reasonably high accuracy and relatively low computational complexity. However, there is still some room for improvement in the classification accuracy of Bio-inspired classifiers for them to be competitive with SVM-like classifiers, which, due to their very high computational complexity, are only suitable as benchmark algorithms. In future, we also plan to carry out a comparative study of algorithmic complexity and memory requirements of different classifiers by implementing them in a unified framework.

Acknowledgement

This work is supported by the National ICT R&D Fund, Ministry of Information Technology, Government of Pakistan. The information, data, comments, and views detailed herein may not necessarily reflect the endorsements of views of the National ICT R&D Fund.

9. REFERENCES

[1] J. Greensmith and U. Aickelin, "Dendritic Cells for SYN Scan Detection", *ACM GECCO*, pp 49-56, 2007.

[2] J. Greensmith, U. Aickelin and J. Twycross, "Articulation and Clarification of the Dendritic Cell Algorithm", *ICARIS*, LNCS, Portugal, 2006.

[3] M. Zubair Shafiq, M. Farooq, S. Ali Khayam, "A Comparative Study of Fuzzy Inference Systems, Neural Networks and Adaptive Neuro Fuzzy Inference Systems for Portscan Detection", *EvoCOMNET*, LNCS, 2008.

[4] S. Forrest, A.S. Perelson, L. Allen, R. Cherukuri, "Self-nonself discrimination in a computer", *IEEE Symposium on Research in Security and Privacy*, IEEE Computer Society Press, 1994.

[5] F. Gonzalez, D. Dasgupta, L.F. Nino, "A randomized real-valued negative selection algorithm", *ICARIS*, LNCS, UK, 2003.

[6] T. Raschke, "The New Security Challenge: Endpoints," IDC/F-Secure, August 2005.

[7] "Symantec Internet Security Threat Report XI – Trends for July – December 07," March 2007.

[8] T. Stibor, J. Timmis and C. Eckert, "On the Use of Hyperspheres in Artificial Immune Systems as Antibody Recognition Regions", *ICARIS*, LNCS, Protugal, 2006.

[9] T. Stibor, J. Timmis, and C. Eckert, "A Comparative Study of Real-Valued Negative Selection to Statistical Anomaly Detection Techniques ", *ICARIS*, LNCS, 2005.

[10] T. Stibor, P. Mohr, J. Timmis and C. Eckert, "Is Negative Selection Appropriate for Anomaly Detection?", *ACM GECCO*, USA, 2005.

[11] J. Twycross and M. M. Williamson, "Implementing and testing a virus throttle," *Usenix Security Symposium*, 2003.

[12] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," *ACM SIGCOMM*, 2005.

[13] Y. Gu, A. McCullum, and D. Towsley, "Detecting anomalies in network traffic using maximum entropy estimation," *ACM/Usenix IMC*, 2005.

[14] Z. Ji, D. Dasgupta, "Real-valued negative selection algorithm with variable-sized detectors", *ACM GECCO*, LNCS, USA, 2004.

[15] Z. Ji, D. Dasgupta, "Applicability Issues of the Real-Valued Negative Selection Algorithms", *ACM GECCO*, July 2006.

[16] Symantec Security Response, `http://securityresponse.symantec.com/avcenter`

[17] C. Shannon and D. Moore, "The spread of the Witty worm," *IEEE Security & Privacy*, 2(4), pp. 46-50, 2004.

[18] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machine", 2001. Available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`

[19] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley-Interscience, June 1991.

[20] Jang, J-S. R.: "ANFIS: Adaptive-Network-Based Fuzzy Inference System". *IEEE Transaction on System, Man and Cybernetics* 23, 1993.

[21] MATLAB, The Mathworks Inc., `http://www.mathworks.com`

[22] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, 1998.

[23] T. Fawcett, "ROC Graphs: Notes and Practical Considerations for Researchers", TR (HPL-2003-4), HP Labs, USA.