Proceedings of the 2005 IEEE
Workshop on Information Assurance and Security
United States Military Academy, West Point, NY

A Machine Learning Framework for Network Anomaly Detection using SVM and GA

Taeshik Shon[1], Yongdae Kim, Cheolwon Lee[2], and Jongsub Moon[1], Member, IEEE

Abstract* – *In today's world of computer security, internet attacks such as Dos/DDos, worms, and spyware continue to evolve as detection techniques improve. It is not easy, however, to distinguish such new attacks using only knowledge of pre-existing attacks. In this paper we concentrate on machine learning techniques for detecting attacks from internet anomalies. Our machine learning framework consists of two major components: Genetic Algorithm (GA) for feature selection and Support Vector Machine (SVM) for packet classification. By experiment we also demonstrate that our proposed framework out performs currently employed real-world NIDS.

Index terms – Intrusion Detection, Network Security, Anomaly Detection, Machine Learning

I. INTRODUCTION

The internet is a crucial aspect of daily life around the world. Businesses, in particular, use the internet as an important aspect of their business model. In addition to using internet applications, such as the web and email, to generate revenue and communicate with customers, they often also store important and proprietary information on machines that are accessible through the internet. While this makes businesses operate more efficiently, it also makes them extremely vulnerable to attacks, both to steal data and to obstruct the operations of the business. In addition to the incentives for attackers, the internet often provides a virtually risk-free environment for attackers involved in malicious activities. A high degree of anonymity, lackluster interest among law enforcement, and absent or easily bypassed attack prevention schemes all factor into this lawless frontier. Thus a multitude of systems have been designed or proposed to thwart internet-based attacks. Many of the most commonly used systems today are based on attack "signatures", putative unique patterns or conditions which indicate an attack. However there are many drawbacks to signature-based Network Intrusion Detection Systems (NIDS). The first problem is that these systems themselves become a single point of failure. If the deployed system is disabled, either surreptitiously or conspicuously, it often gives the attacker the time needed to compromise other systems and possibly gain a foothold in the network. At other times NIDS may provide a false sense of security when they are not properly configured or, more importantly, maintained. Also, research has been done which shows that if the signatures are not sufficiently robust in describing the attack conditions then simple modifications can be made which will allow the attack to succeed. In addition to robustness of the signature, signature-based intrusion detection systems rely on humans to create, test, and deploy the signatures. Thus, it may take hours or days to generate a new signature for an attack, which can be too long when dealing with rapid attacks, such as are often seen in worm propagation. Some effort has been put into automatic signature generation, which does not require human intervention, but these systems are not yet ready for wide scale deployment.

In order to offer a human-independent solution to the above-mentioned problem, anomaly detection systems based on machine learning, data mining or statistical algorithms have been proposed. These systems use a "normal behavior" model for detecting unexpected behavior. There are two categories of these approaches□ supervised methods, which make use of pre-existing knowledge, and unsupervised methods, which do not. Several efforts to design anomaly detection algorithms using supervised methods are described in [1-3]. The research of Debra Anderson at SRI [1] and João B. D. Cabrera [2] deals with statistical methods for intrusion detection. Wenke Lee's research [3] focuses on theoretical measures for anomaly detection. In contrast, unsupervised schemes can make appropriate labels for a given dataset automatically. Anomaly detection methods with unsupervised features are explained in [4-6]. MINDS [4] is based on data mining and data clustering methods. The researches of Eleazar Eskin [5] are used to detect anomalous attacks independent of pre-existing knowledge. Stuart Staniford [6] is the author of SPADE for anomaly-based port scan detection in Snort. SPADE utilizes a statistical anomaly detection method with Bayesian probability. Even when good anomaly detection methods are used, the problems of high false alarm rates, difficulty in finding proper features, and high performance requirements still arise. Therefore, if it were possible to mix the advantages of both learning schemes in machine learning methods according to their characteristics in our problem domain, the combined approach could be used as an efficient means for detecting anomalous attacks.

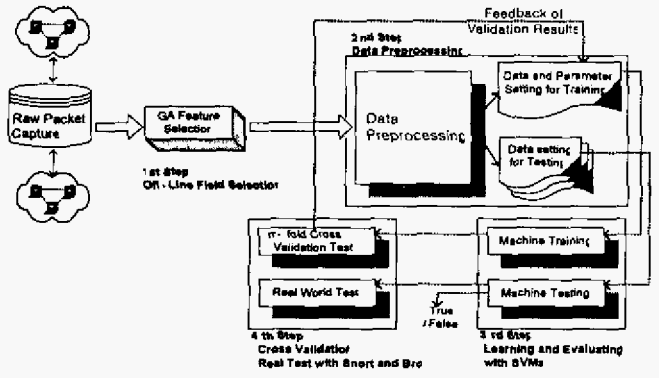
[1]*Center for Information Security Technologies, Korea University,* [2]*National Security Research Institute, Korea*

0-7803-9290-6/05/$20.00 ©2005 IEEE.

II. PROPOSED FRAMEWORK

In this paper, we focus on a Machine Learning Model using a modified Support Vector Machine (SVM) that combines the benefits of supervised and unsupervised learning. Moreover, we provide a preliminary feature selection process using □A to select more appropriate packet fields. Figure 1 illustrates the overall structure of our machine learning framework based on our Enhanced SVM approach. This framework consists of four components as follows□1.feature selection via □A; 2.data preprocessing; 3.data analysis via our modified SVM; and 4.testing and comparison with real-world products. We are confident that this new framework will offer a robust and powerful tool to the anomaly detection arsenal.

Figure 1. Overall Framework



The first component includes field selection using □A. We use □A to choose more appropriate fields from a packet. This offers better performance for packet preprocessing and increases detection rate. The algorithm selects optimized packet fields through the natural evolutionary process. Appropriate fields are first selected off-line and then this knowledge is used for analyzing filtered packets in real-time.

The second component processes data to refine filtered packets for high correction performance. In preprocessing this data, packets that passed the first component are preprocessed to enter our framework as SVM learning inputs.

The third component includes our machine learning approach. We make a model with SVM using two kinds of machine learning methods□ soft margin SVM (supervised method) one-class SVM (unsupervised method). We propose an Enhanced SVM approach to provide the high performance of soft margin SVM and the novelty detection capability of one-class SVM. Therefore, our Enhanced SVM incorporates both supervised and unsupervised features.

SVM is already known to the best algorithm for binary classification [7-9]. It has been successfully applied to a number of pattern recognition applications [10]. Recently, it has also been applied to information security for intrusion detection [11-13]. However, the most significant reason for choosing SVM is that it can be used for either supervised or unsupervised learning. Supervised SVM has relatively fast processing and high correction performance when compared to existing artificial neural networks and unsupervised SVM. However, one serious disadvantage in supervised SVM is that it requires labeled information for learning. Moreover, in all SVM learning, it is difficult to deal with consecutive variations of learning inputs without additional preprocessing. Therefore, in our machine learning framework, we propose an Enhanced SVM approach which incorporates the high performance of supervised SVM and unlabeled capability of unsupervised SVM.

The final component is to prove our approach both experimentally and in terms of current real world solutions. Experimental analysis of our proposed framework considers packet filtering and field selection results, comparison among SVM approaches, and m-fold cross validation. Real world tests compare our framework with well known NIDS such as Snort and Bro. These results verify that our proposed model met our expectations by out-performing current real-world and experimental security solutions.

The rest of this paper is organized as follows□In section III, □A techniques are introduced for providing better performance of our approach. In section IV, we present our major machine learning approach including two existing supervised and unsupervised SVMs. In section V and VI, experimental methods are explained with data preprocessing, data description and parameter settings. In the last section, we conclude with a summary and discussion of future work.

III. AN APPROACH OF □ENETIC AL□ORITHM

□A is a model used to mimic the behavior of evolutionary processes in nature [14-15]. It is known to be an ideal technique for finding solutions to optimization problems. □A uses three operators to produce the next generation from the current□reproduction, crossover, and mutation. Reproduction determines which individuals are chosen for crossover and how many offspring each selected individual produces. The selection uses a probabilistic survival of the fittest mechanism based on a problem-specific evaluation of the individuals. The crossover then generates new chromosomes within the population by exchanging randomly selected segments from within existing chromosomes. Finally, the mutation allows (rarely) the random mutation of existing chromosomes so that new chromosomes may contain parts not found in any existing chromosomes. This whole process is repeated probabilistically, moving from generation to generation, with the expectation that at the end, we are able to choose an individual which closely matches our desired conditions. When the process terminates, the best

chromosome selected from among the final generation is the solution. To apply this "process of evolution" to our problem domain, we have to define the following three components□ individual gene presentation and initialization, evaluation function modeling, and a specific function of genetic operators and their parameters.

For the first operation, we transform TCP/IP packets into binary gene strings. We convert each TCP and IP header fields into a one bit binary gene value, '0' or '1'. '1' means that the corresponding field exists and '0' means it does not. The initial population consists of a set of randomly generated 24-bit strings including 13 bits for IP fields and 11 bits for TCP fields. The total number of individuals in the population should be carefully considered. If the population size is too small, all gene chromosomes soon converge to the same gene string thus making it impossible for the genetic model to generate new individuals. In contrast, if the population size is too large, the model spends too much time calculating gene strings, negatively affecting the overall effectiveness of the method.

Table I. TCP/IP anomaly and communication score

Index	Name of Coefficients	Anomaly Score*	Communication Score**
01	a_{01} (Version)	0	S
02	a_{02} (Header length)	0	De
03	a_{03} (Type of Service)	0	S
04	a_{04} (Total length)	0	De
05	a_{05} (identification)	2	Dy
06	a_{06} (Flags)	5	Dy
07	a_{07} (Fragment offset)	5	Dy
08	a_{08} (Time to live)	1	Dy
09	a_{09} (Protocol)	1	S
10	a_{10} (Header checksum)	0	De
11	a_{11} (Source address)	2	S
12	a_{12} (Destination address)	1	S
13	a_{13} (Options)	1	S
14	a_{14} (Source Port)	1	S
15	a_{15} (Destination Port)	1	S
16	a_{16} (Sequence Number)	2	Dy
17	a_{17} (Ack Number)	2	Dy
18	a_{18} (Offset)	1	Dy
19	a_{19} (Reserved)	1	S
20	a_{20} (Flags)	2	Dy
21	a_{21} (Window)	0	S
22	a_{22} (Checksum)	0	De
23	a_{23} (Urgent Pointer)	1	S
24	a_{24} (Options)	1	S

* By anomaly analysis in [16-17]. ** S□Static, De□Dependent, Dy□Dynamic,

Addressing the second process, we create a fitness function for evaluating individuals. The fitness function consists of an object function $f(x)$ and its transformation function $g(f(X))$.

$$F(X) = g(f(X)) \tag{1}$$

In equation (1), the object function's values are converted into a measure of relative fitness by fitness function $F(X)$ with transformation function $g(x)$. To create our own objective function, we use the anomaly and communication scores shown in Table I.

The anomaly score refers to MIT Lincoln Lab datasets, covert channels, and other anomaly attacks [16-17]. The scores increase in proportion to the frequency of a field being used for anomaly attacks. Communication scores are divided into three kinds of scores in accordance with their importance during communication. 'S' fields have static values, "De" field values are dependent on connection status, and field values for "Dy" can change dynamically. We can derive a polynomial equation which has the above-mentioned considerations as coefficients. As the combination of an anomaly score and a communication score, the coefficients of this derived polynomial equation have the property of being weighted sums. Our objective function $f(X)$ consists of two polynomial functions, $A(X)$ and $N(X)$, shown in (2).

$$\begin{aligned} f(X) &= A(X) + N(X) \\ &= A(X_k(x_i)) + N(X_k(x_i)), \end{aligned} \tag{2}$$

$A(X)$ is our anomaly scoring function and $N(X)$ is our communication scoring function. Variable X is a population, $X_k(x_i)$ is a set of all individuals, and k is the total population. x_i is an individual with 24 attributes. To prevent equation (3) from generating too many features, a bias term μ is used as follows□

$$\begin{aligned} f'(X_k(x_i)) &= f(X_k(x_i)) - \mu \\ &= A(X_k(x_i)) + N(X_k(x_i)) - \mu \end{aligned} \tag{3}$$

Here, μ is the bias term of the new objective function $f'(X_k(x_i))$ whose boundary is $0 < \mu < Max(f(X_k))$. In case of $A(X_k(x_i))$, we can derive the proper equation as follows□

$$\begin{aligned} A(X) &= A(X_k(x_i)) \\ &= A(x_i + \ldots + x_2 + x_1) \\ &= a_i x_i + \ldots + a_2 x_2 + a_1 x_1, \qquad i, j = \{1, \ldots, 24\} \end{aligned} \tag{4}$$

where A=$\{ a_j \ldots a_2, a_1 \}$ is a set of coefficients in the polynomial equation and each coefficient represents an anomaly score. From equation (4), we use the bias term to satisfy the condition (5). Thus, we can choose a reasonable number of features without over-fitting and we can derive the new anomaly scoring function (6) with the bias term μ_A as follows□

$$\begin{aligned} A(X) &= a_j x_i + \ldots + a_2 x_2 + a_1 x_1 \\ &< Max(A(X)) \end{aligned} \tag{5}$$

$$\begin{aligned} &A'(X) = (a_j x_i + \ldots + a_2 x_2 + a_1 x_1) - \mu_A, \\ &0 < \mu_A < Max(A(X)), 0 < A'(X) < Max(A(X)) \end{aligned} \tag{6}$$

For $N(X_k(x_i))$, we develop an appropriate function having the same derivation as equation (6).

$$\begin{aligned} N(X) &= N(X_k(x_i)), \qquad \alpha=1, \beta=2, \gamma=3, \quad i, j = \{1, \ldots 24\} \\ &= N(x_i + \ldots + x_2 + x_1) \\ &= \alpha(x_1 + x_3 + x_9 + x_{11} + x_{12} + x_{13} + x_{14} + x_{15} + x_{19} + x_{21} + x_{23} + x_{24}) + \\ &\quad \beta(x_2 + x_4 + x_{10} + x_{22}) + \gamma(x_5 + x_6 + x_7 + x_8 + x_{16} + x_{17} + x_{18} + x_{20}), \end{aligned} \tag{7}$$

where N is a set of communication scores, and the coefficients α, β, γ mean static(S), dependent(De) and dynamic(Dy) respectively. In equation (7), we give the bias term by the same method as (5) and (6) as follows□

$$N(X) = \alpha(x_\alpha) + \beta(x_\beta) + \gamma(x_\gamma) < Max(N(X)) \quad (8)$$

$$N'(X) = \alpha(x_\alpha) + \beta(x_\beta) + \gamma(x_\gamma) - \mu_N,\ 0 < \mu_n < Max(N(X))$$
$$\therefore\ 0 < N'(X) < Max(N(X)) \quad (9)$$

where $x_\alpha, x_\beta, x_\gamma$ are a set of elements with the coefficients α, β, γ respectively. From equation (6) and (9), we can derive our entire objective equation (10) as follows□

$$\begin{aligned} f'(X_k(x_i)) &= A'(X) + N'(X) \\ &= (a_j x_i + \ldots + a_2 x_2 + a_1 x_1) - \mu_A + \alpha(x_\alpha) + \beta(x_\beta) + \gamma(x_\gamma) - \mu_N \\ &= (a_j x_i + \ldots + a_2 x_2 + a_1 x_1) + \alpha(x_\alpha) + \beta(x_\beta) + \gamma(x_\gamma) - (\mu_A + \mu_N) \\ &= (a_j x_i + \ldots + a_2 x_2 + a_1 x_1) + \alpha(x_\alpha) + \beta(x_\beta) + \gamma(x_\gamma) - \mu_c \\ &\therefore\ 0 < f'(X_k(x_i)) < Max(f(X_k(x_i)) \end{aligned} \quad (10)$$

The relative fitness is calculated using the proposed objective function (10), and converted into the normalized fitness function $F(x_k)$ using a rank-based transformation function. This rank-based transformation overcomes the scaling problems of the proportional fitness assignment. The reproductive range is limited so that no small group of individuals generates an excessive number of offspring. The ranking method introduces a uniform scaling across the population.

The last component for genetic modeling is to define a specific function of genetic operators and their related parameters. We will decide the related parameters in Section VI.

IV. An Enhanced SVM Approach

A. Existing SVMs : Soft Margin and One-Class SVM

SVM is basically used as a supervised learning method. A supervised SVM approach, soft margin SVM, employs slack variables and penalty functions in order to decrease misclassified data and solve non-separable problems [13]. In the case of non-separable problems, forcing zero training error will lead to poor generalization. To take into account the fact that some data points may be misclassified we introduce soft margin SVM using a vector of slack variables $\Xi = (\xi_1, \ldots, \xi_l)^T$ that measure the degree to which the constraints are violated (11).

$$\underset{w,b,\Xi}{\text{Minimize}}\ \Phi(w,b,\Xi) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i^k$$
$$\text{subject to}\ y_i(w^T\phi(x_i) + b) \geq 1 - \xi_i,\quad \xi_i \geq 0,\quad i = 1,\ldots,l \quad (11)$$

where C is a regularization parameter that controls the trade-off between maximizing the margin and minimizing the training error. If C is too small, insufficient stress is placed on fitting the training data. If C is too large, the algorithm will over-fit the dataset. In practice, a typical SVM approach such as soft margin SVM performed better than other machine learning methods. In case of an intrusion detection application, supervised machine learning approaches based on SVM were superior to intrusion detection approaches using artificial neural networks. Therefore, the high classification capability and processing performance of the soft margin SVM approach make it useful tool for anomaly detection. However, because soft margin SVM is a supervised learning approach, it requires that the given dataset be labeled.

On the other hand, single class learning for classifying outliers can be used as an unsupervised SVM [14]. This SVM method does not require pre-existing knowledge for classification. One-class SVM identifies outliers amongst positive examples and uses them as negative examples. In anomaly detection, if we consider anomalies as outliers, the one-class SVM approach can be applied to classify anomalous packets as outliers. As was done in the first paper about one-class SVM, we assume that outliers exist near the origin of a two-dimensional axis mapped from high dimensional space. Then in order to separate the dataset from the origin we need to solve the following quadratic programming problem (12)□

$$\underset{w,b,\Xi}{\text{Minimize}}\quad \Phi(w,b,\Xi) = \frac{1}{2}\|w\|^2 + \frac{1}{vl}\sum_{i=1}^{l}\xi_i^k - \rho$$
$$\text{subject to}\quad y_i(w^T\phi(x_i)) \geq \rho - \xi_i,\ \xi_i \geq 0,\ i = 1,\ldots,l \quad (12)$$

where v is a parameter that controls the trade-off between maximizing the distance from the origin and containing most of the data in the region related by the hyperplane, and corresponds to the ratio of outliers in the training set. Then the decision function $f(x) = \text{sgn}((w \cdot \Phi(x) + b) - \rho)$ will be positive for most examples x_i contained in the training set. In practice, even though one-class SVM has the capability of outlier detection, this approach is more sensitive to a given dataset than other machine learning schemes. It means that deciding on an appropriate hyperplane for classifying outliers is more difficult than in a supervised SVM approach.

B. Our Enhanced SVM Approach

In our problem domain, most of the internet traffic has normal features, and the amount of anomalous traffic is relatively small. In other words, we can see that the number of outliers we hope to detect is extremely small in comparison with normal traffic. Therefore, the single classifier of one-class SVM does not always need to try to maximize the distance from the origin for classifying outliers. Moreover if our SVM approach can be run in soft margin SVM, the proposed SVM approach will show a higher detection rate and processing performance than an unsupervised SVM approach such as one-class SVM.

In one-class SVM learning (12), important parameters for deciding the hyperplane are $\|w\|$, ρ, $\frac{1}{vl}\sum_{i=1}^{l}\xi_i^k$. Each parameter has the following meaning□ $\|w\|$ has to be decreased and ρ has to be increased in order to obtain maximum margin between the origin and hyperplane. In case of $\frac{1}{vl}\sum_{i=1}^{l}\xi_i^k$, this parameter is related to the violation of outlier and has to be decreased. Moreover, in soft margin SVM learning (11), the bias term basically is related to the distance between the origin and hyperplane. If the bias term is decreased, the hyperplane moves closer to the origin like a one-class SVM classifier. If the bias term b of (11) is deleted then we derive equation (13) as follows□

$$\underset{w,b,\Xi}{\text{Minimize}} \quad \Phi(w,b,\Xi) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i^k$$
$$\text{subject to} \quad y_i(w^T\phi(x_i)) \geq 1-\xi_i, \quad \xi_i \geq 0, i = 1,\ldots, l \qquad (13)$$

Therefore, by carefully adjusting the parameters of one-class SVM and soft margin SVM, we can derive an Enhanced SVM that offers the unlabeled classification capability of one-class SVM as well as the high detection performance of supervised soft margin SVM.
If we compare (13) with (12),

$$soft\text{-}SVM\ without\ a\ bias \quad \cong \quad one\text{-}class\ SVM$$
$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i^k \quad \cong \quad \frac{1}{2}\|w\|^2 + \frac{1}{vl}\sum_{i=1}^{l}\xi_i^k - \rho \qquad (14)$$
$$y_i(w^T\phi(x_i)) \geq 1-\xi_i \quad \cong \quad y_i(w^T\phi(x_i)) \geq \rho-\xi,$$
$$0 < v < 1,\ 1 < l,\ 0 \leq \rho \qquad (15)$$

In minimization condition (14), $C\sum_{i=1}^{l}\xi_i^k$ of soft margin SVM is the trade-off value used to adjust the training error in order to obtain the maximum margin between the origin and the hyperplane. $\frac{1}{vl}\sum_{i=1}^{l}\xi_i^k$ of one-class SVM also measures the degree to which outliers are in violation. Thus, by manipulating C in soft margin SVM we can approximate both terms to be equal. In subject condition (15), the ρ of one-class SVM is a parameter used to obtain the maximum margin between the origin and hyperplane. However, we do not need to worry about maximizing the ρ value because anomalous data in our domain can be classified by the hyperplane near to the origin. We can regard the value of ρ as a very small number like '1'. With all of these parameter approximations, we derive an Enhanced SVM in following equation (16).

$$Minimize \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i^k - \rho, \qquad C \cong \frac{1}{vl}$$
$$Subject\ to \quad y_i(w^T\phi(x_i)) \geq \rho-\xi, \quad 0 \leq \rho \cong 1 \qquad (16)$$

The *minimization* condition and *subject* condition of (16) have to satisfy the approximation conditions of $C \cong \frac{1}{vl}$ and $\rho \cong 1$, respectively. Therefore, in our proposed SVM approach, we can expect the unlabeled learning feature of one-class SVM and the relatively low false alarm and high correction rate of soft margin SVM.

V. EXPERIMENT METHODS

Our experiments use the IDS evaluation dataset from MIT Lincoln Lab [17] and known SVM toolkits [18-19]. We describe the MIT Lincoln Lab dataset and SVM setup parameters and kernel functions.

A. Data Description

The 1999 DARPA IDS dataset was collected at MIT Lincoln Lab to evaluate intrusion detection systems, and contained a wide variety of intrusions simulated in a military network environment. All the network traffic, including the entire payload of each packet, was recorded in tcpdump format and provided for evaluation. The data consists of three weeks of training data and two weeks of test data. For this dataset, we use attack-free training data for normal behavior modeling and attack data is used for testing and the construction of the anomaly scores in Table 1. Real network data is used in comparison tests with real NIDS. Moreover, our attack datasets include our own generated attacks such as covert channels, malformed packets, and some Denial of Service attacks. The simulated attacks include one of following five categories and they have DARPA attacks and generated attacks
In order to make the dataset more realistic, we organized many of the attacks so that the resulting data set consisted of 1 to 1.5% attacks and 98.5 to 99% normal traffic. For the supervised learning algorithm, the learning dataset had the characteristics of the dataset described above. This dataset contained 100,000 normal data packets and 1,000 to 1,500 abnormal packets for training and evaluating. For the unsupervised learning methods such as Enhanced SVM and one-class SVM, the dataset contained 100,000 normal packets for training and 1,000 to 1,500 various kinds of packets for testing. The training dataset contained only normal traffic because these methods feature unlabeled learning ability. For testing, the combined dataset containing both normal packets and abnormal packets was used. In our SVM experiments, we used SVMlight [18] for a supervised SVM and Libsvm for an unsupervised SVM [19].

B. Parameter Setup

SVM includes a variety of kernel functions and their parameters. In the case of soft margin SVM, we have to

decide a regularization parameter C. The kernel function transforms a given set of vectors to a possibly higher dimensional space for linear separation. For the parameter selection of our SVM learning, we referred to the experimental results of SVM which include a range of basic SVM parameters, various kernel functions, and their performance arguments. In our experiments, we set the parameters as follows□C between 0.9 and 10, d in the polynomial kernel at 1, σ in a radial basis kernel at 0.0001, and κ *and* θ in a sigmoid kernel at 0.00001, respectively. The SVM kernel functions we considered were linear, polynomial, radial basis kernels, and sigmoid.

VI. EXPERIMENT RESULTS

A. Field Selection using GA

In this experiment, we made preliminary tests using the typical genetic parameter values mentioned in the literature [15] in order to find reasonable genetic parameters. Table II describes the four-time preliminary test results. In this result, the final fitness value refers to the last resultant value calculated by our proposed objective equation.

Table II. Preliminary Test Parameters of □A

	No. of Population	Reproduction Rate(Pr)	Crossover Rate(Pc)	Mutation Rate(Pm)	Fitness Value
Case #1	100	0.100	0.600	0.001	11.72
Case #2	100	0.900	0.900	0.300	14.76
Case #3	100	0.900	0.600	0.100	25.98
Case #4	100	0.600	0.500	0.001	19.12

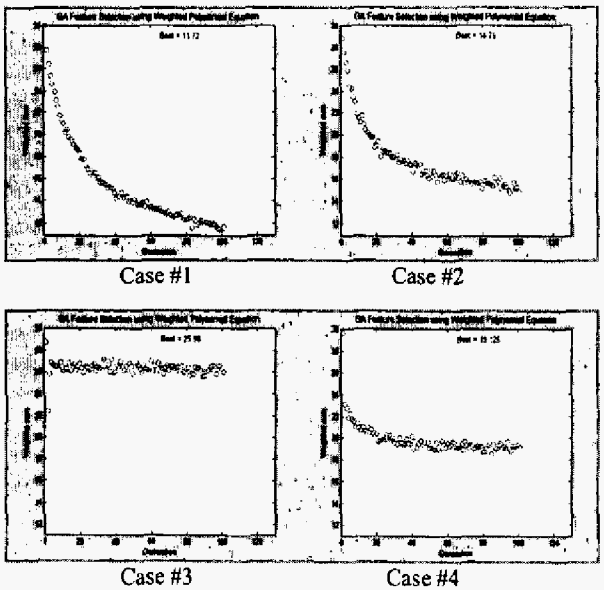


Fig. 2. Evolutionary Process according to preliminary test

Figure 2 shows four graphs of □A feature selection with the fitness function of (10) according to the parameters of Table II. In Case #1 and Case #2, the resultant graphs seem to have rapidly converging values because of a reproduction rate that is too low, and a too high crossover and mutation rate. In Case #3, the graph seems to be constant because of a high reproduction rate. The fourth and final graph, Case #4, seems to be converging to the appropriate values. The detailed experimental results of Case #4 are described in Table III.

Table III. □A Field Selection Results of Preliminary Test #4

□enerati on units	Selected Fields (Total Number)	CR (%)	FP (%)	FN (%)	PT (msec)
01-15	2,5,6,7,8,9,10,11,12,13,14,15,16,17,19,21,22,23,24 (19)	96.68	1.79	7.00	2.27
16-30	2,5,6,7,8,9,10,11,12,16,17,20,21,23,24 (15)	95.00	0.17	16.66	1.90
31-45	2,5,6,7,8,9,10,11,12,16,17,20,21,23,24 (15)	95.00	0.17	16.66	1.90
46-60	1,2,5,6,7,8,9,10,11,12,13,14,17,19,20,22,23,24 (18)	95.12	0.00	16.66	1.84
61-75	2,5,6,7,8,9,10,11,12,13,14,15,16,17,19,20,21 (17)	73.17	0.00	91.60	0.30
76-90	2,5,6,7,8,9,10,11,12,13,14,15,16,17,19,20,21 (17)	73.17	0.00	91.60	0.30
91-100	3,5,6,7,9,12,13,16,17,18,19,21,22,23,24 (15)	97.56	0.00	8.33	1.74

*CR□Correction Rate, FP□False Positive, FN□False Negative, PT□Processing Time (SVM learning time with 2000 packets)

Although we found the appropriate □A condition for our problem domain using the preliminary tests, we tried to select the optimal settings from among all the options. Through the experiment using soft margin SVM learning, we knew the final generations were well-optimized. □enerations 91-100 showed the best correction rate and relatively fast processing time. When comparing generations 16-30 with generations 46-60, it is interesting to note that fewer fields do not always guarantee faster processing because the processing time is also dependant on the field values.

B. Comparison among the Three SVMs and Cross Validation Tests

In this resultant analysis, the three SVMs were tested as follows□soft margin SVM as a supervised method, one-class SVM as an unsupervised method, and our proposed Enhanced SVM. The results are summarized in table IV. Each SVM approach was tested with four kinds of different SVM kernel functions. The high performance of soft margin SVM is not surprising since it is a well-developed supervised method. Also, the supervised capability of soft margin SVM is not suitable for detecting novel attacks. In the case of one-class SVM, the RBF kernel provided the best performance (94.65%); however, the false positive rate was high. Moreover, we could not see the results of the sigmoid kernel experiment because the sigmoid kernel of one-class SVM was overfitted. The overfitting of kernel functions also appeared in the Enhanced SVM experiment. It seems that Enhanced SVM's sensitivity to feature mapping using a kernel function is related to its using two kinds of machine learning methods. In conclusion, our Enhanced SVM experiment with sigmoid kernel performed the best,

producing a correction performance similar to soft margin SVM and a lower false positive rate than one-class SVM.

Table IV. The overall experiment results of SVMs

	Kernels	Correction Rate (%)	False Positive Rate (%)	False Negative Rate (%)
soft margin SVM	Inner product	90.13	10.55	4.36
	Polynomial	91.10	5.00	10.45
	RBF	98.65	2.55	11.09
	Sigmoid	95.03	3.90	12.73
one-class SVM	Inner Product	53.41	48.00	36.00
	Polynomial	54.06	45.00	46.00
	RBF	94.65	20.45	44.00
	Sigmoid	-	-	-
Enhanced SVM	Sigmoid	87.74	10.20	27.27

We evaluated our approaches using an m-fold cross validation test. It is the standard technique used to estimate a method's performance over unseen data. A cross validation test was performed using a 3-fold cross validation method on 3,000 normal packets divided among 3 subsets, and the holdout method is repeated 3 times. Each time we ran a test, one of the 3 subsets was used as the training set and all subsets were combined together to form a test set. The results are illustrated in Table V and showed that the method varied according to the training set.

Table V. 3-fold cross validation results

Training	Test	Correction Rate (%)	False Positive (%)
Validation Set #1	Validation Set #1	79.40	20.60
	Validation Set #2	82.50	17.50
	Validation Set #3	89.80	10.20
	Average	83.90	16.10
Validation Set #2	Validation Set #1	85.40	14.60
	Validation Set #2	88.30	12.70
	Validation Set #3	92.40	7.60
	Average	88.70	11.63
Validation Set #3	Validation Set #1	9.20	90.80
	Validation Set #2	17.00	83.00
	Validation Set #3	13.70	86.30
	Average	13.30	86.70

Even though all validation sets were attack-free datasets from the MIT Lincoln Lab dataset, there were many differences between Validation Sets #1, #2 and Validation Set #3. As a matter of fact, this test depended closely on how well the collected learning sets consisted of a wide variety of normal and abnormal features. The training with Validation Set #2 showed the best correction rate across all three of the cross validation tests and a low false positive rate. In other words, Validation Set #2 has well-organized normal features and was optimal for training.

C. Comparison with Real NIDS

Finally, we compared our proposed approach with real NIDS using real data. Real network traffic was captured from our Institute, and treated as a black box with respect to attacks contained within. Snort and Bro are some of the most well-known and best performance NIDS. Snort and Bro did not require training datasets because they are signature based NIDS. However, for our Enhanced SVM, we used Validation Set #2 as the training set. In this experiment, four kinds of test sets were used as follows. Test #1 was an attack-free MIT Lincoln Lab dataset which was not used in training. Test #2 included ten sorts of attacks from our generated attacks. Test #3 was comprised of nine kinds of attack data from DARPA IDS Test. Test #4 was real data collected by our Institute.

Table VI. Real world test results

	Test sets	Correction Rate (%)	False Positive Rate (%)	False Negative Rate (%)
Enhanced SVM	Test#1 - Normal	92.40	7.60	-
	Test#2 - Attacks	68.70	-	31.30
	Test#3 - Attacks	74.47	-	25.53
	Test#4 - Real	99.99	0.01	-
Snort	Test#1 - Normal	94.77	5.23	-
	Test#2 - Attacks	80.00	-	20.00
	Test#3 - Attacks	88.88	-	11.12
	Test#4 - Real	93.62	6.38	-
Bro	Test#1 - Normal	96.56	3.44	-
	Test#2 - Attacks	70.00	-	30.00
	Test#3 - Attacks	77.78		22.22
	Test#4 - Real	97.29	2.71	-

Table VI shows, as expected, both signature-based NIDS detected known attacks of Test #2 and Test #3 well. However, for Bro, the correction rate may be variable because Bro does not provide a rule configuration file like Snort does for off-line analysis. This means Bro users can construct customized Bro rule sets with individual Bro rule files. When we used the rules selected from Bro programmer Vern Paxson, it worked just as well for specific attacks as Snort did.

As for Test #1 and the attack-free data, Snort and Bro generated many warning messages such as tiny fragment, UDP bad checksum, and so on. Our framework also worked well because our SVM classifier was trained by attack-free data set of MIT Lincoln Lab albeit a different subset. In the case of Test #2 and #3s' known attacks, the two NIDS performed better than our Enhanced SVM. It is interesting to note that their correction rates were proportionate to the number of rules in their rule-set files. This performance is to be expected, when we consider that Snort and Bro are state-of-the-art NIDS based on signatures. Notwithstanding, our Enhanced SVM method performed rather well considering that Snort and Bro had knowledge of preexisting attack signatures and our model did not. Furthermore our Enhanced SVM method would seem more adept at classifying novel attacks.

In the case of Test #4's real and unknown data, even though analysis of the input data was not done to completely ensure these results, Snort and Bro showed a generalized correction rate regardless of the characteristics of the given dataset. Comparing the results of Test #4 with Test #1, correction rates were as follows. In Test #1 and #4, Snort's rates were 94.77% and 93.62%, and Bro's were 96.56% and 97.28%. This means that the construction of real data in Test #4 was similar to the attack-free data of Test #1. In other words, the real data set of Test #4 seems not to have contained serious attacks.

The results of Test #4 using our approach showed that there were virtually no attacks. After analyzing this result we determined that the classifier in our Enhanced SVM was not sufficiently refined. This is in part due to the fact that knowledge for classification was gathered from the MIT Lincoln Lab Dataset. Thus, when we apply our framework to a real environment, it will be necessary to use a more realistic profiling method for defining what constitutes normal data in a given environment.

VII. CONCLUSION

The overall goal of our Enhanced SVM approach was to provide a general framework for detection and classification of novel attacks in network traffic. Four preliminary tests using □A for optimized field selection indicated to us early on that this method had a relatively fast processing time and a better correction rate. As for the SVM component, we proposed our own SVM classification approach and data preprocessing for producing better SVM inputs. Specifically, we designed an Enhanced SVM incorporating both a high-performance supervised scheme and an unsupervised scheme that operates without the use of labels. Moreover we used m-fold cross validation methods and real-world experiments to verify our results and prove the relative effectiveness of our framework over existing NIDS. Snort and Bro are recognized as well developed signature-based detection systems however our Enhanced SVM approach combining two machine learning techniques offers a new and promising method for detecting novel attacks. As expected from our SVM approach, our model, like NIDS, exhibited a low false-positive rate but without the use of labels that signature-based NIDS requires. We assert that our SVM framework is a significant contribution to anomaly detection and justifies deployment in real world environments.

Future work will involve making a profile of normal packets using appropriate methods such as data mining and data clustering. If we better define normal profiles and extract their formalized preventative packet groups, we expect that our approach will be able to classify novel attacks better. After profiling normal packets, we can also apply this framework to real-world TCP/IP traffic and consider a more realistic packet association based on statistical traffic distribution.

VIII. REFERENCES

[1] D. Anderson, et al., "Detecting Unusual Program Behavior Using the Statistical Component of the Next-□eneration Intrusion Detection", SRI-CSL-95-06, Computer Science Laboratory, SRI International, CA. May, 1995

[2] João B. D. Cabrera, et al., "Statistical Traffic Modeling For Network Intrusion Detection", Proc of the 8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, San Francisco, CA, 2000.

[3] W. Lee, D. Xiang, "Information-Theoretic Measures for Anomaly Detection", IEEE Symposium on Security and Privacy, 2001

[4] Ertoz, L. *et al.*, "The MINDS - Minnesota Intrusion Detection System", Next □eneration Data Mining, MIT Press, 2004

[5] E. Eskin, A. Arnold, M. Prerau, Leonid Portnoy and Salvatore Stolfo, "A □eometric Framework for Unsupervised Anomaly Detection□Detecting Intrusions in Unlabeled Data", Data Mining Security Applications, Kluwer, 2002

[6] S. Staniford, J. Hoagland, J. McAlerney, "Practical Automated Detection of Stealthy Portscans", Journal of Computer Security, vol. 10, No. 1-2, 105-136, 2002.

[7] Vapnik V., "the Nature of Statistical Learning Theory", Springer-Verlag, NewYork, 1995.

[8] Cortes. C., et al., "Support-vector network", Machine Learning 20, 273–297. 1995

[9] B. Schölkopf, et al., "Estimating the support of a high-dimensional distribution," Neural Computation, 13, 2001, 1443-1471.

[10] H. Byun and S.W. Lee, "A Survey on Pattern Recognition Applications of Support Vector Machines," International Journal of Pattern Recognition and Artificial Intelligence, Vol. 17, No. 3, 2003, pp. 459-486.

[11] K.A. Heller, K.M. Svore, A. Keromytis, S.J. Stolfo, "One Class Support Vector Machines for Detecting Anomalous Windows Registry Accesses", In the proceedings of the workshop on Data Mining for Computer Security

[12] Wenjie Hu, Yihua Liao, and V. Rao Vemuri, "Robust Support Vector Machines for Anamoly Detection in Computer Security", International Conference on Machine Learning, Los Angeles, CA, July 2003

[13] Andrew H. Sung, et al., "Identifying Important Features for Intrusion Detection Using Support Vector Machines and Neural Networks", SAINT 2003□209-217

[14] J. Holland, "Adaptation in Natural and Artificial Systems", The Univ of Michigan Press, Ann Arbor, 1995.

[15] Mitchell, M., "An Introduction to □enetic Algorithms", MIT Press, Cambridge, MA.

[16] K. Ahsan and D. Kundur, "Practical Data Hiding in TCP/IP", Proc. Workshop on Multimedia Security at ACM Multimedia, 2002, French Riviera, pp. 7

[17] Lincoln Laboratory, MIT, "DARPA Intrusion Detection Evaluation", http□//www.ll.mit.edu/IST/ideval/index.html

[18] Joachmims T, mySVM - a Support Vector Machine, University Dortmund, 2002

[19] Chih-Chung Chang, LIBSVM □a library for support vector machines., 2004