International Journal of Advanced Computer and Mathematical Sciences
ISSN 2230-9624. Vol 3, Issue 3, 2012, pp 379-388
http://bipublication.com

USING CRFS VERY EFFECTIVELY IN IMPROVING THE ATTACK DETECTION RATE AND DECREASING THE FAR FOR INTRUSION DETECTION

P. Hanmanth Reddy and A. Nagesh

Dept of CSE, Mahatma Gandhi Institute of Technology, Hyderabad, A.P, India

[Received-09/09/2012, Accepted-28/09/2012]

ABSTRACT:

Intrusion Detection Systems have inherent limitations which lead to false positives and false negatives; we propose that combining signature and anomaly based IDSs should be examined. This paper contrasts signature and anomaly-based IDSs, and critiques some proposals about hybrid IDSs with signature and heuristic capabilities, which is designed to offer superior pattern analysis and anomaly detection by reducing false positive rates and administrator intervention. In this project, we address these two issues of Accuracy and Efficiency using Conditional Random Fields and Layered Approach.

Index Terms— Intrusion detection, Layered Approach, Conditional Random Fields, network security, Decision trees, naïve Bayes.

I. INTRODUCTION

Intrusion Detection Systems (IDSs) monitor, and detect malicious activities against a computer or a set of computers. With the advance of cloud computing and evidence that traditional host based protection is not a pancea to security they are becoming a necessity [48]. However, they have somelimitations which may lead to false positives and false negatives [49]. For instance, if a new service or application is installed, a heuristic-based IDS may define this behaviour as a false positive whereas if malicious code has been slightly modified, it can bypass the signature-based IDS detection as a false negative [50]. Compromise the entire network security. INTRUSION detection as defined by the SysAdmin, Audit, Networking, and Security (SANS) Institute is the art of detecting inappropriate, inaccurate, or anomalous activity [6]. Today, intrusion detection is one of the high priority and challenging tasks for network administrators and security professionals. More sophisticated security tools mean that the attackers come up with newer and more advanced penetration methods to defeat the installed security systems [4] and [24]. Thus, there is a need to safeguard the networks from known vulnerabilities and at the same time take steps to detect new and unseen, but possible, system abuses by developing more reliable and efficient intrusion detection systems. Any intrusion detection system has some inherent requirements. Its prime purpose is to detect as many attacks as possible with minimum number of false alarms, i.e., the system must be accurate in detecting attacks. However, an accurate system that cannot handle large amount of network traffic and is slow in decision making will not fulfill the purpose of an intrusion detection system. We desire a system that detects most of the attacks, gives very few false alarms, copes with large amount of data, and is fast enough to make real-time decisions. Intrusion detection started in around 1980s after the influential paper from Anderson [10]. Intrusion detection systems are classified as network based, host based, or application based depending on their mode of deployment and data used for analysis [11]. Additionally, intrusion detection systems can also be classified as signature based or anomaly based depending upon the attack detection method. The signature-based systems are trained by extracting specific patterns (or signatures) from previously known attacks while the anomaly-based systems learn from the normal data collected when there is no anomalous activity [11].

There is growing interest in designing generic user data flooding or simplified multicasting forwarding services for use within mobile ad hoc networks (MANETs)[QVL00, CLOV02, WC02, NTCS99.] Also, previous work developing MANET unicast routing protocols often involves the application of a packet flooding service within the protocol control plane to support basic functionality (e.g., router discovery, proactive link state dissemination). Some novel technical work has been done to optimize the flooding process for specific routing protocol designs and requirements [SRS02, HXG02, QVL00, PGC00, OTL04]. [51]

Another approach for detecting intrusions is to consider both the normal and the known anomalous patterns for training a system and then performing classification on the test data. Such a system incorporates the advantages of both the signature-based and the anomaly-based systems and is known as the Hybrid System. Hybrid systems can be very efficient, subject to the classification method used, and can also be used to label unseen or new instances as they assign one of the known classes to every test instance. This is possible because during training the system learns features from all the classes. The only concern with the hybrid method is the availability of labeled data. However, data requirement is also a concern for the signature- and the anomaly-based systems as they require completely anomalous and attack- free data, respectively, which are not easy to ensure.

II .CONDITIONAL RANDOM FIELDS FOR INTRUSION DETECTION

Conditional models are probabilistic systems that are used to model the conditional distribution over a set of random variables. Such models have been extensively used in the natural language processing tasks. Conditional models offer a better framework as they do not make any unwarranted assumptions on the observations and can be used to model rich overlapping features among the visible observations. Maxent classifiers [37], maximum entropy Markov models [34], and CRFs [29] are such conditional models. The advantage of CRFs is that they are undirected and are, thus, free from the Label Bias and the Observation Bias [27]. The simplest conditional classifier is the Maxent classifier based upon maximum entropy classification, which estimates the conditional distribution of every class given the observations [37]. The training data is used to constrain this conditional distribution while ensuring maximum entropy and hence maximum uniformity. We now give a brief description of the CRFs, which is motivated from the work in [29].

Let X be the random variable over data sequence to be labeled and Y the corresponding label sequence. In addition, let G = (V, E) be a graph such that Y = (Y_v)v€ (V)$^{'}$ so that Y is indexed by the vertices of G. Then, (X,Y) is a CRF, when conditioned on X, the random variables Y_v obey the Markov property with respect to the graph: $p(Y_v/X, Y_w, w \neq v)=p(y_v / X, Y_w, w \sim v)$, where w ~ v means that w and v are neighbors in G, i.e., a CRF is a random field globally conditioned on X. For a simple sequence (or chain) modeling, as in our case, the joint distribution over the label sequence Y given X has the following form:

$$p_\theta(y|x) \propto \exp\left(\sum_{e\in E,k} \lambda_k f_k(e, y|_e, x) + \sum_{v\in V,k} \mu_k g_k(v, y|_v, x)\right), \quad (1)$$

Fig1: The Joint distribution [47]

Where x is the data sequence, y is a label sequence, and y|s is the set of components of y associated with the vertices or edges in sub-graph S. In addition, the features f_k and g_k are assumed to be given and fixed. For example, a Boolean edge feature f_k might be true if the observation X_i is “protocol=tcp,” tag Y_{i-1} is “normal,” and tag Y_i is “normal.” Similarly, a Boolean vertex feature g_k might be true if the observation X_i is “service=ftp” and tag Y_i is “attack.” Further, the parameter estimation problem is to find the parameters 0= $(\lambda1, \lambda2,...; \mu1, \mu2,....)$ from the training data D= $(x^i, y^i)^N{}_{i=1}$with the empirical distribution $p^{\sim}(x, y)$ [29].

CRFs are undirected graphical models used for sequence tagging. The prime difference between CRF and other graphical models such as the HMM is that the HMM, being generative, models the joint distribution p(y, x), whereas the CRF are discriminative models and directly model the conditional distribution p(y | x), which is the distribution of interest for the task of classification and sequence labeling. Similar to HMM, the naive Bayes is also generative and models the joint distribution. Modeling the joint distribution has two disadvantages. First, it is not the distribution of interest, since the observations are completely visible and the interest is in finding the correct class for the observations, which is the conditional distribution p(y | x) Second, inferring the conditional probability p(y|x) from the modeled joint distribution, using the Bayes rule, requires the marginal distribution p(x) To estimate this marginal distribution is difficult since the amount of training data is often limited and the observation x contains highly dependent features that are difficult to model and therefore strong independence assumptions are made among the features of an observation. This results in reduced accuracy [40]. CRFs, however, predict the label sequence y given the observation sequence x [15]. This allows them to model arbitrary relationship among different features in an observation x. CRFs also avoid the observation bias and the label bias problem, which are present in other discriminative models, such as the maximum entropy Markov models. This is because the maximum entropy Markov models have a per-state exponential model for the conditional probabilities of the next state given the current state and the observation, whereas the CRFs have a single exponential model for the joint probability of the entire sequence of labels given the observation sequence [29].

The task of intrusion detection can be compared to many problems in machine learning, natural language processing, and bioinformatics. The CRFs have proven to be very successful in such tasks, as they do not make any unwarranted assumptions about the data. Hence, we explore the suitability of CRFs for intrusion detection.

III .LAYERED APPROACH FOR INTRUSION DETECTION

The Layer-based Intrusion Detection System (LIDS) is discussed in detail. The LIDS draws its motivation from what we call as the Airport Security model, where a number of security checks are performed one after the other in a sequence. Similar to this model, the LIDS represents a sequential Layered Approach and is based on ensuring availability, confidentiality, and integrity of data and (or) services over a network. Fig. 2 gives a generic representation of the framework.

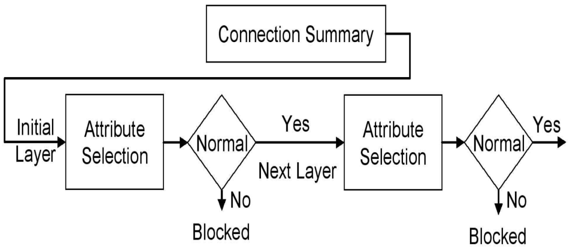


Fig: 2 Layered representation [47]

The goal of using a layered model is to reduce computation and the overall time required to detect anomalous events. The time required to detect an intrusive event is significant and can be reduced by eliminating the communication overhead among different layers. This can be achieved by making the layers autonomous and self-sufficient to block an attack without the need of a central decision-maker. Every layer in the LIDS framework is trained separately and then deployed sequentially. We define four layers that correspond to the four attack groups mentioned in the data set. They are Probe layer, DoS layer, R2L layer, and U2R layer. Each layer is then separately trained with a small set of relevant features. Feature selection is significant for Layered Approach and discussed in the next section. In order to make the layers independent, some features may be present in more than one layer. The layers essentially act as filters that block any anomalous connection, thereby eliminating the need of further processing at subsequent layers enabling quick response to intrusion. The effect of such a sequence of layers is that the anomalous events are identified and blocked as soon as they are detected.

Our second goal is to improve the speed of operation of the system. Hence, we implement the LIDS and select a small set of features for every layer rather than using all the 41 features. This results in significant performance improvement during both the training and the testing of the system. In many situations, there is a trade-off between efficiency and accuracy of the system and there can be various avenues to improve system performance. Methods such as naive Bayes assume independence among the observed data. This certainly increases system efficiency, but it may severely affect the accuracy. To balance this trade-off, we use the CRFs that are more accurate, though expensive, but we implement the Layered Approach to improve overall system performance. The performance of our proposed system, Layered CRFs, is comparable to that of the decision trees and the naive Bayes, and our system has higher attack detection accuracy.

IV .INTEGRATING LAYERED APPROACH WITH CONDITIONAL RANDOM FIELD

We discussed two main requirements for an intrusion detection system: accuracy of detection and efficiency in operation. As discussed in the above, the CRFs can be effective in improving the attack detection accuracy by reducing the number of false alarms, while the Layered Approach can be implemented to improve the overall system efficiency. Hence, a natural choice is to integrate them to build a single system that is accurate in detecting attacks and efficient in operation. Given the data, we first select four layers corresponding to the four attack groups (Probe, DoS, R2L, and U2R) and perform feature selection for each layer.

IP DESIGN APPROACH

In addition to researching various algorithms and approaches for MANET flooding, our goal was to provide a working prototype compatible with existing and emerging IP network protocol frameworks. The implementation approach taken enables the protocol to work as a simplified MANET multicast routing mechanism within a MANET routing area. At present the IP packets are not encapsulated in any additional or specialized IP header, so multicast routing is performed on native IP multicast application packets. One important routing design difference between MANET interfaces and many wired network interfaces is that forwarding out the same interface a packet arrived on is a normal allowed operation. It is important to note that this operation is often disallowed in wired multicast routing designs. Because of this feature, a fairly common requirement in MANET packet flooding is some form of duplicate packet detection. This generally requires some form of packet sequence identification. Our simplified MANET multicast routing implementation is composed of three parts: A sequence id generator and marker to

be used when and if necessary, a duplicate detection module, and a basic multicast packet forwarding module. The sequence generator is responsible for marking each packet with a monotonically increasing unique identification number when existing IP kernel methods are not sufficient or are not predictable. The duplicate detection mechanism is used to remove and detect duplicate packets from both entering the interface forwarding process and from being delivered to upper layer applications. The forwarding module performs basic multicast IP packet forwarding out a particular MANET interface as appropriate.[51]

Feature Selection

Ideally, we would like to perform feature selection automatically. However, as will be discussed later in Section 8, the methods for automatic feature selection were not found to be effective. In this section, we describe our approach for selecting features for every layer and why some features were chosen over others. In our system, every layer is separately trained to detect a single type of attack category.

We observe that the attack groups are different in their impact, and hence, it becomes necessary to treat them differently. Hence, we select features for each layer based upon the type of attacks that the layer is trained to detect.

Probe Layer

The probe attacks are aimed at acquiring information about the target network from a source that is often external to the network. Hence, basic connection level features such as the “duration of connection” and “source bytes” are significant while features like “number of files creations” and “number of files accessed” are not expected to provide information for detecting probes.

DoS Layer

The DoS attacks are meant to force the target to stop the service(s) that is (are) provided by flooding it with illegitimate requests. Hence, for the DoS layer, traffic features such as the “percentage of connections having same destination host and same service” and packet level features such as the “source bytes” and “percentage of packets with errors” are significant. To detect DoS attacks, it may not be important to know whether a user is “logged in or not.”

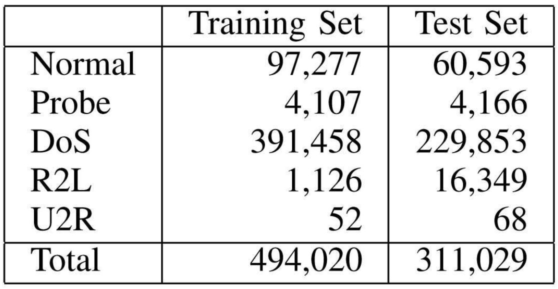
R2L Layer

The R2L attacks are one of the most difficult to detect as they involve the network level and the host level features. We therefore selected both the network level features such as the “duration of connection” and “service requested” and the host level features such as the “number of failed login attempts” among others for detecting R2L attacks.

U2R Layer

The U2R attacks involve the semantic details that are very difficult to capture at an early stage. Such attacks are often content based and target an application. Hence, for U2R attacks, we selected features such as “number of file creations” and “number of shell prompts invoked,” while we ignored features such as “protocol” and “source bytes.” We used domain knowledge together with the practical significance and the feasibility of each feature before selecting it for a particular layer. Thus, from the total 41 features, we selected only 5 features for Probe layer, 9 features for DoS layer, 14 features for R2L layer, and 8 features for U2R layer. Since each layer is independent of every other layer, the feature set for the layers is not disjoint. The selected features for all the four layers are presented. We then use the CRFs for attack detection as discussed. However, the difference is that we use only the selected features for each layer rather than using all the 41 features. We now give the algorithm for integrating CRFs with the Layered Approach.

Table 1 Data Set



	Training Set	Test Set
Normal	97,277	60,593
Probe	4,107	4,166
DoS	391,458	229,853
R2L	1,126	16,349
U2R	52	68
Total	494,020	311,029

Fig 3: Data Set [47]

Our final goal is to improve both the attack detection accuracy and the efficiency of the system. Hence, we integrate the CRFs and the Layered Approach to build a single system. We perform detailed experiments and show that our integrated system has dual advantage. First, as expected, the efficiency of the system increases significantly. Second, since we select significant features for each layer, the accuracy of the system further increases. This is because all the 41 features are not required for detecting attacks belonging to a particular attack group. Using more features than required can result in fitting irregularities in the data, which has a negative effect on the attack detection accuracy of the system.

Algorithm

Training

Step 1: Select the number of layers, n, for the complete system.

Step 2: Separately perform features selection for each layer.

Step 3: Train a separate model with CRFs for each layer using the features selected from Step2.

Step 4: Plug in the trained models sequentially such that only the connections labeled as normal are passed to the next layer.

Testing

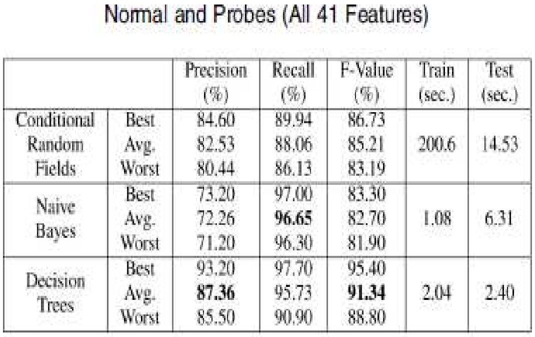
Step 5: For each (next) test instance perform Steps 6 through 9.

Step 6: Test the instance and label it either as attack or normal.

Step 7: If the instance is labeled as attack, block it and identify it as an attack represented by the layer name at which it is detected and go to Step 5. Else pass the sequence to the next layer.

Step 8: If the current layer is not the last layer in the system, test the instance and go to Step 7 Else go to Step 9.

Step 9: Test the instance and label it either as normal or as an attack. If the instance is labeled as an attack, block it and identify it as an attack corresponding to the layer name.



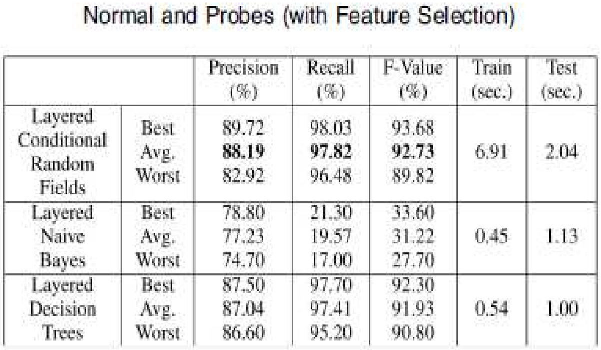
Normal and Probes (All 41 Features)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (sec.)
Conditional Random Fields	Best	84.60	89.94	86.73	200.6	14.53
	Avg.	82.53	88.06	85.21		
	Worst	80.44	86.13	83.19		
Naive Bayes	Best	73.20	97.00	83.30	1.08	6.31
	Avg.	72.26	**96.65**	82.70		
	Worst	71.20	96.30	81.90		
Decision Trees	Best	93.20	97.70	95.40	2.04	2.40
	Avg.	**87.36**	95.73	**91.34**		
	Worst	85.50	90.90	88.80		

Fig 4:Detecting Probe Attacks with All 41 Features [47]

We randomly select about 10,000 normal records and all the Probe records from the training data as the training data for detecting Probe attacks. We then use all the normal and Probe records from the test data for testing. Hence, we have 15,000 training instances and 64,759 test instances. Above table gives the results for the experiments.

Detecting Probe Attacks with Feature Selection

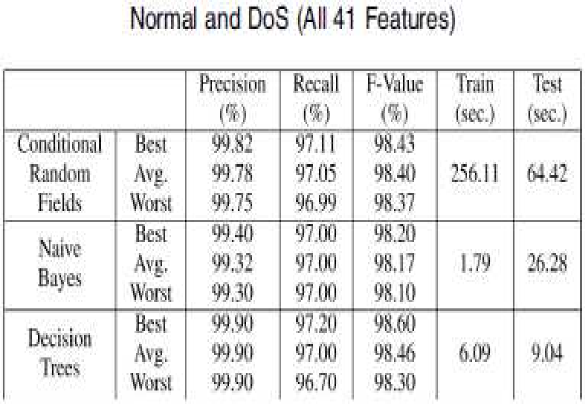


Normal and Probes (with Feature Selection)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (sec.)
Layered Conditional Random Fields	Best	89.72	98.03	93.68	6.91	2.04
	Avg.	**88.19**	**97.82**	**92.73**		
	Worst	82.92	96.48	89.82		
Layered Naive Bayes	Best	78.80	21.30	33.60	0.45	1.13
	Avg.	77.23	19.57	31.22		
	Worst	74.70	17.00	27.70		
Layered Decision Trees	Best	87.50	97.70	92.30	0.54	1.00
	Avg.	87.04	97.41	91.93		
	Worst	86.60	95.20	90.80		

Fig 5: Detecting Probe Attacks with Feature Selection[47]

We used the same set of instances for this experiment as used in the previous experiment. However, we perform feature selection for this experiment. Above Table gives the results for this experiment.

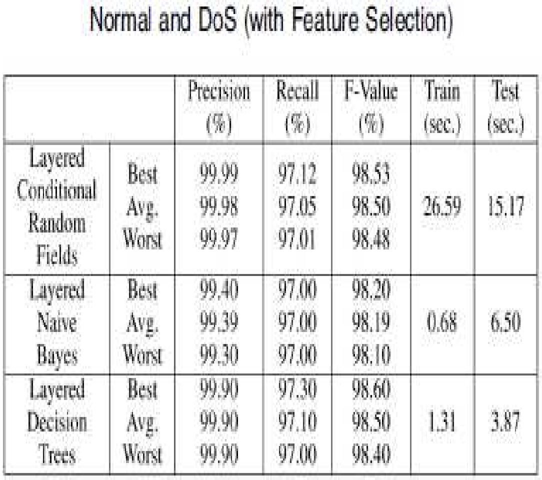


Normal and DoS (All 41 Features)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (sec.)
Conditional Random Fields	Best	99.82	97.11	98.43	256.11	64.42
	Avg.	99.78	97.05	98.40		
	Worst	99.75	96.99	98.37		
Naive Bayes	Best	99.40	97.00	98.20	1.79	26.28
	Avg.	99.32	97.00	98.17		
	Worst	99.30	97.00	98.10		
Decision Trees	Best	99.90	97.20	98.60	6.09	9.04
	Avg.	99.90	97.00	98.46		
	Worst	99.90	96.70	98.30		

Fig 6: Detecting DoS Attacks with All 41 Features[47]

We randomly select about 20,000 normal records and about 4,000 DoS records from the training data as the training data for detecting DoS attacks. We then use all the normal and DoS records from the test data for testing. Hence, we have 24,000 training instances and 290,446 test instances. Above table gives the results for the experiments.

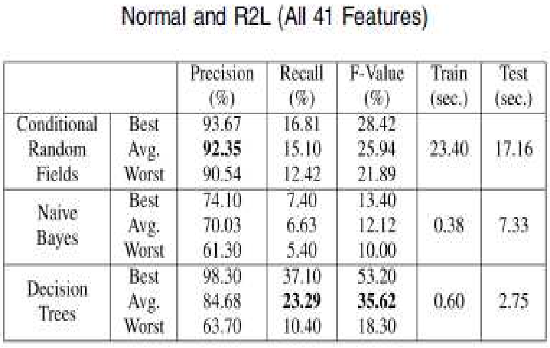


Normal and DoS (with Feature Selection)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (sec.)
Layered Conditional Random Fields	Best	99.99	97.12	98.53	26.59	15.17
	Avg.	99.98	97.05	98.50		
	Worst	99.97	97.01	98.48		
Layered Naive Bayes	Best	99.40	97.00	98.20	0.68	6.50
	Avg.	99.39	97.00	98.19		
	Worst	99.30	97.00	98.10		
Layered Decision Trees	Best	99.90	97.30	98.60	1.31	3.87
	Avg.	99.90	97.10	98.50		
	Worst	99.90	97.00	98.40		

Fig 7:Detecting DoS Attacks with Feature Selection[47]

We used the same data for this experiment as used in the previous experiment. However, we perform feature selection. Above table gives the results.

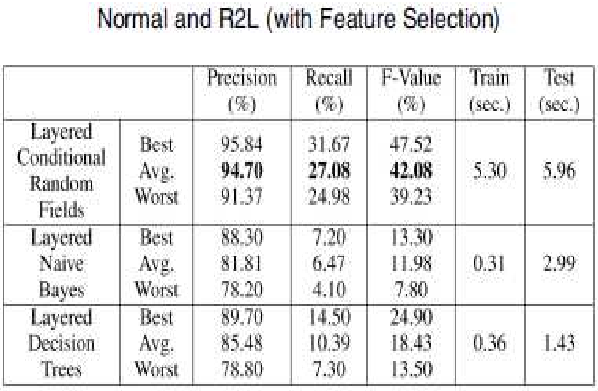


Normal and R2L (All 41 Features)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (sec.)
Conditional Random Fields	Best	93.67	16.81	28.42	23.40	17.16
	Avg.	**92.35**	15.10	25.94		
	Worst	90.54	12.42	21.89		
Naive Bayes	Best	74.10	7.40	13.40	0.38	7.33
	Avg.	70.03	6.63	12.12		
	Worst	61.30	5.40	10.00		
Decision Trees	Best	98.30	37.10	53.20	0.60	2.75
	Avg.	84.68	**23.29**	**35.62**		
	Worst	63.70	10.40	18.30		

Fig 8:Detecting R2L Attacks with All 41 Features[47]

We randomly select about 1,000 normal records and all the R2L records from the training data as the training data for detecting R2L attacks. We then use all the normal and R2L records from the test data for testing. Hence, we have 2,000 training instances and 76,942 test instances. Above table gives the results.



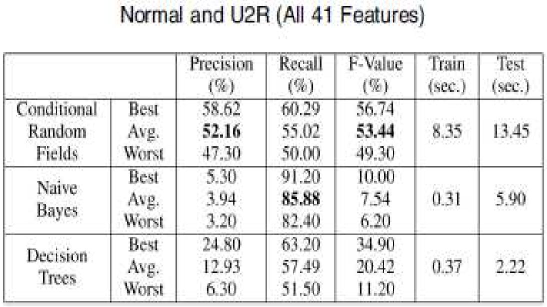
Normal and R2L (with Feature Selection)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (sec.)
Layered Conditional Random Fields	Best	95.84	31.67	47.52	5.30	5.96
	Avg.	**94.70**	**27.08**	**42.08**		
	Worst	91.37	24.98	39.23		
Layered Naive Bayes	Best	88.30	7.20	13.30	0.31	2.99
	Avg.	81.81	6.47	11.98		
	Worst	78.20	4.10	7.80		
Layered Decision Trees	Best	89.70	14.50	24.90	0.36	1.43
	Avg.	85.48	10.39	18.43		
	Worst	78.80	7.30	13.50		

Fig 9 Detecting R2L Attacks with Feature Selection[47]

Above table gives the results when we performed feature selection for detecting R2L attacks. We observe that the time taken to test all the 76,942 instances is only 5.96 seconds. Further, the Layered CRFs perform much better than the CRFs (an increase of about 60 percent), layered decision trees (an increase of about 125 percent), decision trees (an increase of about 17 percent), layered naive Bayes (an increase of about 250 percent), and naive Bayes (an increase of about 250 percent) and are the best choice for detecting the R2L attacks. The Layered CRFs take slightly more time, which is acceptable since we achieve much higher detection accuracy.

Detecting U2R Attacks with All 41 Features



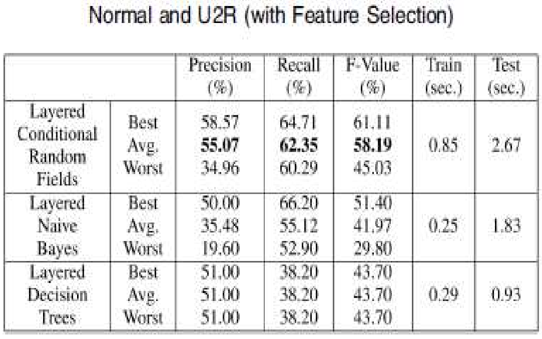
Normal and U2R (All 41 Features)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (sec.)
Conditional Random Fields	Best	58.62	60.29	56.74	8.35	13.45
	Avg.	**52.16**	55.02	**53.44**		
	Worst	47.30	50.00	49.30		
Naive Bayes	Best	5.30	91.20	10.00	0.31	5.90
	Avg.	3.94	**85.88**	7.54		
	Worst	3.20	82.40	6.20		
Decision Trees	Best	24.80	63.20	34.90	0.37	2.22
	Avg.	12.93	57.49	20.42		
	Worst	6.30	51.50	11.20		

Fig 10: Detecting U2R Attacks with All 41 Features[47]

We randomly select about 1,000 normal records and all the U2R records from the training data as the training data for detecting the User to Root attacks. We then use all the normal and U2R records from the test data for testing. Hence, we have 1,000 training instances and 60,661 test instances. Table gives the results.

Detecting U2R Attacks with Feature Selection



Normal and U2R (with Feature Selection)

		Precision (%)	Recall (%)	F-Value (%)	Train (sec.)	Test (sec.)
Layered Conditional Random Fields	Best	58.57	64.71	61.11	0.85	2.67
	Avg.	**55.07**	**62.35**	**58.19**		
	Worst	34.96	60.29	45.03		
Layered Naive Bayes	Best	50.00	66.20	51.40	0.25	1.83
	Avg.	35.48	55.12	41.97		
	Worst	19.60	52.90	29.80		
Layered Decision Trees	Best	51.00	38.20	43.70	0.29	0.93
	Avg.	51.00	38.20	43.70		
	Worst	51.00	38.20	43.70		

Fig 11:Detecting U2R Attacks with Feature Selection[47]

In this experiment, we used exactly the same set of instances as we used in the previous experiment. We also perform feature selection. This table gives the results for this experiment. We observe that the system takes only 2.67 seconds to label all the 60,661 test instances. The Layered CRFs are the best choice for detecting the U2R attacks and are far better than CRFs (an increase of about 8 percent), layered decision trees (an increase of about 30 percent), decision trees (an increase of about 184 percent), layered naive Bayes (an increase of about 38 percent), and naive Bayes (an increase of about 675 percent). We observe that the attack detection capability also increases for the decision trees and the naive Bayes.

V .CONCLUSIONS:

In this paper, we have addressed the dual problem of Accuracy and Efficiency for building robust and efficient intrusion detection systems. Our experimental results show that CRFs are very effective in improving the attack detection rate and decreasing the FAR. Having a low FAR is very important for any intrusion detection system. Further, feature selection and implementing the Layered Approach significantly reduce the time required to train and test the model. Even though we used a relational data set for our experiments, we showed that the sequence labeling methods such as the CRFs can be very effective in detecting attacks and they outperform other methods that are known to work well with the relational data. We compared our approach with some well-known methods and found that most of the present methods for intrusion detection fail to reliably detect R2L and U2R attacks, while our integrated system can effectively and efficiently detect such attacks giving an improvement of 34.5 percent for the R2L and 34.8 percent for the U2R attacks. We also discussed how our system is implemented in real life. Our system can help in identifying an attack once it is detected at a particular layer, which expedites the intrusion response mechanism, thus minimizing the impact of an attack. We showed that our system is robust to noise and performs better than any other compared system even when the training data is noisy. Finally, our system has the advantage that the number of layers can be increased or decreased depending upon the environment in which the system is deployed, giving flexibility to the network administrators.

REFERENCES

[1] Autonomous Agents for Intrusion Detection, http://www.cerias. purdue.edu/research/aafid/, 2010.

[2] CRF++: Yet Another CRF Toolkit, http://crfpp.sourceforge.net/, 2010.

[3] KDD Cup 1999 Intrusion Detection Data, http://kdd.ics.uci.edu/ databases/kddcup99/kddcup99.html, 2010.

[4] Overview of Attack Trends, http://www.cert.org/archive/pdf/ attack_trends.pdf, 2002.

[5] Probabilistic Agent Based Intrusion Detection, http://www.cse.sc. edu/research/isl/agentIDS.shtml, 2010.

[6] SANS Institute—Intrusion Detection FAQ, http://www.sans.org/ resources/idfaq/, 2010.

[7] T. Abraham, IDDM: Intrusion Detection Using Data Mining Techniques, http://www.dsto.defence./gov.au/publications/ 2345/DSTO-GD-0286.pdf, 2008.

[8] R. Agrawal, T. Imielinski, and A. Swami, “Mining Association Rules between Sets of Items in Large Databases,” Proc. ACM SIGMOD, vol. 22, no. 2, pp. 207-216, 1993.

[9] N.B. Amor, S. Benferhat, and Z. Elouedi, “Naive Bayes vs. Decision Trees in Intrusion Detection Systems,” Proc. ACM Symp. Applied Computing (SAC ’04), pp. 420-424, 2004.

[10] J.P. Anderson, Computer Security Threat Monitoring and Surveillance, http://csrc.nist.gov/publications/history/ande80.pdf, 2010.

[11] R. Bace and P. Mell, Intrusion Detection Systems, Computer Security Division, Information Technology Laboratory, Nat’l Inst. of Standards and Technology, 2001.

[12] D. Boughaci, H. Drias, A. Bendib, Y. Bouznit, and B. Benhamou, “Distributed Intrusion Detection Framework Based on Mobile Agents,” Proc. Int’l Conf. Dependability of Computer Systems (DepCoS-RELCOMEX ’06), pp. 248-255, 2006.

[13] Y. Bouzida and S. Gombault, “Eigenconnections to Intrusion Detection,” Security and Protection in Information Processing Systems, pp. 241-258, 2004.

[14] H. Debar, M. Becke, and D. Siboni, “A Neural Network Component for an Intrusion Detection System,” Proc. IEEE Symp. Research in Security and Privacy (RSP ’92), pp. 240-250, 1992.

[15] T.G. Dietterich, “Machine Learning for Sequential Data: A Review,” Proc. Joint IAPR Int’l Workshop Structural, Syntactic, and Statistical Pattern Recognition (SSPR/SPR ’02), LNCS 2396, pp. 15-30, 2002.

[16] P. Dokas, L. Ertoz, A. Lazarevic, J. Srivastava, and P.-N. Tan, “Data Mining for Network Intrusion Detection,” Proc. NSF Workshop Next Generation Data Mining (NGDM ’02), pp. 21-30, 2002.

[17] Y. Du, H. Wang, and Y. Pang, “A Hidden Markov Models-Based Anomaly Intrusion Detection Method,” Proc. Fifth World Congress on Intelligent Control and Automation (WCICA ’04), vol. 5, pp. 4348-4351, 2004.

[18] S. Dzeroski and B. Zenko, “Is Combining Classifiers Better than Selecting the Best One,” Proc. 19th Int’l Conf. Machine Learning (ICML ’02), pp. 123-129, 2002.

[19] L. Ertoz, A. Lazarevic, E. Eilertson, P.-N. Tan, P. Dokas, V. Kumar, and J. Srivastava, “Protecting against Cyber Threats in Networked Information Systems,” Proc. SPIE Battlespace Digitization and Network Centric Systems III, pp. 51-56, 2003.

[20] S. Forrest, S.A. Hofmeyr, A. Somayaji, and T.A. Longstaff, “A Sense of Self for Unix Processes,” Proc. IEEE Symp. Research in Security and Privacy (RSP ’96), pp. 120-128, 1996.

[21] Y. Gu, A. McCallum, and D. Towsley, “Detecting Anomalies in Network Traffic Using Maximum Entropy Estimation,” Proc. Internet Measurement Conf. (IMC ’05), pp. 345-350, USENIX Assoc., 2005.

[22] K.K. Gupta, B. Nath, and R. Kotagiri, “Network Security Framework,” Int’l J. Computer Science and Network Security, vol. 6, no. 7B, pp. 151-157, 2006.

[23] K.K. Gupta, B. Nath, and R. Kotagiri, “Conditional Random Fields for Intrusion Detection,” Proc. 21st Int’l Conf. Advanced Information Networking and Applications Workshops (AINAW ’07), pp. 203-208, 2007.

[24] K.K. Gupta, B. Nath, R. Kotagiri, and A. Kazi, “Attacking Confidentiality: An Agent Based Approach,” Proc. IEEE Int’l Conf. Intelligence and Security Informatics (ISI ’06), vol. 3975, pp. 285-296, 2006.

[25] C. Ji and S. Ma, “Combinations of Weak Classifiers,” IEEE Trans. Neural Networks, vol. 8, no. 1, pp. 32-42, 1997.

[26] D.S. Kim and J.S. Park, “Network-Based Intrusion Detection with Support Vector Machines,” Proc. Information Networking, Networking Technologies for Enhanced Internet Services Int’l Conf. (ICOIN ’03), pp. 747-756, 2003.

[27] D. Klein and C.D. Manning, “Conditional Structure versus Conditional Estimation in NLP Models,” Proc. ACL Conf. Empirical Methods in Natural Language Processing (EMNLP ’02), vol. 10, pp. 9-16, Assoc. for Computational Linguistics, 2002.

[28] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur, “Bayesian Event Classification for Intrusion Detection,” Proc. 19th Ann. Computer Security Applications Conf. (ACSAC ’03), pp. 14-23, 2003.

[29] J. Lafferty, A. McCallum, and F. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data,” Proc. 18th Int’l Conf. Machine Learning (ICML ’01), pp. 282-289, 2001.

[30] W. Lee and S. Stolfo, “Data Mining Approaches for Intrusion Detection,” Proc. Seventh USENIX Security Symp. (Security ’98), pp. 79-94, 1998.

[31] W. Lee, S. Stolfo, and K. Mok, “Mining Audit Data to Build Intrusion Detection Models,” Proc. Fourth Int’l Conf. Knowledge Discovery and Data Mining (KDD ’98), pp. 66-72, 1998.

[32] W. Lee, S. Stolfo, and K. Mok, “A Data Mining Framework for Building Intrusion Detection Model,” Proc. IEEE Symp. Security and Privacy (SP ’99), pp. 120-132, 1999.

[33] A. McCallum, “Efficiently Inducing Features of Conditional Random Fields,” Proc. 19th Ann. Conf. Uncertainty in Artificial Intelligence (UAI ’03), pp. 403-410, 2003.

[34] A. McCallum, D. Freitag, and F. Pereira, “Maximum Entropy Markov Models for Information Extraction and Segmentation,” Proc. 17th Int’l Conf. Machine Learning (ICML ’00), pp. 591-598, 2000.

[35] A.K. McCallum, MALLET: A Machine Learning for Language Toolkit, http://mallet.cs.umass.edu, 2010.

[36] L. Portnoy, E. Eskin, and S. Stolfo, “Intrusion Detection with Unlabeled Data Using Clustering,” Proc. ACM Workshop Data Mining Applied to Security (DMSA), 2001.

[37] A. Ratnaparkhi, “A Maximum Entropy Model for Part-of-Speech Tagging,” Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP ’96), pp. 133-142, Assoc. for Computational Linguistics, 1996.

[38] M. Sabhnani and G. Serpen, “Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context,” Proc. Int’l Conf. Machine Learning, Models, Technologies and Applications (MLMTA ’03), pp. 209-215, 2003.

[39] H. Shah, J. Undercoffer, and A. Joshi, “Fuzzy Clustering for Intrusion Detection,” Proc. 12th IEEE Int’l Conf. Fuzzy Systems (FUZZ-IEEE ’03), vol. 2, pp. 1274-1278, 2003.

[40] C. Sutton and A. McCallum, “An Introduction to Conditional Random Fields for Relational Learning,” Introduction to Statistical Relational Learning, 2006.

[41] E. Tombini, H. Debar, L. Me, and M. Ducasse, “A Serial Combination of Anomaly and Misuse IDSes Applied to HTTP Traffic,” Proc. 20th Ann. Computer Security Applications Conf. (ACSAC ’04), pp. 428-437, 2004.

[42] W. Wang, X.H. Guan, and X.L. Zhang, “Modeling Program Behaviors by Hidden Markov Models for Intrusion Detection,” Proc. Int’l Conf. Machine Learning and Cybernetics (ICMLC ’04), vol. 5, pp. 2830-2835, 2004.

[43] C. Warrender, S. Forrest, and B. Pearlmutter, “Detecting Intrusions Using System Calls: Alternative Data Models,” Proc. IEEE Symp. Security and Privacy (SP ’99), pp. 133-145, 1999.

[44] I.H. Witten and E. Frank, Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, 2005.

[45] Y.-S. Wu, B. Foo, Y. Mei, and S. Bagchi, “Collaborative Intrusion Detection System (CIDS): A Framework for Accurate and Efficient IDS,” Proc. 19th Ann. Computer Security Applications Conf. (ACSAC ’03), pp. 234-244, 2003.

[46] Z. Zhang, J. Li, C.N. Manikopoulos, J. Jorgenson, and J. Ucles, “HIDE: A Hierarchical Network Intrusion Detection System Using Statistical Preprocessing and Neural Network Classification,” Proc. IEEE Workshop Information Assurance and Security (IAW ’01), pp. 85-90, 2001.

[47] Layered Approach Using Conditional Random Fields for Intrusion Detection Kapil Kumar Gupta, Baikunth Nath, Senior Member, IEEE, and Ramamohanarao Kotagiri, Member, IEEE, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, VOL. 7, NO. 1, JANUARY-MARCH 2010

[48] D.J. Day and Z. Zhao, "Protecting Against Address Space Layout Randomization (ASLR) Compromises and Return-toLibc Attacks Using Network Intrusion Detection Systems.,"
International Journal of Automation and Computing, vol. 8, no. 4, pp. 472-483, Dec. 2011.

[49] W. R. Cheswick, S. M. Bellovin, and A.D. Rubin, "Intrusion Detection," in Firewalls and Internet Security: Repelling the Wily Hacker, 2nd ed. Boston: Addison-Wesley, 2003, pp. 279-283.

[50] Ryan Trost, "Intrusion Detection Systems," in Practical Intrusion Analysis: Prevention and Detection for the TwentyFirst Century, Karen Gettman, Ed. Boston, USA: AddisonWesley, 2010, ch. 3, pp. 53-85.

[51] Joseph P. Macker, Simplified Multicast Forwarding in Mobile Ad hoc Networks, http://cs.itd.nrl.navy.mil/pubs/docs/SMF_milcom04.pdf