Defence Science Journal, Vol. 65, No. 6, November 2015, pp. 466-471, DOI : 10.14429/dsj.65.8573

Evolution of a Hybrid Model for an Effective Perimeter Security Device

A.R.Vasudevan and S. Selvakumar*

Department of Computer Science and Engineering,
National Institute of Technology Tiruchirappalli - 620 015, India
**E-mail: ssk@nitt.edu*

ABSTRACT

Clustering and classification models, or hybrid models are the most widely used models that can handle the diverse nature of network intrusion detection systems (NIDS) dataset. Dirichlet process clustering technique is a non-parametric Bayesian mixture model that considers the data distribution of the dataset for the formation of distinct clusters. The number of clusters is not known a priori and it differs across different datasets. Determining the number of clusters based on the distribution of data instances can increase the performance of the model. Naive Bayes model, a supervised learning classification technique, maintains a better computational efficiency, by reducing the training time. In this paper, author proposes a hybrid model to exploit the positive aspect of proper clustering of data instances and the computational efficiency in building a NIDS. RIPPER algorithm is used to extract rules from the traffic description for updation of the rule database. Experiments were conducted in the KDD CUP'99 and SSENet-2011 datasets to study the performance of the proposed model. Also, a comparison of three hybrid methods with the proposed hybrid model was carried out. The results showed that the proposed hybrid model is superior in building a robust perimeter security device.

Keywords: Dirichlet process clustering, naive bayes classifier, RIPPER, hybrid model

1. INTRODUCTION

With the exponential growth of internet and applications, the network threats have increased very significantly. In order to protect the network and the systems from the complex and varied cyber threats[1], the need for an effective defensive mechanism is inevitable.

Network intrusion detection systems (NIDS) is a very important tool for network administrators in protecting the network from cyber threats. Deployable NIDS tools, such as Snort and Bro, maintain a database of signatures of those attack vectors that have already happened. The incoming traffic to a network is checked against these signatures sequentially for alerting the administrator in case of a threat. The efficiency of such NIDS is limited by the inability of NIDS to detect the attack accurately leading to false alarms and the difficulty to craft complex rules[2].

2. EVOLUTION OF HYBRID MODELS

2.1 Need for Hybrid Models

In NIDS domain the complexity of the underlying data distribution and the closeness of normal and attack vectors have rendered the single classifier system ineffective. Also, Wolpert's no-free-lunch[3] concept states that no single component/model can satisfy/solve all the problems. The individual components such as clustering and classification techniques address specific issues. On the other hand, the hybrid model exploits the strength of individual components in building a stringent hybrid classifier system.

2.2 Hybrid Classifiers

As in[4], the hybrid classifiers can be classified into three methods, viz., Cluster + single classifier methods, Cascaded hybrid methods, and Integrated-based hybrid methods.

The idea behind the hybrid classifiers is to improve the accuracy of the model and thereby increase the performance of the system[5]. Most of the hybrid models contain two sub components. The first component takes the dataset as input and outputs intermediate results whereas the second method produces the final results. The usage of clustering as a first stage in the hybrid method decimates the bigger dataset into finer reduced clusters. This enables the second line of classifiers in building a model for the individual clusters and aggregates the detection accuracy. In the cascaded hybrid methods, two or more classifiers are considered for an effective output result. Integrated based hybrid models deploy two different techniques where the first method aims at parameter tuning and the second method aims at prediction.

3. MOTIVATION

The usage of data mining techniques has immensely helped in analysing the effectiveness and efficiency of NIDS datasets. The deployment of single model/classifier based approach has become less effective in the changing scenarios. Hence, the need for a combined strategy of a hybrid modelling of NIDS is inevitable. Such a hybrid model leads to increased performance efficiency. The proposed hybrid model is constructed based on the individual strengths of clustering, classification, and rule

Received 13 March 2015, revised 15 September 2015, online published 10 November 2015

generation techniques there by countering the challenge in containing the network threats.

In the proposed hybrid model a balance is maintained between the output of the data mining/machine learning techniques and its application in an operational environment[6]. The problems associated with such techniques can be mainly attributed to the following:

- Excessive amount of training time required in building the mathematical model
- Mapping of metrics such as detection rate, false alarms, etc., in a real live network

The proposed hybrid model towards building a NIDS addresses the above listed practical issues with the dataset based approach.

4. PROPOSED HYBRID MODEL

The positioning of the proposed Hybrid NIDS model is as shown in Fig. 1. The proposed model can be connected to the SPAN port of a network switch in promiscuous mode. The network traffic is converted into a dataset by mapping the network traffic into features using Tstat tool[7] and labelled by the procedure[8].

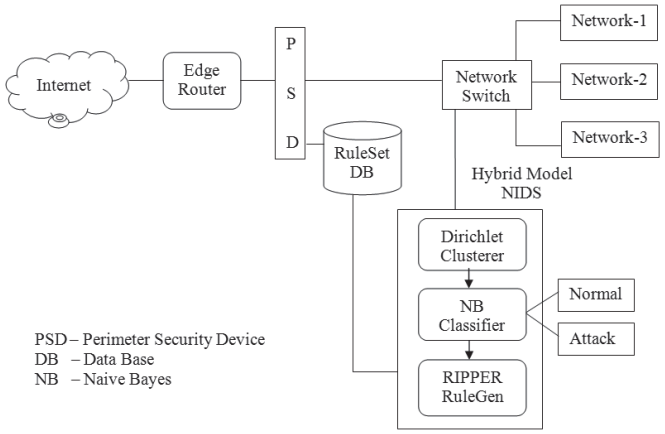

Figure 1. Architecture of proposed hybrid model of NIDS.

Authors proposed work consists of three phases, viz., Dirichlet process clustering, naive bayes classifier, and rule/signature generator. The functionality of the individual models/components in the proposed hybrid model is as follows:

- *Dirichlet Process Clustering*

DP clustering procedure is based on Bayesian mixture models[9]. The input dataset is segregated into different clusters based on the underlying data distribution. Hence, there is no user defined parameters to initialise the value of *k*, the number of clusters.

- *Naive Bayes Classifier*

The Naive Bayes (NB) classifier is a probabilistic model based classifier. It has competitively better performance and less training cost when compared to the contemporary classification models[10,11]. The main advantage of NB classifier is the ability to build a model very quickly and effectively, at network speed[12].

- *RIPPER based rule generation*

The metrics of evaluation such as detection rate, precision, recall, etc., prove the efficiency of the classifiers. In an operational network the rules/signatures coded in the security device prevent the network from malicious packets by checking the parameters in the packet(s) against the signatures. RIPPER generates a series of rules/signatures by which a particular type of attack vector can be recognised.

4.1 Algorithm for Hybrid Model

Figure 2 describes the algorithm for developing the proposed hybrid model of NIDS.

```
Input  : Labelled Dataset
Output : Detection Rate of Classifiers, Rules/
    Signatures for PSD

Hybrid_Model(Dataset)
{
    // Performs DP Clustering algorithm to segregate
    the dataset into different clusters
    // based on the input Model
    [Clus, ClusPoints, ClusCenter, ClusRadius] =
    DPClustering(Dataset, Model, IniClus, Iter)

    // Perform Naive Bayes classification
    for each cluster formed from DP clustering
    do
        Separate the data instances into pair-wise
    target-class (pwtc)
            - pwtc[1] = normal_probe
            - pwtc[2] = normal_priesc
            - pwtc[3] = normal flood
    end
    for each pair-wise target-class in the cluster
    do
        [DetectRate, Precision, Recall] =
    NBClasifier(pwtc[i])
    end
    // Perform RIPPER rule extraction algorithm
    for each pair-wise target-class in the cluster
    do
        [NumRules, Rules] = RuleGen(pwtc[i])
    end
}
```

Figure 2. Algorithm of the proposed hybrid model.

4.2 Dirichlet Process Clustering

DP clustering performs Bayesian non-parametric mixture modelling[13]. Non-parametric does not mean a parameter less model, but a model in which representations grow as more data are observed. A probabilistic mixture of a number of models is used to explain the observed data. Each data point is assumed to have come from one of the models in the mixture but not known from which model it came from. The algorithm for DP clustering is as shown in Fig. 3. Given the observed data and the prior distributions of model parameters, the algorithm assigns data instance to a model. The process is initialised

```
DPClustering(Dataset, Model, numIter, numClus)
{
    Compute DirichletState()
    for each interation
        build a new array of models
        update DirichletState()
        compute normalised vector of probabilities
    using pdfs
    end
    return clusterSamples
}
DirichletState()
{
    Compute or update parameters for each model
    return adjusted probability for sample x in the k^th
    model
}
```

Figure 3. Algorithm for Dirichlet process clustering.

by considering models at random from a prior distribution of models. Then each sample (data instance) is assigned to a model based on the degree of fit and probability. The DP clustering plays the crucial role of estimating the number of clusters[14]. The result of DP clustering are three, viz., number of samples with mixed probabilities, models, and assignment of points to models. These samples provide interesting information that are lacking in normal clustering algorithm. The number of models, having data points assigned, describe the number of clusters the dataset support. In[15], Incremental Infinite Generalised Dirichlet Mixture model with Feature Selection is applied on KDD CUP'99 and Kyoto datasets for detecting anomalies but does not discuss the number of clusters formed and the composition of target-class in each cluster.

4.3 Naive Bayes Classifier

The Naive Bayes classification model is one of the most popular models because of its simplicity and computation efficiency as well as its good performance on datasets. Such advantages are inherited from its conditional independence assumption property[16]. One of the leading examples of the application of Bayesian methods in IDS is the audit data analysis and mining (ADAM)[17]. Also, Naive Bayes has the advantage of having simple structure and efficient learning which make it fit for use in NIDS domain[18].

The algorithm for Naive Bayes classifier is as shown in Fig. 4. Naive Bayes model that assumes all variables are independent. Bayes classifier is based on the Bayesian rule which uses joint probability distribution model for decisions about uncertain variables. During the model build phase, the prior probabilities are calculated. The prediction of target-class label for a test instance is based on the maximum posterior probability. The test instance is classified to a target-class that has the highest posterior probability. In NIDS dataset, KDD CUP'99 has 41 features[19] and SSENet-2011 has 28 features.

4.4 RIPPER Rule Generation

RIPPER, an advanced version of incremental reduced

Input:
Each data instance, $X = (x_1, x_2, x_3, \ldots\ldots, x_n)$
// n is the number of features
Target-Class, $C = (C_1, C_2, \ldots\ldots C_m)$
// m target-classes
Dataset, $D = \langle X, C \rangle$
Output
To calculate the highest posterior probability, $p(C_i/X)$ such that $p(C_i/X) > p(C_j/X)$; for all $1 \leq j \leq m$ and $j \neq i$

NB_Classifier (Dataset D)
{
(1) Model Build Phase
- Calculate prior probability of target-classes, $p(C)$
- Calculate conditional probability of data instance X, given C_i

$$p(X/c_i) = \prod_{k=1}^{n} p(x_k/c_i)$$

(2) Test Phase
// To predict the target-class label for test instance X
- For each target-class C_i

Calculate $p(X/C_i) * P(C_i)$
- The data instance X belongs to the target-class which has the highest posterior probability

}

Figure 4. Naive Bayes algorithm.

error pruning (IREP) algorithm, increases the accuracy of ruleset by revising or replacing the individual rules[20]. RIPPER algorithm has two phases, viz., the building stage and the optimisation stage. The building stage can be further divided into growing phase and pruning phase. In the growing phase, the rule is grown incrementally by adding conditions to the rule until it is accurate. The phase continues for every attribute/feature based on the condition of highest information gain. In the next sub phase, pruning phase, each rule is pruned as long as the accuracy of the rule increases. In the optimisation stage, for each rule, R, for each target-class, the dataset is split afresh into growing and pruning sets. The data instances are removed that are not covered by the rules and are added to a newly split data. Growing and pruning phases are again applied on the newly formed dataset. The rules so generated are added to the resultant ruleset.

5. EXPERIMENTS AND RESULTS

The three different components of the hybrid model were applied sequentially on the two NIDS datasets, viz., SSENet-2011 and KDD CUP'99. The procedure involved the application of the Dirichlet Process clustering for segregating the dataset into different clusters based on the underlying distribution of data. In the second phase, on each cluster the NB classifier was applied on pair wise target-classes. For each pair wise target-class datasets, the metrics such as detection rate, precision, recall, ROC area, and execution time were measured. Also, in the last phase, the execution of RIPPER algorithm on the pair wise target-class subsets provided the rules in differentiating

the attack class from normal class. These rules may be added to the RuleSet DB for signature based detection. The number of clusters formed for each dataset is as shown in Table 1. The Table also indicates the target-class data in each cluster.

5.1 SSENet-2011

Table 1 shows the number of clusters formed by applying the DP clustering technique. The SSENet-2011 dataset was clustered into four different clusters. Table 1 also shows the distribution of data (target-classes) in the four clusters. For a dataset representing a real network it is very difficult to distinguish between attack vectors and normal instances. The NB classifier classifies the data instances, as shown in Table 2, in each cluster and builds the model in less time. The RIPPER rules generated for each pair wise target-classes can be identified from the last column of the Table. The signatures so generated are stored in RuleSet DB for future reference.

Table 1. DP clustering output on the NIDS datasets

Dataset	No. of Clusters	Data distribution	
		Cluster No.	Target-Classes found
SSENet-2011	4	Cluster-0	normal, probe, priesc, flood
		Cluster-1	normal, probe, priesc, flood
		Cluster-2	normal, probe, priesc
		Cluster-3	normal, probe, priesc, flood
KDD CUP'99	5	Cluster-0	normal
		Cluster-1	normal, probe, priesc, flood
		Cluster-2	normal, probe, priesc, flood
		Cluster-3	normal
		Cluster-4	normal

5.2 KDD CUP'99

Table 1 shows that KDD CUP'99 dataset had five clusters after the application of DP clustering technique. Upon investigation it is found that in three clusters there were only normal instances. The clusters, viz., Cluster-0, Cluster-3, and Cluster-4, as shown in Table 1, are mono target-class that contain only normal instances. Hence, the need for binary classification using NB classifier does not arise for Cluster-0, Cluster-3, and Cluster-4 of Table 3 and the evaluation metrics are kept blank. However, the cluster radius and cluster center are well enough to decide whether a future instance falls into the cluster or not. The remaining three clusters have data instances from all the target-classes. The NB classifier differentiates the data instances with a greater accuracy, as in Table 3.

To summarise the experiment, DP Clustering algorithms were applied on NIDS datasets such as KDD CUP'99 and SSENet-2011. From the experiments it is evident that the underlying data distribution models, and hence the number of clusters, were different for the datasets.

Table 2. NB Classifier and Generation of RIPPER rules on the clusters of SSENet-2011

Cluster-ID	Pair-wise Target-Class	Detection Rate	Precision	Recall	ROC Area	Execution Time (s)	No. of Rules
Cluster-0	Normal-probe	90.13	95.1	90.1	92.9	0.03	6
	Normal-priesc	90.27	91.3	90.3	95.6	0.03	9
	Normal-flood	99.47	99.5	99.5	99.8	0.03	5
Cluster-1	Normal-probe	96.14	96.6	96.1	97.8	0.06	10
	Normal-priesc	88.71	91.7	88.7	93.3	0.06	14
	Normal-flood	98.79	99.7	98.8	86.9	0.03	3
Cluster-2	Normal-probe	96.78	97.5	96.8	99.6	2.06	14
	Normal-priesc	93.98	96.4	94.0	99.0	1.86	22
Cluster-3	Normal-probe	94.33	95.3	94.3	97.3	0.06	9
	Normal-priesc	71.33	84.3	71.3	83.9	0.08	17
	Normal-flood	99.06	99.1	99.1	99.1	0.36	9

Table 3. Naive Bayes Classifier and Generation of RIPPER rules on KDD CUP'99

Cluster-ID	Pair-wise target-class	Detection rate	Precision	Recall	ROC area	Execution time (s)	No. of rules
Cluster-0	Normal	-	-	-	-	-	-
Cluster-1	Normal-probe	98.78	98.8	98.8	99.8	0.17	7
	Normal-priesc	91.60	92.1	91.6	97.9	0.02	6
	Normal-flood	96.74	96.8	96.7	98.1	0.02	4
Cluster-2	Normal-probe	99.01	99.0	99.0	99.8	0.10	7
	Normal-priesc	97.96	98.2	98.0	99.5	0.07	9
	Normal-flood	98.59	98.6	98.6	98.6	0.09	4
Cluster-3	Normal	-	-	-	-	-	-
Cluster-4	Normal	-	-	-	-	-	-

The proposed hybrid technique was compared against the existing hybrid methodologies. For performing the comparison, the methodology as shown in Fig. 5 was considered. The rule generator, RIPPER algorithm, was considered the same across all the methods. The first and second stages differ across the hybrid models. Tables 4 and 5 give the comparison of the performance metrics in terms of accuracy, precision, recall, execution time, and the rules generated across the four hybrid models for SSENet-2011 and KDD CUP'99 datasets.

From the tables it is evident that the proposed model (M4) has a better accuracy with much lesser execution time than the other three methods. Also, the number of rules generated was comparatively lesser than the other three methods. Apparently, smaller ruleset size with higher detection rate is more desirable in a signature based NIDS.

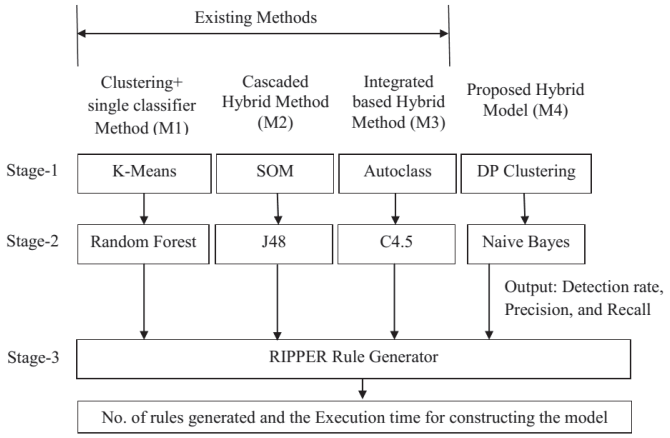

Figure 5. Experimental hierarchy for comparison of hybrid models.

Table 4. Comparative results of the hybrid models on SSENet-2011 dataset

Evaluation Parameters	M1	M2	M3	M4
No. of clusters formed	4	4	5	4
Detection Accuracy	97.7	99.5	96.3	99.8
Precision	97.7	99.6	96.3	99.7
Recall	97.6	99.7	96.8	99.9
ROC Area	97.9	99.8	96.8	99.9
Execution Time (sec.)	1723	760.21	630.8	4.79
No. of rules generated	166	150	99	118

Table 5. Comparative results of the hybrid models on KDD CUP'99 dataset

Evaluation Parameters	M1	M2	M3	M4
No. of clusters formed	4	4	3	5
Detection Accuracy	99.5	98.8	95.4	98.8
Precision	99.5	98.7	95.5	98.8
Recall	99.6	98.7	95.5	98.8
ROC Area	99.8	98.8	96.1	99.2
Execution Time (sec.)	613.2	779.2	372.6	2.35
No. of rules generated	111	214	81	37

6. CONCLUSIONS

The problems of effective clustering of data instances and computational efficiency in the classification process of a huge dataset such as NIDS have been addressed and the hybrid model consisting of DP Clustering, Naive Bayes classifier, and RIPPER rule generator has been proposed as a solution to it. The innovations are the usage of DP clustering, Naive Bayes classifier, and RIPPER rule generator for the formation of distinct data clusters and a computationally efficient classifier. The contribution of this paper is a hybrid model combining the clustering and classification algorithms in a sequential manner for building a robust model. The RIPPER rule generator forms signatures that are deployable in a perimeter security device.

The proposed solution has been experimented on the SSENet-2011 and KDD CUP'99 frameworks and has been compared to the other three different hybrid models. From the experiments it is evident that the proposed cascaded hybrid model generates smaller ruleset and has a higher detection rate compared to other three hybrid models, which are very much the desirable properties of a signature based NIDS.

REFERENCES

1. Igino, Corona; Giorgio, Giacinto & Fabio, Roli. Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues. *Info. Sci.*, 2013, **239**(1), 201–225. doi:10.1016/j.ins.2013.03.022
2. Samaneh, Rastegari; Philip, Hingston & Chiou-Peng, Lam. Evolving statistical rulesets for network intrusion detection. *Appl. Soft Comput.*, 2015, **33**, 348–359. doi:10.1016/j.asoc.2015.04.041
3. Wolpert, D. The supervised learning no-free-lunch theorems. *In* Proceeding of 6th Online World Conf. on Soft Computing in Industrial Applications, 2001, pp. 25–42.
4. Chih-Fong, Tsaia; Yu-Feng Hsub; Chia-Ying Linc, & Wei-Yang Lin. Intrusion detection by machine learning: A review. *Expert Sys. Applications*, 2009, **36**(10), 11994–12000. doi:10.1016/j.eswa.2009.05.029
5. Wozniaka, Michal; Granab, Manuel & Emilio, Corchadoc. A survey of multiple classifier systems as hybrid systems. *Info. Fusion*, 2014, **16**, 3-17. doi:10.1016/j.inffus.2013.04.006
6. Summer, Robin & Paxon, Vern. Outside the closed world: On using machine learning for network intrusion detection. *In* IEEE Symposium of Security and Privacy, May 2010, pp. 305-316. doi:10.1109/SP.2010.25
7. Mellia, Marco; Carpani, Andrea & Renato, Lo Cigno. TStat: TCP statistic and analysis tool. *Quality Service Multiservice IP Networks*, 2003, **2601**, 145-157. doi:10.1007/3-540-36480-3_11
8. Vasudevan, A. R.; Harshini, E. & Selvakumar, S. SSENet-2011: A network intrusion detection system dataset and its comparison with KDD CUP 99 dataset. *In* Proceeding of Second Asian Himalayas International Conference on Internet (AH-ICI), 2011, 1-5. doi:10.1109/AHICI.2011.6113948

9. Vlachos, Andreas; Korhonen, Anna & Zoubin, Ghahramani. Unsupervised and constrained dirichlet process mixture models for verb clustering. *In* Workshop on Geometrical Models of Natural Language Semantics (GEMS '09), 2009, 74-82.
10. Jason, D.M. Rennie; Shih, Lawrence; Teevan Jaime, & David R. Karger. Tackling the poor assumptions of naive bayes text classifiers. *In* Proceeding Twentieth Int. Conf. on Machine Learning (ICML), Washington DC, 2003, 1-8.
doi:10.1.1.13.8572
11. Colas, F.P.R. Data mining scenarios for the discovery of subtypes and the comparison of algorithms. Leiden University, 2009. PhD Thesis.
12. Karthick, R. R.; Hattiwale, V.P. & Ravindran, B. Adaptive network intrusion detection system using a hybrid approach. *In* Proceeding of COMSNETS, 2012, 1-7.
doi:10.1109/COMSNETS.2012.6151345
13. Yangfeng, Ji; Tong, Lin & Hongbin, Zha. CDP mixture models for data clustering. *In* Proceeding of International Conference on Pattern Recognition (ICPR 2010), 2010, 637- 640.
14. Daume III, Hal & Marcu, Daniel. A Bayesian model for supervised clustering with the dirichlet process prior. *J Mach. Learning Res.*, 2005, **6**, 1551-1557.
15. Fan, Wentao; Bouguila, Niza, & Sallay, Hassen. Anomaly intrusion detection using incremental learning of an infinite mixture model with feature selection. *In* Proceeding 8th International Conference on RSKT 2013, Halifax, NS, Canada, 2013, 364-373.
doi:10.1007/978-3-642-41299-8_35
16. Koc, Levent; Thomas, A. Mazzuchi & Sarkani, Shahrami. A network intrusion detection system based on a Hidden Naive Bayes multiclass classifier. *Expert Sys. Applications*, 2012, **39**(18), 13492–13500.
doi:10.1016/j.eswa.2012.07.009
17. Barbará, Daniel; Couta, Juli, & Jajodia, Sushil. ADAM: detecting intrusions by data mining. *In* Proceeding of IEEE Workshop on Information Assurance and Security, 2001, 11-16.
doi:10.1.1.138.8956.
18. Khor, Kok-Chin; Ting, Choo-Yee & Amnuaisuk, Somnuk-Phon. From feature selection to building of bayesian classifiers: A network intrusion detection perspective. *American J. Appl. Sci.*, 2009, **6**(11), 1948-1959.
19. DARPA KDD CUP dataset. http://www.ll.mit.edu/mission/communications/CST/ darpa.html, [Accessed on June 2013].
20. Chauhan, Himadri; Kumar, Vipin; Pundir, Sumit & Pilli, Emmanuel S. Comparative analysis and research issues in classification techniques for intrusion detection. *In* Proceedings of the International Conference on Advanced Computing, Networking, and Informatics, India, 2013, 675-685.
doi:10.1007/978-81-322-1665-0_68.

ACKNOWLEDGEMENTS

The authors are grateful for the sponsorship provided by National Technical Research Organisation (NTRO), New Delhi, India under the Collaborative Directed Basic Research (CDBR) - Smart and Secure Environment (SSE) Project.

CONTRIBUTORS

Dr S. Selvakumar received his PhD from the Indian Institute of Technology Madras (IITM), Chennai in 1999. He is a Professor in the Department of Computer Science and Engineering, National Institute of Technology, Tiruchirappalli, Tamil Nadu. His research interests include : Network security, computer networks, high-speed networks, mobile networks, and wireless sensor networks. He has to his credit of publishing 68 research papers.

Mr A.R. Vasudevan received his BTech (CSE) from Nehru College of Engineering and Research Center, Thiruvilwamala, Kerala, in 2006 and MTech (Cyber Security) from Amrita Vishwa Vidyapeetham, Coimbatore, Tamil Nadu, in 2009. Currently pursuing his PhD in the Department of Computer Science and Engineering, National Institute of Technology, Tiruchirappalli, Tamil Nadu. His areas of interest include: Network security, computer networks, and digital forensics.