Sādhanā Vol. 40, Part 6, September 2015, pp. 1737–1767. © Indian Academy of Sciences

Multi-level anomaly detection: Relevance of big data analytics in networks

SAAD Y SAIT*, AKSHAY BHANDARI, SHREYA KHARE,
CYRIAC JAMES and HEMA A MURTHY

Department of Computer Science and Engineering, Indian Institute of Technology
Madras, Chennai 600 036, India
e-mail: saad@cse.iitm.ac.in; akshayb@cse.iitm.ac.in; shreya@cse.iitm.ac.in;
cyriac83@gmail.com; hema@cse.iitm.ac.in

MS received 31 August 2014; revised 1 April 2015; accepted 6 June 2015

Abstract. The Internet has become a vital source of information; internal and external attacks threaten the integrity of the LAN connected to the Internet. In this work, several techniques have been described for detection of such threats. We have focussed on anomaly-based intrusion detection in the campus environment at the network edge. A campus LAN consisting of more than 9000 users with a 90 Mbps internet access link is a large network. Therefore, efficient techniques are required to handle such big data and to model user behaviour. Proxy server logs of a campus LAN and edge router traces have been used for anomalies like abusive Internet access, systematic downloading (internal threats) and DDoS attacks (external threat); our techniques involve machine learning and time series analysis applied at different layers in TCP/IP stack. Accuracy of our techniques has been demonstrated through extensive experimentation on huge and varied datasets. All the techniques are applicable at the edge and can be integrated into a Network Intrusion Detection System.

Keywords. Machine learning; big data; anomaly detection; one class classification; time series analysis.

1. Introduction

In the age of Internet, it is vital to maintain the integrity of network resources. The current upward trends in Internet access for over a decade have brought forth the problem of securing network resources from external and internal threats. A typical educational institution like Indian Institute of Technology Madras with about 9000 users is faced with threats like unacceptable use of Internet — like systematic downloading of documents from publishers. In this paper, we describe mechanisms to control threats encountered at educational institutions like IIT Madras.

*For correspondence

Traditionally, Network Intrusion Detection System (NIDS) (Mukherjee *et al* 1994) has been used to detect unusual network behaviour. Network intrusion detection systems may be *signature-based* or *anomaly-based*. While the former aims to identify attacks by matching attack patterns with signatures, the latter tries to define what comprises normal usage, and triggers an alarm when deviations from normal usage are observed. The former is unable to identify new attacks while the latter has the potential to do so, but at the cost of a larger number of false positives. In our work, we have focused on anomaly-based network intrusion detection (A-NIDS).

A-NIDS techniques may be statistical, knowledge based or machine learning based (Garcia-Teodoro *et al* 2009). In the statistical-based technique, a normal profile of traffic is maintained; current traffic is compared with the normal traffic to obtain an anomaly score, which when it crosses a threshold, then an anomaly is flagged. Expert-systems based approaches intend to classify audit data based on a set of rules obtained from training data. Machine-learning based approaches are based on building a behavioral model from labelled data which can be used to categorize examples.

A-NIDS may be applied at different levels. For e.g., protocol analysis, behavior analysis, flow analysis and application semantics. A more recent trend in A-NIDS is the use of distributed architecture to detect intrusions by means of detectors co-operating together with a central console to supervise the overall detection process (Seresht & Azmi 2014). Another trend is to make these detectors as pluggable modules, and detection capability is obtained by combining and correlating the information from different detectors. This last type of IDS has been referred to as a *multi-level IDS* (Al-Nashif *et al* 2008).

While prior techniques have combined the output of detectors at various levels — viz. protocol analysis, flow analysis, application semantics (Al-Nashif *et al* 2008), and in some cases output from kernel-level and user-level detectors (Dini *et al* 2012), the novelty of our approach lies in the fact that only one of our detectors is active all the time — the detector at the network (IP) layer; this basically models the traffic at the edge using coarse-grained (linear) time series models which are not computationally intensive. When this detector detects anomalous behaviour, it triggers the detectors in upper layers to confirm the type of attack (viz. abusive usage, systematic downloading or DDoS). The detectors in the upper layers are typically more complex in nature; they are started only when an attack is detected.

Typically LAN topology in educational instituitions is as shown in figure 1. We have designed detectors using machine learning techniques for abusive usage, systematic downloading and DDoS attacks which can be incorporated at the IDS.

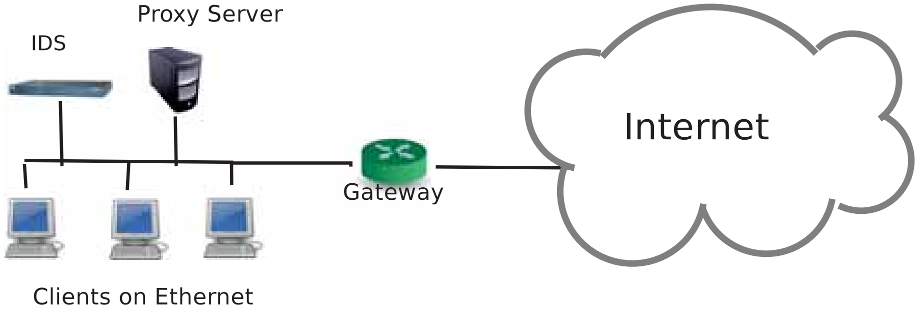


Figure 1. LAN topology.

Section 3 discusses detection of abusive Internet access and URL classification, while section 4 discusses malicious attacks like systematic downloading and DDoS detection. Section 5 deals with anomaly detection. Finally, conclusions are provided in section 6.

2. Multi-level anomaly detection

The system we have designed targets educational institutions which are prone to attacks like abusive usage, systematic downloading and DDoS. In what follows we discuss some of the approaches to detection, and suggest our solution to each individual problem. We have considered only anomaly-based detection schemes. For signature-based schemes, an exhaustive sequence of events that specify normal behaviour need to be maintained; even minor variations in this behaviour may not be detected; further, they can detect only known attacks.

(i) *Systematic downloading*: Typically a campus pays a fee to the publishers (like IEEE, ACM, etc.) for subscription to its research articles, which covers access for personal usage. Systematic downloading is the excessive download of licensed publications from a publisher which violates the acceptable usage policy. It culminates in the denial of access for the entire institution. Detection of such an attack using a threshold is not practical because of the dynamic nature of the traffic to these publishers. We have used time series models to learn the traffic model and devised an automated technique which can capture the intent of the user and pro-actively detect systematic downloading.

Tracking significant changes in the traffic pattern provide an insight for detecting anomalies. Thottan & Ji (1998); Wu & Shao (2005) deploy time series data obtained from management information base variables (MIB) to raise alarms for network anomalies. Time series analysis have been used in many applications like spam detection, system failure detection, biosurveillance, etc. Rule based, pattern matching and statistical analysis approaches are the popular methods of detecting anomalies in time series (Thottan & Ji 2003). Rule based detection systems require an exhaustive database containing the rules of behaviour of the faulty system (Ndousse & Okuda 1996), but these systems are too slow for real time applications. In pattern matching based techniques, anomalies are defined as a deviation from normal behaviour that can be associated with parametric and non-parametric changes evolving gradually in the system (Chin *et al* 2005). Statistical pattern matching techniques attempt to address variability in traffic, by building traffic profiles using online learning (Thottan & Ji 2003). Divakaran *et al* (2006a); James & Murthy (2012); Thottan & Ji (1998); Wu & Shao (2005) and Liu & Kim (2010a) implement various statistical approaches to detect anomalies and intrusions in different systems.

Our work performs anomaly detection using time series models in tandem with pattern analysis techniques (One-class classification) for evolving environments. This hybrid approach has been used to design an agent that can be used at the campus proxy to detect systematic downloading on the fly. While signal processing provides a good set of features, pattern analysis makes use of these features for modelling and detection.

(ii) *Abusive usage*: Abusive usage is another common problem with campus networks. Bommepally *et al* (2010) has found that a large fraction of the Internet access bandwidth is consumed by a small number of users. Prior work in this area (Chu & Chang 2007; Kumar *et al* 2000; Lin *et al* 2004; Paine & Griggs 2008) simply counts the bytes for each user to detect abusive usage. However, detection of such users is not as straightforward as counting the bytes. This is because downloading 1GB data during lean hours is not the same as downloading during peak hours. A more accurate way would be to compute the measure

$$\varphi_u = \frac{b_u}{\frac{tot}{N}}, \tag{1}$$

where N is the number of Internet users during the day, b_u is the bytes downloaded by user u in a day and tot is the total bytes downloaded by all users in that day. However, this requires monitoring and computation across all users. Given a small set of suspicious users, it is far easier to determine if they fit a pre-trained model of abusive users; for this again, machine learning techniques prove useful.

(iii) *DDos detection*: Distributed Denial of Service (DDoS) is a common malicious attack on computer systems. Many signature-based approaches have been proposed in order to detect these attacks (Syndefender and Syn-cookies). The disadvantages of these schemes have already been mentioned.

Time series modelling, has been used in the past to detect anomalies (Divakaran *et al* 2006b; Siris & Papagalou 2006; Wang *et al* 2002b). Modelling using *fine-grained* time series becomes all the more important in the context of such attacks, that are shown to affect the dynamics of the system at fine scales (Guirguis *et al* 2005a). These dynamics get averaged out during aggregation for coarse-grained feature(s) and can be studied only by modelling with fine-grained features(s).

Two approaches have been used in time series modelling; parametric and non-parametric. Parametric approach is used when a model is valid over the entire input space. So, the model coefficients are estimated from the training data to build a single static model. This is possible when the series is stationary. If the above conditions are not satisfied, then model coefficients are updated using data collected over regular time intervals to the order of a few seconds or minutes. This is the non-parametric approach. Most studies (Arshadi & Jahangir 2011; Divakaran *et al* 2006b; Siris & Papagalou 2006; Wang *et al* 2002b; Ye *et al* 2003) use *non-parametric* time series methods to model fine-grained network time series extending to days and weeks. The benefits of the parametric approach are that model parameters can be determined using standard parameter estimation techniques rather than empirically; moreover, they need not be recomputed at regular intervals. This work aims to decouple the stationary and non-stationary components in the time series, thereby enabling the computation of simple parametric models for the former which will remain valid for months ahead.

Two known techniques for intrusion detection are signature-based and anomaly-based. While the former aims to identify attacks by matching attack patterns with signatures, the latter tries to define what comprises normal usage, and triggers an alarm when deviations from normal usage are observed. The former is unable to identify new attacks while the latter has the potential to do so, but at the cost of a larger number of false positives.

(iv) *Anomaly detection at IP layer:* Figure 2 describes the proposed system. Our technique comprises detection at various layers; the network volume analyzer builds a coarse-grained time series (using linear models) of bytes transferred in each sampling interval. The efficacy of using linear prediction to detect network anomalies has been demonstrated in Thottan & Ji (2003). Due to the simplicity of the model, predicted values can easily be computed and anomalies flagged off for deviations that are above a threshold. Alarms at this layer are used to trigger detectors at higher layers viz. DDoS detection at transport layer, systematic downloading and abusive Internet access at the application layer. This scheme has the benefit that simple coarse grained time

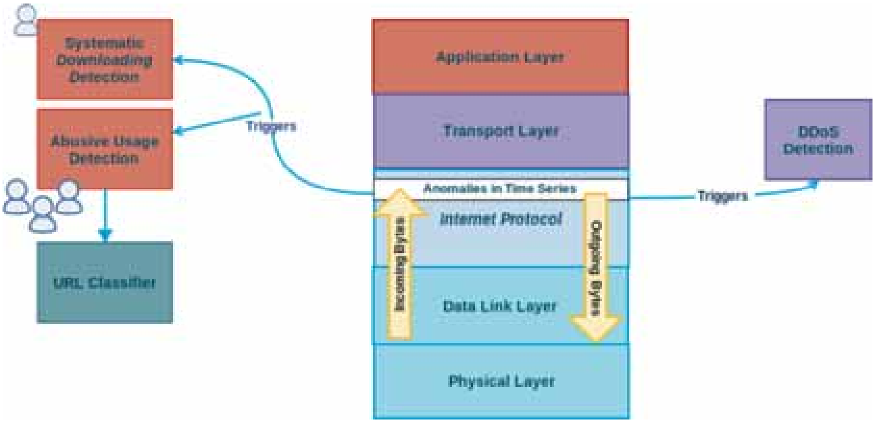


Figure 2. Architecture of the multi-level anomaly detection system.

series, are used to detect the attacks which can be confirmed by further analysis in the upper layers. The agents in the upper layers are typically more complex in nature; they are started only when an attack is detected.

Further, we have designed an innovative technique to classify URLs based on the meta-data obtained from the web-pages. This can be used by abusive usage detector in order to categorize URLs which would enable more accurate feature vectors.

3. Detection of abnormal usage

Use of Internet in college campuses has increased drastically in recent years, with proportionate increase in abusive usage. On college campuses, there has been a drift in the primary use of the Internet from academics to videos and entertainment based content which consume a major share of the bandwidth. This has forced the administrators in some cases to exercise rate limitation policies to ensure that all users get a fair share of the bandwidth. This shaping of network traffic e.g., by blocking Youtube, Facebook, etc. or restricting the time of usage, prevents users from accessing useful (academic) information that is available through these channels. The idea is to apply user based policies, and rate limit flows belonging to abusive users, thereby saving bandwidth for other users. We have done this by innovatively classifying URLs as entertainment, academic or any other category. This provides valuable information which can be used to select features for classification and characterization of users. Data from proxy logs has been used to build models for URL as well as user classification.

3.1 *Abusive usage control*

We consider the problem of abnormal Internet access during peak hours. LANs in university campuses and commercial institutions that access the Internet are faced with the problem of bad Internet experience during peak hours. Bommepally *et al* (2010) has found that a large fraction of the Internet access bandwidth is consumed by a small number of users. These users need to be controlled so that Internet access can be fair to everyone. For this, suspicious users may be categorized as *high* or *low* users. Then suitable control policies may be applied to control the high users.

Previous work with respect to controlling abusive Internet access may be found in Chu & Chang (2007); Kumar *et al* (2000); Lin *et al* (2004); Paine & Griggs (2008); while Kumar *et al* (2000) controls peak hour usage, Lin *et al* (2004); Paine & Griggs (2008) control abusive usage by scheduling packets of misbehaving nodes on a low priority queue. Chu & Chang (2007) extends the work in Lin *et al* (2004) with time-of-day pricing (TDP), and obtains significant improvements in peak-hour usage and fairness, and load balancing across the time periods. Although Lin *et al* (2004); Chu & Chang (2007); Paine & Griggs (2008) provide effective mechanisms to control Internet access, their solutions assume static IP addresses, which may no longer hold due to the widespread use of wireless devices; they also assume that every user has his own (unshared) node.

Using the proxy server logs of IITM, suitable machine learning models may be built for high and low usage at the end of each month (see figure 3). These models are then used during the course of the month to categorize suspicious users based on their past week's usage into high and low categories; then policies are accordingly applied depending on the category. The advantage of such models is that suspicious usage may quickly be classified based on the machine learning model rather than performing extensive calculations on the usage across all users.

In what follows, we have analysed the suitability of NB and GMM models for identifying high users and characterizing them. We have outlined two models — *usage-based* model and the other *user-based* model. While the usage based model is suitable for identifying high users, the user-based model may be used for characterizing them. We recommend weekly features rather than daily features for classification; intutively, this is better as it is unfair to penalize based on a day's usage. Further, we have shown empirically that a week's usage is more reflective of the behavior of a high user than a day's usage.

3.1a *Data set used and features*: The data set we used consists of one month of proxy server logs of a university campus which had a 90 Mbps Internet access link, with 6906 users on the LAN. The features used for building the models are shown in table 1. A bag of words model is used as a feature vector i.e., $\langle f_1, f_2, \ldots f_n \rangle$, where each feature f_i is the number of occurrences of the feature i, or, in the case of a statistical feature - its count in a certain time period (typically a day or a week).

3.1b *Labelling*: Labelling is the task of ground truthing. Examples have to be labelled before a machine learning model can be built. Each example is a vector of the kind mentioned in the

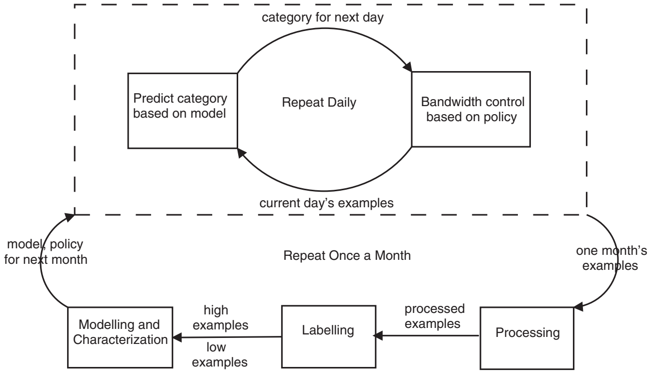


Figure 3. Internet access control framework.

Table 1. Features extracted from proxy logs.

Feature type	Feature Name
URL type	Social Networking, Video Hosting, Academics, Email, Web Search, Movies, News, Sports, Shopping, Travel, Sharing
Content type	Text, image, video, audio, compressed format, executables, javascript, real time audio/video, rss feeds, pdf, xml
HTTP request type	Get, post, head, put, delete
Action	Hit, miss, tcp denied
Size	Different sizes
Statistical	Total bytes downloaded, number of packets, number of hours of access, bytes per hour, average bytes per packet, variance of bytes per packet

previous section. The label would be whether the example is representative of high or low usage. Before we label an example, we first obtain a metric called *usage fairness index* for each usage example. This is computed as

$$\varphi_u = \frac{b_u}{\frac{tot}{N}}, \tag{2}$$

where N is the number of Internet users during the day, b_u is the bytes downloaded by user u in a day and tot is the total bytes downloaded by all users in that day. In other words, this is the ratio between the bytes downloaded by a user and the number of bytes he is eligible to download assuming equal sharing between the users. However, simply counting the total number of Internet users during the day and substituting in place of N would give an inaccurate estimate of φ_u. This is because a user who accesses the Internet for only 5 minutes cannot be treated on par with the one who uses it for hours on end. Therefore, the number of users is counted by taking into account the time spent on the Internet; if T_{avg} represents the average time spent in a day by the average user on the Internet, a user i contributes an amount

$$\frac{T_i}{T_{avg}}$$

to the number of users. Thus, we compute effective number of users N_{eff} as

$$N_{eff} = \frac{\sum_i T_i}{T_{avg}}. \tag{3}$$

Substituting N_{eff} in 3 in place of N in 2 we get

$$\varphi_u = \frac{b_u}{tot} * \frac{\sum_i T_i}{T_{avg}}.$$

The value of T_i can be approximated by breaking up the time into 5-minute intervals. In that case T_i would be the number of slots during which user i accessed the Internet, and T_{avg} would represent the average number of slots during which an average user accessed the Internet. φ_u is calculated for each user u for each day. Based on this value, the data is divided into high and low usage classes by choosing a threshold. Hereafter we shall refer to the usage fairness index as simply *fairness*.

It may be noted here that although the users may be classified based on the usage fairness index, this requires computation across all the users. There is no doubt that given the set of features of a suspicious user, it is simpler to classify him/her based on a machine learning model.

3.1c *Feature selection*: Mutual information, which is a measure of predictability, has been used in this work for feature selection. The algorithm used is called Minimum Redundancy Maximum Relevance (Peng *et al* 2005) (mRMR). The idea is to select a set of features that have high predictability for the category of the example (relevance) but have low redundancy amongst themselves. The latter condition is required in order to obtain better classification accuracy. Therefore a set of features S is selected so as to maximize

$$max\ \phi = D - R,$$

where (relevance)

$$D = \frac{1}{|S|} \sum_{F_i \in S} I(F_i; C)$$

and (redundancy)

$$R = \frac{1}{|S|^2} \sum_{F_i, F_j \in S} I(F_i; F_j),$$

where F_i and C are random variables corresponding to feature f_i and the category/class of the example, respectively. The features are selected by the forward selection algorithm (Peng *et al* 2005).

3.1d *Experiments*: Daily and weekly feature vectors of users for the month of January 2012 were extracted from IITM proxy server logs. All feature vectors were labelled by computing the *fairness*, then clustering based on this, and choosing a threshold to separate the vectors into 2 categories. This constitutes a usage-based label for the vector. In the case of user-based labelling, the fairness was computed by adding the fairness values of all days of the month for each user to obtain a cumulative fairness. This cumulative fairness of all users was clustered, and a threshold obtained for separating the users into categories. Subsequently, all daily/weekly examples of that user were grouped in the category of that user. These concepts have been elaborated in table 2. NB and *gaussian mixture* models (GMMs) were built using the above labelled examples. NB models were used in conjunction with mRMR; mRMR was used first to determine the best features for classification after which the NB model was built using those features. GMMs were used in conjunction with PCA; PCA was first used to transform the features to obtain an orthogonal set of features, which were used for training the GMM. GMMs were built using the

Table 2. Usage and user based labelling.

Labelling Type	Data Duration	Basis of Labelling
Usage-Based	Daily	That day's usage fairness index for that user
	Weekly	That week's usage fairness index for that user
User-Based	Daily	That month's usage fairness index for that user
	Weekly	That month's usage fairness index for that user

Universal Background Model with adapt-means-only approach (Reynolds *et al* 2000). While testing, 10-fold cross-validation was used, and the classification accuracies were averaged over all the runs. Table 3 shows the approximate execution time of the different steps.

Tables 4 and 5 indicate the following: (a) daily features are more suitable for usage-based labelling. The reason for this, as shown in figure 4 is due to the larger separation among features (here total bytes feature shown) of the 2 categories for daily data as compared to weekly data. (b) On the other hand, weekly features are more suitable for user-based labelling. As shown in figure 5, weekly features are more well separated than daily data for user-based labelling. (High and low users are likely to have similar daily features but less likely to have similar weekly features) (c) It follows from the previous point that weekly features are more representative of the characteristics of high users. This supports our use of weekly features for classification of

Table 3. Execution times for various steps.

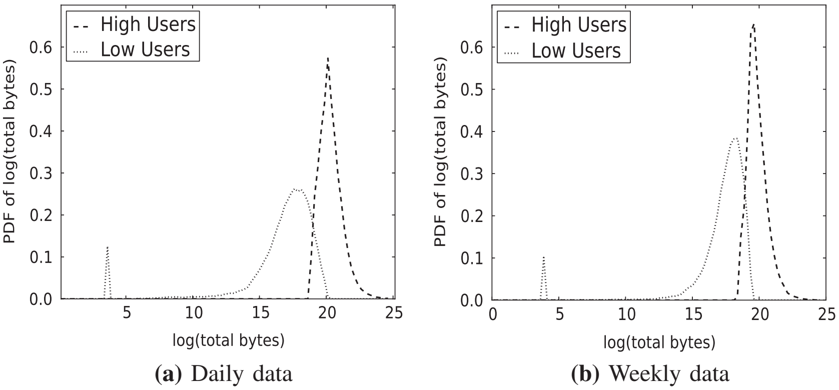
Action		Step no.	Execution Time
Extraction of features from proxy logs		1	1 day
Processing		2	26 secs
Labelling		3	1 min
GMM	PCA	4	8 secs
	Modelling	5	25 mins
	Classification	6	20 mins
NB+mRMR	MI Computation	4	1 day
	mRMR feature ranking	5	1 sec
	Candidate Feature Set	6	6 mins
	Forward Selection	7	A few hours
	Modelling	8	30 secs
	Classification	9	15 secs

Table 4. Classification accuracy for NB classifier.

Labelling	Duration	Class. Accuracy	Std. Dev.
Usage-Based	Daily	89.11%	0.22%
	Weekly	87.25%	0.18%
User-Based	Daily	71.29%	0.38%
	Weekly	79.36%	0.19%

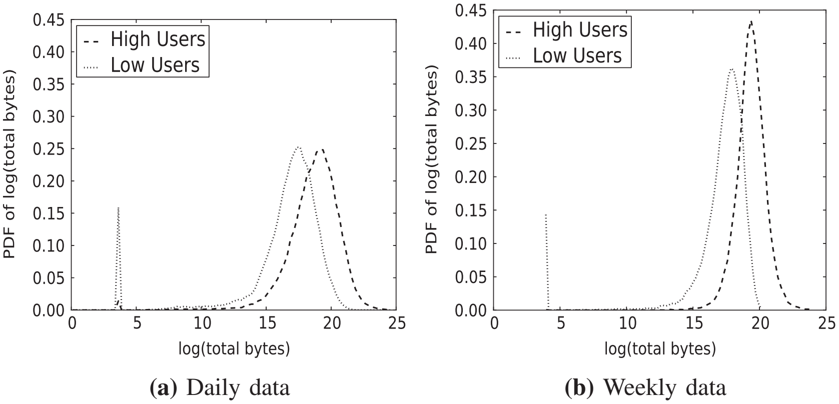
Table 5. Classification accuracy for GMM.

Labelling	Duration	Modes	Class. Accuracy	Std. Dev.
Usage-Based	Daily	2	90.10%	3.44%
		4	89.34%	3.82%
	Weekly	2	89.61%	1.16%
		4	89.56%	1.12%
User-Based	Daily	2	67.31%	4.41%
		4	65.27%	4.29%
	Weekly	2	80.60%	2.74%
		4	79.23%	2.99%



(a) Daily data **(b)** Weekly data

Figure 4. PDFs of total bytes downloaded across usage categories.



(a) Daily data **(b)** Weekly data

Figure 5. PDFs of total bytes downloaded across user categories.

usage. Next, we will compare the characteristics of high as compared to low users, which can be done using user-based models.

Table 6 shows the behavior of groups of (weekly) user examples. These groups correspond to each mixture of the GMM. The mean of the examples belonging to the mixture gives us the average behavior of the group in terms of Internet accesses; the weights of each mixture tell us what percentage of examples in that category belong to that group. Note the same percentage-wise distribution among both categories into groups. This arises because of the use of the UBM coupled with the adapt-means-only approach (Reynolds *et al* 2000). The meanings of the codes in the table are listed in table 7. Since the z-score is obtained by subtracting the mean and dividing by the square root of variance, it is suitable to estimate the ‘bigness’ of a value and the other advantage is that it can be compared across the features. It indicates different patterns of usage based on number of accesses to academic and entertainment sites. Both groups of high user examples have high access to video hosting, movie and social networking sites. Group 3 consists of low users who exhibit a preference for academic over entertainment sites; group 4 consists of low users who have overall low usage for all URL categories.

Table 6. Groups of users.

Features	High users groups		Low users groups	
	1	2	3	4
	66%	34%	66%	34%
MAIL	h	h	h	vl
ACADEMIC	h	l	h	vl
SHARING	h	l	h	vl
SEARCH	h	h	h	vl
VIDHOST	vh	h	l	vl
MOVIE	h	h	l	vl
SOC. NETW.	h	h	h	vl
NEWS	h	l	h	vl
GAMING	h	l	l	l
SHOPPING	h	l	h	vl
SPORT	h	l	h	vl

Table 7. Explanation of extent of usage.

Usage Code	Expansion	Z-score Range
vl	very low	upto −0.5
l	low	−0.5 to 0.0
h	high	0 to 0.5
vh	very high	more than 0.5

3.1e *Discussion*: In this section, we have offered a feasible solution for detecting abusive usage of suspicious users. Models are built on a monthly basis. Suspicious users are monitored, and their past week's usage is used to identify abusive usage. We have shown by way of experiments that weekly features are more reflective of the characteristics of abusive users than daily features. We have experimented with 2 different labelling techniques — usage-based and user-based. While usage based models can be used for identifying abusive usage, user-based models help us to identify the characteristics of abusive users.

3.2 *URL classification*

Today, video hosting websites like Youtube are not primarily entertainment websites, as huge amount of academic content is also available. URL classification is expected to play a critical role in understanding the user behaviour. This can be deemed a problem of web-page classification. In the present context, non-academic content are considered as anomalies.

Choi & Yao (2005), Qi & Davison (2009) have emphasized importance of Web-specific features for web-page classification. Kan & Thi (2005) proposed a mechanism using lexical and host based features associated with a URL for classification purposes. Further, He & Liu (2008) represented web-page with a vector of features for classification.

In this study, meta-data embedded in a HTML page has been used to perform web-page classification (Khare *et al* 2014). Data present in a web-page is huge, noisy and difficult to process. Moreover, for video hosting websites, where the textual content in a HTML page is less, meta-data can provide some of the necessary characteristics of the content embedded in the flash objects or video.

(i) *Dataset:* Proxy logs of IITM were parsed for extracting URLs. For each URL, the meta-data was obtained by parsing the HTML web-page. A labelled dataset was developed using the

McAfee(TM) Trusted source web database. The database contains URLs organized into 104 categories. The database includes categories like entertainment, social networking, pornography, etc. URLs under similar categories were grouped together to form a more general category. The number of categories were reduced to 44 manually to ensure enough members across categories. A total of about 10 gigabytes (2 lakh URLs) of proxy data was used for preparing the labelled dataset.

(ii) Proposed method: Meta-data obtained from the URLs is very noisy and contain a lot of irrelevant information. Figure 6 shows the proposed framework for URL classification.

3.2a *Text pre-processing*: Text pre-processing involves a series of steps to remove noisiness and to structure the data. It comprises of extracting words from the character stream, also known as tokenization, which is usually followed by removal of stop words such as articles, prepositions, etc. as they do not have any discriminating power. Stemming (Porter 1980) is then used to map inflected forms of the words like parts of speech and plural sense to the root of the word. Categorization of entities and proper nouns helps in structuring the data which can then be used to obtain features.

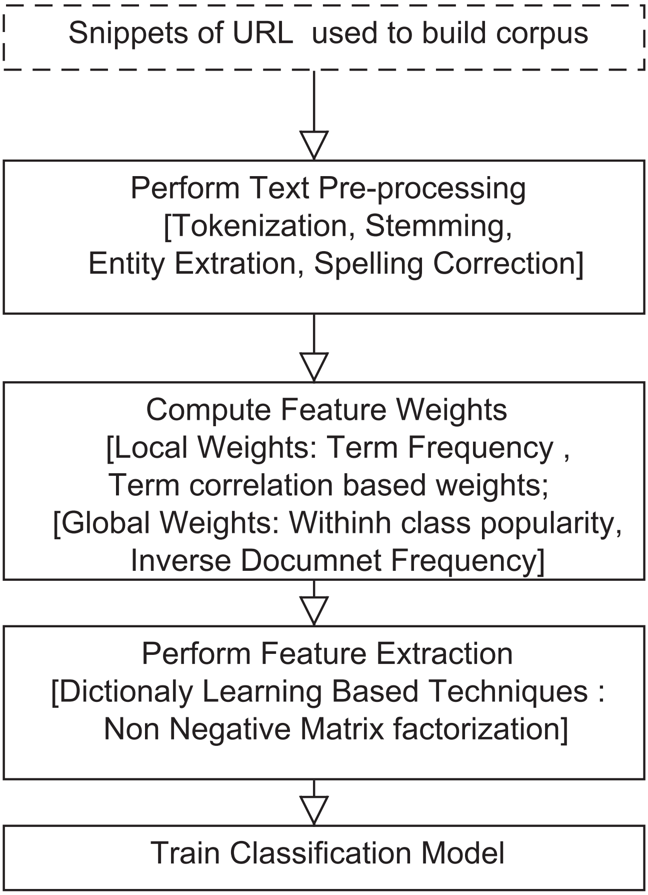

Snippets of URL used to build corpus
Perform Text Pre-processing [Tokenization, Stemming, Entity Extration, Spelling Correction]
Compute Feature Weights [Local Weights: Term Frequency , Term correlation based weights; [Global Weights: Withinh class popularity, Inverse Documnet Frequency]
Perform Feature Extraction [Dictionaly Learning Based Techniques : Non Negative Matrix factorization]
Train Classification Model

Figure 6. Proposed framework for URL classification.

3.2b *Text representation using feature weights*: In a vector space model, a 'document' is represented as a vector (Salton & Buckley 1988). In this work, a document refers to the meta-data of a particular URL. Henceforth, a document (D_j) is represented as

$$D_j = \{w_1, w_2, w_3, w_4, ...w_m\}, \tag{4}$$

where w_i represents the frequency of a term in the document.

Frequency of a term in a document (Term Frequency) and Inverse Document Frequency (IDF) are used to characterize documents. IDF of a term w in a document d can be calculated as

$$idf(w) = log \frac{|D|}{df(w, D)}, \tag{5}$$

where $|D|$ is the number of documents in the corpus and $df(w, D)$ is the number of documents in which the word w appears.

Within Class popularity (WCP) seems to be a useful technique for the purpose of document classification. This feature weight addresses the issue of uneven distribution of prior class probabilities (Singh *et al* 2010). Using WCP, global goodness of a term is derived as a weight representing the distribution of a term across all classes of documents. WCP is calculated as

$$wcp(w, c_i) = \frac{Pr(w|c_i)}{\sum\limits_{k=1}^{|C|} Pr(w|c_i)}, \tag{6}$$

where C is the set of class labels and $Pr(w|c_i)$ is given as

$$Pr(w|c_i) = \frac{1 + N(w, c_i)}{|V| + \sum\limits_{f \epsilon V} N(w, c_i)}, \tag{7}$$

where $N(w, c_i)$ is the number of occurrences of a term w in all the documents in c_i and $|V|$ is the cardinality of the vocabulary set.

To quantify the term co-occurrence within a document, a graph is constructed with unique terms as the vertices. Each term is assigned a score using a random walk algorithm as mentioned in Blanco & Lioma (2007). For a vertex (V_a), let $In(V_a)$ be a set of predecessor vertices, and $Out(V_a)$ be the set of successors vertices . The score of a vertex (term) is defined as

$$S(V_a) = (1 - d) + d. \sum_{V_b \epsilon In(V_a)} \frac{S(V_b)}{|Out(V_b)|}, \tag{8}$$

where d is a damping factor, which is set between 0 and 1 (Hassan *et al* 2007).

Combinations of these weighing techniques quantifying both local and global contribution of a term within a document of a particular class have been formulated by multiplication of the local and global descriptor to represent URLs for classification.

3.2c *Feature extraction based on non-negative matrix factorization*: The feature vectors obtained from the text span a very high-dimensional space. Dimensional reduction is required for meaningful classification (Deerwester *et al* 1990). Since input data reduces to a matrix, low rank approximation algorithms based on dictionary learning can be exploited for dimensional reduction. Non negative matrix factorization (NMF) forms additive parts-based representation of the data (Lee & Seung 2001). NMF decomposes a $m \times n$ (X) matrix, into two matrices W and H of dimension $m \times k$ and $k \times n$ respectively, where $k << (m, n)$.

In our problem, size of the feature vector space, called the document term matrix is, $m \times n$, with m and n as the number of features and documents, respectively. W matrix is called the basis document vector spanning m dimensional space, and H is called the weight matrix which gives relevance to different elements of the basis vectors of W. NMF is usually used to approximate X by computing a pair of W and H to minimize the Frobenius norm, as

$$min_{W,H}\|X - WH\|_f^2. \tag{9}$$

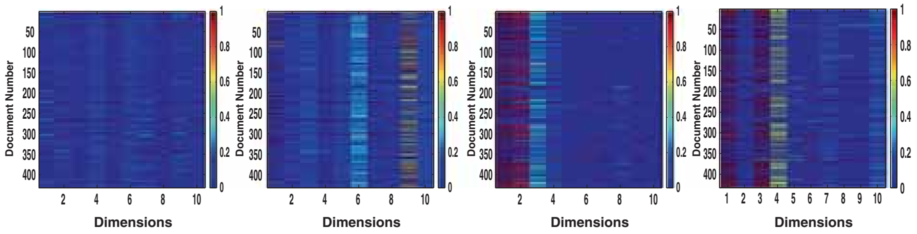
Matrices W and H thus obtained are not unique and are initialized to random values with non negativity constraint. Our study uses Gradient Descent with Constrained Least Squares (GD-CLS) algorithm for NMF. It computes the weight matrix H using constrained least square mode in order to penalize non-smoothness and non-sparsity in H. The particulars of this algorithm can be obtained from Berry *et al* (2007). This algorithm uses smaller number of iterations than other NMF algorithms.

3.2d *Experiments and results*: Textual data obtained was processed to build the vocabulary and form the document term matrix. The following composite schemes of the features were considered for evaluation:

(i) Term frequency and inverse document frequency (TF-IDF)
(ii) Term frequency and within class popularity (TF-WCP)
(iii) Random walk term weights and inverse document frequency (RW-IDF)
(iv) Random walk term weights and within class popularity (RW-WCP).

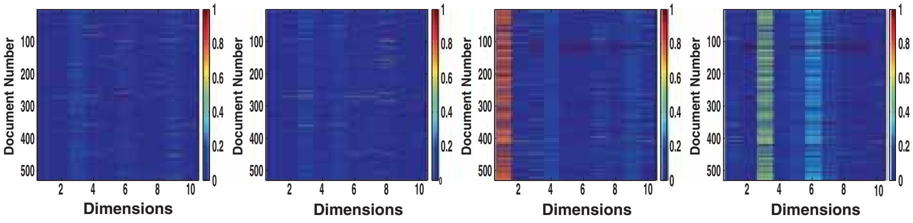
In order to predict the categories of a web-page using the above mentioned feature weighing and extraction technique, Naive Bayes Classifier was used.

To appreciate the dimensional reduction performed by NMF, heat maps of the activations in the reduced space for 2 categories have been plotted (figures 7 and 8). Heat maps are general graphical representation of a matrix (data) where each value of the matrix is shown as a colour in the heat map. For separation between classes in the reduced dimensional space, all the documents in the class should have activations in the same dimension. Moreover, different classes should have activations in different dimensions which ensure good separation between classes.



(a) TF-IDF **(b)** TF-WCP **(c)** RW-IDF **(d)** RW-WCP

Figure 7. Heat map of the weight matrix H of category 4.



(a) TF-IDF **(b)** TF-WCP **(c)** RW-IDF **(d)** RW-WCP

Figure 8. Heat map of the weight matrix H of category 16.

Table 8. Accuracy and F_1 measures for model evaluation.

Feature weights	Feature extraction	Accuracy (%)	F Measure
TF-IDF	NMF	89.62	0.8135
	SVD	74.49	0.7819
TF-WCP	NMF	86.69	0.8456
	SVD	74.49	0.7819
RW-IDF	NMF	91.35	0.8700
	SVD	75.82	0.7821
RW-WCP	NMF	**92.96**	**0.8736**
	SVD	75.82	0.7821

After extraction of features, evaluation of every feature weighting scheme was performed using Naive Bayes classifier. 10 fold cross validation was performed and $F_1 - measure$ was computed for comparative evaluation of each scheme. F_1 measure is computed as

$$F_1 = 2.\frac{precision.recall}{precision + recall}, \tag{10}$$

where precision is the fraction of retrieved instances that are of relevant, while recall is the fraction of relevant instances that are retrieved. A large number of experiments were conducted by changing the initialization parameters and varying the number of reduced dimension from 5 to 100 in NMF and SVD. From these experiments it was observed that the best result was obtained at $k = 10$ and $\lambda = 0.2$. The results (table 8) clearly indicate that the random walk weights in tandem with NMF shows considerable increase in accuracy in comparison with other weighing techniques. However, the accuracy of SVD is less as compared to that of NMF. Higher accuracy and F_1 score in NMF based classification indicates that presence of negative coefficients in SVD reduces the classification performance.

In the framework suggested, the new features, random walk term weights and within-class popularity as feature weight along with NMF for feature extraction and Naive bayes classification has shown promising results. An overall accuracy of 92.96 % has been achieved in classification of the access to the Web. The primary reason for better performance is that the term correlation captured by random walk term weights, and popularity of this correlation in a particular category captures the separability among all the categories.

4. Detection of malicious attacks

With the growing use of network for Internet, the dynamics of traffic changes quite rapidly. This makes the management of network traffic a difficult task. It is important for automatically managing a network to decipher and understand the dynamics of the network. Traffic models can be used to understand the behaviour and help in pro-actively detecting network faults. The traffic at any given point of time is influenced by the traffic in the past. So, network traffic can be perceived as a *time series*, where observations are *dependent*. This property enables the estimation of certain traffic parameters which can be used to model normal traffic; deviations from the normal traffic can be regarded as anomalies.

We present a multi level anomaly detection scheme whose mechanisms may be incorporated at various layers of an IDS. Network traffic modelling at lower layers (i.e., IP layer) raises an alarm when unusual behaviour is detected. Due to a large number of false alarms at the IP layer, a confirmation may be obtained regarding the nature of the attack (viz. DDoS or systematic downloading) by modelling at the TCP and application layers. At the IP layer, we record the bytes data downloaded and packet arrivals at an edge router. The higher-level mechanisms are initiated only when an alarm is raised, thereby saving system resources.

4.1 *Theoretical background*

Network characteristics continuously change with time. Treating network data as a time series, where observations are dependent, helps in modelling and detecting anomalous traffic behaviour as noted earlier. *Time series analysis* helps in analysing such a series of dependent observations.

For a linear, discrete-time system, the objective of linear prediction (Rabiner & Gold 1975) is to estimate the output sequence from a linear combination of input samples from past output samples or both. It can be expressed as,

$$\tilde{y}(n) = \sum_{j=0}^{q} b_j x(n-j) + \sum_{i=1}^{p} a_i y(n-i), \tag{11}$$

where $y(n)$ and $x(n)$ refer to the output and input sample sequence, respectively, and a_i and b_j are coefficients associated with the output and input. Taking Z transform on both sides of Eq. (11),

$$H(z) = \frac{Y(z)}{X(z)} = \frac{1 + \sum_{j=1}^{q} b_j z^{-j}}{1 + \sum_{i=1}^{p} a_i z^{-i}}. \tag{12}$$

This model is termed as autoregressive moving average (ARMA) model. If predictor uses only output samples i.e., all $b_j = 0$, it called an *all-pole* model. An all-pole model has several advantages over the ARMA model. It is based primarily on the fact that a single cause results in a significant change in the output. To give a perspective to this model in the context of network time series, a single burst at a router can affect a number of different routers.

4.2 *DDoS attack detection (transport layer)*

Traffic analysis and anomaly detection are extensively used to understand and characterize network traffic behaviour as well as to identify abnormal operational conditions such as malicious attacks. This study aims to understand and model the long-term fine-grained traffic characteristics of network time series at the edge network for different time scales and deploy the model developed in low intensity DoS attack detection (TCP SYN attack specifically), by studying the stability and stationarity of the underlying process.

The most important aspect of the study is to perform fine-grained time series analysis. Such analysis becomes more important in the context of such attacks, that are shown to affect the dynamics of the system at fine scales (Guirguis *et al* 2005b). These dynamics get averaged out during aggregation in the case of coarse-grained analysis. The fine-grained analysis is performed using parametric methods. In the literature, attempts have been made to use either adaptive and non adaptive mechanism, whose performance is found to be extremely sensitive towards empirically determined parameters of the model and hence difficult to determine. Most techniques assume stationarity and predictability of the given time series.

The analysis is performed on a real traffic trace, collected over a period of three months, from one of the edge routers of IITM as shown in figure 9.

The network trace is collected over a period of three months from July 15^{th} 2010 to September 30^{th} 2010, from the edge router using TCPDUMP utility (TCPDUMP 1999). From this trace, traffic to/from four servers which provide HTTP and SMTP services are identified and the Half-Open Count (HOC) feature is extracted at equally spaced time intervals of 10 s, 60 s and 120 s. Sampling interval in the order of seconds are chosen for two reasons. Firstly, the sampled HOC values were insignificant for smaller time intervals. Secondly, the exponential back-off and retransmission of SYN/ACK packets during a SYN attack happens in the order of seconds. It is this property of TCP which is being exploited for modeling the mechanism of a SYN attack (Divakaran *et al* 2006c; Ranjan *et al* 2010; Wang *et al* 2002a). For simplicity and to provide a representation of the traffic over 3 months, three data sets are created, each consisting of 5 days (i.e., 24 h × 5 days) of continuous traffic, one from each month.

- Data Set-1: 26^{th} July 2010 to 30^{th} July 2010
- Data Set-2: 23^{rd} August 2010 to 27^{th} August 2010
- Data Set-3: 20^{th} September 2010 to 24^{th} September 2010

These data sets are checked for any existing TCP SYN DoS attack by verifying whether all the connection requests are valid. This is done to demonstrate the effectiveness of our approach in detecting SYN DoS attacks. Prior to using this time series for TCP-SYN attack detection, analysis on the stationarity and predictability of time series is essential. Therefore, we will first discuss the characteristics of long range network time series

4.2a *Analysis on long range network time series*: Initially, visual inspection is carried out to characterize HOC network time series. A closer look at the data is shown in figure 10, depicts similarity in the plots across different sampling intervals, which indicates a possibility of *self-similar or scale invariant* behavior (Leland *et al* 1994; Paxson & Floyd 1995). The red dotted line in the case of sampling interval of 120 s is expanded wholly for 60 s and partially for 10 s.

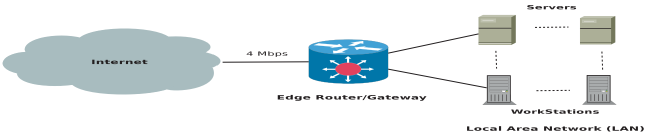


Figure 9. Edge network scenario.

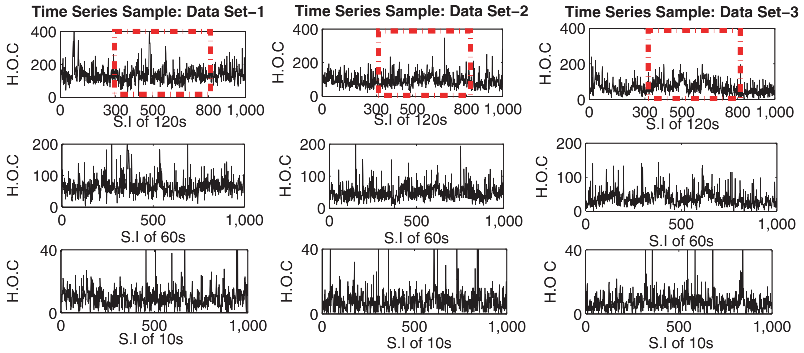


Figure 10. Half-open count characteristics.

Further exploration reveals the presence of transient shocks and structural breaks in the time series, irrespective of sampling interval. Time span for transient breaks in the process is small whereas for structural breaks is throughout the process. Figure 11(a) shows some of the pronounced structural breaks across various sampling intervals. This trend is seen across all three data sets. Such structural breaks cause permanent *mean shifts* in the series and introduces non-stationarity in the data. Further significance of structural breaks is shown by observing Auto Correlation Function (ACF) across different datasets as in figure 11(b). A slowly decaying ACF is an indication of non-stationarity in the data.

Figure 11(b) shows non-deterministic and slowly decaying trend with increasing lag values. This is quantified using Canny Edge Detector Algorithm (Canny 1986). Probability of structural

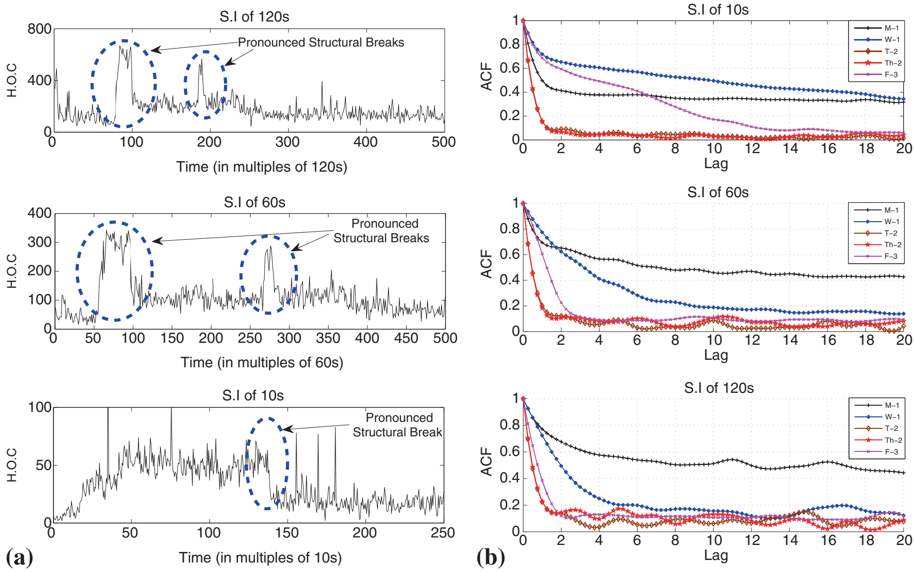


Figure 11. (**a**) Visible structural breaks in the series; (**b**) Sample ACF Showing non-stationarity. M-1, T-1, W-1, Th-1 and F-1 correspond to Monday, Tuesday, Wednesday, Thursday and Friday data of data set-1. Similar conventions are used for other datasets.

breaks with increasing time duration is shown in figure 12. It is essential to decouple the stationary component from the non-stationary component of the process for modeling purposes. An absolute differencing operation is done to remove the non-stationary effects caused by the structural breaks and model the transient shocks. The derived series can be modeled using regression based time series models, provided that the adjacent values in the series have significant correlations and the process is stable as well as stationary.

For our scenario, the time series is modeled using ARIMA process. An AR process is stable if the absolute value of the root(s) of the AR polynomial are within the unit circle. In addition, ACF of the stable process decays rapidly, while that of an unstable process decreases at a rate less than that of an exponential distribution with increasing lag.

The initial approximation on the model orders for the AR and MA parts, is obtained from the estimates of ACF and PACF. Absolute values of the roots of the AR polynomial being well within the unit root circle (figure 13), along with the fastly decaying ACF and PACF, rule out the possibility of long range persistence of anti-persistence. Further, MA component is found to be absent in the series. Hence, FARIMA model is not considered in this work.

4.2b *TCP syn attack detection*: The rationale behind detection is that the prediction error during a normal period will be different from that of an abnormal period. The prediction errors obtained from the model are found to follow a normal distribution with mean 0. A threshold is fixed on the absolute value of the prediction error to detect the attacks. This is equivalent to folding the distribution of error at the mean value, which is also called *half-normal* distribution. The value of the threshold is estimated as sum of the mean and thrice the standard deviation. The proposed method termed as ‘AR’ method is compared with non-adaptive and adaptive methods,

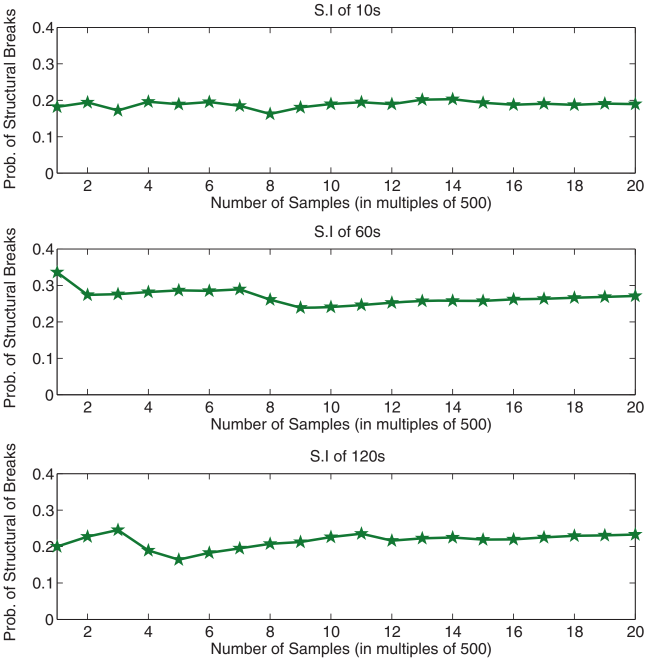


Figure 12. Probability of structural breaks.

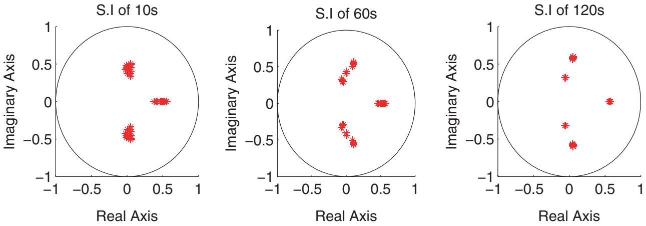


Figure 13. Roots of difference series.

referred to as CUSUM and Entropy methods. The details of these techniques can be found in our paper (James & Murthy 2012).

In addition, we also propose a new estimate called the poles of transfer function which measures stability of the discrete linear systems under study. Sequential tracking of one or mroe roots of the AR polynomial at regular intervals of time enables distinguishing between normal and anomalous periods. During an attack, there will be change in the expected mean and variance, which can be captured by estimating the roots over each window of samples.

4.2c *Attack simulation*: Low intensity TCP SYN DoS attacks, similar to the ones discussed in Siris & Papagalou (2004) and Liu & Kim (2010b) are studied in this work. In such attacks, the attacker aims to degrade the efficiency of the service rather than denying the service completely, by flooding the network with large number of spoofed SYN packets.

Attacks of different rates and varied natures have been simulated: (i) Attacks with rate uniformly varying between 10 and 20 syns/s (mean rate is 13.26 syns/s). (ii) Attacks with rate uniformly varying between 2 and 8 syns/s (mean rate is 3.19 syns/s) and (iii) Slowly increasing/decreasing attacks with rate varying between 0.5 and 5 syn/s. The rate is incremented/decremented in a step fashion by 0.005 between 0.5 and 5, after sending each SYN packet (mean rate is 0.91 syns/s). This is a modification on the stealth attack discussed in Liu & Kim (2010b), where the attack rate always increases. Each type of attack is simulated for over 50 times with duration of nearly 3 h. Traffic trace at the victim server is collected and is mixed with normal traffic for sampling intervals of 10 s, 60 s and 120 s across all three data sets to create the attack traffic.

4.2d *Results*: It is empirically observed that for Auto Regressive models, the appropraite model order 3. Performance for all the three models are compared in terms of False Positives (FP), False Alarm Time (FAT) which is a mean distance between two consecutive alarms, Detection Rate (DR) and Detection Delay (DD). The evaluation for one of the SYN Attacks is mentioned in table 9, others are mentioned in James & Murthy (2012) help us in concluding that the AR method, in terms of False Positives, performs better than the other two with 100% DR. CUSUM method gives 100% DR but has larger FP. It is also concluded that the entropy method appears to be inefficient. Another desirable feature of AR is that false alarms are wide apart, thus making it desirable for sequential decision techniques. As the proposed technique models the dynamics of the system at finer time scales, it is a strong contender for detecting low rate attacks. It can be concluded that the proposed approach gives 100 % detection rate with false positive as low as 0.9 %.

To quantify the concept of the poles of the transfer function as its stability measure, analyses are conducted on sampling intervals of 10 s and for attacks with rate varying between 10 and

Table 9. SYN attack of 10 to 20 syns/second.

(a) Sampling Interval: 10s

Method	FP(%)	DD(sampling units)	DR(%)
AR	2.79	1	100
CUSUM	6.39	1	100
Entropy	0.5	37.9	96.67

(b) Sampling Interval: 60s

Method	FP(%)	DD(sampling units)	DR(%)
AR	1.5	1	100
CUSUM	6.84	1	100
Entropy	0.5	1	95

(c) Sampling Interval: 120s

Method	FP(%)	DD(sampling units)	DR(%)
AR	0.9	1	100
CUSUM	6	1	100
Entropy	0.5	1	80

(d) False Alarm time for different Sampling Intervals

Method	10s	60s	120s
AR	44.39 (7.39 min)	64.85 (64.85 min)	146.31 (292.62 min)
CUSUM	17.17 (2.86 min)	15.59 (15.59 min)	16.40 (32.8 min)
Entropy	250.25 (41.70 min)	177.54 (177.54 min)	93.65 (187 min)

20 syns/s. The roots of the AR polynomial are computed over a window size of 2000 samples at a time, which is shifted by 1000 samples during the next estimation. The sequential change in the estimated values of the root during an attack and non-attack period is shown in figure 14(a). The absolute value of the roots during an attack is relatively higher compared to that of a non-attack period. The spread of the roots during attack and non-attack period in a complex plane is shown in figure 14(b). Since the attacks simulated are not high rate in nature, the system does not become unstable (i.e., service getting completely disrupted), but shows a tendency to move towards the unstable region (i.e., efficiency of the service is affected). This may cause some misclassification due to overlap of attack and non-attack regions, which can be observed in figure 14(b), where a number of points belonging to both the regions lie near the boundary of separation.

4.3 *Systematic downloading (application layer)*

Academic institution libraries are now subscribed to a large number of publishers like IEEE, ACM, etc. for scientific content. Researchers at these academic institutions are allowed access to resources for their personal use. Systematic downloading refers to an attempt to steal subscribed content by downloading for indefinite periods of time, with the malicious intent of selling it to others or for other personal benefits. This act violates the terms of usage between the publisher and subscriber, followed by cessation of access to the content. Baker & Tenopir (2006) reported past significant incidents and consequences of systematic downloading. Vendors are reluctant to reveal specifics about detection, as this information may be used in turn to bypass detection. We have modeled the access to publishers as a time series, and thereby detect systematic downloads.

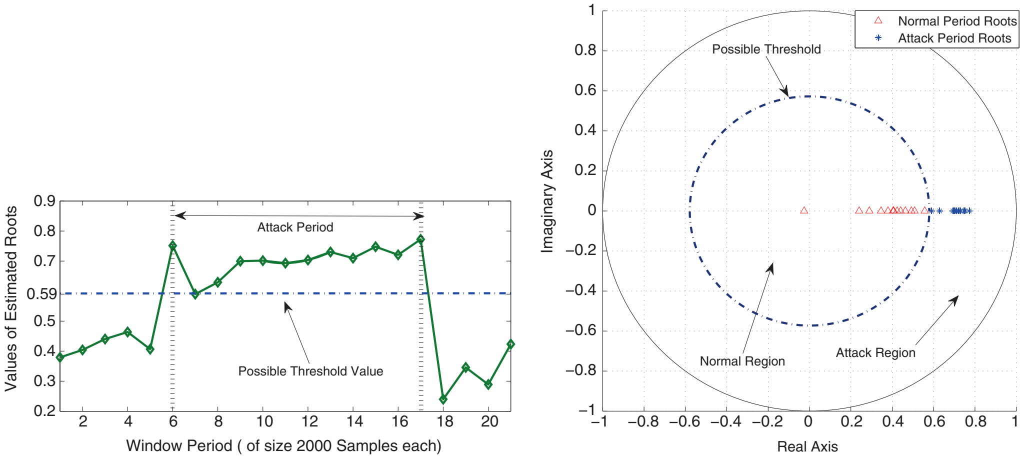


Figure 14. (**a**) Sequential behavior of roots; (**b**) Region of normal and attack period.

The location of the *roots* of the AR model may be used to detect systematic downloads. This is similar to our observation of the roots for DDoS attacks.

4.3a *Dataset*: At an institutional level, the proxy server acts as an intermediary for requests. It logs the details of all the requests from the client. User access pattern for the subscribed content, which is available in the logs, can be used to train models for detection of systematic downloading (Bhandari *et al* 2014). For this task, Normal and Systematic download traffic request are analysed as mentioned below.

- *Normal traffic request*: Proxy logs were pre-processed to obtain traffic requests of the entire institute to the subscribed publishers, over a period of four months. Traffic to each publisher was modeled separately.
- *Systematic download traffic request*: Due to unavailability of systematic downloading traffic in logs, an emulation architecture as shown in figure 15 was designed. Configuration of emulation parameters such as download speed of files, number of documents in Web server, distribution of file size, were obtained from the proxy logs. A large number of simulations were carried out with varying download rate to make emulation a true replica of the real time behavior. The order in which the files were downloaded was randomized to remove correlation between requests.

4.3b *Features*: Appropriate features must be identified which can distinguish between normal and abnormal traffic. We have used number of requests (to a particular publisher) per sampling interval for detecting systematic downloads. Fixing a threshold on number of requests to detect systematic downloads will generate false alarms for bursts in traffic. On the other hand, a per-session threshold may cause interrupted access to subscribed content. Hence, suitable models are required to distinguish systematic downloading from normal access. The number of requests to a publisher was modeled as a time series, and an AR model was used to capture the relationship between successive values in the time series. The *roots* of the AR model are then used to distinguish systematic downloading from the normal case. In addition, *Line Spectrum Pairs* has also

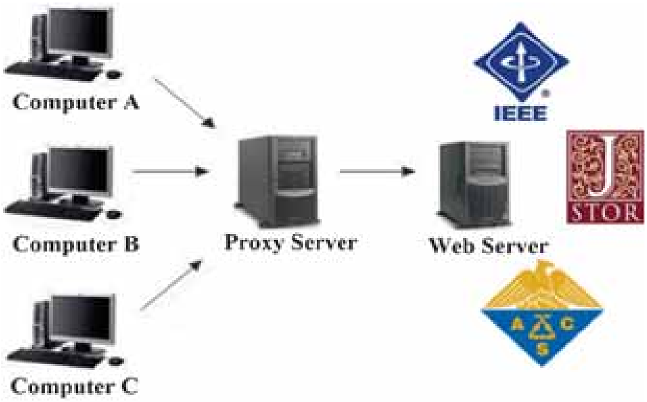


Figure 15. Systematic downloading emulation topology.

been used as a feature for model creation. A small perturbation in the LPC coefficients leads to significant changes in the roots of the AR model. Line Spectrum Pairs (LSPs) have better sensitivity to noise and show better interpolation properties when compared to LPC coefficients. An all-pole filter $H(z) = \frac{1}{A(z)}$, where $A(z)$ is given by

$$A(z) = 1 + a_1 z^{-1} + ... + a_p z^{-p}, \tag{13}$$

where p is the order of the LP analysis and a_i represents the LP coefficients. The inverse filter polynomial can be decomposed into two polynomials

$$P(z) = A(z) + z^{-(p+1)} A(z^{-1}) \tag{14}$$

and

$$Q(z) = A(z) - z^{-(p+1)} A(z^{-1}). \tag{15}$$

Contrary to the roots of $A(z)$ which lie inside the unit circle (set of complex numbers of the form $z = e^{i\theta}$), roots of $P(z)$ and $Q(z)$ lie on the unit circle. Paliwal (1992) explains how LSPs are obtained and its advantages.

4.3c *Systematic download detection*: Systematic downloads have been detected by modeling normal behavior using signal processing and one class classification techniques.

- *Signal Processing Method*: The poles of an all-pole filter are the roots of the inverse filter. As mentioned before, the position of the roots have been used to detect systematic downloading.
- *Data description*: As normal traffic is the norm, a large number of examples of normal traffic are available from the proxy logs. Since details about distribution of normal data cannot be discerned, it is necessary to model the boundary of the normal class. This method is different from the standard classification methods and is termed as *one class* classification. Tax & Duin (2004) presents Support Vector Data Description (SVDD), a method which is robust against anomalies and provides spherically shaped boundaries around a dataset without estimating data density.

SVDD translates the one class scenario to that of finding a minimum enclosing hypersphere (of center a and radius R) that contains normal input data. For a function,

$$f(x) = \| x_i - a \|^2 - R^2, \tag{16}$$

where x_i is a training object, the boundary of the hypersphere is described by the set $x : f(x) = 0 \wedge x \in \chi$. This set is termed as the set of support vectors. The parameters of $f(x)$ are to be chosen such that $f(x) \leq 0$ for normal data and $f(x) > 0$ for anomalous points. The parameters (i.e., center a and radius R) can be calculated by solving the optimization problem as shown in Eq. (17):

$$\begin{aligned} &\min_{R,c,\xi} \quad R^2 + C \sum_{i=1}^{n} \xi_i \\ &\quad s.t. \forall_{i=1}^{n} f(x) = \| x_i - a \|^2 - R^2 - \xi_i \\ &\forall_{i=1}^{n} : \xi_i \geq 0. \end{aligned} \tag{17}$$

In Tax & Duin (2004), the details of solving the above mentioned optimization problem is discussed. The parameter C controls the trade-off between the volume and the errors. Omission of some data points might lead to concise description of data. Slack induced by discarded data points is observed by variables ξ_i. To test an object z, the distance from the center of the sphere is calculated. A test object is classified as normal by calculating distance from the center, which is given as

$$\| z - a \|^2 \leq R^2. \tag{18}$$

4.3d *Experiments and results*: Time series of the number of requests for subscribed content to different publishers was extracted from the proxy server logs over a period of four months. The data generated, as discussed in section 4.3a, was used for analysis purposes. Time series of number of requests was obtained for each publisher separately, and roots of the AR model and LSP were calculated. Two methods for detection were used:

(i) *Signal processing method*

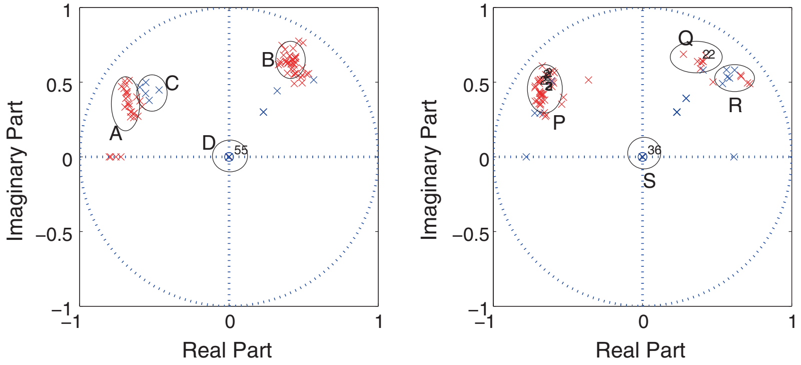
By sequentially tracking one or more roots of the AR polynomial at regular intervals, systematic downloads can be distinguished from normal downloads. At the time of systematic downloading, roots tend to move towards the boundary of the unit circle. During systematic downloading, there will be a change in expected mean and variance, which can be captured by estimating the roots over each window of samples.

Figure 16 shows the clustering of roots at the time of systematic downloading, which suggests that roots pertaining to the same download behavior get clustered together. It is also evident that roots belonging to systematic downloading traffic are closer to the unit circle.

This is similar to our observation in the case of DDoS attacks, that anomalous behavior leads to models with roots close to the unit circle, which indicate instability.

(ii) *Data description method*

A large number of experiments were performed with varying parameters like different polling interval of time series, and C parameter (see eq. (17)), kernel functions and its



(a) Polling Interval– 5 seconds **(b)** Polling Interval– 10 seconds

Figure 16. Clustering of roots at 5 and 10 seconds.

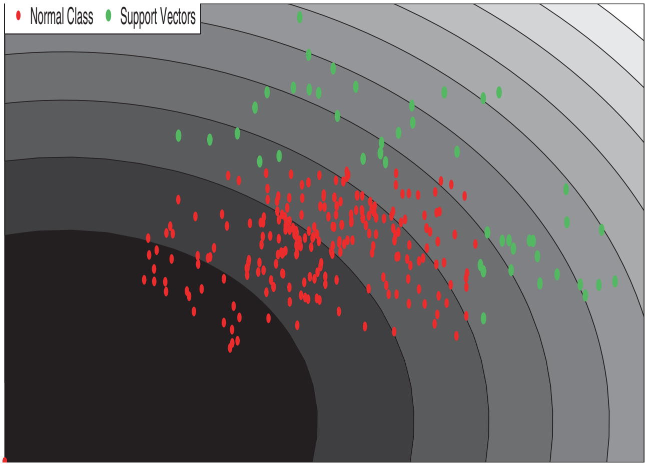


Figure 17. Description of normal data along with support vectors using Gaussian kernel.

parameters in SVDD, to detect systematic downloading. Performance of SVDD has been investigated on the following set of features derived from roots of the AR model and LSPs:

(a) AR model roots
(b) Magnitude and Phase of AR roots (MPAR)
(c) LSP roots
(d) Distance between LSP roots (DLSP).

Figure 17 indicates that normal data is enclosed in a hypersphere bounded with support vectors, represented as green solid points. F_1 measure has been computed for comparative evaluation of features. Figure 18 shows the F_1 measure for AR roots and LSPs using linear kernel for different values of C, with polling interval of the time series being 5 seconds. A lot of variation is observed with different C values for the same feature. It can be inferred that with finely tuned SVDD parameters, distance between LSP roots outperforms other features. This fact is further verified in table 10.

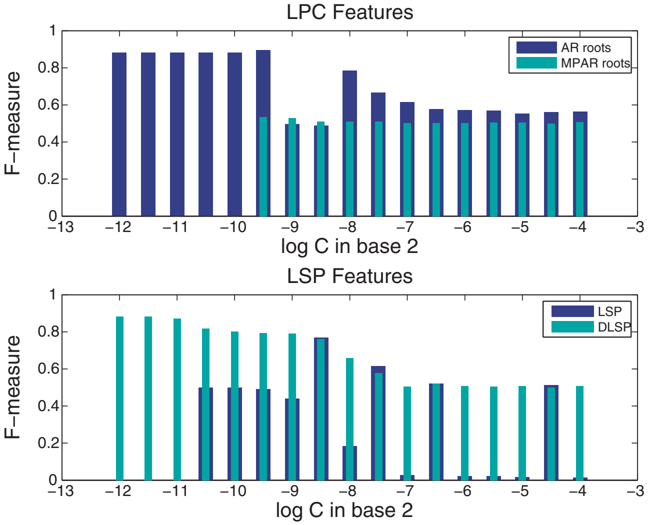


Figure 18. F-measure for linear kernel at different values of *C* for all the four features.

Table 10. Accuracy and F-measure using the derived features for linear, polynomial and Gaussian kernel at 5 and 10 seconds polling interval.

		Linear		Polynomial		Gaussian	
		5 sec	10 sec	5 sec	10 sec	5 sec	10 sec
AR roots	*F-Measure*	0.88	0.85	0.88	0.86	0.89	0.85
	Accuracy	99.5	99.2	99.5	99.3	99.5	99.39
MPAR	*F-Measure*	0.59	0.69	0.55	0.70	0.54	0.69
	Accuracy	98.4	99.0	99.4	99.4	98.1	99.3
LSP roots	*F-Measure*	0.89	0.87	**0.90**	0.87	**0.90**	0.87
	Accuracy	99.4	99.4	**99.5**	99.3	**99.5**	99.3
DLSP	*F-Measure*	0.88	0.85	0.88	0.80	0.89	0.86
	Accuracy	99.4	99.2	99.4	99.1	99.4	99.3

4.3e *Discussion*: In this section, we discussed how time series analysis and pattern analysis techniques can be used in tandem to detect anomalies at different layers in the TCP/IP stack. LP model is a very simple model that is able to capture differences in the network characteristics, adjust itself based on these characteristics, and estimates the future from the past. This prediction can be used to detect faults in the system. Moreover, features are extracted from the time series which are further used in classification techniques to make detection system more robust.

5. Anomaly detection at IP layer

5.1 *Experimental set-up*

Figure 2 shows the complete architecture used for simulation. Proxy server logs were utilised in order to perform a trace-driven emulation. A client was used to generate these requests to a web server, and the incoming and outgoing bytes were recorded at the client in each sampling interval in order to obtain a time series. The time series thus obtained for 50 random normal users was used to build the normal model.

Similarly, each attack (abusive usage, systematic downloading and DDoS) was emulated, and a time series recorded for each. For abusive usage of Internet access, 5 low bandwidth users were replaced with 5 random high bandwidth users and emulation was performed with the same configuration as the normal data simulation. For systematic downloading, one of the low bandwidth users was made as the attacker, and received and transmitted bytes were recorded at the interface. Various attack rates were emulated for testing our techniques. Similarly, abnormal data was obtained for DDoS attacks by attacking the Web-server and data recorded.

5.2 *Results*

Figure 19 shows the auto correlation function (ACF) of the differenced time series of the number of bytes at the interface. It is very clear from the plot that the differenced time series is stationary and thus it can be modelled using auto regressive (AR) modelling. Model order for prediction was obtained using the Partial ACF. Model parameters were then estimated, and predictions computed. The difference between the estimated and observed values was used as a cue for detection of various anomalies. For each experiment, false alarm rate and miss rate were calculated for varying thresholds on the difference between the estimated and observed values.

5.2a *Abusive usage and network layer*: Figure 20 shows False Alarm rate, Miss rate and ROC curve. A suitable threshold may be chosen so as to have a balance between the number of misses and false alarms.

5.2b *Systematic downloading and IP layer*: Various systematic downloading attacks were simulated and false alarm rate and miss rate were plotted as shown in figure 21. As compared to abusive usage, this detection mechanism should not fail to detect any attack even if it makes more false alarms as these will be filtered at higher layers.

5.2c *DDoS attack and IP layer*: FAR and MR plot for various rates of DOS attack. Similar to the case of systematic downloading, this detection mechanism also should not fail to detect any attack even if it makes more false alarms, as shown in figure 22 these will be filtered at higher layers.

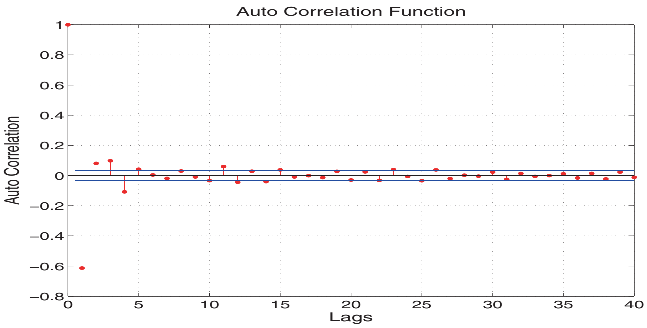
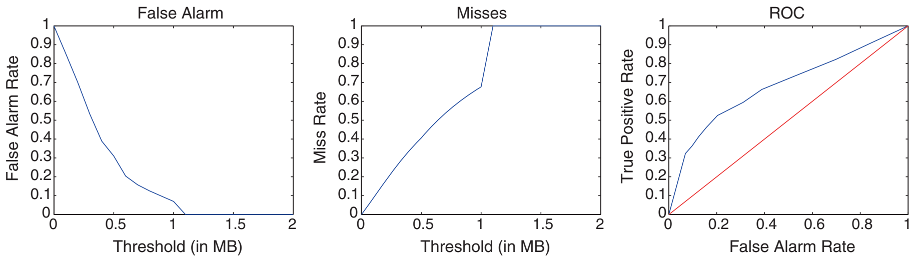
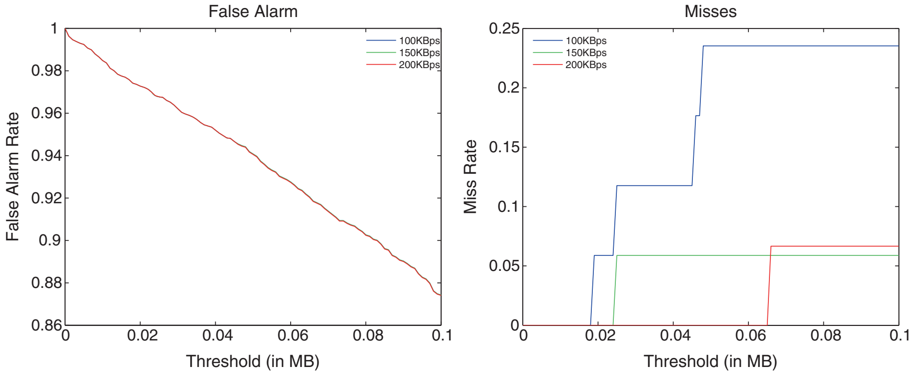


Figure 19. Auto correlation function for differenced time series.



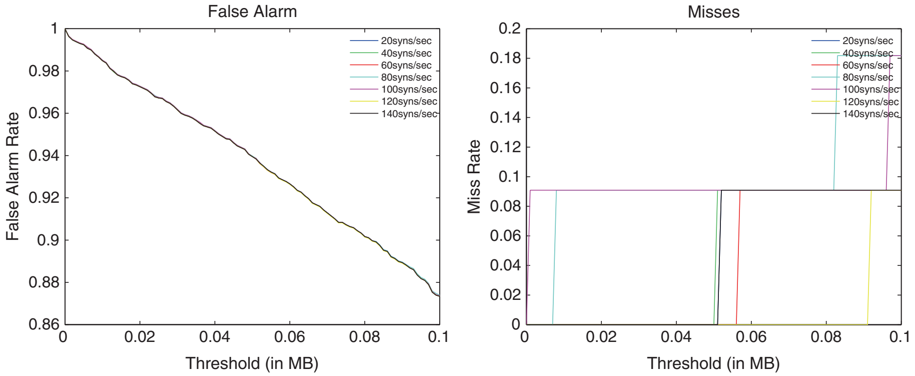
(a) False alarm rate **(b)** Miss rate **(c)** ROC curve

Figure 20. (**a**) The False alarm rate; (**b**) the miss rate and at different thresholds; (**c**) the receiver operating characteristics for detection of abusive Internet access.



(a) False alarm rate **(b)** Miss rate

Figure 21. (**a**) The false alarm rate and (**b**) the miss rate for detection of various rates systematic downloading at various thresholds.



(a) False alarm rate **(b)** Miss rate

Figure 22. (**a**) The false alarm rate and (**b**) the miss rate for detection of various rates DOS attack at various thresholds.

5.3 *Discussion*

In general, there are different thresholds for different types of intrusions. For the case of systematic downloading and DDoS attacks, the threshold has to be low so that the possibility of a miss is less. The threshold for abusive usage may be chosen so as to have a balance between the number of misses and false alarms.

6. Conclusion

In this paper, we have described anomaly detection techniques at different layers of TCP/IP stack — two internal threats (systematic downloading and abusive Internet access), and one external threat (DDoS attacks). We showed how anomalies in traffic (systematic downloading, DDoS attacks, traffic volumes) can be detected with good accuracy using time series analysis. We also showed how anomalies in Internet access may be detected accurately using machine learning. Further we described an innovative technique to classify URLs. All are applicable at the edge and may be incorporated into an NIDS.

We propose three scenarios in which these techniques may be combined at the NIDS. (i) Traffic volumes are modeled and analysed on an ongoing basis at the network layer, whereas detection of systematic downloads and DDoS attacks are only triggered when an anomaly is detected at the network layer, thereby saving system resources; essentially the traffic volume analyzer filters out the abnormal traffic. (ii) The IP address of a user requesting a download is known at the network layer; systematic downloads are detected at the application layer, and URLs may be classified as belonging to subscription websites at the application layer. Putting together the three pieces of information, the location of the user performing a systematic download may be known. (iii) Detection of abusive bandwdith usage along with URL classification leads to better user modeling and detection of abusive Internet access.

Acknowledgements

We thank Mr. A Ramasamy for contributing resources from his thesis for this work. This work was carried out under the IU-ATC project funded by the Department of Science and Technology (DST), Government of India and the UK EPSRC Digital Economy Programme.

References

Al-Nashif Y, Kumar A A, Hariri S, Qu G, Luo Y and Szidarovsky F 2008 Multi-level intrusion detection system. *In: International Conference on Autonomic Computing*

Arshadi L and Jahangir A-H 2011 Entropy based syn flooding detection. *In: Local Computer Networks (LCN), 2011 IEEE 36th Conference on*. IEEE

Baker G and Tenopir C 2006 Managing the unmanageable: Systematic downloading of electronic resources by library users. *J. Library Admin.* 44: 11–24

Berry M W, Browne M, Langville A N, Pauca V P and Plemmons R J 2007 Algorithms and applications for approximate nonnegative matrix factorization. *Comput. Stat. Data Anal.* 52(1): 155–173

Bhandari A, Khare S, Murthy H *et al* 2014 Systematic downloading: Analysis and detection. *In: Signal Processing and Communication Systems (ICSPCS), 2014 8th International Conference on*. IEEE

Blanco R and Lioma C 2007 Random walk term weighting for information retrieval. *In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM

Bommepally K, Glisa T, Prakash J, Singh S and Murthy H 2010 Internet activity analysis through proxy log. *In: National Conference on Communications (NCC)*

Canny J 1986 A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)* 8(6): 679–698

Chin S C, Ray A and Rajagopalan V 2005 Symbolic time series analysis for anomaly detection: A comparative evaluation. *Signal Process.* 85(9): 1859–1868. ISSN 0165-1684. URL http://www.sciencedirect.com/science/article/pii/S0165168405001039

Choi B and Yao Z 2005 Web page classification*. *In: Foundations and Advances in Data Mining*. Springer, 221–274

Chu S-I and Chang S-C 2007 Time-of-day internet-access management by combining empirical data-based pricing with quota-based priority control. *IET Commun.* 1: 587–596

Deerwester S, Dumais S T, Furnas G W, Landauer T K and Harshman R 1990 Indexing by latent semantic analysis. *J. Am. Soc. Inform. Sci.* 41: 391–407

Dini G, Fabio M, Saracino A and Sgandurra D 2012 MADAM: A multi-level anomaly detector for android malware. *Lecture Notes in Computer Science* 7531: 240–253

Divakaran D, Murthy H and Gonsalves T 2006a Detection of syn flooding attacks using linear prediction analysis. *In: Networks, 2006. ICON '06. 14th IEEE International Conference on*, volume 1. ISSN 1556–6463

Divakaran D M, Murthy H A and Gonsalves T A 2006b Detection of syn flooding attacks using linear prediction analysis. *In: Networks, 2006. ICON'06. 14th IEEE International Conference on*, volume 1. IEEE

Divakaran D M, Murthy H A and Gonsalves T A 2006c Detection of SYN flooding attacks using linear prediction analysis. *In: International Conference on Networks (ICON)*

Garcia-Teodoro P, Verdejo J D, Fernandez G M and Vazquez E 2009 Anomaly-based network intrusion detection: Techniques, systems and challenges. *Comput. Security* 28: 18–28

Guirguis M, Bestavros A, Matta I and Zhang Y 2005a Reduction of quality (roq) attacks on internet end-systems. *In: INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 2. IEEE

Guirguis M, Bestavros A, Matta I and Zhang Y 2005b Reduction of Quality (RoQ) Attacks on Internet End-Systems. *In: International Conference on Computer Communication (INFOCOM)*, volume 2

Hassan S, Mihalcea R and Banea C 2007 Random walk term weighting for improved text classification. *Int. J. Semantic Comput.* 1(04): 421–439

He Z and Liu Z 2008 A novel approach to naive bayes web page automatic classification. *In: Fuzzy Systems and Knowledge Discovery. 2008 FSKD '08. Fifth International Conference on*, volume 2

James C and Murthy H A 2012 Decoupling non-stationary and stationary components in long range network time series in the context of anomaly detection. *In: Local Computer Networks (LCN). 2012 IEEE 37th Conference on*. IEEE

Kan M-Y and Thi H O N 2005 Fast webpage classification using url features. *In: Proceedings of the 14th ACM international conference on Information and knowledge management*, CIKM '05. ACM. ISBN 1-59593-140-6

Khare S, Bhandari A and Murthy H A 2014 Url classification using non negative matrix factorization. *In: Communications (NCC), 2014 Twentieth National Conference on*. IEEE

Kumar A, Hegde M, Anand S, Bindu B, Thirumurthy D and Kherani A 2000 Nonintrusive TCP connection admission control for bandwidth management of an internet access link. 38: 160–167

Lee D D and Seung H S 2001 Algorithms for non-negative matrix factorization. *In: NIPS*. MIT Press

Leland W E, Taqqu M S, Willinger W and Wilson D V 1994 On the self-similar nature of Ethernet traffic (extended version). *IEEE/ACM Trans. Netw.* 2(1): 1–15

Lin T-C, Sun Y, Chang S-C, Chu S-I, Chou Y-T and Li M-W 2004 Management of abusive and unfair internet access by quota-based priority control. *Comput. Netw. Int. J. Comput. Telecommun. Netw.* 44: 441–462

Liu H and Kim M S 2010a Real-time detection of stealthy ddos attacks using time-series decomposition. *In: Communications (ICC), 2010 IEEE International Conference on*. ISSN. 1550–3607

Liu H and Kim M S 2010b Real-time detection of stealthy DDOS attacks using time-series decomposition. *In: International Conference on Communications (ICC)*

Mukherjee B, Heberlein L T and Levitt K N 1994 Network intrusion detection. *IEEE Netw.* 8: 26–41

Ndousse T and Okuda T 1996 Computational intelligence for distributed fault management in networks using fuzzy cognitive maps. *In: Communications, 1996. ICC '96, Conference Record, Converging Technologies for Tomorrow's Applications. 1996 IEEE International Conference on*, volume 3

Paine T A and Griggs T J 2008 Directing traffic: Managing internet bandwidth fairly. *EDUCAUSE Q.* 3: 66–70

Paliwal K 1992 On the use of line spectral frequency parameters for speech recognition. *Digital Signal Process.* 2(2): 80–87

Paxson V and Floyd S 1995 Wide- Area traffic: The failure of Poisson modeling. *IEEE/ACM Trans. Netw.* 3(3): 226–244

Peng H, Long F and Ding C 2005 Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. 27: 1226–1238

Porter M F 1980 An algorithm for suffix stripping. *Program: electronic library and information systems* 14(3): 130–137

Qi X and Davison B D 2009 Web page classification: Features and algorithms. *ACM Comput. Surv.* 41(2): 12:1–12:31. ISSN 0360-0300. URL http://doi.acm.org/10.1145/1459352.1459357

Rabiner L R and Gold B 1975 Theory and application of digital signal processing. Englewood Cliffs, NJ, Prentice-Hall, Inc., 1975, 777 p., 1

Ranjan N, Murthy H A and Gonsalves T A 2010 Detection of SYN flooding attacks using Generalized Autoregressive Conditional Heteroskedasticity (GARCH) modeling technique. *In: National Conference on Communications (NCC)*

Reynolds D, Quatieri T and Dunn R 2000 Speaker verification using adapted gaussian mixture models. *Digital Signal Process.* 10: 19–41

Salton G and Buckley C 1988 Term-weighting approaches in automatic text retrieval. *In: Information processing and management*

Seresht N A and Azmi R 2014 MAIS-IDS: A distributed intrusion detection system using multi-agent AIS approach. *Eng. Appl. Artif. Intell.* 35: 286–298

Singh S R, Murthy H A and Gonsalves T A 2010 Feature selection for text classification based on gini coefficient of inequality. *In:* H Liu, H Motoda, R Setiono and Z Zhao (eds.), FSDM, volume 10 of *JMLR Proceedings*. JMLR.org

Siris V A and Papagalou F 2004 Application of anomaly detection algorithms for detecting SYN flooding attacks. *In: Global Telecommunications Conference (GLOBECOM)*

Siris V A and Papagalou F 2006 Application of anomaly detection algorithms for detecting syn flooding attacks. *Comput. Commun.* 29(9): 1433–1442

Tax D M J and Duin R P W 2004 Support vector data description. *Mach. Learn.* 54(1): 45–66. ISSN 0885-6125. URL http://dx.doi.org/10.1023/B:MACH.0000008084.60811.49

TCPDUMP 1999 http://www.tcpdump.org/

Thottan M and Ji C 1998 Proactive anomaly detection using distributed intelligent agents. *Netw. IEEE* 12(5): 21–27

Thottan M and Ji C 2003 Anomaly detection in ip networks. *IEEE Trans. Signal Process.* 51(8): 2191–2204. ISSN 1053-587X

Wang H, Zhang D and Shin K 2002a SYN-dog: Sniffing SYN flooding sources. *In: Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS)*

Wang H, Zhang D and Shin K G 2002b Syn-dog: Sniffing syn flooding sources. *In: Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*. IEEE

Wu Q and Shao Z 2005 Network anomaly detection using time series analysis. *In: Autonomic and Autonomous Systems and International Conference on Networking and Services, 2005. ICAS-ICNS 2005. Joint International Conference on*

Ye N, Vilbert S and Chen Q 2003 Computer intrusion detection through ewma for autocorrelated and uncorrelated data. *Reliability, IEEE Transactions on* 52(1): 75–82