A Comparative Study of Anomaly Detection Algorithms for Detection of SIP Flooding in IMS

M. Ali Akbar, Zeeshan Tariq and Muddassar Farooq
Next Generation Intelligent Networks Research Center (nexGIN RC)
National University of Computer & Emerging Sciences (NUCES)
Islamabad, Pakistan
Email: {ali.akbar, zeeshan.tariq, muddassar.farooq}@nexginrc.org

Abstract—**The IP Multimedia Subsystem (IMS) framework uses Session Initiation Protocol (SIP) for signaling and control of sessions. In this paper, we first demonstrate that SIP flooding attacks on IMS can result in denial of service to the legitimate users. Afterwards, we report our comparative study of three well-known anomaly detection algorithms, Adaptive threshold, Cumulative sum, and Hellinger distance) for detection of flood attacks in IMS. We evaluate the accuracy of the algorithms using a comprehensive traffic dataset that consists of varying benign and malicious traffic patterns.**

I. Introduction

We are witnessing the integration of cellular networks and the Internet. The tremendous rise in popularity of IP telephony and an ever-increasing demand for novel Internet-based multimedia applications have resulted in a new all IP standard known as IMS. It is used for provisioning of multimedia services on fixed and mobile networks. IMS uses common Internet-based protocols to provide service control architecture for multimedia services [1]. IMS only deals with signaling and control of sessions and it is not responsible for transport of the multimedia traffic.

IMS uses SIP for signaling and control of multimedia sessions. Since IMS is based on SIP, its main elements are SIP proxies/servers, known in the IMS core as Call Service Control Functions (CSCF) [2]. The Proxy CSCF (P-CSCF) acts as a gateway for all SIP session requests from users. P-CSCF forwards the incoming SIP requests to the Serving CSCF which ultimately controls the session. The smooth and uninterrupted operation of CSCFs is vital for successful provision of services in the IMS framework. With an increase in the number of telecom operators which support IMS for service provisioning, the fear of assault from malicious hackers (illegitimate users or compromised legitimate users) has also increased. Such an attack may disrupt services and cause significant financial losses to the service operators and users.

SIP proxies/servers are vulnerable to a wide variety of attacks. The most common attack is the Denial of Service (DoS) attack. It can be launched in two ways: (1) In flooding attacks, a malicious user (or users) sends a large number of SIP messages that overload the SIP server[1] and this subsequently results in significant delays, and (2) malformed SIP requests are used to exploit vulnerabilities in SIP parser implementations to crash the server or force it to execute malicious code. We maintain the focus of this study on flooding attacks only.

In 2003, the 3rd Generation Partnership Project 2 (3GPP2) released a comprehensive security framework for IMS [3]. This framework addresses many vulnerabilities in the SIP protocol. But this framework does not specify any mechanism for the detection of flooding attacks. In IMS community, the anomaly detection algorithms remain relatively unexplored. The authors in [4] have suggested using a threshold on CPU usage to detect flooding attacks launched on IMS components. A malicious hacker can simply vary the patterns of attack traffic to circumvent such naive security mechanisms. A sophisticated anomaly detection algorithm can provide significantly better defense against such malicious attacks. The authors in [5], [6] have proposed anomaly detection schemes for detection of flooding attacks on general SIP based Voice over IP (VoIP) applications. In the last decade, web servers have faced similar Transport Control Protocol (TCP) based flooding attacks. Many anomaly detection algorithms have been explored for protecting the web servers against TCP `SYN` flooding attacks [7], [8].

In this paper, we customize some well known existing anomaly detection algorithms to provide security against SIP flooding attacks in IMS and evaluate their accuracy on a synthetic traffic data. The SIP traffic brings in some additional requirements for anomaly detection. The detection accuracy needs to be higher as even a relatively small rate of flood traffic can clog a SIP server. Moreover, the time-constrained nature of SIP traffic makes time-consuming computations infeasible. Therefore, the criteria for selection of three algorithms are based on high detection accuracy and low complexity. We choose *adaptive threshold* and *cumulative sum* which were originally proposed for providing security at transport layer in the Internet [7]. Both of these algorithms provided high accuracy in `SYN` flooding attacks on TCP [7]. We also choose *Hellinger distance* algorithm from the category of anomaly detection algorithms which were proposed for detection of VoIP flooding attacks [5]. The important contributions of this work are listed below.

1) Adaptation of well known anomaly detection algorithms, Adaptive threshold and Cumulative sum, from transport

[1] In this study, we do not distinguish spoofed flood attacks because three anomaly detection algorithms selected for our study perform their analysis only on aggregate values of the features.

layer to application layer.

2) Generation of comprehensive traffic datasets to evaluate these algorithms under malicious traffic datasets.
3) Comparing accuracy of three algorithms for SIP flood detection in IMS using our traffic datasets. We discuss the effect of design parameters of algorithms on their detection accuracy.

II. Related Work

Security in IMS has received little attention [9]. Some security frameworks (such as [4], [9] and [10]) have been proposed to detect flooding attacks in IMS. The authors of [4] proposed detection of flooding attacks by monitoring the CPU usage of the IMS components. A careful hacker can evade this security measure by crafting an attack that keeps the CPU usage below the threshold. The authors of [9] have proposed an artificial immune system based algorithm for detection of flood attacks on IMS. The authors have compared their framework's performance with a signature-based algorithm. The authors of [5] and [6] have used anomaly detection algorithms for security in VoIP networks. The authors of [5] have proposed a SIP-based flooding attack detection using Hellinger distance. The experimental results are very promising. However, the experiments are limited to detection accuracy against variation in flood attack intensity only. The authors of [6] proposed an Application Layer Attack Sensor (ALAS), which is able to detect SIP flooding attacks with high accuracy. In [7], adaptive threshold and cumulative sum algorithms have been applied for detection of TCP SYN flood attacks.

III. Anomaly Detection Algorithms

In this paper, we compare accuracy of three well-known anomaly detection algorithms: Adaptive threshold, Cumulative sum and Hellinger distance, utilized to provide protection against flooding attacks in IMS. Malicious users can launch these attacks on P-CSCF by sending bursts of INVITE packets. We define accuracy of an anomaly detection algorithm in terms of detection rate and false positive rate. We now provide a brief overview of three algorithms that will help the reader in understanding our enhancements and adaptations for providing security in P-CSCF.

1) Adaptive Threshold: In this algorithm, we simply calculate moving average of a given feature in a predefined time-window. We calculate current value of $\bar{\mu}_n$ as follows [7]:

$$\bar{\mu}_n = \beta\bar{\mu}_{n-1} + (1-\beta)x_n, \tag{1}$$

where x_n is the value of feature in time-window n, $\bar{\mu}_n$ and $\bar{\mu}_{n-1}$ are the moving averages in time-windows n and $n-1$ respectively, and β is the weight assigned to past and current moving averages. Now the adaptive threshold is $(\alpha+1)\bar{\mu}_{n-1}$, where $\alpha > 0$ is the percentage above moving average that raises an initial suspicion about a malicious behavior. If this suspicion is raised for k consecutive time-windows then an alarm about the malicious activity is raised. Mathematically, we represent it as [7]:

$$\sum_{i=n-k+1}^{n} 1_{\{x_i \geq (\alpha+1)\bar{\mu}_{i-1}\}} \geq k \tag{2}$$

At the transport layer, the feature is the number of SYN packets in a fixed time-window. At the application layer, we map the number of SYN packets with the number of INVITE packets. We took x_n as the number of INVITE packets arrived. The time-window is of 10 seconds. Moreover the design parameters of the algorithm, α and β at the transport layer are to be mapped on the application layer. We empirically evaluated the algorithm on SIP traffic and then fine tuned the values of α and β to 0.3 and 0.8 respectively.

2) Cumulative Sum: Cumulative sum (CUSUM) is an algorithm that is used in data mining for detection of change in a statistical distribution between two hypotheses. It detects a change by computing the difference, g_n, between log-likelihood ratio, S_n, of feature values for the two hypothesis and its current minimum value m_n. We used a simple method to compute this difference proposed in [7]. The moving average of a feature in a time-window is computed using the adaptive threshold. The simplified iterative formula [7] is

$$g_n = [g_{n-1} + \frac{\alpha\bar{\mu}_{n-1}}{\sigma^2}(x_n - \bar{\mu}_{n-1} - \frac{\alpha\bar{\mu}_{n-1}}{2})]^+, \tag{3}$$

where g_n and g_{n-1} are the estimated differences in time-windows n and $n-1$, α is the percentage above moving average for suspicious activity, $\bar{\mu}_{n-1}$ is the moving average for time-window $n-1$, σ^2 is the variance of the feature, and x_n is the current value of the feature. If g_n exceeds the given threshold value h, an alarm is raised.

The adaptation of CUSUM algorithm from transport layer to application layer was done similar to the adaptation of Adaptive threshold algorithm. The feature was mapped from the number of SYN packets to the number of INVITE packets. We took x_n as the number of INVITE packets arrived. The time-window is of 10 seconds. The values of design parameters of algorithm were mapped to the application layer. We empirically evaluated the algorithm on SIP traffic and then fine tuned the values of α, β and σ^2 to 0.4, 0.7 and 100 respectively.

3) Hellinger Distance: The Hellinger distance (HD) is "a metric that quantifies the deviation between two probability measures" [11].

The authors of [5] have used HD to detect anomalies in SIP. They took four attributes of SIP which are the number of INVITE, 200 OK, ACK and BYE packets arrived in a predefined time-window. The algorithm consists of training and testing phases. In the training phase, the normalized frequencies $p_{\texttt{INVITE}}$, $p_{\texttt{200OK}}$, $p_{\texttt{ACK}}$, $p_{\texttt{BYE}}$ for INVITE, 200 OK, ACK and BYE respectively are calculated over the training dataset. Similarly, the normalized frequencies $q_{\texttt{INVITE}}$, $q_{\texttt{200OK}}$, $q_{\texttt{ACK}}$, $q_{\texttt{BYE}}$ are calculated in the testing phase for each time-window n. The HD between these frequency distributions of two phases is:

$$\begin{aligned}\mathrm{HD} = \;& (\sqrt{p_{\mathtt{INVITE}}} - \sqrt{q_{\mathtt{INVITE}}})^2 + (\sqrt{p_{\mathtt{200OK}}} - \sqrt{q_{\mathtt{200OK}}})^2 \\ & + (\sqrt{p_{\mathtt{ACK}}} - \sqrt{q_{\mathtt{ACK}}})^2 + (\sqrt{p_{\mathtt{BYE}}} - \sqrt{q_{\mathtt{BYE}}})^2\end{aligned}$$

The threshold value is a function of the average of observed HDs and their mean deviation [5]. The authors in [5] have described a mechanism for determining the threshold dynamically. If an observed HD is greater than this threshold value then an alarm is raised. The feature values are calculated in a time-window of 10 sec. The training dataset has a duration of 120 seconds.

IV. Performance Evaluation

In this section, we describe our strategy for performance evaluation and comparison of the anomaly detection algorithms. First we define the metrics used for comparing performance. Then we describe the different traffic patterns used by us to study the impact of different design options of algorithms on their performance. After this, we discuss in detail different traffic sets with varying traffic patterns, generated for our experiments. Finally, we illustrate how we injected malicious traffic into benign traffic sets.

A. Performance Metrics

In a comparative study, the metrics used for comparison of algorithms must be carefully selected. In practice, the performance of anomaly detection schemes is mostly reported in terms of *detection rate* (DR) and *false alarms rate* (FAR). In the context of this study, we define *detection rate* as the fraction of anomalous traffic that was successfully detected. Similarly, we define *false alarms rate* as the fraction of benign traffic that was incorrectly labeled as anomalous. In our case, a malicious user launches flooding attack by exploiting hardware or software vulnerabilities in our VoIP user agent in a soft phone. Our objective is to identify an anomaly detection algorithm with high detection rate and low false alarms rate.

B. Generation of Traffic Sets

We did not find a publicly available traffic dataset of a real world SIP server. The important reason is the privacy concern of the users. Therefore, we were left with no option but to synthetically generate traffic data.

The SIP traffic was generated using an open source *SIPp* [12] tool. It can generate SIP messages and has the ability to simulate customized SIP scenarios. The scenarios are described in eXtensible Markup Language (XML) syntax. SIPp gives a user complete control of the simulated scenarios at a predefined UDP port. The basic SipStone [13] SIP server and client scenarios are distributed with SIPp. We modified these scenarios according to our requirements.

The SIP server and client scenarios simulate a typical SIP call flow with INVITE, 180 RINGING, 200 OK, ACK, BYE and ACK packets as shown in Fig. 1. The call rate is controlled at runtime by sending commands to SIPp at the predefined UDP port. The call length is fixed at 60 seconds. Random packet loss is also simulated in the scenario. The traffic traces are collected using Wireshark [14].

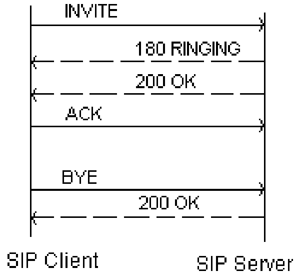


Fig. 1. Typical SIP call setup flow [13]

TABLE I
Characteristics of Traffic Sets

Traffic	Type	Duration	Mean	Distribution
LLS	Normal	60 min	500 calls/min	normal
MLS	Normal	60 min	750 calls/min	normal
HLS	Normal	60 min	1000 calls/min	normal
VLHA	Attack	1 min	25 calls/sec	constant
LHA	Attack	1 min	50 calls/sec	constant
MHA	Attack	1 min	100 calls/sec	constant
HHA	Attack	1 min	300 calls/sec	constant
VHHA	Attack	1 min	500 calls/sec	constant
VLCA	Attack	10 min	25 calls/sec	constant
LCA	Attack	10 min	50 calls/sec	constant
MCA	Attack	10 min	100 calls/sec	constant
HCA	Attack	10 min	300 calls/sec	constant
VHCA	Attack	10 min	500 calls/sec	constant

C. Experimental Characteristics

We analyze the performance of three algorithms in three scenarios: (1) varying the average traffic load on P-CSCF, (2) varying the intensity of attack, and (3) varying the duration of attack.

The traffic characteristics of different scenarios are tabulated in Table I. Normal scenarios for SIP servers serving three different benign traffic loads are simulated. These are labeled as Low Load Server (LLS), Medium Load Server (MLS) and High Load Server (HLS) in Table I. The traffic load is modeled as a normal distribution. The mean of this distribution represents the average traffic load on the corresponding P-CSCF. The mean traffic loads are 500 calls/min, 750 calls/min and 1000 calls/min for LLS, MLS and HLS respectively. Fig. 2 shows normally distributed traffic loads of LLS, MLS and HLS in an hour.

In Table I, harmonic attack traffics are labeled with suffix 'HA' and are for one minute each. VLHA is the Very Low intensity Harmonic Attack traffic with a constant traffic load of 25 calls/sec and duration of 1 minute. Similarly, LHA stands for Low intensity Harmonic Attack traffic, MHA for Medium intensity Harmonic Attack traffic, HHA for High intensity Attack traffic and VHHA for Very High intensity Harmonic Attack traffic. Similar convention has been followed for labeling prolonged (chunk) attacks. The prolonged attacks are labeled with suffix 'CA' and have duration of ten minutes. VLCA stands for Very Low intensity Chunk Attack traffic, LCA for Low intensity Chunk Attack traffic and so on.

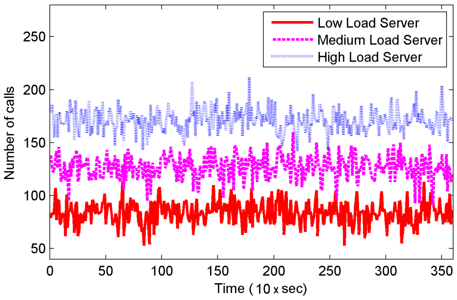
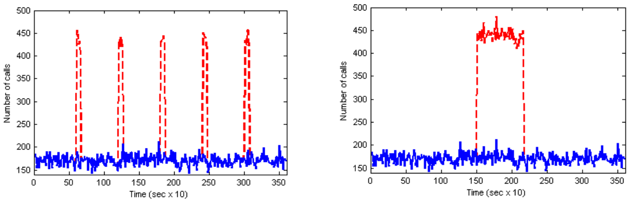


Fig. 2. Normal traffic load



(a) Harmonic attack

(b) Chunk attack

Fig. 3. Anomaly injection - dashed curves represent attack traffic

D. Anomaly Injection

We inject attack traffic into benign traffic to generate our final dataset. The harmonic attack traffic is injected at multiple locations in the benign traffic to mimic the behavior of an attacker who launches flood attacks of a short duration but at a high frequency. The prolonged attack traffic is injected at a single location in the benign traffic so that it models an attacker who launches prolonged flood attacks but at a low frequency. In Fig. 3, one can see a dataset in which both harmonic and prolonged attacks were launched. The solid curves represent the benign traffic while the dashed curves represent the injected attack traffic. The total collection period of dataset with embedded anomalies is one hour. The datasets for all possible combinations of benign and malicious traffic given in Table I have been generated. This resulted in 30 different test scenarios for our experiments.

For all datasets, we calculate the number of `INVITE`, `ACK`, `200 OK` and `BYE` packets sent or received in a time interval of 10 seconds. As a result, we can generate histograms of each feature and afterwards different classification algorithms are applied on each histogram for an appropriate range of threshold values. Each algorithm's detection rate and false alarms rate are calculated for each dataset. The results of these experiments are described in detail in the next section.

V. Results

To detect flooding attacks, the anomaly detection algorithms measure the deviation of selected features from their benign values. In the context of P-CSCF, the average traffic load defines its benign behavior. The adaptive threshold algorithm sets its threshold using $(\alpha + 1)\bar{\mu}_{n-1}$ (see section III), therefore, slight change in traffic volume is expected to have a large impact on estimated mean value. Consequently, adaptive threshold algorithm results in significantly higher false alarm rate under high traffic load. In comparison, cumulative sum detects the point of change in feature space and hellinger distance measures deviation from mean feature value; therefore, these schemes have no impact with an increase in traffic load.

The intensity of attack significantly alters the deviation from normal behavior. Therefore, we expect that increase in attack intensity should make it easier for all algorithms to detect the attack. In other words, the high intensity attacks should have higher detection rate and lower false alarms rate compared with medium intensity attacks. Similarly, medium intensity attacks should have higher detection rate with lower false alarms rate as compared with low intensity attacks.

Adaptive threshold and cumulative sum algorithms update their notion of benign behavior from dataset, as a result, prolonged attacks can mislead them to detect anomalous behavior as normal. The adaptive threshold algorithm sets its threshold directly proportional to the mean value of the feature, we expect its performance for prolonged attacks to be relatively poor. The cumulative sum is a 'point of change detection' algorithm, therefore, it will suffer less deterioration in performance for prolonged attacks. Hellinger distance scheme requires initial training on normal data and does not change its sense of normal behavior on the basis of testing dataset; therefore, we expect it to have very slight performance change with variation in attack duration and frequency.

The experimental results are summarized in Table II. Figures 4, 5, 6 and 7 show the receiver operating characteristic (ROC) curves for the three algorithms for varying attack intensity, attack duration and normal traffic load. ROC curves display the trade-off between false alarm rate and detection rate. Now we discuss the experimental results in detail.

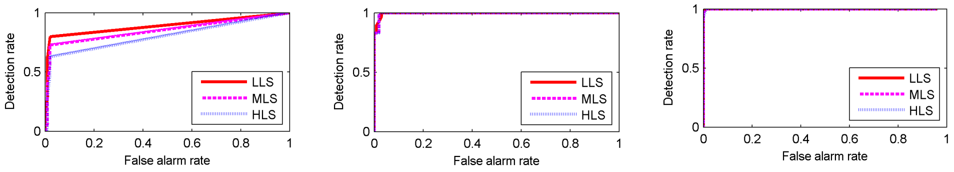
A. Adaptive Threshold

1) Effect of variation in normal traffic load: Fig. 4 shows the variation in performance when normal traffic load of server is changed. As expected, the performance of adaptive threshold algorithm degrades with increase in normal traffic load.

2) Effect of variation in attack intensity: The performance of adaptive algorithm varies significantly with variation in attack intensity. As expected, high intensity attacks are relatively accurately detected with less false alarms than that of low intensity attacks. From Table II, we can see that for a false alarm limit of 2%, the detection rate for very low intensity harmonic attack (25 calls/sec) is 77% while very high intensity harmonic attack (500 calls/sec) has a 100% detection rate. The variation in attack intensity for chunk attacks produces similar results. The algorithm's detection rate increases from 7% to 28% when attack intensity is increased from 25 calls/sec to 500 calls/sec. Fig. 5 clearly shows that the trade-off between detection rate and false alarm rate improves with increase in attack intensity for both harmonic and chunk attacks.

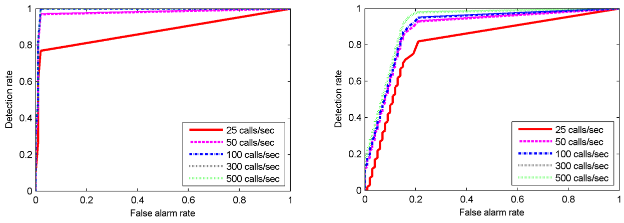
TABLE II
DETECTION RATE (IN %) FOR VARIOUS ATTACK INTENSITIES AND DURATIONS - RESULTS ENCLOSED IN PARENTHESIS ARE OBTAINED BEFORE RE-TUNING OF ALGORITHMIC PARAMETERS

Algorithm	FAR	VHHA	HHA	MHA	LHA	VLHA	VHCA	HCA	MCA	LCA	VLCA
Adaptive threshold	0	33	33	33	30	10	18(5)	18(5)	13(3)	12(3)	0(0)
	2	100	100	100	97	77	28(12)	28(12)	23(10)	22(10)	7(7)
	5	100	100	100	100	85	43(14)	43(14)	38(12)	37(12)	22(10)
	10	100	100	100	100	100	68(17)	65(17)	63(14)	62(14)	47(12)
Cumulative sum	0	67	67	83	83	83	0(18)	0(18)	0(13)	10(12)	28(12)
	2	83	83	83	83	93	33(28)	33(28)	38(23)	45(22)	58(52)
	5	83	83	100	100	100	87(43)	88(43)	92(38)	97(37)	97(85)
	10	100	100	100	100	100	93(68)	95(68)	97(63)	98(62)	98(90)
Hellinger distance	0	100	100	100	100	97	28	28	27	23	5
	2	100	100	100	100	100	100	100	100	100	98
	5	100	100	100	100	100	100	100	100	100	100
	10	100	100	100	100	100	100	100	100	100	100



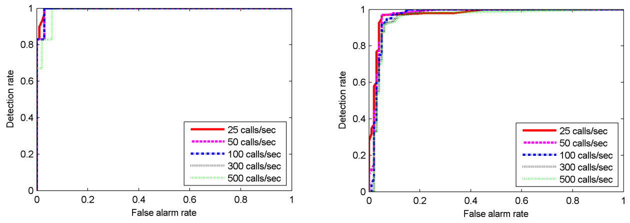
(a) Adaptive threshold (b) Cumulative sum (c) Hellinger distance

Fig. 4. ROC curves for different normal traffic loads



(a) Harmonic attacks (b) Chunk attacks

Fig. 5. ROC curves for adaptive threshold

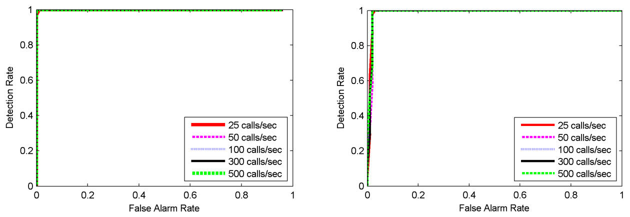


(a) Harmonic attacks (b) Chunk attacks

Fig. 6. ROC curves for cumulative sum

3) Effect of variation in attack duration: The adaptive threshold algorithm performs better for harmonic attacks as compared to chunk attacks. At a cost of 2% false alarms rate, its detection accuracy for medium intensity harmonic attack is 100%. Whereas, in case of chunk attacks, for the same false alarms rate, its detection rate is only 10%. We tried to improve this performance by changing the parameters (α and β), but the detection rate could not be improved beyond 23%.

As the parameters are constant at run-time, the algorithm can not provide its best performance for both harmonic and chunk attacks at the same time. Figure 5 shows the best performance of adaptive algorithm for both harmonic and chunk attacks. The graph clearly indicates the superior performance of adaptive algorithm for harmonic attacks as compared to chunk attacks. The graphs have been obtained by using different parameters for the algorithm.



(a) Harmonic attacks (b) Chunk attacks

Fig. 7. ROC curves for hellinger distance

B. Cumulative Sum

1) Effect of variation in normal traffic load: Fig. 4 shows that the performance of cumulative sum algorithm is independent of normal traffic load.

2) Effect of variation in attack intensity: As expected, cumulative sum shows variation in performance against variation in attack intensity. However, quite surprisingly, *cumulative sum shows better performance against low intensity attacks as compared to high intensity attacks*. According to Table II, for 5% false alarm rate, cumulative sum has a detection rate of 100% for very low intensity harmonic attacks as compared to 83% detection rate for very high intensity harmonic attacks. Similar trend is obvious for chunk attacks too. We observe that for 5% false alarm rate, cumulative sum has a detection rate of 97% for very low intensity chunk attacks as compared to 87% detection rate for very high intensity chunk attacks. A close look at Fig. 6 confirms that cumulative sum has slightly better performance for low intensity attacks as compared to high intensity attacks.

The reason for this unusual behavior is that cumulative sum tries to detect the negative drift in feature to mark beginning of attack and positive drift for marking end of attack. A sudden change in drift cannot be accurately followed as the drift detection function has been modeled linearly. This leads to relatively large number of false alarms. As the intensity of attack increases, the slope becomes steeper (greater drift); hence the number of false alarms is increased.

3) Effect of variation in attack duration: The performance of cumulative sum algorithm is significantly better under harmonic attacks compared with prolonged attacks. A comparison between harmonic and chunk attacks in Table II reveals that the detection rate for cumulative sum falls from 100% to 38% for medium intensity attack and 5% false alarms rate. But by re-tuning of algorithmic parameters, the detection rate increases to 92%. This is an indication that the algorithm cannot perform optimally both for harmonic and chunk attacks for same set of parameter values. Fig. 6 shows the difference in performance of the algorithm in case of harmonic and chunk attacks. The graphs are obtained for different sets of parameter values of the algorithm. If the parameters are kept constant, the algorithm's performance is significantly degraded for chunk attacks (see values enclosed by parentheses in Table II).

C. Hellinger Distance

1) Effect of variation in normal traffic load: Fig. 4 shows that there is no variation in performance of Hellinger distance once we change the benign traffic load of the server.

2) Effect of variation in attack intensity: It is evident from Table II that for 0% false alarm rate, the detection rate is 97% for very low intensity harmonic attack and 100% for very high intensity harmonic attack. We observe that for 2% false alarms rate, the detection rate is 98% for very low intensity chunk attack and 100% for very high intensity chunk attack. Fig. 7 shows that the curves for different attack intensities overlap. Thus, Hellinger distance algorithm is robust to variations in attack intensity.

3) Effect of variation in attack duration: Hellinger distance algorithm significantly outperforms adaptive rate threshold and cumulative sum but does not require re-tuning of algorithmic parameters. Table II shows that for a false alarms rate of 2%, hellinger distance algorithm has a detection rate of 100% for very low intensity harmonic attack and 98% for very low intensity chunk attack. Fig. 7 shows that difference in performance is very small between harmonic and chunk attacks. Both graphs use same set of algorithmic parameters.

VI. Conclusion & Future Work

In this paper, we compared three well-known anomaly detection algorithms and evaluated their detection accuracy for malicious traffic datasets. The experiments were designed to show the effect of design parameters on the detection accuracy of algorithms for different attack patterns. From the results, we can see that Hellinger distance algorithm outperforms Adaptive threshold and Cumulative sum. It has a better detection accuracy and does not require retuning of its parameters. It is robust to variations in benign and attack traffic patterns.

In future work, we plan to include the DoS attacks using malformed SIP packets in our study. We also plan to include more anomaly detection algorithms in our study and perform experiments on real world IMS traffic dataset.

Acknowledgments

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

References

[1] Poikeselka, Mayer, Khartabil, and Niemi, *The IMS IP Multimedia Concepts and Services*, 2nd ed. John Wiley & Sons, Ltd., 2006.

[2] A. Cuevas, J. Moreno, P. Vidales, and H. Einsiedler, "The IMS Service Platform: A Solution for Next-Generation Network Operators to Be More than Bit Pipes," *IEEE Comm. Mag.*, pp. 75–81, August 2006.

[3] 3rd Generation Partnership Project 2 (3GPP2), "IMS Security Framework," *http://www.3gpp2.org*, December 2003.

[4] M. Sher and T. Magedanz, "Secure Service Provisioning Framework (SSPF) for IP Multimedia System and Next Generation Mobile Networks," *IWWST'05 Proceedings*, pp. 101–106, April 2005.

[5] H. Sengar, H. Wang, D. Wijesekera, and S. Jajodia, "Detecting VoIP Floods using the Hellinger Distance," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 6, pp. 794–805, June 2008.

[6] B. Reynolds and D. Ghosal, "Secure IP Telephony Using Multi-Layered Protection," *Proc. Net. and Distributed Sys. Sec. Symp.*, Feb 2003.

[7] V. Siris and F. Papagalou, "Application of Anomaly Detection Algorithms for Detecting SYN Flooding Attacks," *Computer Communications*, vol. 29, no. 9, pp. 1433–1442, 2006.

[8] H. Wang, D. Zhang, and K. Shin, "Detecting SYN flooding attacks," *INFOCOM 2002. Twenty-First Annual Joint Conf. of the IEEE Comp. and Comm. Soc. Proceedings. IEEE*, vol. 3, 2002.

[9] A. Awais, M. Farooq, and M. Javed, "Attack analysis & bio-inspired security framework for IP Multimedia subsystem," *Proceedings of the 2008 GECCO conference companion on Genetic and evolutionary computation*, pp. 2093–2098, 2008.

[10] Y. Rebahi, M. Sher, and T. Magedanz, "Detecting flooding attacks against IP Multimedia Subsystem (IMS) networks," *IEEE/ACS Intl. Conf. on Comp. Sys. and App., 2008.*, pp. 848–851, 2008.

[11] D. Pollard, *Asymptopia*, 1st ed. http://www.stat.yale.edu/∼pollard/, 2000.

[12] R. Gayraud and O. Jacques, "SIPp," 2007, http://sipp.sourceforge.net.

[13] H. Schulzrinne, S. Narayanan, J. Lennox, and M. Doyle, "SIPstone-benchmarking SIP server performance," *Columbia University*, 2002.

[14] G. Combs, "Wireshark," 2008, http://www.wireshark.org.