A Technique for Detecting New Attacks in Low-Interaction Honeypot Traffic

S. Almotairi, A. Clark, G. Mohay, and J. Zimmermann
Information Security Institute, Queensland University of Technology
Brisbane, Queensland, Australia
{s.almotairi, a.clark, g.mohay, j.zimmerm}@isi.qut.edu.au

***Abstract*—Honeypots are flexible security tools for gathering artefacts associated with a variety of Internet attack activities. While existing work on honeypot traffic analysis focuses mainly on identifying existing attacks, this paper describes a technique for detecting new attacks based on principal component analysis. The proposed technique requires no prior knowledge of attack types and has low computational requirements that makes it suitable for online detection systems. Our method of detecting new attacks is based on measuring changes in the residual space using square prediction error (SPE) statistics. When attack vectors are projected onto the residual space, attacks that are not presented by the main hyperspace will create new directions with high SPE values. We demonstrate the usefulness of our technique by using real traffic data from the Leurré.com project, a world-wide deployment of low-interaction honeypots, where several examples of new traffic detected by the system are illustrated.**

I. Introduction

Monitoring and characterizing Internet threats is very critical in order to better protect production systems by gaining an understanding of how attacks work, and consequently protecting systems from them. A honeypot is a security resource whose value lies in being probed, attacked, or compromised [1]. Honeypots are very valuable for collecting different types of attack traffic. However, characterizing attackers' activities present in honeypot traffic data can be challenging due to the high dimensionality of the data, (or large number of variables,) and the large volumes of traffic data collected. The large amount of background noise, such as scans and backscatter, adds to the challenge by hiding interesting abnormal activities that require immediate attention from security personnel. Detecting these activities can potentially be of high value and give early signs of new vulnerabilities or breakouts of new automated malicious codes, such as worms, if the honeypot data is handled in time.

Principal component analysis (PCA) is a widely used multivariate statistical technique for reducing the dimensionality of variables and unveiling latent structures and detecting outliers in data sets [2], [3]. This paper presents a technique for detecting new attacks in low-interaction honeypot traffic using PCA. Our method of detecting attacks draws its roots from anomaly intrusion detection, through building a model of honeypot profile, and multivariate statistical technique capabilities, namely principal component analysis, in detecting different types of outliers. New observations are projected onto the residuals' space of the least significant components and their distances from the k-dimensional hyperspace defined by the PCA model are measured using the square prediction error (SPE) statistic. A higher value of SPE indicates that the new observation represents a new direction that has not been captured by the PCA model of attacks seen in the historical honeypot traffic.

The rest of the paper is organized as follows. Section II overviews related work. Section III provides a brief summary of principal component analysis. The dataset used in this study and the pre-processing are described in Section IV. The detection model and the process of applying PCA to the pre-processed honeypot data are presented in Section V. The results and the evaluation of the detection technique is discussed in Section VI. Finally, the paper is concluded in Section VII.

II. Related work

The application of PCA to computer network traffic falls roughly in three categories: detecting the latent structure, reducing the dimensionality of the data, and identifying anomalies. A number of researchers have used principal component analysis (PCA) to reduce the dimensionality of variables and to identify attacks. Labib et al. [4] utilized PCA in reducing the dimensionality of the traffic data and visualizing and identifying attacks. Bouzida et al. [5] presented a performance study of two machine learning algorithms, namely nearest neighbours and decision trees, when used with traffic data with or without PCA.

Previous research in low-interaction honeypot include designing a low-interaction honeypot daemon [6], improving the interactivities of the honeypot [7], and generating intrusion detection signatures of unknown attacks automatically [8]. Traffic analysis of low-interaction honeypots include the manual clustering of traffic that shares similar activity fingerprints, with the port sequence as a main clustering feature [9], the use of packet inter-arrival times for investigating unsolicited internet traffic [10], and identification of repeated use of attack tools and attack processes [11].

The use of PCA to structure network traffic flow was introduced by Lakhina [12] where PCA is used to decompose the structure of Origin-Destination flows, from two backbone networks, into three main constituents, namely periodic trends, bursts and noise. In our previous work [13], we have applied principal component analysis (PCA) to traffic flows of low-interaction honeypots to detect the structure of attackers' activities and to break honeypot traffic into seven dominant clusters. Shyu et al. [14] proposed an anomaly detection scheme based on robust principal component analysis. Two classifiers were implemented to detect anomalies, one was based on the major components that capture most of the variation in the data and the second was based on the minor components or the residuals. A new observation is considered anomalous when the sum of squares of the weighted principal components exceeds a threshold in any of the two classifiers.

Lakhina et al. [15] applied the principal component analysis technique to Origin-Destination (OD) flow traffic counts of link data bytes. Anomaly is flagged if the value of the square prediction error exceeds a predefined limit. The subspace method was extended by the same authors [16] to detect anomalies in multivariate time series of OD flow traffic with three features (number of bytes, number of packets, and number of flows).

Finally, previous applications of PCA treat the network traffic as either normal or anomalous, and then the detection model is built on what is believed normal. The notion of normal and anomalous does not apply in honeypot traffic where all traffic is potentially malicious. Thus, our technique is fundamentally different from the above techniques in the following ways: traffic features are extracted from aggregated flows, where standard flows from a single IP address are grouped together to provide sufficient information of attack patterns; secondly, PCA is used to build a model of existing attacks that has been seen in the past rather than normal behaviors, and then any large deviation from the attack model is considered either a new attack vector or an attack that is not present in the model. To the best of our knowledge this is the first time PCA has been used to detect new attacks using honeypot traffic. As we will show, the technique shows much promise and pave the way for further applications of the technique.

III. Principal component analysis

Principal component analysis (PCA) is a multivariate statistical technique that has been widely used in multi-disciplinary research areas such as Internet traffic analysis, economics, image processing, and genetics. PCA is mainly used to reduce the dimensionality of a data set into a few uncorrelated variables, principal components (PCs), which retain most of the variation in the original data. The resulting principal components are a linear combination of the original variables, are orthogonal, and ordered with the first principal component having the largest variance. Although the number of resulting principal components is equal to the number of original variables, much of the variance in the original set of p variables can be retained by the first k PCs, where $k < p$. Thus, the original p variables can be replaced by the new k principal components.

Given the p-dimensional random variables $X = (X_1, .., X_p)^T$ with a sample mean $\bar{X}$ and a sample covariance matrix R, we seek to find a lower dimension vector $A = (A_1, .., A_k)^T$of R that has the maximum variance of the original data with all the Eigenvalues l being greater than zero. Thus, the first linear function Z_1 of X having maximum variance, the second linear function Z_2 is uncorrelated with Z_1 and having the second largest variance and so on until the k^{th} function Z_k is found which is uncorrelated with Z_1,...,Z_{k-1} (see Eq. 1)

$$\begin{aligned} Z_1 &= a_{11}X_1 + a_{12}X_2 + \cdots + a_{1p}X_p \\ &\vdots \\ Z_k &= a_{k1}X_1 + a_{k2}X_2 + \cdots + a_{kp}X_p \end{aligned} \tag{1}$$

For the interested reader, a full discussion of principal component analysis can be found in [2], [3].

IV. Dataset and pre-processing

A. Dataset

The honeypot traffic data used in this analysis comes from the Leurré.com project [17]. The Leurré.com project was launched in 2004 for collecting malicious traffic using globally distributed, identical honeypot environments; currently 50 platforms are deployed in 30 different countries. The Leurré.com honeypot sensor is based on the open source low-interaction honeypot *honeyd* [6]. Each sensor runs on a single host and emulates three operating systems at the same time (on different IP addresses): Windows 2003 Professional; Windows 2003 Server; and Linux Red Hat. For the purpose of this study, only one low-interaction honeypot sensor's data is used due to the ready availability of log files from that sensor. Traffic data for the period of September 15 until November 30, 2007 for two of the honeypot environments was included, namely Windows 2003 Professional and Windows 2003 Server. Both environments are identical in terms of open TCP and UDP ports. Two data sets of traffic data were extracted for the purpose of this study. Traffic data sets that have been used in this study are: Data set I for constructing the PCA model; Data set II for evaluating the model. Table I gives a brief summary of the used data sets.

Table I
Summary of the data sets used in the study

Data Set	Start Date	End Date	Packets	Standard Flows	Activity Flows
I	15/09/2007	30/11/2007	839663	562470	5401
II	01/12/2007	31/03/2008	2231245	1586715	7343

B. Pre-processing

Before applying the PCA to the traffic data, the following steps were performed to process the raw traffic data. First, raw tcpdump files of daily honeypot data were collected and merged into a single traffic file. Then packets were grouped together into basic flows (according to the notation of flow). Our basic flow conforms to the standard definition of an IP flow of packets that share the five keys: source IP address, destination IP address, source port, destination port, and protocol type. If a packet differs from another packet by any key field, it is considered to belong to another flow [18].

Other features associated with flows were also extracted to enrich the analysis. These features include number of packets, number of bytes, total activities, and durations. For the purpose of this study, we set the timeout of basic flows to a maximum of five minutes. The five-minute timeout parameter was selected based on our experiments and the nature of low-interaction honeypots where the majority of flows were less than 300 seconds; a higher value of time out has little influence in the final results.

The second step was to group the basic flows again into what we call activity flows, where the newly generated flows were combined based upon the source IP address of the attacker with a maximum of sixty minutes inter-arrival time between basic flows. Finally, the data was filtered to remove Internet backscatter by examining each flow individually against common backscatter flags, such as TCP RST and TCP SYN/ACK [19].

C. Candidate feature selection

We have extracted 18 features from the activity flows. These traffic features were selected as being representative of the behavior of the three protocols that are monitored by the honeypot, namely TCP, UDP and ICMP. Traffic features computed from the activity flows include: the total number of basic flows generated by individual IPs and aggregated based on sixty minutes; total number of open TCP ports targeted; total number of distinct open TCP ports targeted; total number of open UDP ports targeted; total number of distinct open UDP ports targeted; total number of closed UDP ports targeted; total number of distinct closed UDP ports targeted; total number of ICMP flows; number of machines targeted per attack; total duration of basic flows; total number of source packets sent per IP; total number of source bytes sent; total source rates which is the sum of the source rates of all the basic flows, where a source rate is number of source packets in a basic flow divided by the duration of that flow; sum of the average packet size per basic flow; total activities as the summation of source and destination rates; and summation of inter-arrival times between basic flows.

V. Model architecture

The architecture of the detection model is depicted in Figure 1. As the figure shows, the model consists of three main components:

- **Traffic Flow Aggregator:** The traffic flow aggregator accepts Argus traffic flows [20], set to 5-minute maximum expiration, and then groups the traffic flows into the activity flows. The newly generated flows, activity flows, are combined by the source IP address of the attacker with a maximum of 60 minutes inter-arrival time between original flows. Internet noise, such as backscatter, is filtered out in this model.
- **PCA Model Extraction:** is where the PCA profile is built from historical honeypot data. This includes the calculation of the correlation matrix, the extraction of the Eigenvectors and Eigenvalues, and the generation of principal components.
- **Detection:** in the detection model, new observations are tested against the predefined PCA model parameters for detecting new attacks.

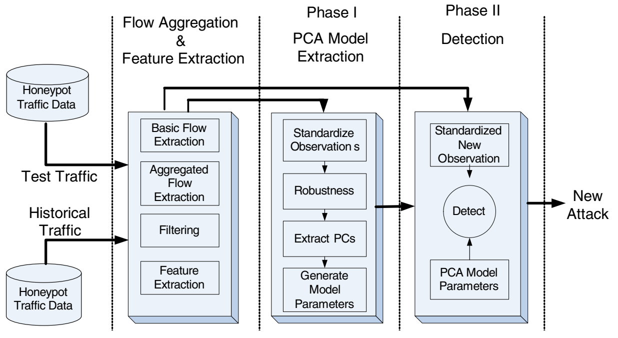

Figure 1. Detection model architecture

Our methodology of detecting new attacks in low-interaction honeypot traffic is adapted from multivariate statistical process control (MSPC), a widely used statistical technique in monitoring production processes in industries, such as chemical industries, to detect manufacturing process faults. The detection model proposed in this paper is performed in two phases:

- Phase I: building a PCA profile of the honeypot traffic from historical data over a defined period of time. This includes the calculation of the correlation matrix, the extraction of the Eigenvectors and Eigenvalues, and the generation of principal component scores.
- Phase II: Detecting new attacks where new observations are projected onto the residuals of the predefined PCA model and are tested against a predefined threshold.

A. Processing the flow traffic via PCA

Principal component analysis can be performed using either the covariance matrix or the correlation matrix. However, PCs defined using the covariance matrix are very sensitive to the unit of measurements and are difficult to interpret. In addition, when the variance of the variables differs widely, which is the case for the honeypot data, the first few PCs will be dominated by variables with high variances, as they contribute little information to the structure of the data set. Thus, the use of the correlation matrix rather than the covariance matrix for deriving the PCs was preferred in our analysis. To calculate the PCs from the correlation matrix, the p-dimensional vector$X = (X_1, .., X_p)^T$ is first standardized by:

$$C_i = \frac{X_i - \bar{X}_i}{\sqrt{S_i}} \tag{2}$$

for i=$1,\ldots,p$, where $\bar{X}_i$ is the sample mean and S_i is the sample variance for X_i. Let R be the sample correlation matrix of C with Eigenvalue vector l=$(l_1,\ldots,l_p)$, then the principal component analysis

$$Z = A^T C \tag{3}$$

Table II
EXTRACTED PRINCIPAL COMPONENTS

Principal Component	Eigenvalues	% of Variance	Cumulative %
1	5.410	30.054	30.054
2	2.374	13.190	43.244
3	2.153	11.959	55.204
4	1.681	9.339	64.543
5	1.432	7.954	72.497
6	1.362	7.567	80.064
7	.959	5.329	85.393

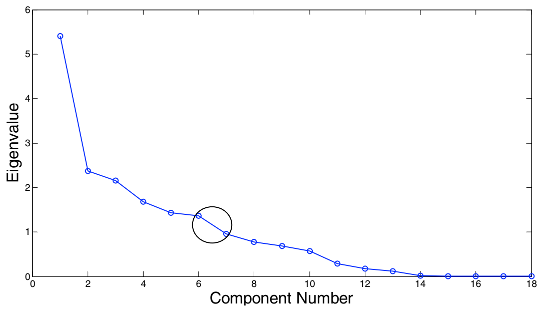

Figure 2. Scree plot of Eigenvalues

where $A= (A_1, .., A_k)^T$ is the Eigenvectors of R, with the first component equal to:

$$Z_1 = a_{11}C_1 + a_{12}C_2 + \cdots + a_{1p}C_p \tag{4}$$

Three factors were considered when we selecting the number of principal components (PCs) that are representative of the variables: the Kaisers' rule [2] for eliminating PCs with Eigenvalue less than one (see Table 2 for the Eigenvalues); the extracted commonalities of variables or the amount of variance within each variable accounted for by the components; the Scree plot of energy contributed by each PC, see Figure 2. All of the above suggest that seven principal components retain over 90% of the total variance of the original data.

B. Robustness

Building a staple detection model requires that the model parameters to be estimated from a clean data set. Extracting principal components from a standard correlation matrix is very sensitive to outliers, where the resulting principal components might be determined by their directions. An effective technique of improving the principal component analysis and reducing the effect of these outliers is through the robustification of the correlation matrix during the model building phase, Phase I. The robustification works by eliminating observations with large mahalanobios distance in an iterative process until the data is believed to be clean or the given number of iterations is reached [21].

Given a p-dimensional random vector$X = (X_1, .., X_p)^T$ of n samples, where $\bar{X}$ is the sample mean and S is the sample variance of X_i. Then T^2, given by Eq. 6, is an ellipsoid in the p-dimensional space which is centered at the mean $\bar{X}$ and the distance to its surface is given by T^2 values(a contour of constant probability). The constant probability contour for the distribution of X satisfies $T^2 \leq \chi^2_p(\alpha)$, where $\chi^2_p(\alpha)$ is the percentile of a chi-square distribution with p degrees of freedom. Setting a threshold for detecting outlying observations based on $\chi^2_p(\alpha)$ requires the distribution of X to be a multivariate normal. However, since we do not make any assumptions about the distribution of our data, the population ellipsoid is still valid despite any normality assumptions, but the ellipsoid loses its interpretation as contours of constant probability of the distribution of X [2]. Accordingly , the threshold for the robustification process can be determined from the empirical distribution of T^2.

C. Setting up model parameters

A critical step in designing a detection approach is setting the limit for judging new observations since it has a dramatic effect on the quality of the detection. When the limit value is very narrow, it will frequently be exceeded resulting in a high rate of false positive alarms, and when the limit is very wide the limit will never be exceeded, resulting in many false negative alarms.

Let X be a data matrix of n samples of p-dimensional random variables, where $\bar{X}$=$(\bar{X}_1,..,\bar{X}_p)^T$ is the sample mean vector of X and R is the sample correlation matrix. The sum of the squares of the weighted principal component scores of the last q principal components, the residual space, in detecting outliers is given by:

$$Q_i = \sum_{k=p-q+1}^{p} \frac{Z_{ik}^2}{l_k} \tag{5}$$

where q<p and Z_{ik} is the score of the k^{th}PC of the i^{th} observation and $l_k$is the k^{th}Eigenvalue. When q=p, the previous equation can be represented by of the distance of the i^{th}observation from the mean of the data, which is given by:

$$T_i^2 = (x_i - \bar{x})^T S^{-1} (x_i - \bar{x}) \tag{6}$$

Then T^2 follows a chi-square distribution χ^2, for larger sample size, with p degrees of freedom [22]. Thus the upper control limit becomes:

$$UCL = \chi^2_{1-\alpha,p} \tag{7}$$

where $\chi^2_p(\alpha)$is the percentile of a chi-square distribution with p degrees of freedom.

Although we do not make any assumption about the exact distribution of each of the p variables and we are only interested in a large values of T^2, the upper limit would be computed from the empirical distribution of the T^2 population as follows:

$$UCL = u + 3s \tag{8}$$

where u is the sample mean and s is the standard deviation.

Setting the control limit as a multiple of a standard deviation is a common practice and gives good practical results [23]. The T^2 test is equivalent to using all principal components in Eq. 5. We use the T^2 test in Phase I to reduce the effect of outliers. Square prediction error (SPE), or the Q-statistic, is a test

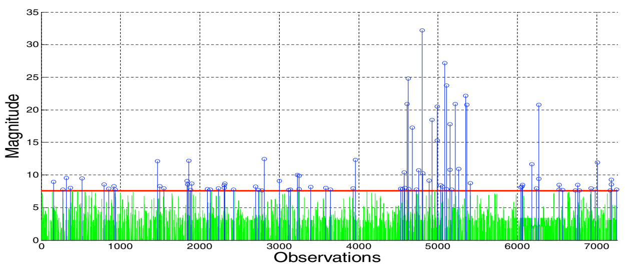

Figure 3. Plot of four-month attack data projected onto the residual space

of how a particular observation fits the principal component model. SPE is calculated from the sum of squares of the residuals and it measures the distance from the observation to the k-dimensional hyperspace defined by the PCA model. A high value of SPE indicates that the new observation represents a new direction that is not included in the PCA model. The Q-statistic of the residual space can be represented by the sum of the squares of the weighted principal component scores of the last $p-q$ principal components in Eq. 5. The upper limit for Q is given by [3]:

$$Q_\alpha = \theta_1 \left[\frac{C_\alpha \sqrt{2\theta_2 h_0^2}}{\theta_1} + 1 + \frac{\theta_2 h_0 (h_0 - 1)}{\theta_1^2} \right]^{\frac{1}{h_0}} \tag{9}$$

where $C_\alpha$is the normal deviate corresponding to the upper $(1-\alpha)$ percentile, $\theta_i = \sum_{i=k+1}^{p} l_i^j$, for *j= 1, 2, 3*, and $h_0 = 1 - \frac{2\theta_1\theta_3}{3\theta_2^2}$

The use of the upper limit in Eq. 9 assumes that the data is normally distributed. Alternatively we set the upper limit based on the empirical distribution of Q of the sample data. The Q statistic is used in Phase II to detect new attacks with our detection model.

VI. Results

A. Detection and identification

Four months of real attack evaluation data was extracted from the honeypot environment and projected into the residual space, Data set II in Section IV-A. Figure 3 illustrates the SPE projection of the data stet.

The projection shows observations that have high SPE values as spikes that rise above the threshold value. These observations are possibly new attacks and require further investigations. As the figure shows, there are 81 observations that have been flagged by our detection algorithm which violate the structure of the attack model. Moreover, the figure shows intense attack activities that are obvious in the figure along the *X axis* around observation 5000. These activities reflect a single class of attacks that one of our honeypot sensors had experienced in late February and early March of 2008. Details of these attacks and the rest of the attack activities are discussed in section VI-D, Evaluation by manual inspection of the detected traffic.

B. Stability of the monitoring model over time

As our technique uses a historical block of data to construct the PCA detection model, it is very important to evaluate the stability of the PCA model over time. Our preliminary investigation, which include the stability of the estimated mean vectors and the correlation matrix, suggests that there is a slight variation in some of the variables' means and also in the amount of variance contained in the first seven components. The number of significant components is still the same for both of the data sets, however the amount of variation in the first seven components increased slightly in data set II. While we consider that these changes have little effect on the residual space, a further investigation is required to assess how these changes would affect the reliability of the model in terms of detection rates. Moreover, further work is required in order to study ways to adaptively update the model parameters, such as the mean and the correlation matrix over time.

C. Computational requirements

The detection model was developed using a combination of the open-source programming language Python and the high level scientific computing environment Matlab. The Python language was utilized in developing the flow aggregator while Matlab was used in the development of the detection engine. Tasks that are required by the detection model include: standardizing the data, calculating the correlation matrix, finding the Eigenvectors and the Eigenvalues, extracting the PC scores, and computing the T^2 and Q statistics. Let X be a data matrix of n samples of p dimensional random variables, then the computational requirements for computing the correlation matrix of X is $O(np^2)$ and the extraction of the Eigenvectors and Eigenvalues are $O(p^3)$ [24]. The computational requirements are mainly matrix manipulations and are not considered expensive taking into account the massive reduction in data records using our aggregation technique of flows. Table III shows the empirical execution times required by a number of components of the detection model. The detection model was tested on a personal computer with a 2.0 GHz Intel dual core processor.

Table III
Average execution times of the major tasks (Seconds)

Tasks involved in the detection	Time
Calculating the correlation matrix for all data	0.070073
Finding the Eigenvectors and Eigenvalues of the correlation matrix	0.001322
Calculating the PC scores of X	0.003049
T^2 Test of one vector for all PCs (18 PCs)	0.000142
Q Test of one vector for the residual (11 PCs)	0.000088

To detect new attacks, only the Q statistics needs to be calculated using parameters from the historical data.

D. Evaluation

Broadly speaking, there are two methods for validating an attack detection model. The first validation method consists of manual labeling of attacks in a data set and then testing the model performance, false positive and false negative criteria, against this labeled data. The second validation approach is based on testing the model performance against synthetically crafted attacks that are manually injected into the data set. Applying these classical validation methods to a low-interaction honeypot detection model is challenging as the notion of normal and abnormal does not apply, the nature of the collected traffic which is considered malicious per se, and the low level of detail available from the collected traffic. In this section, we detail our evaluation methodology of the detection model using the data set described in Section IV-A.

Evaluation by manual inspection of the detected traffic: To help better understand the nature of the detected observations and judge their significance, we have carried out a manual inspection of every observation that was flagged by our detection algorithm, 81 observations in total. Our aim is to explain the reasons that these observations were flagged by our detection algorithm and to group them according to their similarities into different classes. The manual validation process consists of manual inspection and manual classification of these detected attack observations.

Firstly, we examined all of the 18 traffic features, that have been used by the algorithm, and then went further by checking the basic flows for other patterns of attacks such as destination ports, protocols, and flags. After that, observations were also checked against the original logs. Secondly, observations were grouped together into different types of attack clusters based on their attack port similarities, or port sequences. The port sequence is a list of targeted honeypot ports that are generated by a single IP address during the attack period. An example of a port sequence of an attack that is generated by an IP address which targeted TCP port 139 ,445, 998, 139, 445, and UDP 137 with ICMP traffic would be: {T139,T445,T998,U137,I}.

Our manual inspection of the detected traffic has found eight clusters of attack activities. Table IV provides a brief summary of the results. As Table IV shows, there are four types of activities that were classified as worm attacks. The first class, Worm Activity I, is the largest with a port sequence (T139,T445,T9988,ICMP). This class represents repeated attempts that target two open TCP ports, 139 and 445, and a single TCP closed port, 9988. These activities resemble a well known Rahack worm [25], which targets Microsoft OSs.

The second class of worm activities is distinguished by its port sequence (T1080, T3128, T80, T8080, T10080, ICMP). The pattern of these activities' port sequence is similar to Mydoom worm family [26]. The third class of worm activity, port sequence (T445, T135, T1433, T139, T5000), is another automated exploit that targets a Microsoft Windows LSASS vulnerability [27]. The last worm class of activities targets TCP port 5900. This class of activities is mainly scans for Trojans that listen for remote connections on TCP Port 5900, such as Backdoor.Evivinc [28].

The denial of service activities class comes second in terms of number of observations. The attacking IPs targeted a single machine on a single open TCP port, port 80, with very short time between packets. These attacks were detected by our algorithm as the total activities of the source IP were huge in addition to other parameters such as number of source packets sent. The attack mainly caused by a few IP addresses over the period from 20/02/2008 to 04/03/2008.

Table V
SYNTHETIC ATTACK CLASSES

Attack Class	Description
SYN-Flooding	A Combination of low and high rate of SYN-Flooding attacks, using Hping3
Scan	Default Nmap SYN scans
OS fingerprinting	Operating system identification and TCP/IP signatures using Xprobe2
UDP attack	UDP buffer overflow attack against open UDP port
Vulnerability scan	Nessus vulnerability scanns

The third class of activities that is detected by our model is scan activities. While low to moderate scanning activities are very common in our log files, these activities were flagged by our algorithm as they generated large values on single or multiple features. Mis-configuration class of activities is mainly a DHCP request on UDP port 53. The last class of activities, Miscellaneous, consists of all observations that we were not able to explain and did not fit in any class. This class of activities represent short attacks on non standard single TCP, single UDP ports or both.

Evaluation by synthetic data: Our evaluation methodology to validate our attack model consists of two parts, manual generation of attacks in a controlled environment and the evaluation of the detection model against these attacks. The synthetic attack scenarios are played against our test environment, a replica of our honeypot environment that runs under VMware, logged using Tcpdump, and then processed using the technique detailed in Section IV. Five attack scenarios were played against our test environment, Table V details these scenarios.

Due to the large amount of traffic data and the low interactivity of the honeypot, it is very difficult to be assured that the synthetic attack data does not exist in the training data. We have conducted a simple, yet an effective methodology of inserting every synthetic attack that was flagged by the detection algorithm into the training data set, reconstructing the attack model, and then projecting these attacks again into the residual space. For all of the synthetic attacks that were detected as new, the new projection has produced very low *SPE* values, after their inclusion in the training data.

The validation results show that all the synthetic attacks were detected by our technique as not being seen in the training data. As mentioned earlier, manual confirmation of the non existence of these attacks in the training data is not feasible through manual inspection. As a result, we included these attacks in the original training data and reconstructed a new attack model, which we are certain that these attacks were accounted for. When these attacks were projected into the residual space of the new model, the Q values were small and were under the threshold value. These validation results

Table IV
CLASSES OF THE DETECTED ATTACK ACTIVITIES

Activities Class	Distinct Behaviors	Possible Type	No.
Worm Activity I	Moderate: TF, TCP_O Low: TCP_C, IAT	W32.Rahack.W worm	26
Worm Activity II	Moderate: TF, TCP_C Low: TCP_O	Mydoom worm family	6
Worm Activity III	Moderate: TF, TCP_C High: AVG_PK_SIZE	Bobax worm family	1
Worm Activity V	Low:TF High: IAT	Backdoor.Evivinc	2
Denial of Service	High: TF, Dur, TCP_O, SPackets,T_ACT Short: IAT	Distributed DOS or DOS	21
Scan Activities	Large: TF, TCP_C, SPackets Moderate: TCP_O, ICMP	Horizontal scan or machine detection	2
Mis-configuration	Low: TF,UDP_C	DHCP request	8
Miscellaneous	LOW: TF, TCP_C,UDP_C	Unknown	15

confirm that our detection model is capable of detecting new attacks that are either new or not present in the training data.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a technique for detecting new attacks in low-interaction honeypot traffic. The proposed detection method is performed in two phases. Firstly, traffic flows are grouped based on IP addresses and then PCA profile of honeypot traffic is built. Secondly, new traffic vectors are projected onto the residual space of the PCA attack model and the square prediction error (SPE) statistic is used to flag new attacks based on their large deviations from the attack model. The effectiveness of the proposed technique is demonstrated through the analysis of real traffic data from the Leurré.com project and is validated through the use of synthetic attack data. Our evaluation results show that our technique is capable of detecting different types of attacks with no prior knowledge of these attacks and the technique has low computational requirement that makes it suitable for online detection systems.

Future work includes automating the extraction of the detection parameters and improving the model capability to adapt over time to changes in the correlation structure. Another area for improvement is to develop attack classification models to help in finding the actual class of the detected attacks automatically, which would ease the interpretations.

REFERENCES

[1] L. Spitzner, *Honeypots: Tracking Hackers*. Addison-Wesley, 2003.

[2] I.T.Jollif, *Principal Component Analysis*, 2nd ed., ser. Springer Series in Statistics. New York: Springer, 2002.

[3] J. E. Jackson, *A User's Guide to Principal Components*, 1st ed. Wiley-Interscience, 2003.

[4] K. Labib and V. R. Vemuri, "An application of principal component analysis to the detection and visualization of computer network attacks," *Annals of Telecommunications*, pp. 218–234, 2005, Nov-Dec Issue.

[5] Y. Bouzida, F. Cuppens, N. Cuppens-Boulahia, and S. Gombault, "Efficient intrusion detection using principal component analysis," in *3éme Conférence sur la Sécurité et Architectures Réseaux (SAR)*, La Londe, France, Jun. 2004.

[6] N. Provos, "A virtual honeypot framework," in *13th USENIX Security Symosium*, Aug 2004.

[7] C. Leita, K. Mermoud, and M. Dacier, "Script gen: An automated script generation tool for honeyd," in *21st Annual Computer Security Applications Conference (ACSA)*, Dec 2005.

[8] C. Kreibich and J. Crowcroft, "Honeycomb - creating intrusion detection signatures using honeypots," in *Proceedings of the Second Workshop on Hot Topics in Networks*, 2003.

[9] F. Pouget and M. Dacier, "Honeypot-based forensics," in *AusCERT Asia Pacific Information technology Security Conference*, 2004.

[10] J. Zimmermann, A. Clark, G. Mohay, F. Pouget, and M. Dacier, "The use of packet inter-arrival times for investigating unsolicited internet traffic," in *First International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE'05)*, 2005 2005.

[11] S. Almotairi, A. Clark, M. Dacier, C. Leita, G. Mohay, V. H. Pham, O. Thonnard, and J. Zimmermann, "Extracting inter-arrival time based behaviour from honeypot traffic using cliques," in *The 5th Australian Digital Forensics Conference*, Perth, Australia, 2007.

[12] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. Kolaczyk, and N. Taft, "Structural analysis of network traffic flows," in *ACM SIGMETRICS*, 2004.

[13] S. Almotairi, A. Clark, G. Mohay, and J. Zimmermann, "Characterization of attackers' activities in honeypot traffic using principal component analysis," in *Network and System Security NSS 2008*. IEEE Computer Society Proceedings, Oct 2008.

[14] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang, "A novel anomaly detection scheme based on principal component classier," in *Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop, in conjunction with the Third IEEE International Conference on Data Mining (ICDM'03)*, 2003, pp. 172–179.

[15] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2004.

[16] A. Lakhina, M. Crovella, and C. Diot, "Characterization of network-wide anomalies in traffic flows," in *ACM-SIGCOMM Internet Measurement Conference*, 2004.

[17] Network Security Team - Eurecom, "The leurre.com project home page," available http://www.leurrecom.org/ Last Visited 30/10/2008.

[18] Cisco Systems. (2007) Cisco ios netflow. [Online]. Available: http://www.cisco.com/en/US/products/ps6601/products_ios_protocol_group_home.html

[19] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, "Inferring internet denial-of-service activity," *ACM Transactions on Computer Systems*, vol. 24, no. 2, pp. 115–139, 2006.

[20] qosient, LLC. (2007) Argus-client 2.0.6. [Online]. Available: http://www.qosient.com/argus/

[21] R. Gnanadesikan, *Methods for Statistical Data Analysis of Multivariate Observations*, 2nd ed. New York: Wiley-Interscience Publication, 1997.

[22] S. Bersimis, S. Psarakis, and J. Panaretos, "Multivariate statistical process control charts: an overview," *Quality and Reliability Engineering International*, vol. 23 Issue 5, no. 5, pp. 517–543, 2006.

[23] D. C. Montgomery, *Introduction to Statistical Quality Control*, 4th ed. New York: John Wiley & Sons, Inc, 2004.

[24] S. Roweis, "Em algorithms for pca and spca," in *Advances in Neural Information Processing Systems*, M. press, Ed., vol. 10, 1998, pp. 626–632.

[25] Symantec Security Response. (2007, Jan.) W32.Rahack.W. [Online]. Available: http://www.symantec.com/security_response/writeup.jsp?docid=2007-011509-2103-99&tabid=2

[26] Symantec Security Response. (2004, Jan.) W32.Mydoom.B. [Online]. Available: http://www.symantec.com/security_response/writeup.jsp?docid=2004-012816-3647-99&tabid=2

[27] Bitdefender. (2004, May) Win32.Worm.Bobax. [Online]. Available: http://www.bitdefender.com/VIRUS-1000045-en--Win32.Worm.Bobax.A-C.html

[28] Symantec Security Response. (2007, Feb.) Backdoor.Evivinc. [Online]. Available: http://www.symantec.com/security_response/writeup.jsp?docid=2004-042518-0520-99&tabid=2